INTRUDER DETECTION SYSTEM FOR RPL NETWORK

PROJECT REPORT

PHASE-II

Submitted by

VISAKA C

Register No: 15MCO014

in partial fulfillment for the requirement of award of the degree

of

MASTER OF ENGINEERING

in

COMMUNICATION SYSTEMS

Department of Electronics and Communication Engineering

KUMARAGURU COLLEGE OF TECHNOLOGY

(An autonomous institution affiliated to Anna University, Chennai)

COIMBATORE-641049

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2017

BONAFIDE CERTIFICATE

Certified that this project report titled **"INTRUDER DETECTION SYSTEM FOR RPL NETWORK"** is the bonafide work of **VISAKA C [Reg. No. 15MCO014]** who carried out the research under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr. S.N. SHIVAPPRIYA

PROJECT SUPERVISOR

Department of ECE

Kumaraguru College of Technology

Coimbatore-641 049

SIGNATURE

Prof. K. RAMPRAKASH

HEAD OF PG PROGRAMME

Department of ECE

Kumaraguru College of Technology

Coimbatore-641 049

ACKNOWLEDGEMENT

First, I would like to express my praise and gratitude to the Lord, who has showered his grace and blessings enabling me to complete this project in an excellent manner.

I express my sincere thanks to the management of Kumaraguru College of Technology and Joint Correspondent **Shri Shankar Vanavarayar** for the kind support and for providing necessary facilities to carry out the work.

I would like to express my sincere thanks to our beloved Principal **Dr.R.S.Kumar,** Kumaraguru College of Technology, who encouraged me with his valuable thoughts.

I would like to thank **Prof.K.Ramprakash,** Head of PG programme, Electronics and Communication Engineering, for his kind support and for providing necessary facilities to carry out the project work.

I am greatly privileged to express my heartfelt thanks to my project guide **Dr.S.N.Shivappriya,** Assistant Professor II, Department of Electronics and Communication Engineering, for her expert counseling and guidance to make this project to a great deal of success.

In particular, I wish to thank with everlasting gratitude to the project coordinator **Dr.M.Bharathi,** Professor, Department of Electronics and Communication Engineering, throughout the course of this project work. I wish to convey my deep sense of gratitude to all teaching and non-teaching staff of ECE Department for their help and cooperation.

Finally, I thank my parents and my family members for giving me the moral support and abundant blessings in all of my activities and my dear friends who helped me to endure my difficult times with their unfailing support and warm wishes.

ABSTRACT

The Routing for Low Power and Lossy Networks (RPL) is a standard routing protocol, has been proposed by the IETF ROLL working group for addressing the properties and constraints of the networks. However, in this protocol there is security concerns like being exposed to a large variety of attacks. Their consequences can be significant in terms of network performance and resources. In this project, development of an Intruder Detection System (IDS) against RPL attacks is done. As a first step, communications between motes (wireless sensor nodes in the COOJA network simulator) are established. Most common attacks in RPL are flooding attack, versioning attack and sinkhole attack. As the second step, any one kind of or combination of two or more attacks (as mentioned earlier) are been created by one of the mote to act as an intruder in the network, considering three categories such as attacks targeting network resources, attacks modifying the network topology and attacks related to network traffic. The description of these attacks, analyzes is done and comparison of various properties with the help of contiki os using cooja simulator. As the final step, resilience against the attack is been carried out with the intruder detection system (IDS). The discussion of the counter measures and usage of the IDS from risk management perspective are explained.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Overview of Wireless Sensor Network

A Wireless Sensor Network (WSN), also known as wireless sensor and actor network (WSAN) are spatially distributed autonomous sensors to monitor physical or environmental conditions, such as temperature, pressure, sound, etc and to cooperatively pass their data through the network to a main location. Figure 1.1 shows one of the examples of wireless sensor network. Power is stored either in batteries or capacitors. Batteries (both rechargeable and non-rechargeable) are the main source of power supply for sensor nodes.

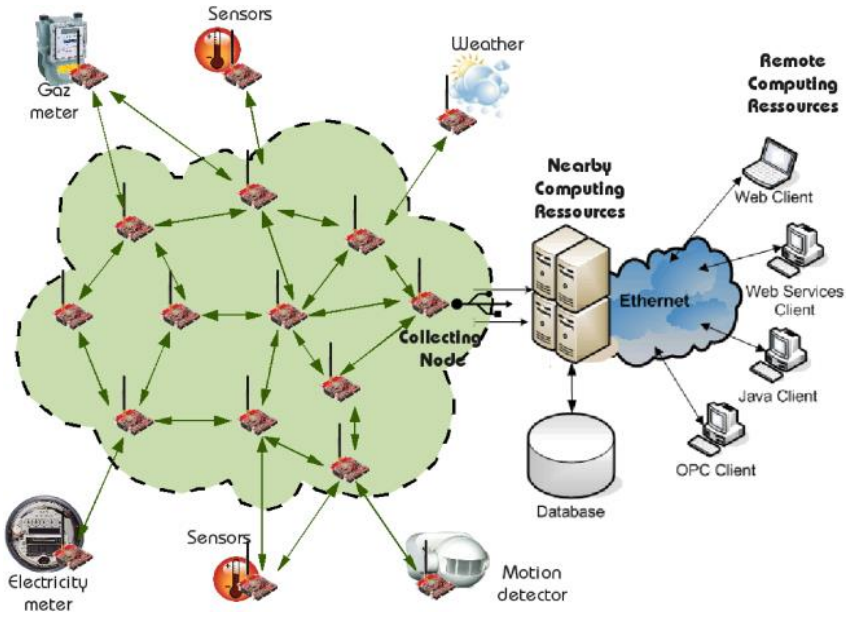

Figure 1.1 Wireless Sensor Network

In the current scenario, the Low Power and Lossy networks (LLNs) have a wide scope of applications in military and civilian, such as healthcare, environmental monitoring and military surveillance [1]. LLNs comprises of embedded devices with constrained resources in terms of processing power, memory size and energy. Due to this specific routing requirement of LLNs, IETF ROLL working group has proposed Routing Protocol for Low power and Lossy Networks (RPL).

1.2 Routing Protocol for Low power and Lossy Networks (RPL)

RPL is a distance-vector and source routing protocol (uses IPv6) designed to work on multiple link layers, which is hierarchical and proactive. The RPL topology is based on Directed Acyclic Graph (DAG) which represents a tree like structure; unlike trees its nodes can be associated with more than one parent. RPL network organizes as Destination Oriented DAG's (DODAG's). Therefore RPL uses IPv6 control messages like DODAG Information Solicitation (DIS), DODAG Information Object (DIO), and Destination Advertisement Object (DAO) [2].

1.3 Contiki Operating System

1.3.1 Introduction

Contiki is a wireless sensor network operating system and consists of the kernel, libraries, the program loader, and a set of processes. It is used in networked embedded systems and smart objects.

Contiki provides mechanisms that assist in programming the smart object applications. It provides libraries for memory allocation, linked list manipulation and communication abstractions. It is the first operating system that provided IP communication. It is developed in C, all its applications are also developed in C programming language, and therefore it is highly portable to different architectures like Texas Instruments MSP430.

Contiki is an event-driven system in which processes are implemented as event handlers that run to completion. A Contiki system is partitioned into two parts: the core and the loaded programs. The core consists of the Contiki kernel, the program loader, the language run-time, and a communication stack with device drivers for the communication hardware [3].

The Program loader loads the programs into the memory and it can either obtain it from a host using communication stack or can obtain from the attached storage device such as EEPROM.

The Contiki operating system provides modules for different tasks (layers). It provides the routing modules in a separate directory "contiki/core/net/rpl" and consists of a number of

files. These files are separated logically based on the functionalities they provide for instance rpl-dag.c contains the functionality for Directed Acyclic Graph (DAG) formation, rpl-icmp6.c provides functionality for packaging ICMP messages etc.

1.3.2 COOJA simulator

Cooja is a Java-based simulator designed for simulating sensor networks running the Contiki sensor network operating system. The simulator is implemented in Java but allows sensor node software to be written in C.

One of the differentiating features is that Cooja allows for simultaneous simulations at three different levels: Network Level, Operating System Level and Machine code instruction level. Cooja can also run Contiki programs either compiled natively on the host CPU or compiled for MSP430 emulator.

In Cooja all the interactions with the simulated nodes are performed via plugins like Simulation Visualizer, Timeline, and Radio logger. It stores the simulation in an xml file with extension 'csc' (Cooja Simulation Configuration). This file contains information about the simulation environment, plugins, the nodes and its positions, random seed and radio medium etc. Cooja Simulator runs the Contiki applications whose files are placed in another directory and may also contain a "project-conf.h" file which provides the ability to change RPL parameters in one place.

1.3.3 Attack in RPL

For introducing attack against the RPL network, it is necessary to investigate the protection capabilities of the RPL protocol against the well known security attacks presented for WSNs. Multiple attacks will be created in the RPL network to check whether the protocol can counter these attacks and/or mitigate their impact using the IDS.

1.3.4 Attack Implementation

Routing attacks are implemented in the RPL network which makes use of Contiki OS, therefore Contiki RPL storing nodes uses network routing where nodes keep track of descendants. The μIP is an IP stack in the Contiki OS, is utilized for providing IP communication

in RPL network. For demonstrating attacks against a simulated RPL network Cooja simulator is being used. In this simulation, emulated Tmote Sky nodes running Contiki RPL in the Contiki OS is implemented [4].

1.3.5 Intrusion Detection Systems (IDS)

An Intrusion Detection System (IDS) analyzes activities or processes in a network or in a device and detects attacks, reports them and/or mitigates the harmful effect of the detected attacks. Due to the diversity of attacks and the unpredictable behavior of novel attacks, IDSs are subjected to false positives (indicating false information when there is no attack) and false negatives (not indicating any information when there is an attack). Generally, there are two categories of IDSs: anomaly based and signature based.

Anomaly based detections determine the ordinary behavior of a network or a device, use it as a baseline, and detect anomalies when there are deviations from the baseline. This approach can detect new attacks but has comparatively high false positive and false negative rates because it may raise false report and/or cannot detect attack when attacks only show small deviations from the baseline. The functionality of the IDS is to build a global view of the network and as a consequence the possibility to detect incoherence in the network [5].

Signature based detections compare the current activities in a network or in a device against predefined and stored attack patterns called signatures. This approach cannot detect new attacks, needs specific knowledge of each attack, has a significant storage cost that grows with the number of attacks, and has a high false positive but low false negative rate.

In this project, the signature based intruder detection is done, multiple attacks are combined to form a single attack and the IDS is designed in such a way that it detects the attack detected and mitigates the harmful effects created by the intruder.

CHAPTER 2

LITERATURE SURVEY

2.1 Introduction

The literature survey gives a brief on the resources that have been studied for better understanding of the current research trends in the area of proposed research.

2.2 Contiki Operating System Study

Contiki OS is an open source, highly portable, multi-tasking operating system for memory-efficient networked embedded systems and wireless sensor networks. Contiki is designed for microcontrollers with small amounts of memory. A typical Contiki configuration is 2 kilobytes of RAM and 40 kilobytes of ROM. Contiki provides IP communication, both for IPv4 and IPv6. Many key mechanisms and ideas from Contiki have been widely adopted in the industry. The uIP embedded IP stack, released in 2001, is today used by hundreds of companies in systems such as freighter ships, satellites and oil drilling equipment. Contiki and uIP are recognized by the popular nmap network scanning tool. Contiki's proto threads, first released in 2005, have been used in many different embedded systems, ranging from digital TV decoders to wireless vibration sensors.

Contiki introduced the idea of using IP communication in low-power sensor networks. This subsequently led to an IETF standard and the IPSO Alliance, an international industry alliance. Contiki is developed by a group of developers from industry and academia lead by Adam Dunkels from the Swedish Institute of Computer Science. The Contiki team currently consists of sixteen developers from SICS, SAP AG, Cisco, Atmel, New AE and TU Munich [6].

Contiki contains two communication stacks: uIP and Rime. uIP is a small RFC-compliant TCP/IP stack that makes it possible for Contiki to communicate over the Internet. RIME is a lightweight communication stack designed for low-power radios. Rime provides a wide range of communication primitives, from best-effort local area broadcast, to reliable multi-hop bulk data flooding. Contiki runs on a variety of platform ranging from embedded microcontrollers such as the MSP430 and the AVR to old home computers. Code footprint is on the order of kilobytes and

memory usage can be configured to be as low as tens of bytes. Contiki is written in the C programming language and is freely available as open source under a BSD-style license [5, 6].

2.3 Enhancing RPL Resilience against Routing Layer Insider Attacks

Resilience is needed to mitigate such inherent vulnerabilities and risks related to security and reliability. Routing Protocol for Low-Power and Lossy Networks (RPL) is studied in presence of packet dropping malicious compromised nodes. Random behavior and data replication have been introduced to RPL to enhance its resilience against such insider attacks [7].

The classical RPL and its resilient variants have been analyzed through COOJA simulations and hardware emulation. Resilient techniques introduced to RPL have enhanced significantly the resilience against attacks providing route diversification to exploit the redundant topology created by wireless communications.

2.4 Establishment of resilience for RPL

To establish the resilience, Randomized RPL and secure RPL have to be used. In Randomized RPL, parent should be selected for each node in a random manner. In Secure RPL, secure control messages such as SDIO, SDIS, and SDAO are used with the help of lightweight cryptographic algorithms. In this paper, the attacks are classified based on exploiting network resources, modifying network topology and access to network traffic and express the scenario of security solutions available only in the proof-of-concept level. The impact of the attacks results in significant degradation of network performance in terms of delivery ratio, fairness and connectivity, convergence time, power consumption and network life time. The proposed IDS is expected to exhibit better network performance with optimum energy utilization and protect the network from multiple attacks.

2.5 RPL Protocol Study - IETF Draft

The basic construct in RPL is a Destination Oriented Directed Acyclic Graph (DODAG). In a converged LLN, each RPL router has identified a stable set of parents, each of which is a potential next-hop on a path towards the root of the DODAG, as well as a preferred parent. Each router, which is part of a DODAG will emit DODAG Information Object (DIO) messages, using

link-local multicast, indicating its respective rank in the DODAG (i.e. distance to the DODAG root according to some metric(s), in the simplest form hop-count).Upon having received a number of such DIO messages, a router calculates its own rank such that it is greater than the rank of each of its parents, select a preferred parent and then start emitting DIO messages [8].

The DODAG formation starts at the DODAG root (initially, the only router which is part of a DODAG), and spreads gradually to cover the whole LLN as DIOs are received, parents and preferred parents are selected and further routers participate in the DODAG. The DODAG root also includes, in DIO messages, a DODAG Configuration Object, describing common configuration attributes for all RPL routers in that network (including their mode of operation, timer characteristics etc). RPL routers in a DODAG include a verbatim copy of the last received DODAG Configuration Object in their DIO messages, permitting also such configuration parameters propagating through the network [7, 8].

2.6 RPL optimization for precise green house management using wireless sensor network

Wireless sensor networks (WSN) have been employed in many applications, due to their simplicity in connectivity, cooperative networking ability and long battery life in deployed nodes. Due to the proactive nature, memory constraints and processing capability issues in Routing Protocol for Low Power Lossy Networks (RPL), the Directed Acyclic Graph (DAG) formation with root node at apex and the node discovery using the routing table repair is done using RPL triple timer mechanism. And also the DIO Interval minimum, DIO Doubling Rate and DIO Redundancy Rate parameters of RPL plays a critical role in deciding network convergence and control overhead. Optimization involves obtaining the level of above parameters using Contiki OS, it is developed specially for networks with memory constrained and power constrained systems and routing protocol has to be optimized for any specific deployment.

The precision farming techniques provide possibility to grow money crops like capsicum, nursery plants when the temperature and humidity is under controlled condition. Wireless sensor networks (WSN) can be used for automatic monitoring and controlling soil humidity and temperature inside the green house. The complete architecture required for practical implementation of green house monitoring is proposed and validated using real time simulation.

CHAPTER 3

ROUTING PROTOCOL FOR LOW POWER AND LOSSY NETWORK (RPL)

3.1 RPL Introduction

RPL (Routing Protocol for low power and Lossy network) is routing protocol used at the network layer. RPL is a distance-vector and a source routing protocol. The key aspect of RPL is to maintain network state information using one or more directed acyclic graphs (DAGs).

3.2 DODAG Construction

A DAG is a directed graph wherein all edges are oriented in such a way that no cycles exist. Each DAG created in RPL has a root node which acts as a gateway. Each node (client node) in the DAG is assigned a rank that is computed on the basis of an objective function. Figure 3.1 shows a simple DODAG.

