Security Analysis of RPL Protocol

Project Report submitted in partial fulfillment of the requirement for the degree of

B.Tech

In

Information Technology

under the Supervision of

Dr. Geetanjali

By

RAJAT KUMAR AGGARWAL

Roll No. 141405

Jaypee University of Information Technology

Waknaghat, Solan – 173234, Himachal Pradesh

Certificate

This is to certify that project report entitled "**Security Analysis of RPL Protocol**", submitted by **Rajat kumar Aggarwal** in partial fulfillment for the award of degree of Master of Technology in Computer Science & Engineering to Jaypee University of Information Technology, Waknaghat, Solan has been made under my supervision.

This report has not been submitted partially or fully to any other University or Institute for the award of this or any other degree or diploma.

Date: **Dr. Geetanjali**

Assistant Professor

Acknowledgement

It is always said that God show the path in any challenge and any walk of life, so firstly my heartiest reverence to my Guru and God whose blessings have filled my life with wisdom, joy and prosperity.

I would like to express my immense gratitude and appreciate to my Supervisor for his advice, support, guidance and inspiration throughout the course of my study. His enthusiasm and practical views on research have made a deep impression on me. I owe him my deepest gratitude for showing me the way to study and to conduct research.

I would also like to thank all the staff of JUIT, Solan for their friendly behavior and assistance during my research study. Special thanks also due to my Friends for their assistance and expert advice.

Finally, without the support of my Family, their understanding and patience, it would have been impossible for me to finally complete my study.

Date:

Signature:

RAJAT KUMAR AGGARWAL

Table of Content

Chapter-1

Chapter-2

Chapter-3

Chapter-4

Chapter-5

List of Figures

List of Tables

Abstract

Internet of Things is one of the leading domains of research in the distributed and secured computing using wireless technologies. There are assorted attacks and vulnerability factors which are frequently analyzed. In this work, the implementation of security protocol for higher effectiveness in of RPL refers to Routing Protocol for Low Power and Lossy Networks is done with the integration of dynamic hash security is presented with the implementation in IoT platform Contiki Cooja. It is found from the results that the dynamic hash based security is performance aware approach that can escalate the overall integrity of the wireless based IoT environment. The observations and implementation are done using Cooja code and the results are effectual in the assorted versions of dynamic hash based security.

Keywords: Cooja, Contiki Platform for IoT, Security in Internet of Things, Internet of Things, IoT Security, RPL Protocol

Chapter-1

INTRODUCTION

1.1 Introduction

With the increasing traffic on network based applications, the security is becoming a prominent issue so that the network environment can be safe from different types of attacks. The probability of vulnerabilities in network or web based applications increases if the vulnerability testing is not done properly. In traditional implementations, the network administrators use their own set of tools for the testing of their network environment but such tools can be restricted to specific types of attacks. It is always desired that the administrators should use different types of penetration and vulnerability testing tools which are meant to assorted attacks. It is done to check the overall deployment on different types of attacks. This methodology ensures that the network or web based environment is secured from multiple attacks without any compromise on security.

A number of frameworks and software tools are available which provides the features to evaluate the network or web based application on different aspects and parameters of security audit. In security audit of web application or network devices, the loopholes or vulnerabilities are checked from different dimensions so that the attackers or sniffers cannot destroy their environment. Traditionally, the use of penetration testing tools is done by the network administrators and application developers to analyze the weak-points or vulnerabilities. In penetration testing, the application or devices are put to the pre-programmed attacks so that the actual behavior of hardware or software can be checked. If the application or hardware devices react in abnormal way during penetration testing, the suitable remedial actions and troubleshooting is done to cope up with the attacks. Penetration testing can be implemented on any type of deployment including network, devices, websites, servers or software installations and the figure1.1 is below.