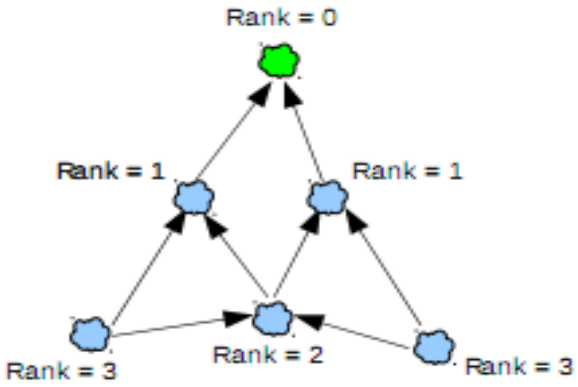

Figure 3.1 Simple DODAG

The rank monotonically increases in the downward direction (DAG root has the lowest rank) and represents a node's virtual position to other nodes with respect to the DAG root [9]. A node in DAG can only be associated with other nodes having same or smaller rank compared to its own rank in order to avoid cycles. RPL does not specify any particular objective function for DAG rank computation.

3.3 RPL Control messages

3.3.1 DODAG Information Solicitation (DIS)

This DIS message is mapped to 0x01, and it is used to solicit a DODAG Information Object (DIO) from an RPL node. The DIS may be used to probe neighbor nodes in adjacent DODAGs. The current DIS message format contains non-specified flags and fields for future use. Table 3.2 shows the DIS message format [10].

Flags (8)	Reserved (8)	Options (8)

Table 3.1 DIS message Format

3.3.2 DODAG Information Object (DIO)

It is issued by the DODAG root to construct a new DAG and then sent in multicast through the DODAG structure. The DIO message carries relevant network information that allows a node to discover a RPL instance, learn its configuration parameters, select a DODAG parent set, and maintain the DODAG. Table 3.2 shows DIO message format [10, 11].

<table>
<tr><td colspan="4">RPLInstanceID
(8)</td><td>Version Number
(8)</td><td colspan="2">Rank
(16)</td></tr>
<tr><td>G
(1)</td><td>0</td><td>MOP
(2)</td><td>Prf
(3)</td><td>DTSN
(8)</td><td>Flags
(8)</td><td>Reserved</td></tr>
<tr><td colspan="7">DODAGID
(128)</td></tr>
</table>

Table 3.2 DIO message Format

3.3.3 Destination Advertisement Object (DAO)

It is used to propagate reverse route information to record the nodes visited along the upward path. DAO messages are sent by each node, other than the DODAG root, to populate the routing tables with prefixes of their children and to advertise their addresses and prefixes to their parents. Table 3.3 shows DAO message format [12].

RPLInstanceID (8)	K (1)	D (1)	Flags	Reserved	DAOSequence
DODAGID (128)					
Options					

Table 3.3 DAO message Format

3.3.4 Destination Advertisement Object Acknowledgement (DAO-ACK)

It is sent as a unicast packet by a DAO recipient (a DAO parent or DODAG root) in response to a unicast DAO message [12].

CHAPTER 4

ATTACK AND IDS IN RPL

4.1 Attacks against RPL network security

Attacks against RPL network have concern about making the legitimate nodes that performs unnecessary processing in order to exhaust their resources. This category of attacks aims at consuming node energy, memory or processing time. This may impact on the availability of the network by congesting available links and therefore on the lifetime of the network which can be shortened.

There are two subcategories of attacks against resources in the RPL network. The first type targets on direct attacks where a malicious node will directly generate the overload in order to degrade the network [12]. The second one contains indirect attacks where the attackers will make other nodes generate a large amount of traffic. For instance, such an attack can be performed by building loops in the RPL network so that make other nodes produce traffic overhead.

Denial of Service (DoS) attacks aims to make a machine or a network unavailable for a certain period. The general principle of DoS attacks results in sending data or packets whose size or content is unusual to cause unexpected reactions of the network or targeted node, sometimes they can even cause the service interruption. This kind of attack is very common on networks because it is simple to implement and can nevertheless have devastating consequences [13].

In addition, the detection and prevention of these attacks are very difficult because they can take many forms. The basic attacks are mostly targeting on consumption of the processor time or the storage capability, congestion in communication links, disruption of service or routing information in the system, etc.

4.2 Types of Attacks

4.2.1 Flooding Attacks

Flooding attack generates a large amount of traffic in a network and makes nodes and links unavailable. These attacks can be performed by an external or internal attacker. In a worst case scenario, there is a possibility of exhaustion of resources in the network. More specifically, when the attack is created using solicitation messages to perform the flooding then it is called as HELLO flood attack. In RPL networks, an attacker can either broadcast DIS messages to its neighboring nodes which have to reset their trickle timer or unicast DIS message to a node which has to reply with a DIO message. In both cases, flooding of network traffic leads to congestion and exhaustion of resources in the RPL network. The consequences of such attacks results in increased control message overhead even though the delivery ratio is not affected [14].

4.2.2 Version Number Attacks

The version number is an important part of each DIO message. It is propagated unchanged down the DODAG graph and is incremented on reconstruction of the DODAG by the DAG root only during global repair mechanism. An older value indicates that the node has not migrated to the new DODAG graph and cannot be used as a parent node. An attacker can change the version number illegitimately by increasing the corresponding field in DIO messages while forwarding the same to its neighbors. Such an attack causes an unnecessary rebuilding of the whole DODAG graph [14]. This attack results in indefinite looping and ends up in loss of data packets. Also the successive unnecessary rebuilding of the graph increases the control message overhead exhausting nodes resources and congesting the network.

4.2.3 Sinkhole Attacks

Sinkhole attack takes place in two ways; first the malicious node manages to attract a lot of traffic by advertising false information data and on receiving the same in an illegitimate manner can modify or drop it. In RPL networks, the attack can be easily performed through the manipulation of control messages and data. Because of this false advertisement the malicious node is more frequently chosen as a preferred parent by the other nodes, resulting in poor network performance and frequent topology change [15].

4.2.4 Wormhole Attacks

Wormhole attacks use a pair of RPL attacker nodes linked via a private network connection. In this scenario, every packet received by intruder node is forwarded through the wormhole to other node that can be replayed later. Since the roles of the pair of nodes in the wormhole are interchangeable, both can act as intruder node. In the case of wireless networks with wormhole attack, transfer of packets with self addressing and interception of any network traffic is possible. This attack creates a predominant problem especially in RPL networks by distorting the routing path [14, 15]. If an attacker tunnels routing information to another part of the network, nodes which are actually at distance appears to be in neighborhood. As a result they can create non-optimized routes for the intruder based on the objective function.

4.2.5 Black hole Attacks

In a black hole attack, a malicious intruder drops all the packets that it is supposed to be forwarded. It can be seen as a type of denial-of-service attack. If the attacker is located at a strategic position in the graph it can isolate several nodes from the network. There is also a variant of this attack which can discard a specific subpart of the network traffic. This attack can be very damaging when combined with a sinkhole attack causing the loss of a large part of the traffic [14].

4.2.6 Intrusion Detection System

While designing an IDS for RPL network, the prima-facie job is to protect the network resources in terms of memory, storage capacity and bandwidth. Their batteries are more constrained and very sensible to energy exhaustion. In various RPL applications the monitored data is very specific. Developing an IDS to prevent all attacks in one shot is impossible. Also the availability of resources varies with the corresponding application which limits the zone of generalized IDS for any scenario. However to secure the RPL network the IDS must implement detection mechanisms that deal with Denial of Service (DoS) threats. The most of DoS attacks aim directly for damaging the routing protocol in layer 3.

Developed IDS must avoid misuse of resources consuming less of it. Anomaly-based is more suitable because it has the possibility to detect new attacks by developing its threshold in

its training phase and it consumes less energy. Also anomaly-based intrusion detection results in high false alarm rate due to external or internal factors and results by itself in a malicious operation. Specification based detection usually consumes less energy as they are based on pre-defined patterns, but is inflexible in upgrading.

The aim of this work is to create a hybrid IDS that combine the advantages of both anomaly-based technique and specification-based technique and there by significantly decrease the false-alarm rate with flexibility for upgradation. Specification-based IDS should incorporate the RPL specifications for its normal operation. As majority of the DOS attacks target the routing protocol, the emphasis for IDS has been increased with a compensation of little network resources for its functioning.

CHAPTER 5

RESILENCE AGAINST RPL ATTACKS

5.1 Introduction

The salient nature of LLNs and inherent RPL vulnerabilities has raised momentous challenges in designing secure RPL to defense intrusions. Intrusion refers to unauthorized or unapproved actions, which attempts to compromise the system. Intruders can be divided into external intruders and internal intruders. External intruders have no right to access the network but can create limited intrusion impacts. Once they obtain authorization to become internal intruders, they are seen as legitimate by the system, which results in severe damage. Due to the harsh and unattended environment, RPL mainly focuses on self- organizing capability, such as auto-optimized routing. This creates an avenue for vulnerability in LLNs with any internal intruder trying to be the part of the self organizing RPL network and creates the hour of need towards IDS.

Thus the objective of this project is to develop a novel Hybrid Intrusion Detection System (HIDS) that appropriately identifies an internal intrusion breaking the monotony of RPL auto-optimized routing construction [15].

5.2 Methodology

5.2.1 Intrusion response

The predominant role of IDS is to detect the intrusion and take appropriate corrective action, such as designing a new rule in a firewall or disconnection of suspicious links. An IDS can be designed to detect and prevent different kind of attacks as single or in group depending on the availability of resources for establishing itself. Once trained for a particular detection has the capacity to detect and prevent the future identical intrusions [16].

5.2.2 Development of HIDS

Developing HIDS for a RPL network may require a proper analysis of the network by several monitoring techniques. Various network parameters are consistently monitored before the attack is created in the network. In the chosen simulation environment, an intruder is artificially introduced in to the RPL network, i.e. one of the nodes is programmed to act as an intruder. Parameter changes that occur due to intrusion have to be measured and analysed though proper monitoring system in the sink node. The establishment of resilience for multiple attacks in the RPL network is executed by creating an HIDS in the DAG root. The HIDS can there upon be activated to have control over that situation. Figure 5.1 shows flow chart for developing HIDS.

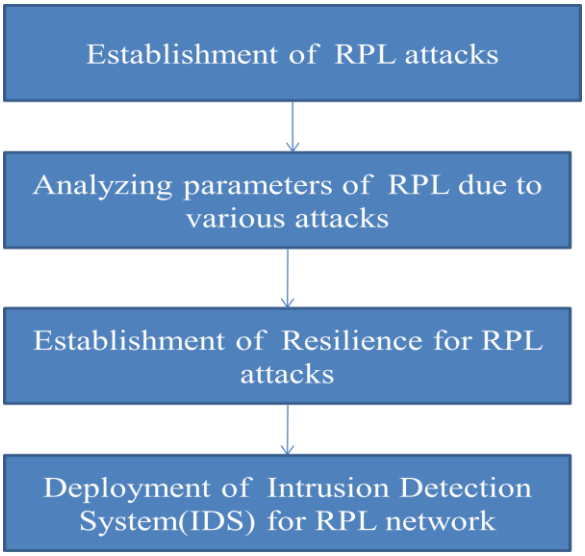

Figure 5.1 Flow chart for developing HIDS

5.2.3Algorithm for HIDS

The algorithm of the HIDS present in the sink node is following the sequential activity shown below.

Require: *Packet*—The IPv6 packet received

Require: *OwnIP*—The IPv6 address of this node

if *Packet.protocol* != RPL *Packet.protocol* and

Packet.destination != *OwnIP* then

Drop packet

end if

5.2.4 Flow chart for HIDS

The RPL network is deployed with number of nodes and each node is made to communicate with one and another. One node is constructed to act as sink node whereas the other nodes act as source nodes to perform any type of monitoring activity as per necessity. Out of the multiple source nodes one is converted to behave as an intruder by changing its characteristics so as to create attacks in the network. On identifying the attack through continuous network monitoring, the sink node activates the HIDS. The nature of the attack and its behaviour is studied by analysing the characteristics of the network parameters from the generated log files.

One such problem is the memory buffer overflow problem created due to the allocation of memory to intruder node assuming it to be a legitimate node. On activation of HIDS, the memory allocation to the intruder node is totally prevented during self-organising themselves to form the network. Attack creating segment of the intruder node is removed using the HIDS, which would make the intruder node become a legitimate node.

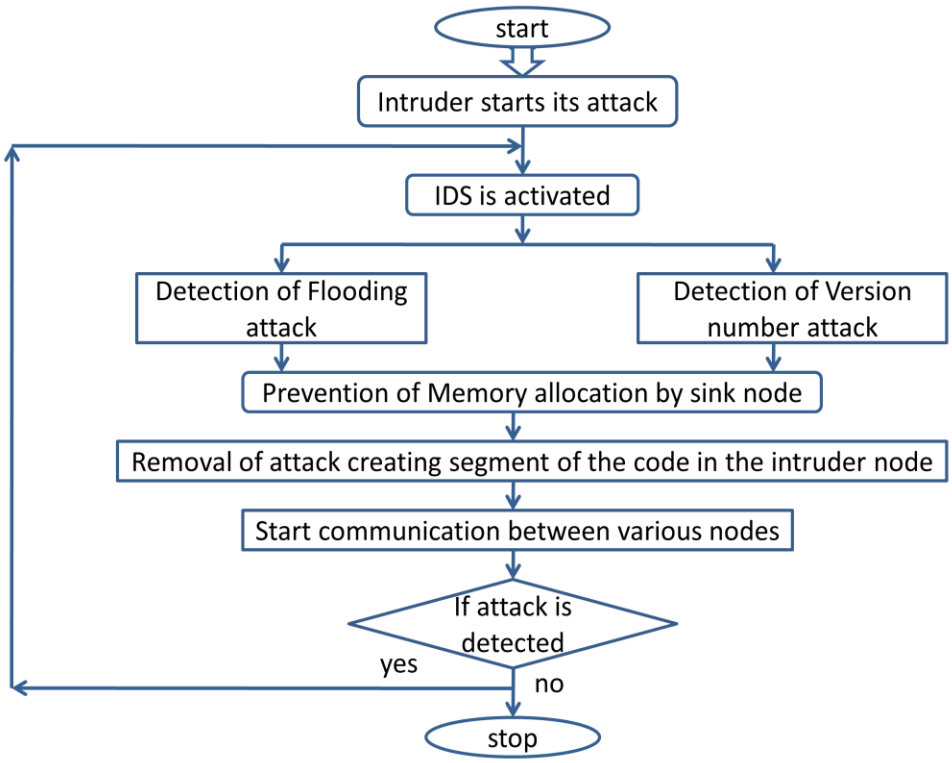

Figure 5.2 Flow chart for HIDS

The usual way of communication is initiated among the nodes and the characteristics of the network are analysed continuously until the identification of any malicious activity. Then the role of HIDS is initiated to identify the culprit in the network and the type of attack it has introduced. Then the memory allocation of that node in the sink node is totally prevented. Also HIDS removes the part of the coding which is responsible for introducing the malicious activity. The routine process of network establishment and data communication continues with the available nodes, until the intruder node becomes a legitimate node and the network is free from intruder node activities. The figure 5.2 shows the steps involved in flowchart of HIDS to detect and prevent the various attacks that dominate a RPL network.

CHAPTER 6

SIMULATION ENVIRONMENT

6.1 ContikiOS

The ContikiOS is written in basic C language. The figure 6.1 shows the different folders come under Contiki 3.0. The apps folder consists of application that can be run on Contiki. The CPU folder has source codes for processors that can run Contiki. The platform consists of readily available motes that can be used to build Contiki. The Makefile.include compiles and builds the entire Contiki system [17].

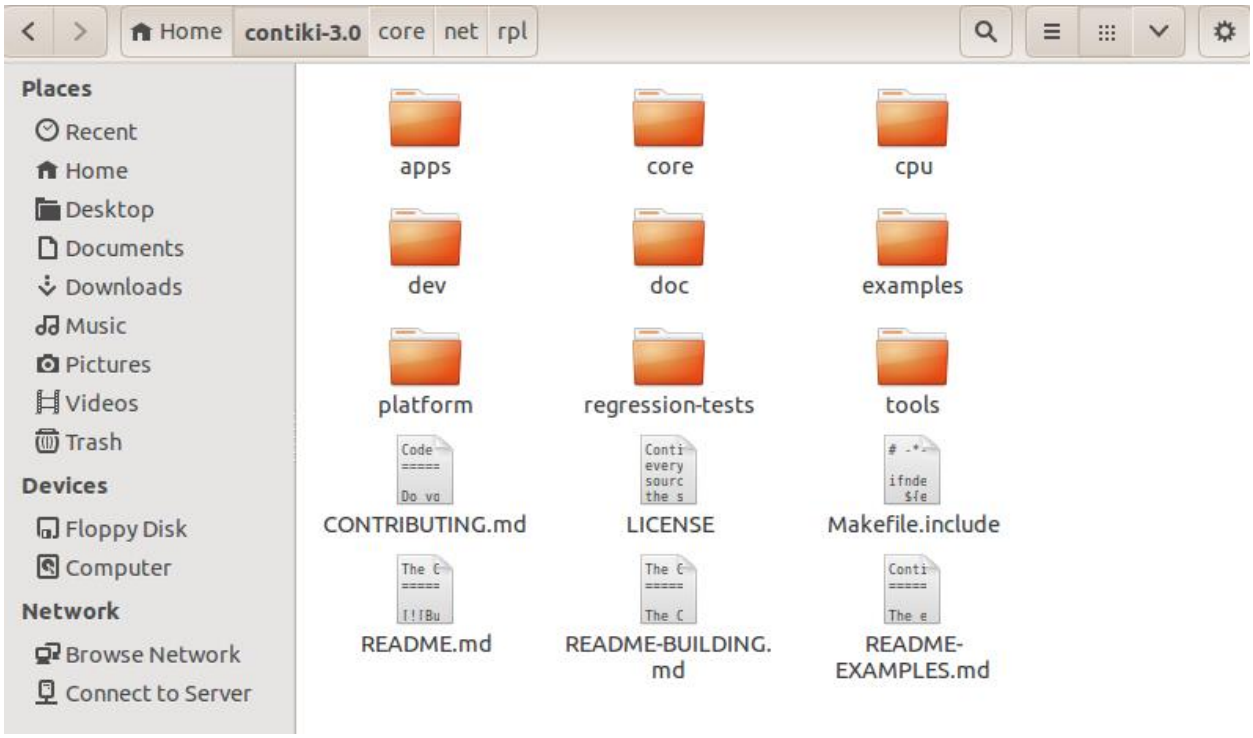

Figure 6.1 Folders that comes with Contiki OS

6.2 Contiki Folder Structure

The Contiki OS has following folder structure, the RPL routing protocol is present in /Contiki-3.0/core/net/rpl/rpl-icmp6.c

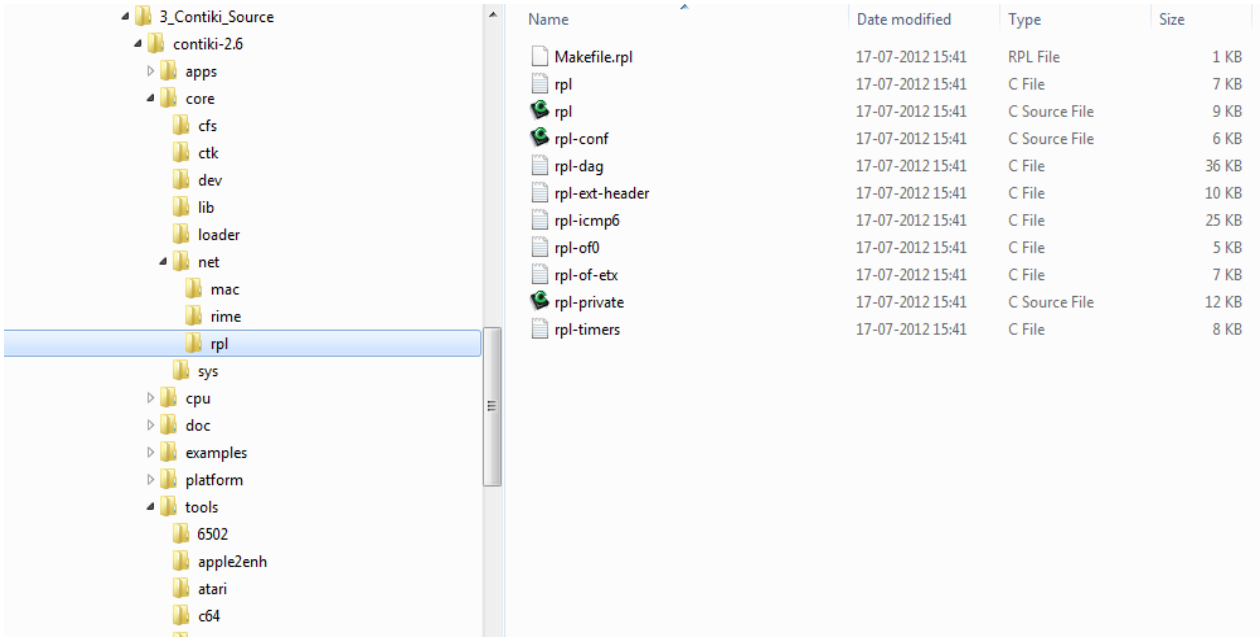

Figure 6.2 Contiki-3.0 Folder Structure

6.3 Cooja Simulator: Simulation Steps

The Cooja is a java based real time simulator for the Contiki OS. It can simulate the actual working of wireless sensor nodes. The serial port logs from the nodes can be viewed in the mote output window, which can be stored in the file for later analysis.

The following are the steps to simulate the wireless sensor network on Cooja simulator,

- Start the Cooja simulator by entering following commands in the path /Contiki-3.0/tools/Cooja.

```
$cd …./Contiki-3.0/tools/Cooja
$ant run
```

- The above command will show the Cooja simulator default window, from the file menu, create a new simulation.
- Save the simulation in file system.
- Add motes from Motes->Add Mote->choose the type of mote.
- Choose the program file to be compiled.

- Once the file is compiles, choose the number of motes and arrangement.
- Important Note: Increase the buffer size from tools->buffer size. The buffer size denotes the no of log output lines buffer memory allocated.

The default is Unit Disk Graph Medium (UDGM): Distance loss which denotes that the medium will have loss if the motes are far apart. There is also an option to include constant loss medium or multipath ray tracer medium. Change the simulation parameter as per requirements and click create button to create a simulation [16].

When a new simulation is created, the window shown in figure 9 appears. The figure shows multiple windows such as

- Network: AN area used to create and position motes. The view menu in this window has option that can be used for better understanding of the motes being simulated.
- Simulation control: It is used to start, stop, pause or reload the simulation when relevant changes are made in the source code.
- Mote output: The motes running in the network window will send outputs which are generally printing statements given in the source code. These outputs are captured in the Mote output window.
- Timeline: This window displays the active time of each mote in which it sends or receives data.

  The above said folder contains the routing protocol logic and code for the routing layer. The WSN nodes can be created in real time simulator called Cooja simulator [15-17].

6.4 Simulation Setup

The network consisting of one server mote (mote 1) and five clients motes (mote 1,2,3,4,5), among those clients only one mote is being an intruder(mote 2).

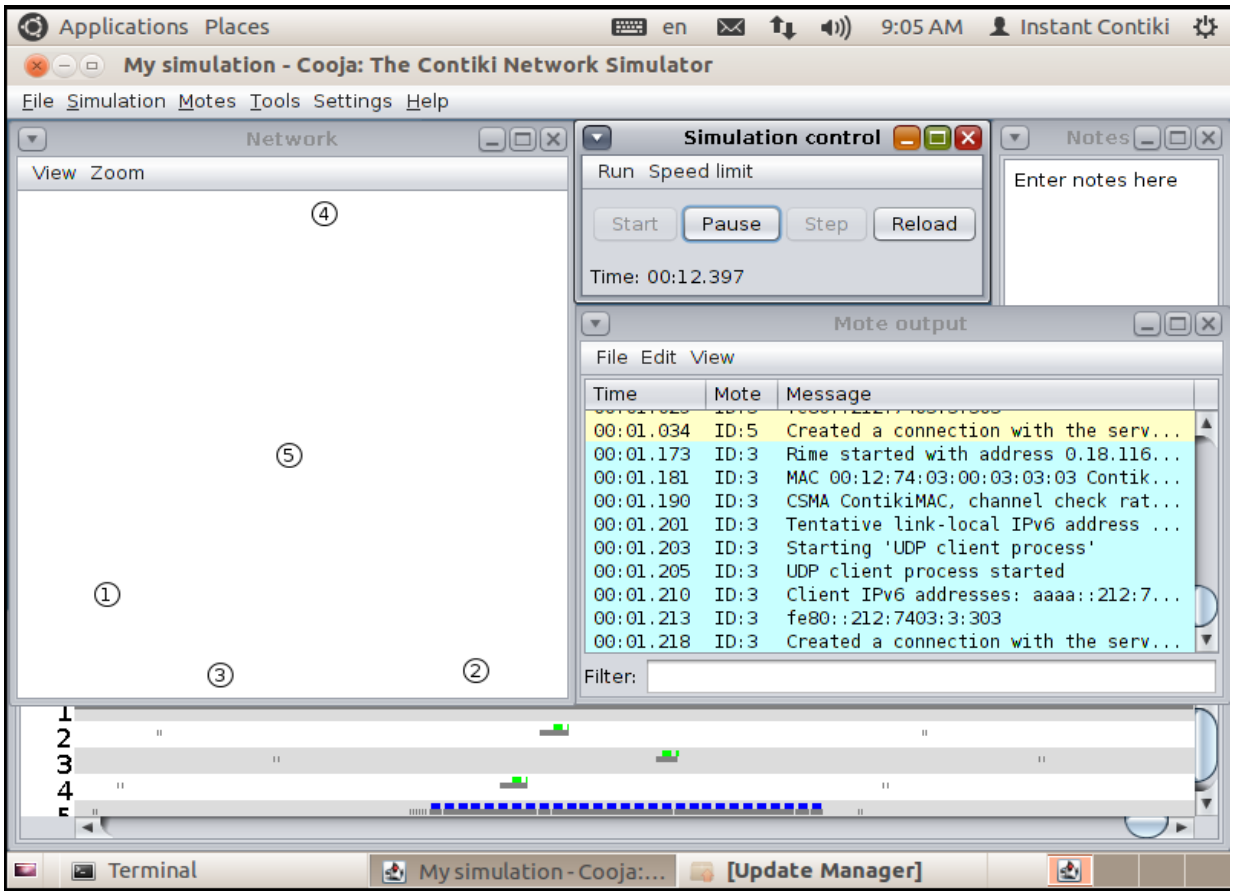

Figure 6.3 Cooja Simulator Screen Shot

Figure 6.3 shows the simulation of UDP on sky motes (1 server mote and 2, 3, 4 and 5 are client motes).

6.4.1 Analytical Equation for the Metrics Calculation

The convergence time, control overhead and packet delivery ratio is calculated using the following formulas. The debug prints are included in the Contiki RPL layer code. When executed the debug prints will be available in mote output window, the result log is stored. The log file is analyzed using the automatic PERL script. The PERL script is developed to act on a set of files and the result is collected in result.log file.

The simulator output has time resolution in milliseconds, the time difference is being subtracted and corresponding values are converted to seconds.

CHAPTER 7

RESULTS AND DISCUSSIONS

7.1 Simulation Results

The Cooja is a java based real time simulator for the Contiki OS, it can simulate the actual working of wireless sensor motes. The serial port logs from the motes can be viewed in the mote output window, which can be stored in the file for later analysis.

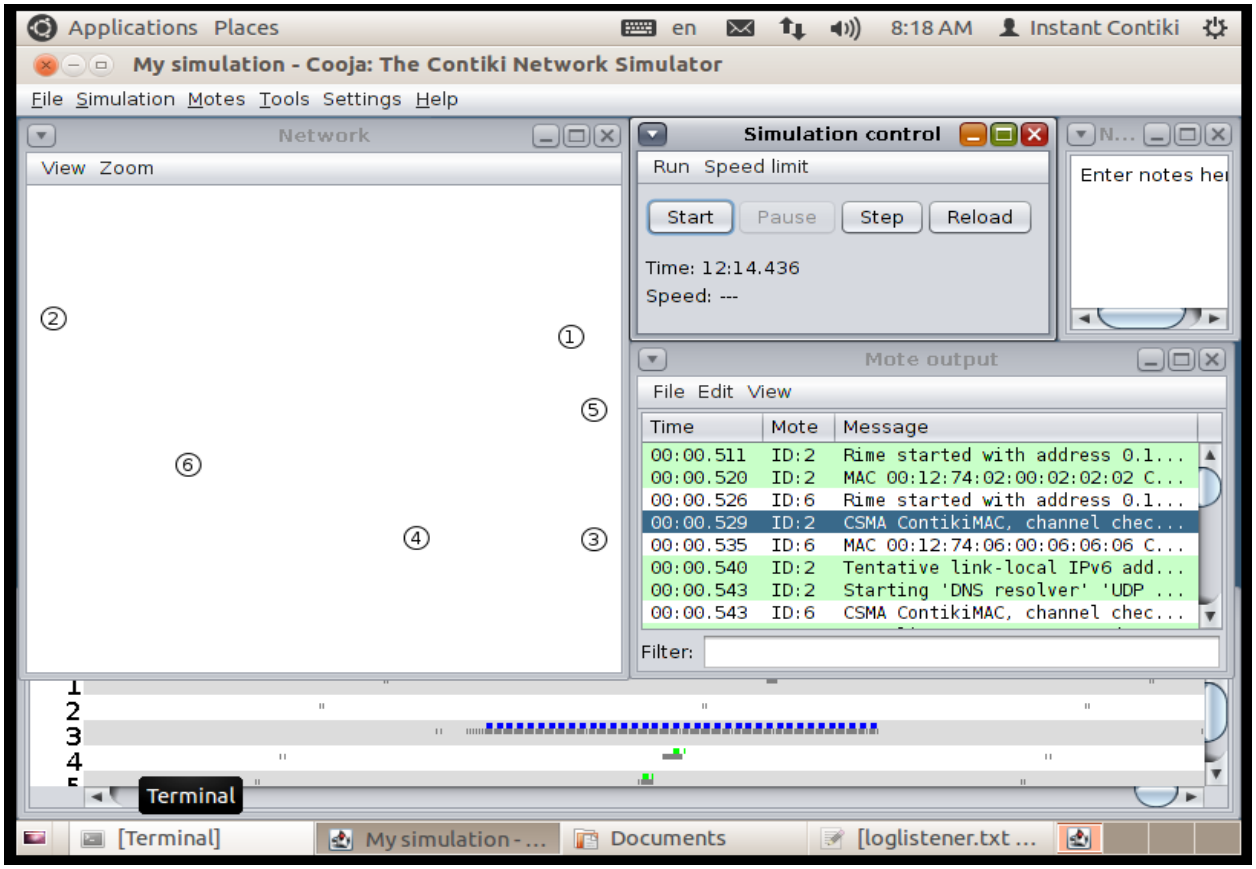

Figure 7.1 Mote 2 starts acting like an intruder mote

In the above figure 7.1, mote 2 acts as an intruder which will start its communication in the network. The mote 2 will starts its communication only after few seconds of the data communication, so that the attack creating mote will be identified by the intruder detection system.

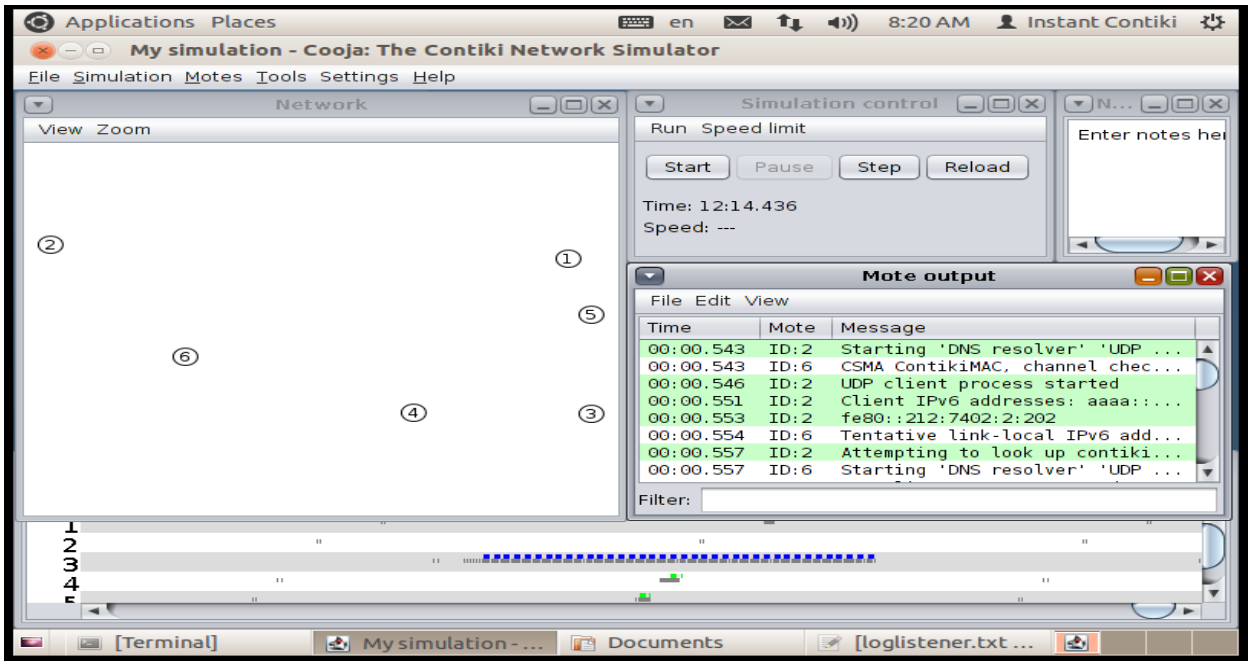

Figure 7.2 Mote 2 communicates with other mote

In Figure 7.2, the intruder mote 2 communicates with a genuine mote 6, in order to broadcast itself as neighboring mote to mote 6 and creates a false routing table with the help of information received during the communication.

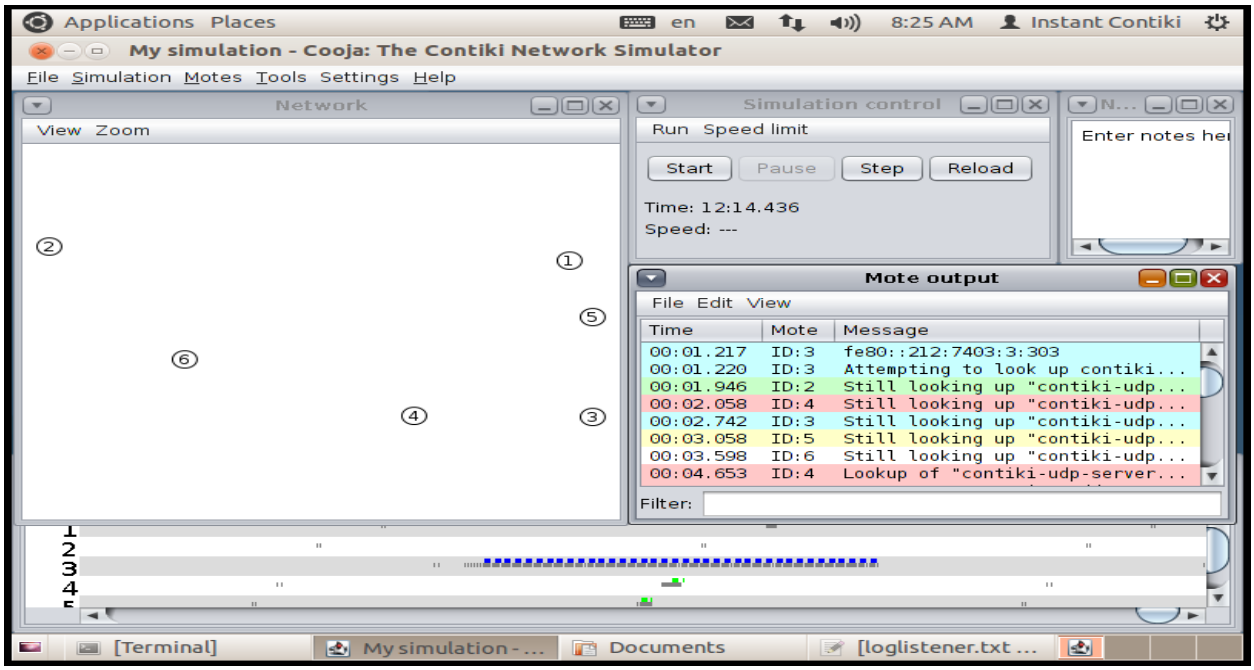

Figure 7.3 Intruder detection system detects mote 2 as an intruder

In figure 7.3, intruder mote 2 will collect all the necessary information from genuine user mote 6, in order to starts its communication in the network. The attack will be created in the network by the mote 2, which creates unwanted communication between the intruder mote 2 and the server mote 1 and not allowing any other genuine mote to communicate with the server mote 1. The unnecessary data sent by intruder mote 2 leads to memory buffer filling which does not contain any of the useful information and communication between other genuine motes to the server mote 1 is completely stopped. This leads to detection of intruder mote 2, in the network by the intruder detection system. The network activity is being kept in track with the help of the intruder detection system which detects the intruder mote 2 by analyzing the parameter changes taking place in the entire network. Depending upon the parameter changes the intruder detection system categorizes the type of the attack being taken place in the network. The intruder detection system follows the corresponding resilience method depending upon the type of attack taken place.

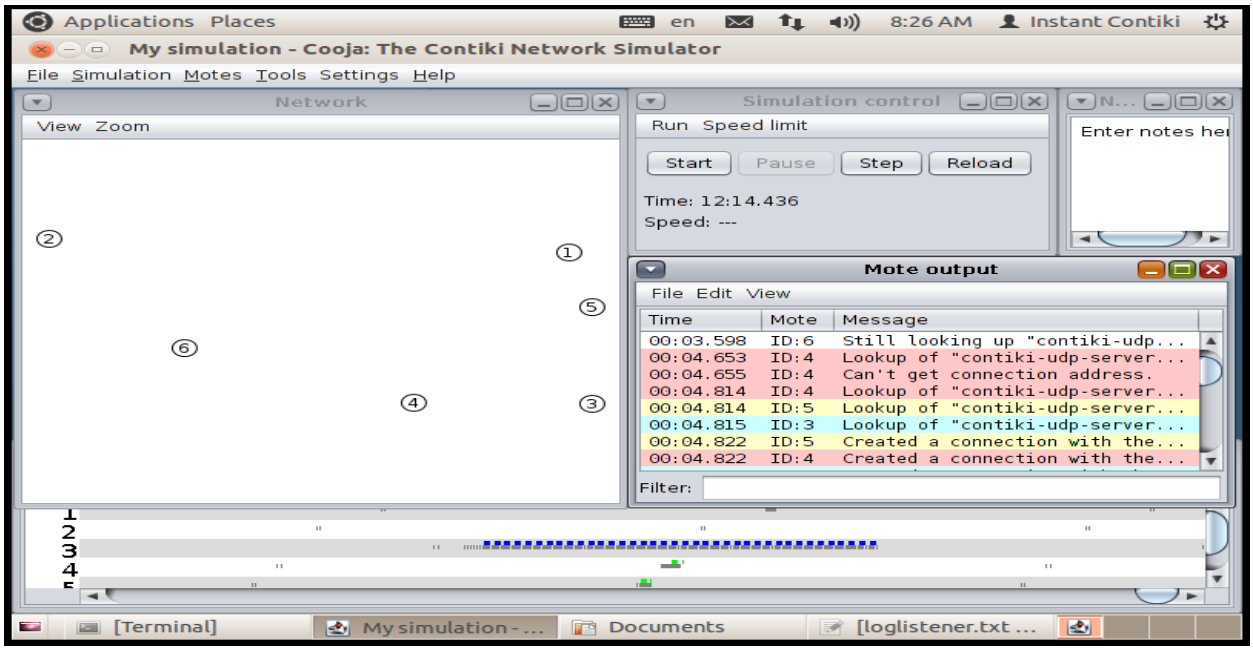

Figure 7.4 Resilience against the attack takes place in the network using the intruder detection system

In this figure 7.4, it is seen that the intruder detection system analyses the network parameters changes and takes preventive measures against the attack being created. Based on the type of attack existing in the network the corresponding algorithm will be followed.

The intruder detection system identifies the mote 2 as the intruder since communication takes place between intruder mote 2 and the server mote 1 for longer time duration with unnecessary data communication. Malicious activity is taken place in the network unnecessarily by the mote 2 which is analyzed by the parameter changes. Thereby mote 2 will be isolated temporarily from the network and the routing table content will be analyzed to check whether the mote is genuine user or it is being an intruder having information which is collected from any one of the genuine motes. The code sequence will be run in order to find out the processes taking place in the network due to this mote. Depending upon the characteristics of the mote and analyzing the parameters the type of attack created will be determined. If the mote is identified to be an intruder mote then the resilience method will be followed. The attack creating segment would be removed from the mote source coding by knowing the general functioning of the network genuine motes. Thus mote 2 is identified to be an intruder in the network so the attack creating segment code will be removed by the intruder detection system.

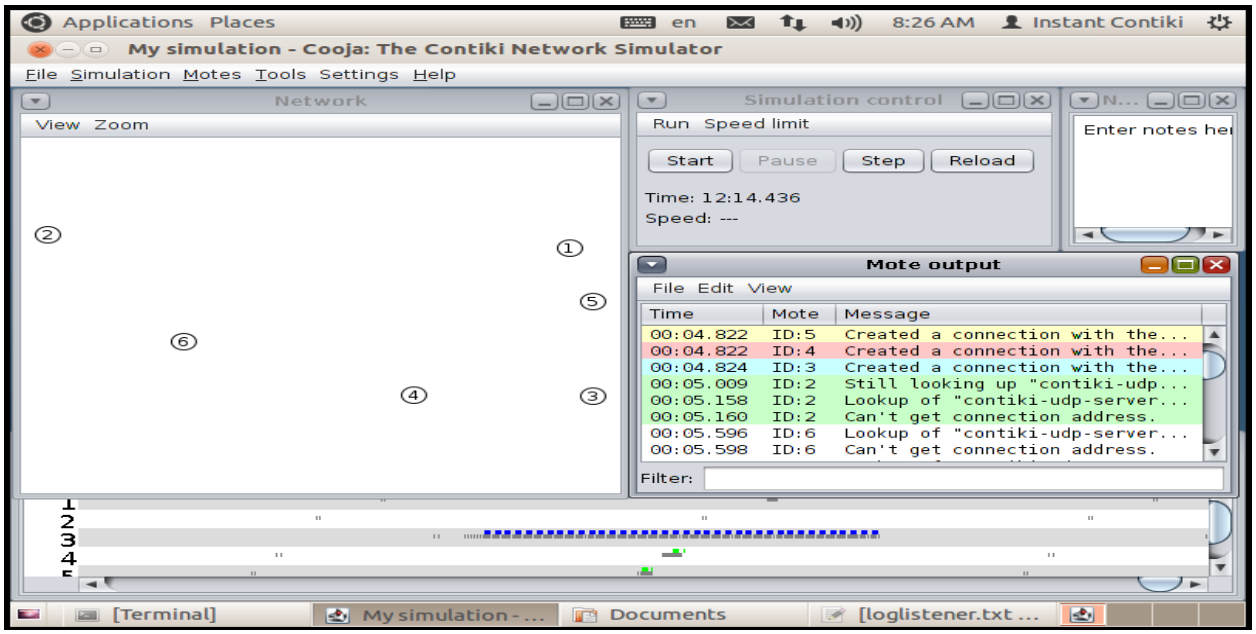

Figure 7.5 Mote 2 still acts as an intruder which is detected by the Intruder detection system

In figure 7.5, it is shown that after taking the preventive measures against the attack, the communication will be started as soon as the intruder detection system removes the attack creating code segment in mote 2.

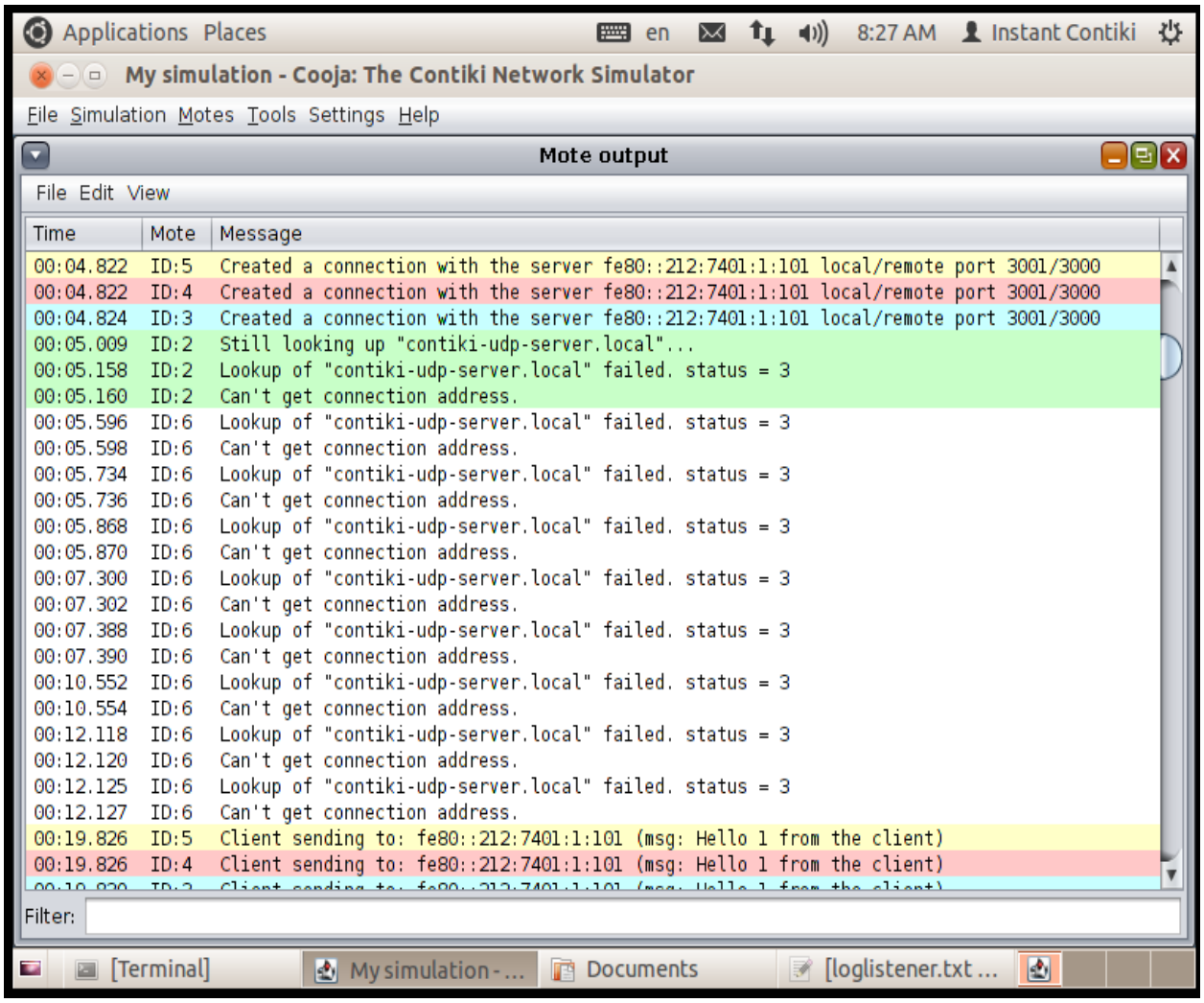

Figure 7.6 Intruder detection system isolates the attack creating node from the network

The intruder detection system again starts its activity of analyzing the parameter changes taking place in the network due to the intruder mote 2 is done. The mote 2 does not change its characteristics and behavior which still continues to act as an intruder. Therefore the intruder detection system removes the mote 2 from the entire network. Communication with mote 2 will be permanently stopped and data will not be shared with that mote 2. The mote 2 will be isolated from the network thus the intrusion activity will be perfectly removed.

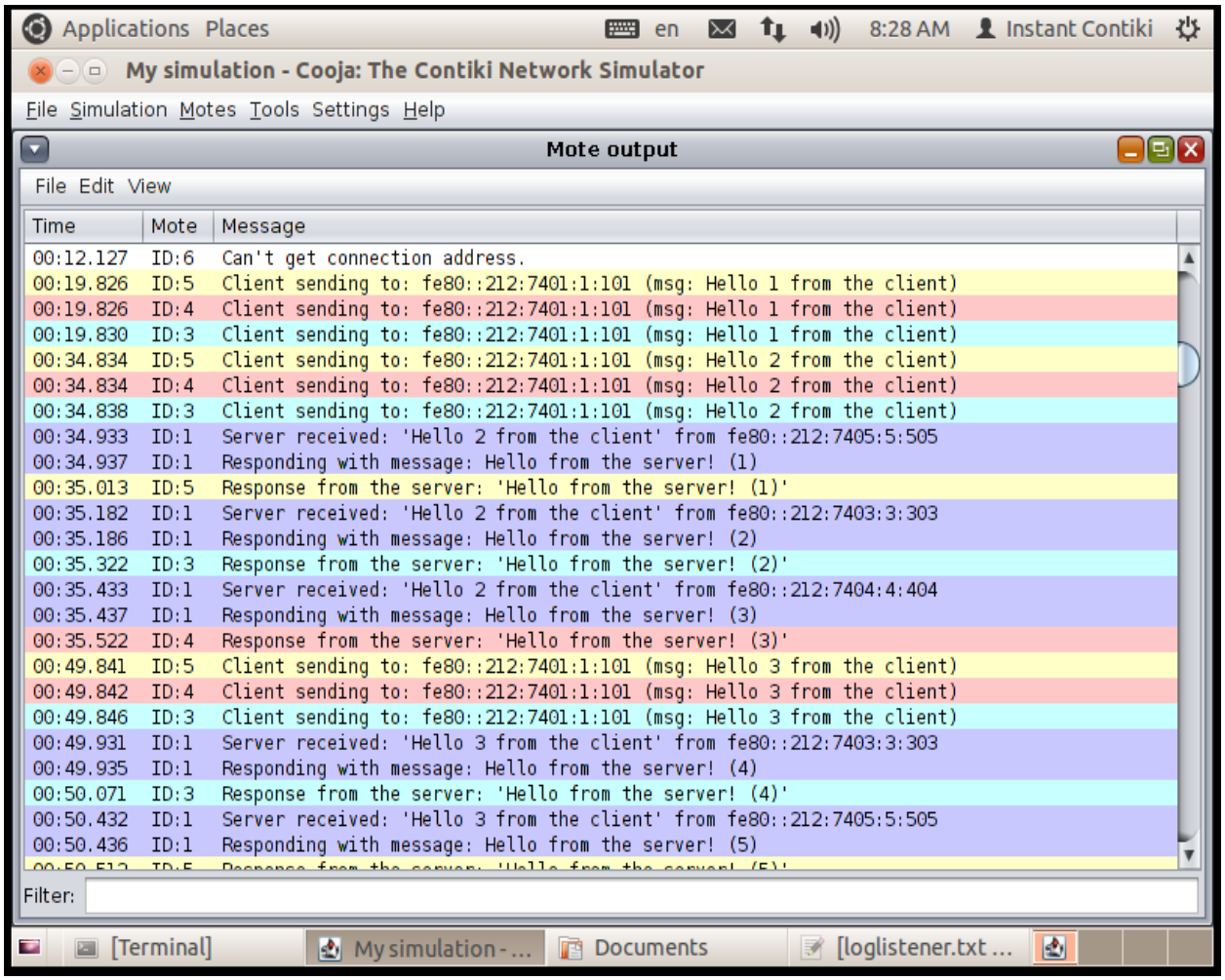

Figure 7.7 Communication takes place between various motes except the attack creating mote

The communication will be started after isolating the intruder node 2 permanently from the network. Intruder detection system will not allow the mote 2 to communicate with the server mote 1 or with any other motes throughout the entire communication. Therefore the network will be free from any type of attack being created by the intruder. The intruder detection system will be monitoring the parameter changes taking place in the network, when there is no intruder activity detected the intruder detection system will be deactivated and will not take any action against mote 2. Therefore the motes will be communicating useful information with the server mote 1. The intruder detection system will be activated when any kind of malicious activity takes place in the network.

7.1.1 Sensor Data Collect with Contiki

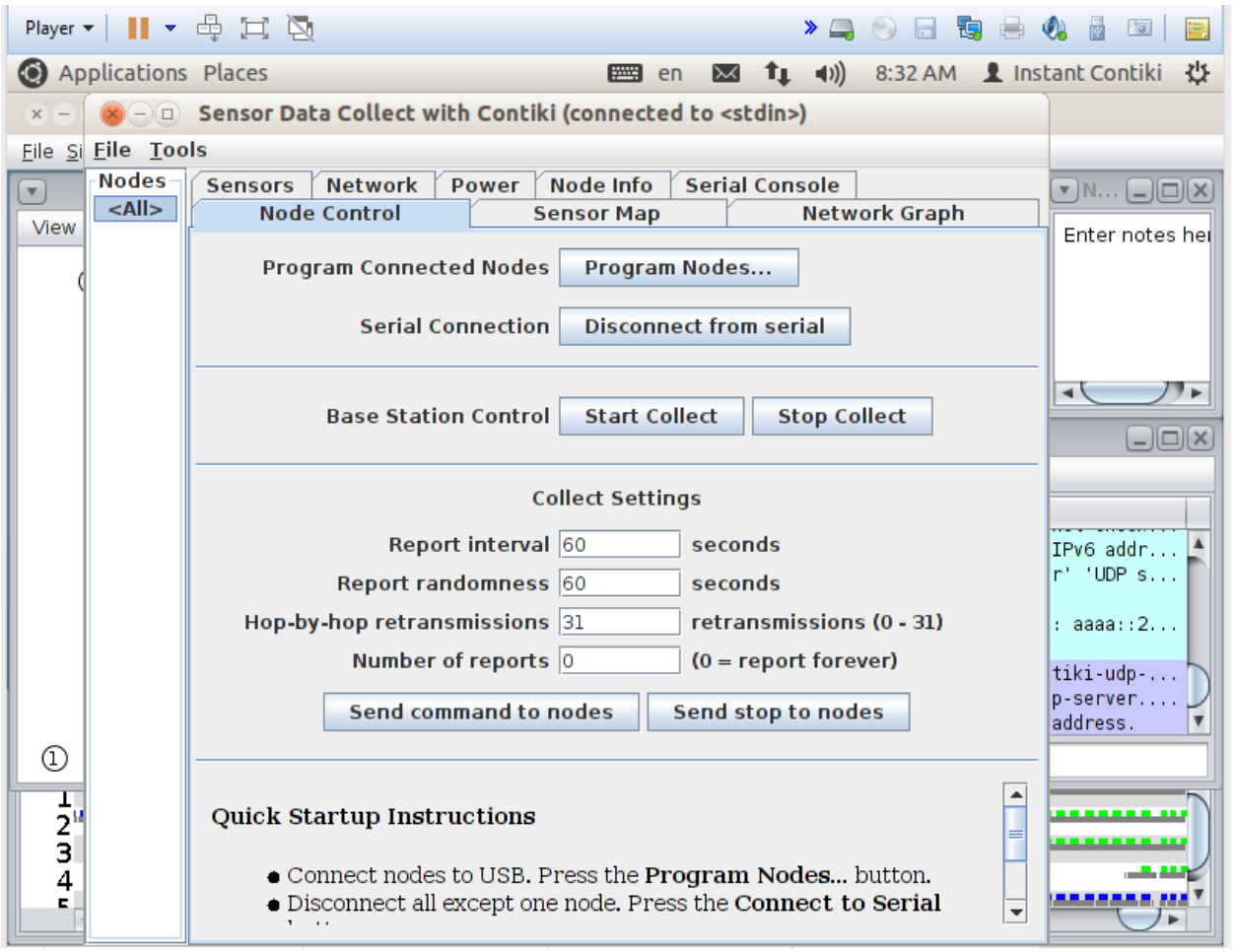

Figure 7.8 Sensor Data Collect with Contiki

The sensor data collect with contiki will be useful in measuring and analyzing various types of network parameters as follows.

7.1.2 Analyses of various RPL network parameters for the established Resilience - Intruder Detection System

For simulation 1 server (mote 1) and 4 clients (mote 2, 3, 4, 5) - mote 2 as an intruder Sensor Data Collect with Contiki for Sky mote 1 and the data received is shown in the figure 7.9.

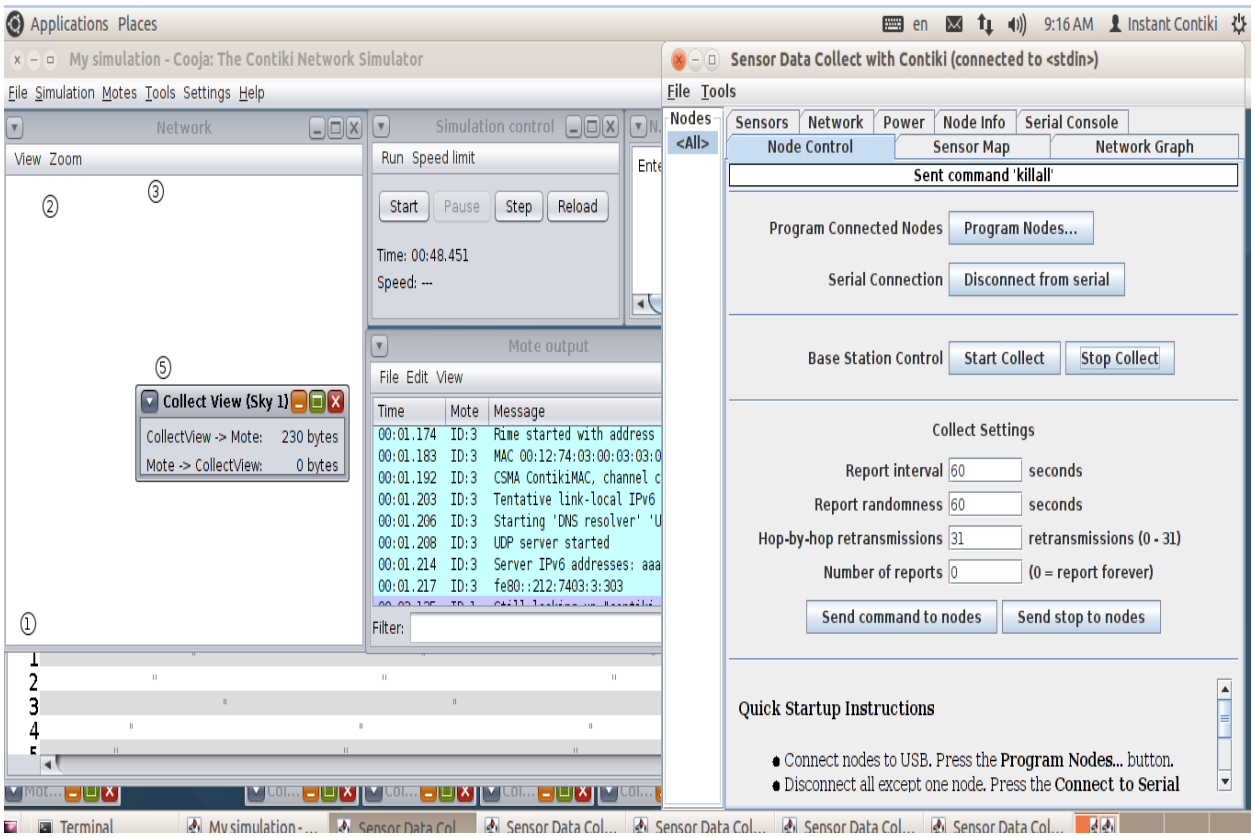

Figure 7.9 Sensor Data Collect with Contiki for Sky mote

The sensor data collect uses node control for obtaining the memory occupation taking place in each every mote being communicating in the network.

Figure 7.10 Collect View- sky mote 1

In the figure 7.10, the collect view provides the total number bytes of memory space consumed by the server mote 1 during the entire communication (sky mote). The total memory consumption of the server mote increases if intruder detection takes place in the network. This would be one of the important disadvantages of a network, when an intruder starts attacking. This could be overcome by implementation of intruder detection system at the begging of the attack creation. The memory consumption must be minimized efficiently by a proper intruder detection system.

Figure 7.11 Collect View- sky mote 2

The Sensor data collect with Contiki for Sky mote 2 is been shown in the figure 7.11. The mote 2 will be isolated from the network by the intruder detection system. Thereby the memory occupation for the intruder Sky mote 2 will not be allocated. Thus the memory will not be allotted for the entire communication since it has been detected to be an intruder in the network.

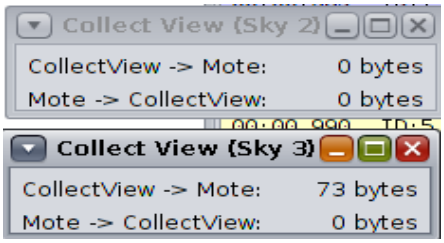

Figure 7.12 Collect View- sky mote 3 (client motes 2 and 3)

The Sensor data collect with Contiki is used for showing the comparison between Sky mote 2 and 3 are being shown in the figure 7.12.

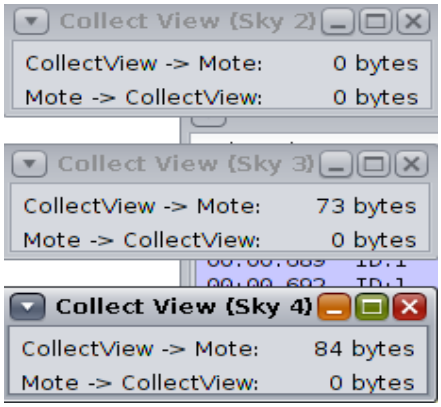

Figure 7.13 Collect View- sky mote 4 (client motes 2, 3 and 4)

The Sky mote 2, 3 and 4 are shown in the figure 7.13, with the help of the Sensor data collect with Contiki.

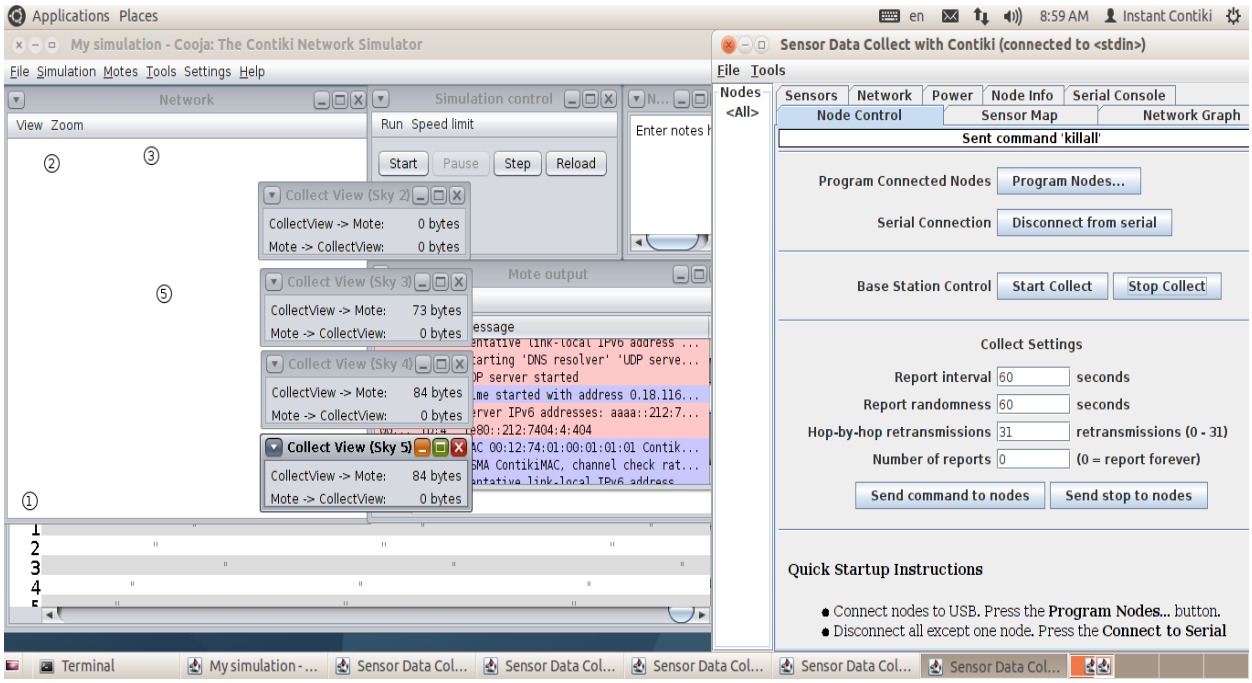

Figure 7.14 Sensor Data Collect with Contiki- Sky mote 5

In the figure 7.14, the memory occupation of sky mote 5 is given by sensor data collect with contiki

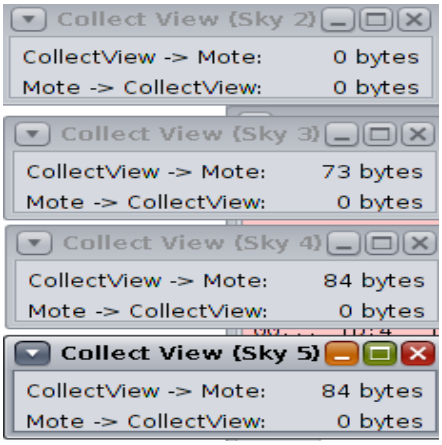

Figure 7.15 Collect View- sky mote 5 (client motes 2, 3, 4 and 5)

The above figure 7.15, it shows the memory consumption of sky mote 5 along with mote 2,3,4 and 5.

7. 1.3 Power Tracking

7.1.3.1 RPL network with attacks:

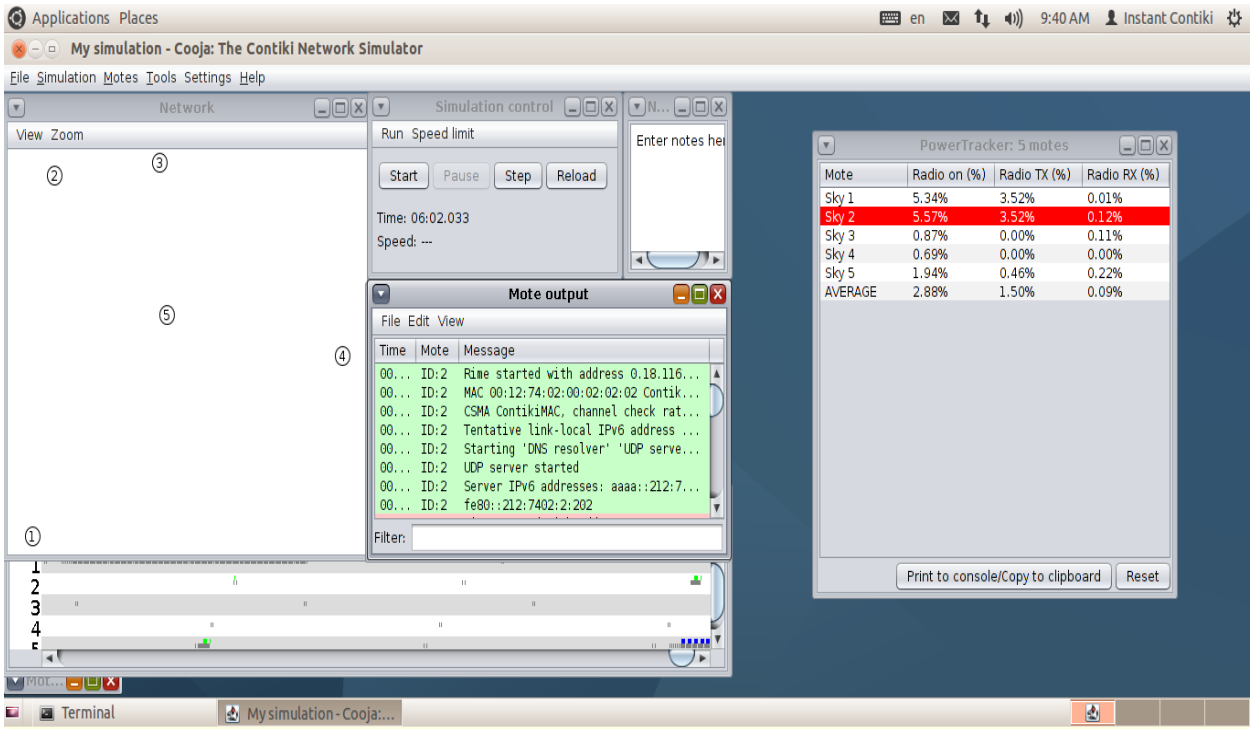

Figure 7.16 One server (mote 1) and four clients (mote 2, 3, 4 and 5) - mote 2 as an intruder

The figure 7.16, it shows the power tracker which provides the radio on, radio tx and radio rx power in percentages.

Radio on, Radio TX and Radio RX Percentages:

The Sky mote 2 is the intruder which consumes more radio on, radio TX and radio RX percentage than requirement, while it is transferring data to all other motes because server mote 1 and remaining 4 clients (mote 2, 3, 4 and 5) - mote 2 as an intruder Radio on, Radio TX and Radio RX Percentages are given in the Table 7.1.

Mote	Radio on (%)	Radio TX (%)	Radio RX (%)
Sky 1	5.34%	3.52%	0.01%
Sky 2	5.57%	3.52%	0.12%
Sky 3	0.87%	0.00%	0.11%
Sky 4	0.69%	0.00%	0.00%
Sky 5	1.94%	0.46%	0.22%
AVERAGE	2.88%	1.50%	0.09%

Table 7.1 Power tracking measured in RPL network which is being attacked by intruder mote 2

7.1.3.2 RPL network with IDS:

Same scenario is followed for this simulation of RPL network with IDS i.e, server as mote 1 and 4 clients (mote 2, 3, 4 and 5) - mote 2 as an intruder. IDS detects the sky mote 2 as intruder, so consumption of radio on, radio TX and radio RX percentage are made to be reduced.

Mote	Radio on (%)	Radio TX (%)	Radio RX (%)
Sky 1	0.70%	0.00%	0.00%
Sky 2	0.69%	0.00%	0.00%
Sky 3	0.72%	0.00%	0.01%
Sky 4	1.20%	0.38%	0.00%
Sky 5	0.72%	0.00%	0.01%
AVERAGE	0.81%	0.08%	0.01%

Table 7.2 Power tracking measured in RPL network after activation of IDS against the intruder mote 2

Mote	Radio on (%)	Radio TX (%)	Radio RX (%)
Sky 1	0.71%	0.00%	0.00%
Sky 2	0.69%	0.00%	0.00%
Sky 3	0.71%	0.00%	0.00%
Sky 4	0.70%	0.00%	0.00%
Sky 5	0.70%	0.00%	0.00%
AVERAGE	0.70%	0.00%	0.00%

Table 7.3 Power tracking measured in RPL network after isolating the intruder mote 2

IDS will stop the sky mote 2 (intruder) to communicate further so that power consumption is remaining same. The radio on, radio TX and radio RX percentage are not been changed since no communication is taken after isolating the sky mote 2 from the network. While all other sky motes are allowed to consume equal radio on, radio TX and radio RX percentage, while transferring data which is detailed in Table 7.3.

7.1.4 Analyzing parameter changes that occurred due to RPL attacks

Radio messages

The radio messages are used for viewing the packet number, packet transmission time, sender and receiver number and also the data communication taking place between the motes of the network. A network is been created with one server(mote 1) and four clients(mote 2, 3, 4 and 5) among those mote 5 as an intruder.

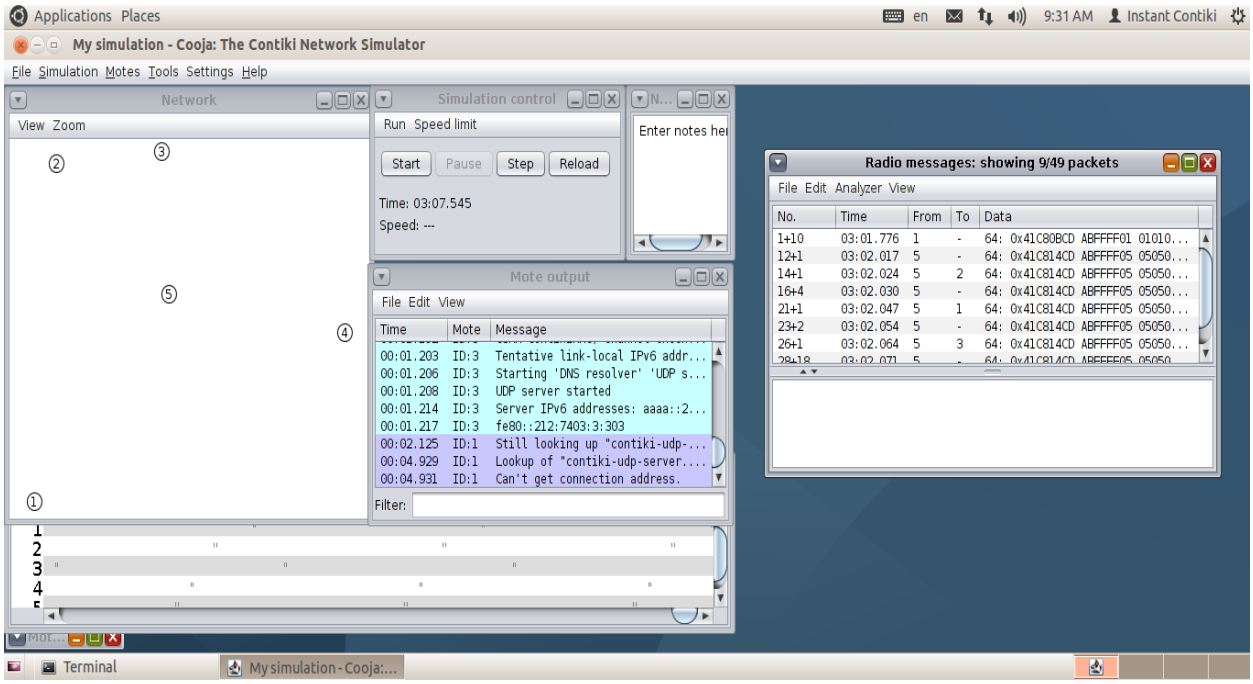

Figure 7.17 Radio messages view in cooja the contiki network simulator

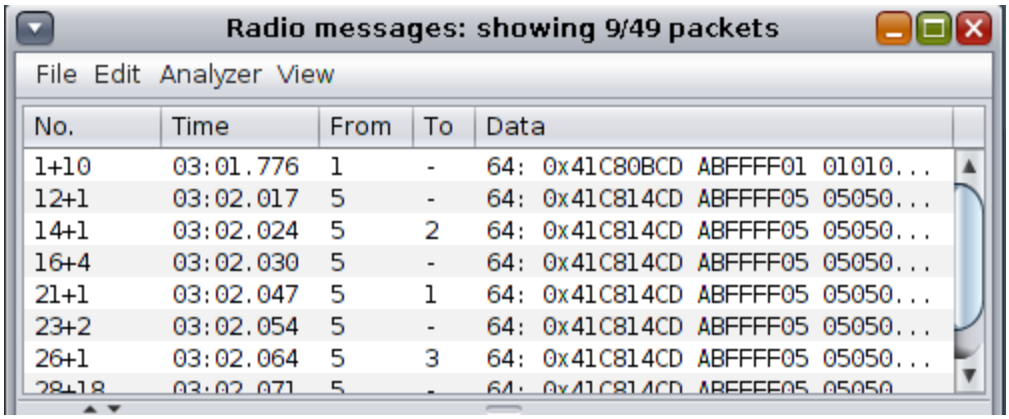

Figure 7.18 Radio messages showing 9/49 packets

The Radio messages are being sent from client mote (mote 5) to intruder mote (mote 2) while communicating with genuine user mote 1, 3 and 4. At the time of 03:02.024 the data communication takes place from mote 5 to mote 2, during this communication the information are sent to the intruder mote 2. The intruder does not allow mote 5 to communicate any information further with any other motes other than the acknowledgement message sent to intruder mote 2. Therefore the network gets heavy and messages are not being sent to the corresponding mote. The network connection will be made with the intruder mote 2 alone throughout the communication process.

7.1.5 Stack watcher

7.1.5.1 RPL network without attacks:

The Msp stack watch viewer is used for monitoring the stack overflows in the network. The stack overflow takes place at the faster rate when the network is being attacked by an intruder mote. The communication takes place unnecessarily with the intruder mote which has only unwanted information. In the network, one server (mote 1) and four clients (mote 2, 3, 4 and 5) are made to communicate for analyzing the stack overflow with the Msp stack watcher.

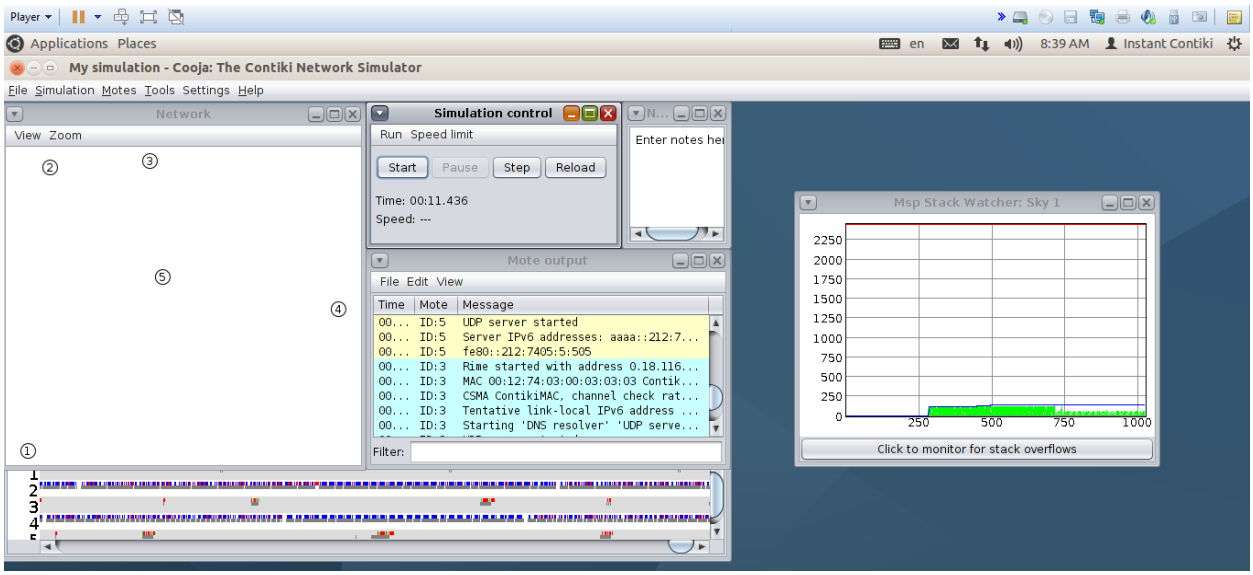

Figure 7.19 Msp stack watcher view in the cooja the contiki network simulator

The Msp stack watcher viewer showing actual network data transmission graph without attacks is shown in the figure 7.19.

7.1.5.2 RPL network with attacks:

In the network, one server mote(mote 1) and four clients motes(mote 2, 3, 4 and 5) where mote 5 is an intruder. The stack watcher viewer will provide network data transmission graph with attack existing in the network by an intruder mote 5.

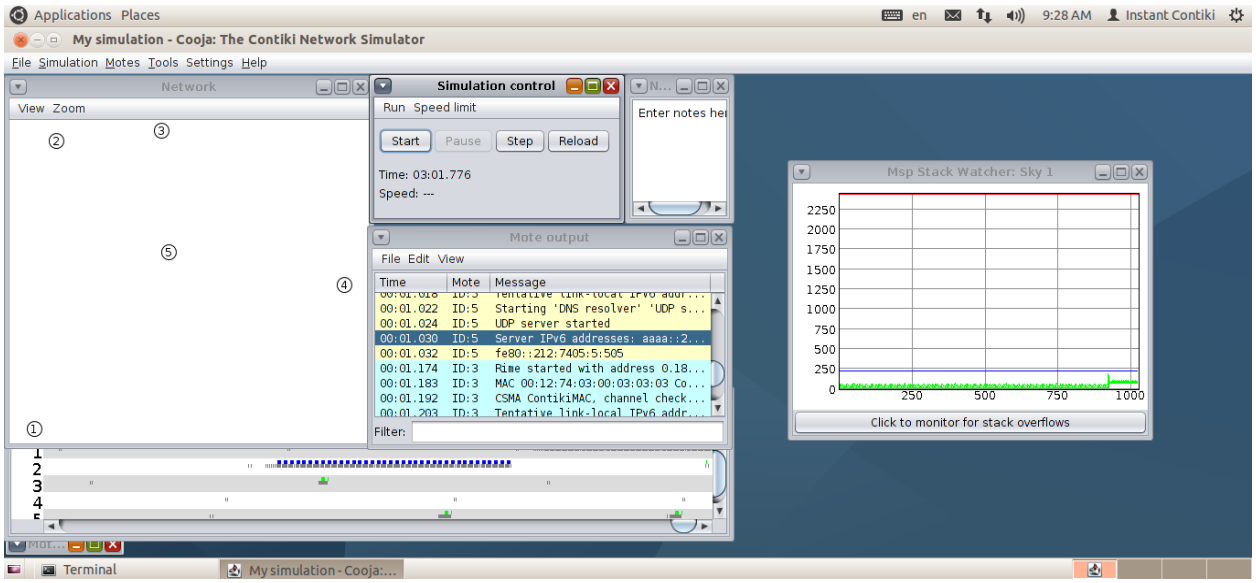

Figure 7.20 Msp stack watcher view in the cooja the contiki network simulator

The Msp stack watcher viewer showing actual network data transmission graph without attacks is shown in the figure 7.20.

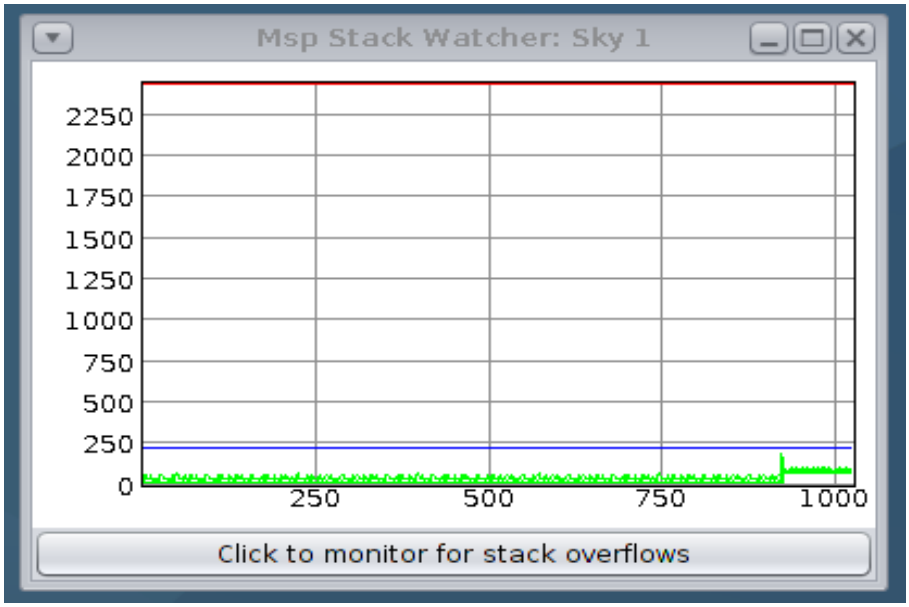

Figure 7.21 Stack watcher showing actual network data transmission rate graph with attacks

Formulas

Convergence time = last dag joined – first dio sent.

Control overhead = (control packets / data packets).

*Packet delivery ratio= (successful received / transmitted packets)*100.*

*Memory consumption = (total nodes memory * code memory + attack segment memory).*

Power consumption = (total nodes power consumption + attacker node power).

7.1.6 Average Power consumption

The LLN network with RPL routing protocol is usually constrained in terms of network resources such as power and memory. Power consumption is one of the important factors that deteriorate progressively in Wireless Sensor Networks. Power consumption can easily be calculated using Cooja simulator by using Collect view application which is not possible with many other network simulators. For each sensor node, Power consumption of each node can be evaluated from the collect view option and the same is tabulated in the form of bar chart for the three types of nodes viz. Sink node, Attacker node and Source node.

Figure 7.22 depicts clearly the average power consumption in a RPL network under three different scenarios such as network without attack, network affected by intruder (flooding attack) and after IDS activation in the network [17].

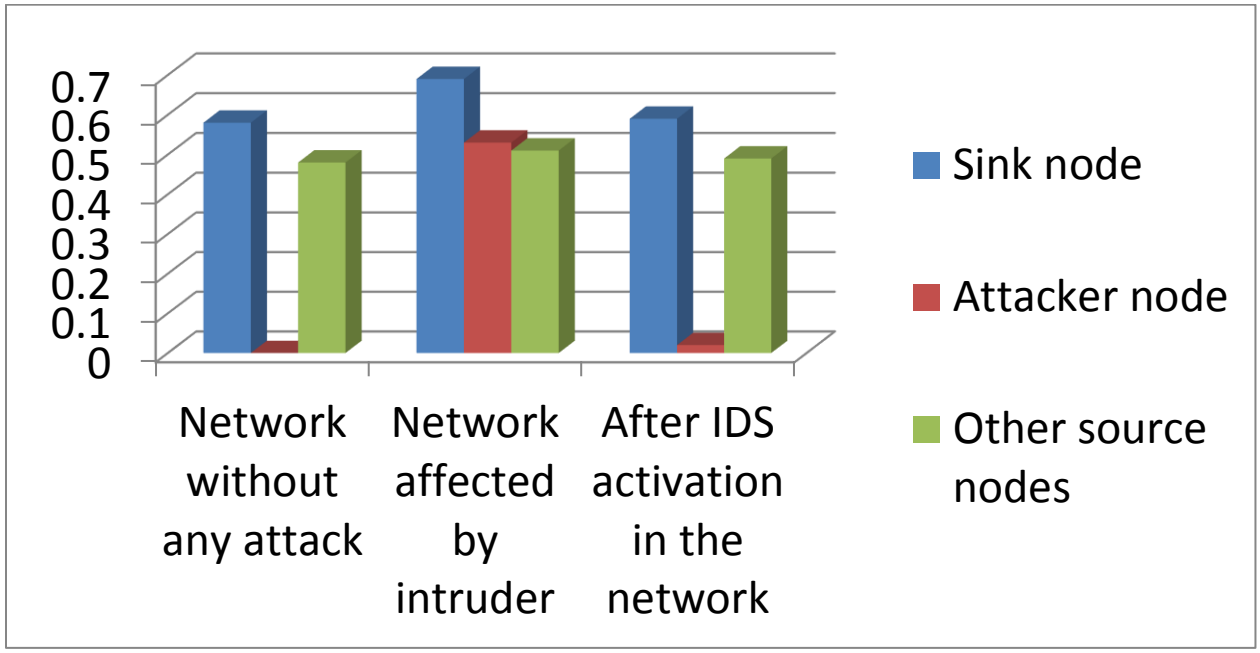

Figure 7.22 Average Power Consumption (mW) in flooding attack

From the bar chart it is obvious that the average power consumption of the sink node without attack is the least compared to the one with attack and after IDS activation. The power consumption of the attacker node is predominant during attack and has a drastic reduction after activation of HIDS in the network. The power consumption of the sink node, attacker node and the source nodes is comparatively high for the network with attack and HIDS activated network.

When HIDS is activated, the power consumption is reduced than that with the network with flooding attack. The same process is executed for the version number attack as well.

Figure 7.23 shows the average power consumption of the RPL network without attack, network affected by intruder (version number attack) and after IDS activation in the network.

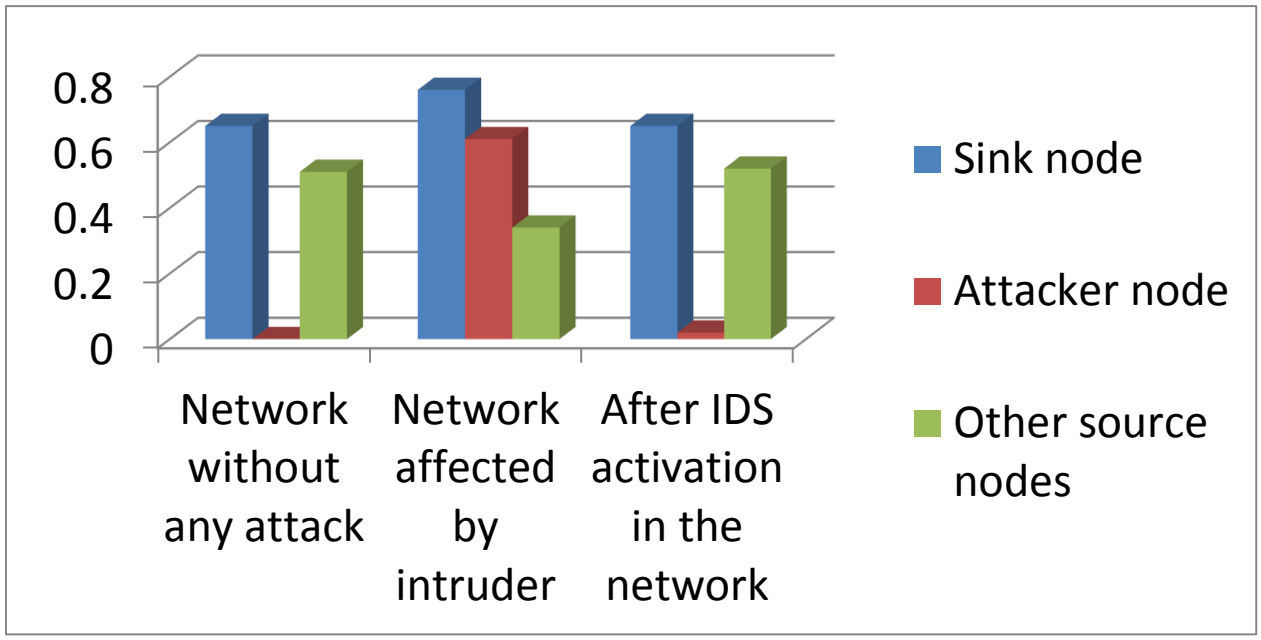

Figure 7.23 Average Power Consumption (mW) in version number attack

7.1.7 Memory consumption

Contiki is designed for systems with only a few kilobytes of memory. Therefore Contiki is highly memory efficient and provides memory allocation mechanism: memb- memory block allocation, mmem- a managed memory allocator, and supports malloc. Implementing Contiki-MAC consumes a lot of memory, but reduces power consumption to approximately 1 mW from around 60 mW. Different algorithms can be implemented for application layer security when Contiki-MAC and RPL are not included. Since the current work revolves around battery motes and securing RPL, both Contiki-MAC and RPL can be implemented [17].

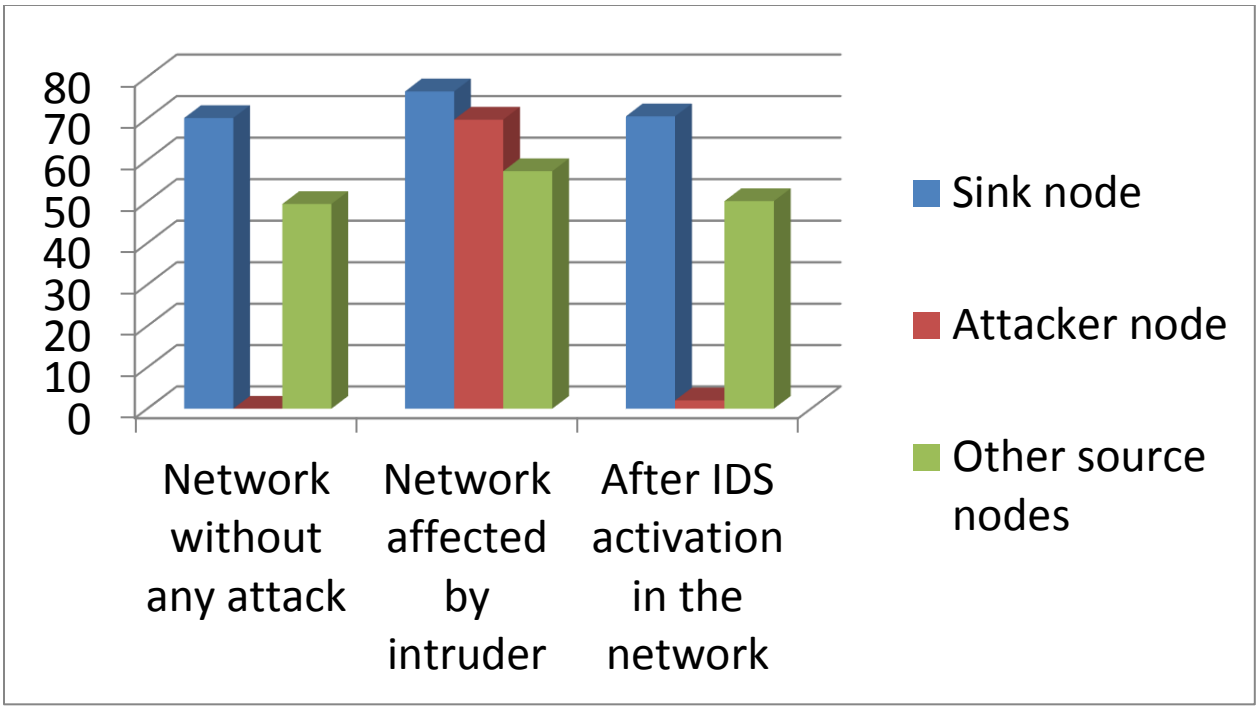

Figure 7.24 Memory allocations (Size in Kb) in flooding attack

Figure 7.24, shows the memory consumption of the RPL network without attack, network affected by intruder (flooding number attack) and after IDS activation in the network.

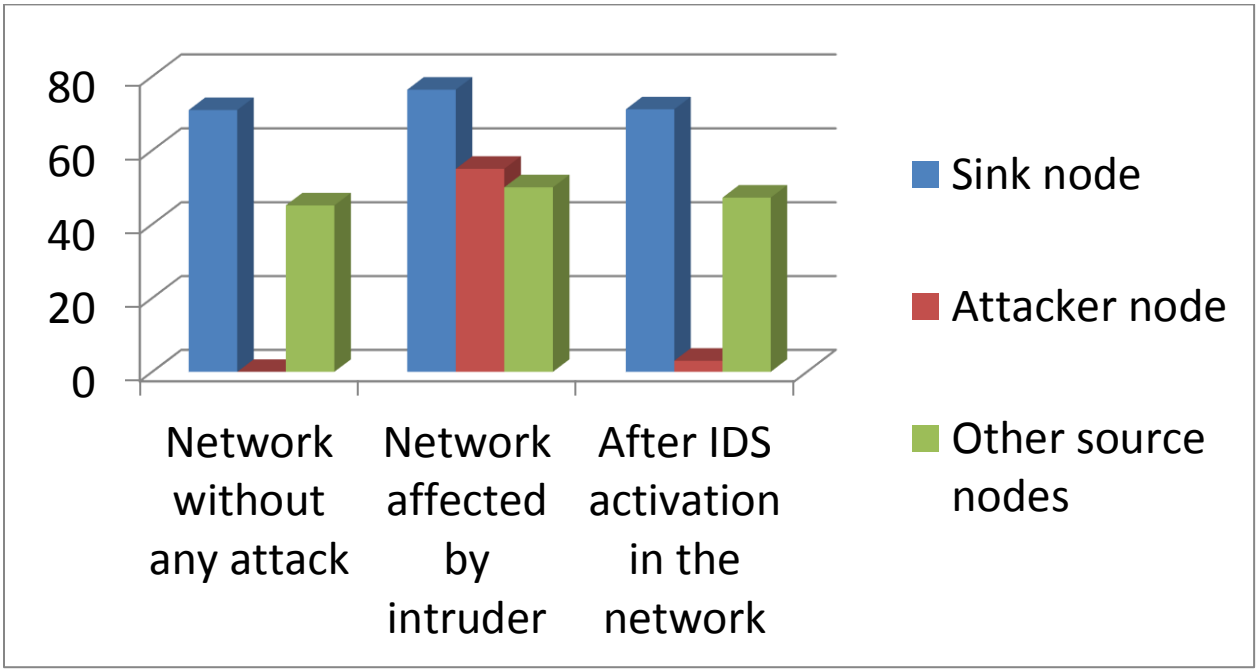

Figure 7.25 Memory allocations (Size in Kb) in version number attack

Figure 7.25, shows the memory consumption of the RPL network without attack, network affected by intruder (version number attack) and after IDS activation in the network. The memory consumed by the attacker is the highest of all the other types of nodes in the network. On activation of HIDS, it is obvious that the memory allocation is reduced for all the nodes affected by flooding attack viz. sink node, attacker node and source node.

7.1.8 Average Radio Duty Cycling

Several MAC duty-cycle protocols have been proposed during the last decade to address specific WSNs requirements and constraints such as a low power consumption related to battery operated nodes.

Radio Duty-Cycle(RDC) MAC protocols try to reduce the energy consumption by allowing a node to keep its radio-transceiver off most of the time. Contiki has three duty cycling mechanisms: Contiki MAC, X-MAC and LPP. Contiki MAC is a radio duty cycling protocol that uses periodical wake-ups to listen for packet transmission from neighbors.

Contiki's X-MAC is based on the original X-MAC protocol, but has been enhanced to reduce power consumption and maintain good network conditions. Contiki's LPP is based on the Low-Power Probing (LPP) protocol but with enhancements that improve power consumption as well as provide mechanisms for sending broadcast data. Average radio duty cycling is related to power.

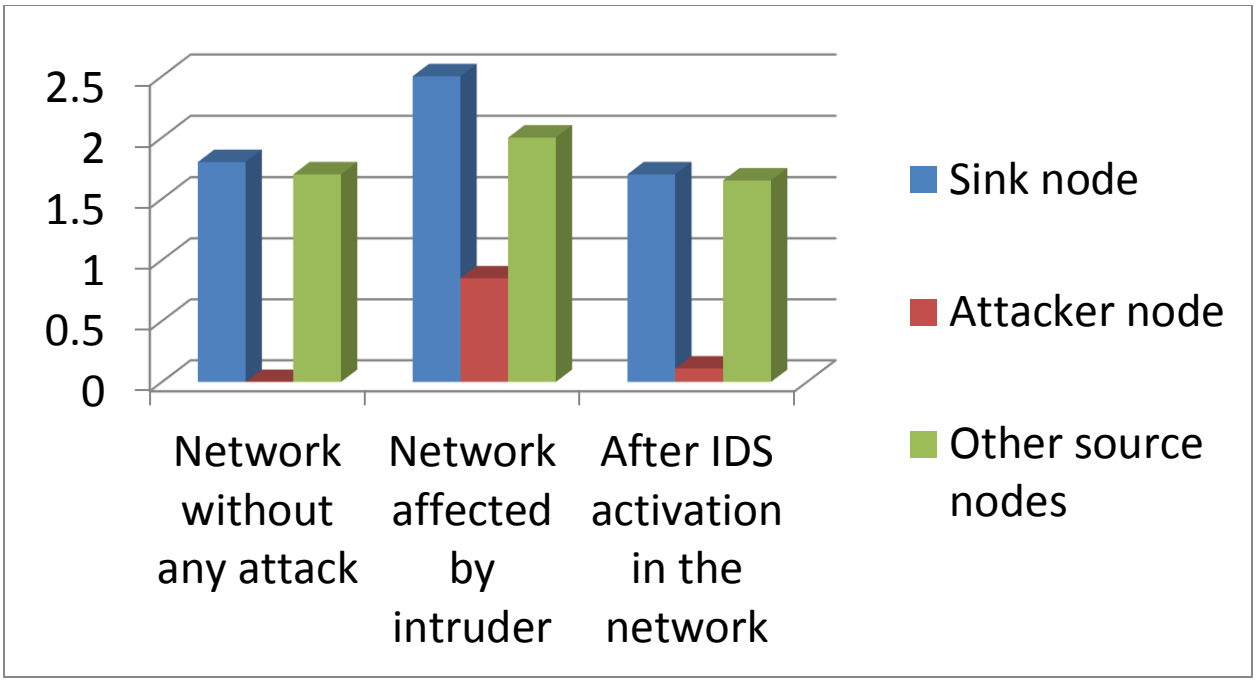

Figure 7.26 Average Radio Duty Cycle in flooding attack

Figure 7.26, represents the average radio duty cycle with flooding attack in percentage (%). The network with IDS will have a good average radio duty cycle when compared to network without IDS activation.

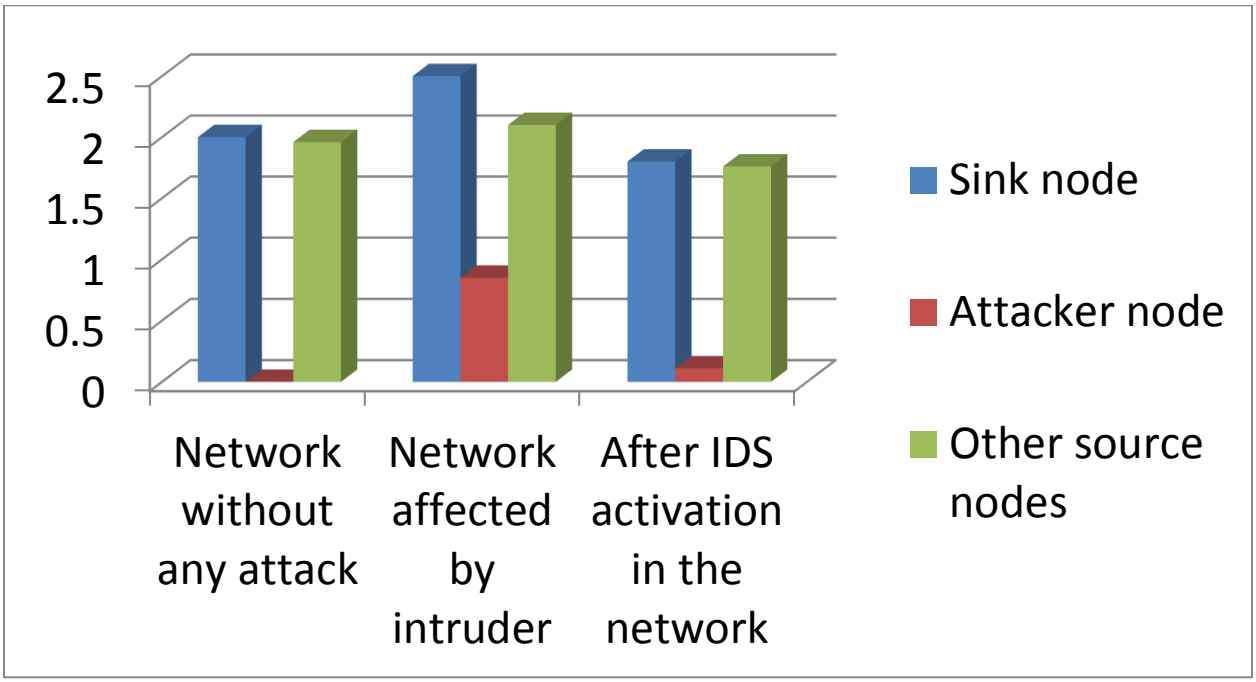

Figure 7.27 Average Radio Duty Cycle in version number attack

Figure 7.27, represents the average radio duty cycle with version number attack in percentage (%). The network with IDS will have a good average radio duty cycle when compared to network without IDS activation. The average radio duty cycle for different nodes of RPL network with attack and without attack is been plotted.

7.1.9 Packet delivery ratio

Packet delivery ratio is a ratio of the number of successfully received to the number of transmitted packets. The formula for calculating packet delivery ratio is

Packet delivery ratio = (successfully received/transmitted packets)*100

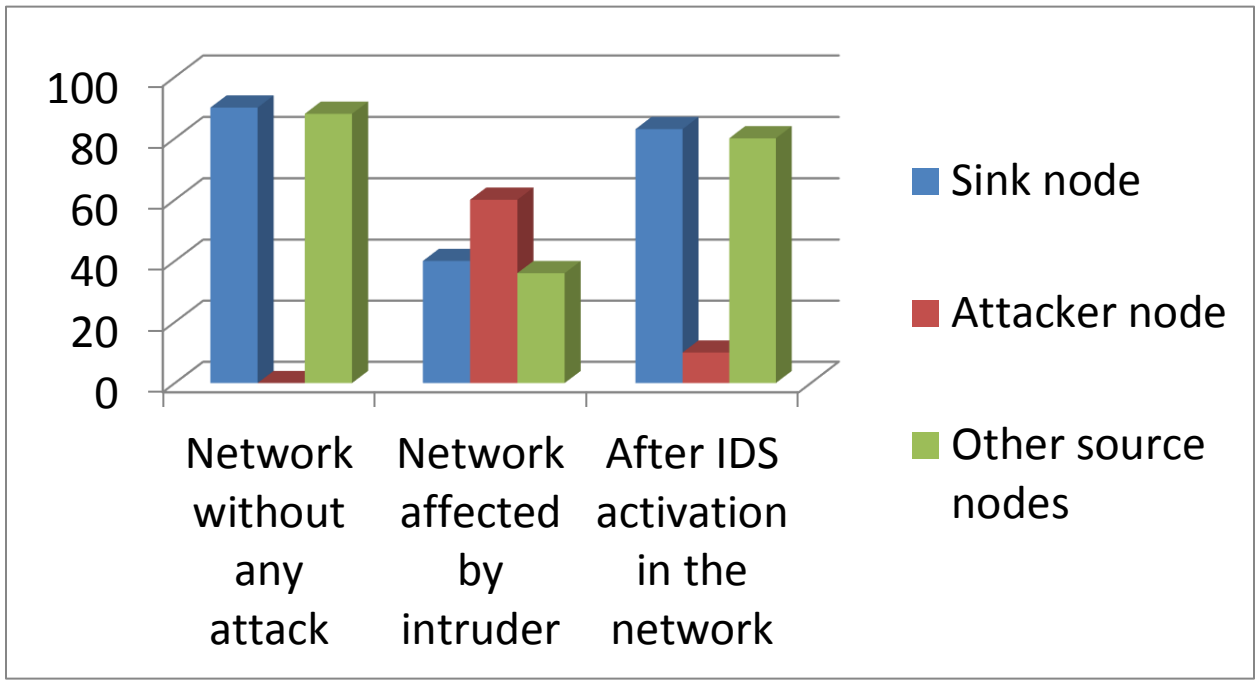

Figure 7.28 Packet delivery ratio in flooding attack

Figure 7.28, represents the packet delivery ratio in flooding attack in percentage (%). The network with IDS will have a good packet delivery ratio when compared to network without IDS activation. The packet delivery ratio of the different nodes for RPL network with attack and without attack is been plotted.

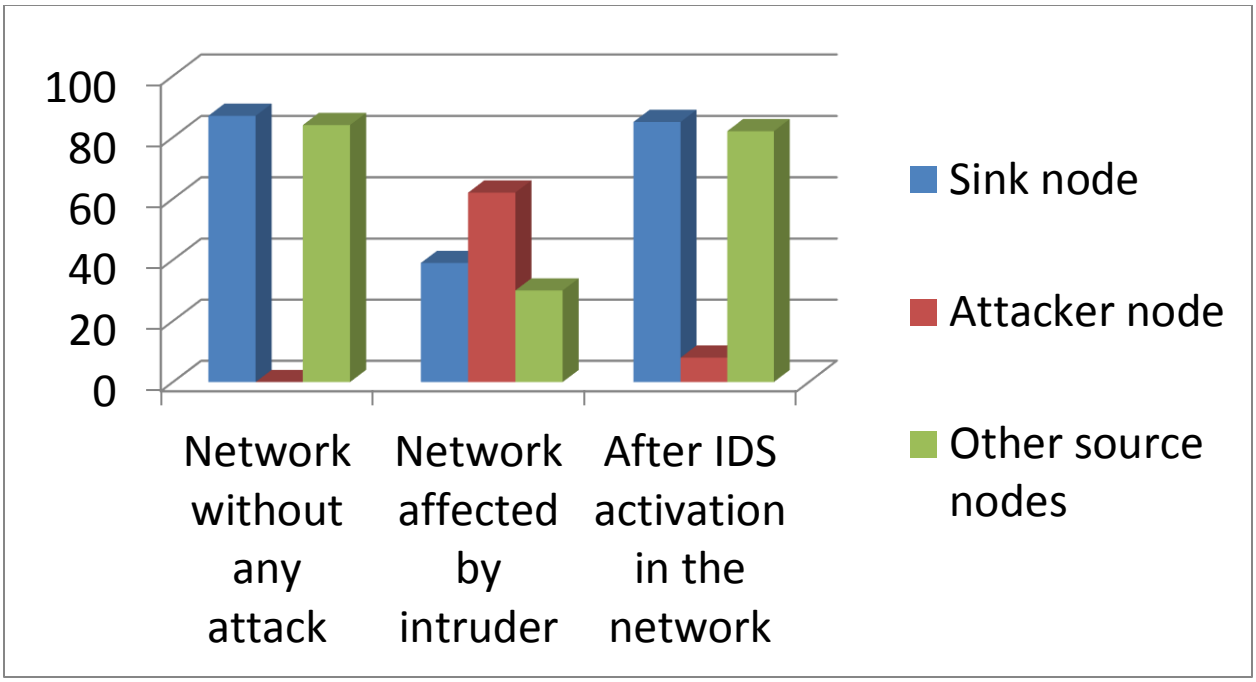

Figure 7.29 Packet delivery ratio in version number attack

In the figure 7.29, the packet delivery ratio in version number attack in percentage (%). The network without any attack will have a better packet delivery ratio when compared to IDS activation in the network. The packet delivery ratio of the different nodes for RPL network with attack and without attack is been plotted.

CHAPTER 8

CONCLUSION AND FUTURE WORK

8.1 Conclusion

The LLN's are always constrained in resources such as energy, processing time and memory. The IETF has proposed a standardized routing protocol known as RPL. Due to the nature of these wireless networks while being connected with real world are affected by various types of intruder attacks. Attacks created by the intruder results in multiple looping and data packets losses leading to repeated unnecessary re-organization of the network. This ends up with predominant increase in control message overhead and exhaustion of nodes resources due to network congestion.

The proposed methodology is based on the detection of intruder, classification of attacks and protection against those attacks. The HIDS developed has better network performance in the following three different categories. In the first category, IDS works against the attacks created by the intruder mote which affects the network life time by generating fake control messages or by rebuilding of loops. In the second category, IDS is against the attacks created towards the topology which make the network convergence delayed and resulting in suboptimal configuration with isolation of the malicious motes. Finally in the third category, IDS works against attacks creating the enormous fake network traffic by the malicious mote. Based on this taxonomy, the HIDS provides necessary detection methods and techniques to avoid or prevent the network from further attacks.

Though most of the security systems in literature are still at proof-of-concept level, which proceeds in the path of network performance degradation and some wireless networks are even limited in connectivity with Internet of Things. Hence this step of designing the proposed HIDS provides new perspectives in providing solution to these problems. The proposed HIDS can be implemented dynamically in any node in the network and can be exploited to the core for its cent percent co-operation to establish the required security in any real time scenario.

8.2 Future work

The future work could be the hardware implementation of the tested simulation. The Hardware requirements for the implementation of the network could be possible using Processor board (like raspberry pi 2, ardunio, etc), motes with battery. There are few hardware suitable for Contiki OS using Cooja simulator such as Atmega 128rfa1 a complete integrated circuit (IC) which comes with inbuilt radio frequency (RF) transceivers. These are more compactable but are quiet costlier.

REFERENCES

1. Anhtuan Le, Jonathan Loo, AboubakerLasebae, Alexey Vinel, Yue Chen, and Michael Chai, "The Impact of Rank Attack on Network Topologyof Routing Protocol for Low-Power And Lossy Networks," IEEE Sensors Journal, Vol. 13, No. 10, ,pp. 3685-3692, October 2013.

2. Adnan Aijaz, and A. Hamid Aghvami, "Cognitive Machine-to-Machine Communications for Internet-of-Things: A Protocol Stack Perspective Cognitive Machine-to-Machine Communications for Internet-of-Things: A Protocol Stack Perspective," IEEE Internet of Things Journal, Vol. 2, No. 2, pp.103-112, April 2015.

3. Jan Kantert, Christian Ringwald, Georg von Zengen, Sven Tomforde, Lars Wolf, and Christian Müller-Schloer, "Enhancing RPL for Robust and Efficient Routing in Challenging Environments,"9th International Conference on Self-Adaptive and Self-Organizing Systems Workshops,Cambridge, MA, USA, pp. 7-12, September 2015.

4. Lan Zhang, Gang Feng, and Shuang Qin, "Intrusion Detection System for RPL from Routing Choice Intrusion,"IEEE ICC 2015-Workshop on Security and Privacy for Internet of Things and Cyber-Physical Systems, London, pp. 2652- 2658, June 2015.

5. AntheaMayzaud, RemiBadonnel, and Isabelle Chrisment, "A Taxonomy of Attacks in RPL-based Internet of Things," International Journal of Network Security, vol.18, no.3, pp.459-473, May 2016.

6. Emilio Ancillotti, Raffaele Bruno, and Marco Conti,"Reliable Data Delivery With the IETF Routing Protocol for Low-Power and Lossy Networks," IEEE Transactions on Industrial Informatics, vol. 10, no. 3, pp. 1864-1877, August 2014.

7. Mohan V. Pawar, and J.Anuradha, "Network Security and Types of Attacks in Network," International Conference on Intelligent Computing, Communication & Convergence, Bhubaneswar, Odisha,pp. 503-506, 2015.

8. Linuswallgren, ShahidRaza, and Thiemo Voigt, " Routing Attacks and Countermeasures in the RPL-Based Internet of Things," International Journal of Distributed Sensor Networks vol. 9 no. 8, pp. 1-12, August 2013.

9. KarishmaChugh, Aboubaker Lasebae, and Jonathan Loo,"Case Study of a Black Hole Attack on 6lowpan-RPL," SECURWARE 2012: The Sixth International Conference on Emerging Security Information, Systems and Technologies, Rome, pp. 157-162,August 2012.

10. KarelHeurtefeux, OchirkhandErdene-Ochir, NasreenMohsin and Hamid Menouar, "Enhancing RPL Resilience Against Routing LayerInsider Attacks," IEEE 29th International Conference on Advanced Information Networking and Applications (AINA), 24-27,Gwangju, pp. 802 – 807, March 2015.

11. TariqahmadSherasiya, and HardikUpadhyay,"Intrusion Detection System for Internet of Things," International Journal of Advance Research and Innovative Ideas in Education, vol. 2 no. 3,pp. 2244-2249, 2016.

12. Kevin Weekly, and Kristofer Pister, "Evaluating Sinkhole Defense Techniques in RPL Networks,"20th IEEE International Conference on Network Protocols (ICNP), Austin, Texas, pp. 1-6, October 2012.

13. HassenRedwan and Ki-Hyung Kim,"Survey of Security Requirements, Attacks and Network Integration in Wireless Mesh Networks," 2008 Japan-China Joint Workshop on Frontier of Computer Science and Technology, NTMS, pp.3-9, November2008.

14. AhmetAris, Sema F. Oktug, and S. BernaOrsYalcin,"RPL Version Number Attacks: In-depth Study," IEEE/IFIPNetwork Operations and Management Symposium (NOMS), Istanbul, Turkey, pp. 776-779, April 2016.

15. PavanPongle, and GurunathChavan, "Real Time Intrusion and Wormhole Attack Detection in Internet of Things," International Journal of Computer Applications, Vol. 121, no. 9, pp. 1-9, July 2015.

16. Anhtuan Le, Jonathan Loo, Yuan Luo, and AboubakerLasebae,"Specification-based IDS for securing RPL from topology attack," IFIP Wireless Days (WD), Niagara Falls, ON, pp. 1-3, October 2011.

17. http://anrg.usc.edu/contiki/index.php/Contiki_Programming_Guide.