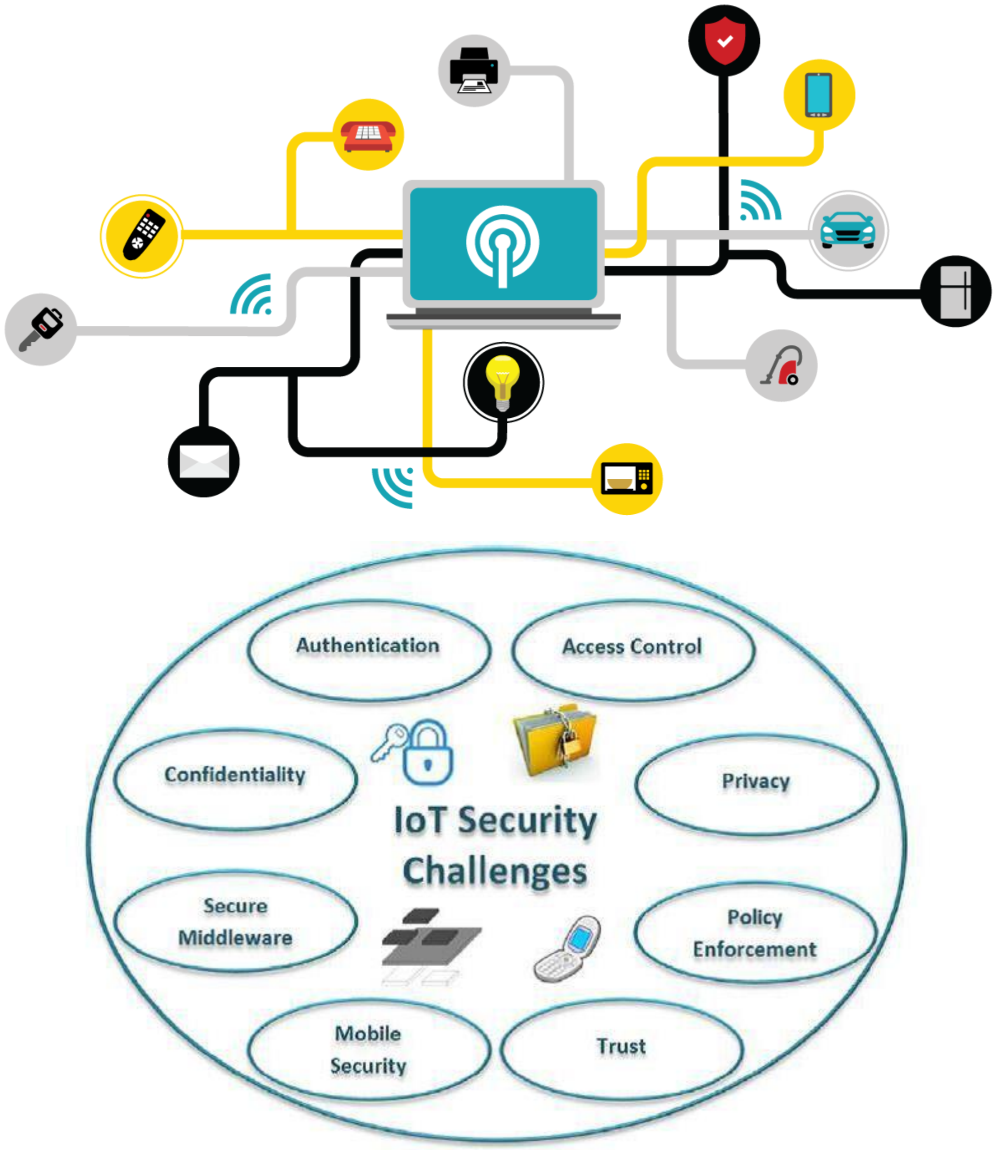


Figure 1.1. IoT and Security Challenges [1]

RPL in IoT

RPL refers to Routing Protocol over Low Power and Lossy Networks that is the standardized protocol specifically for Internet of Things (IoT) with higher degree of efficiency and transmission rate but that is susceptible to assorted attacks [1]. This is the key aspect of work done in this dimension so that the overall efficiency in terms of security, latency and throughput can be reduced against different types of attacks.

A number of algorithms and approaches are devised and worked out by number of researchers, scientists and network practitioners still there is huge scope of research work with the integration of security protocols and layers of the RPL environment and the figure1.2 as shown.

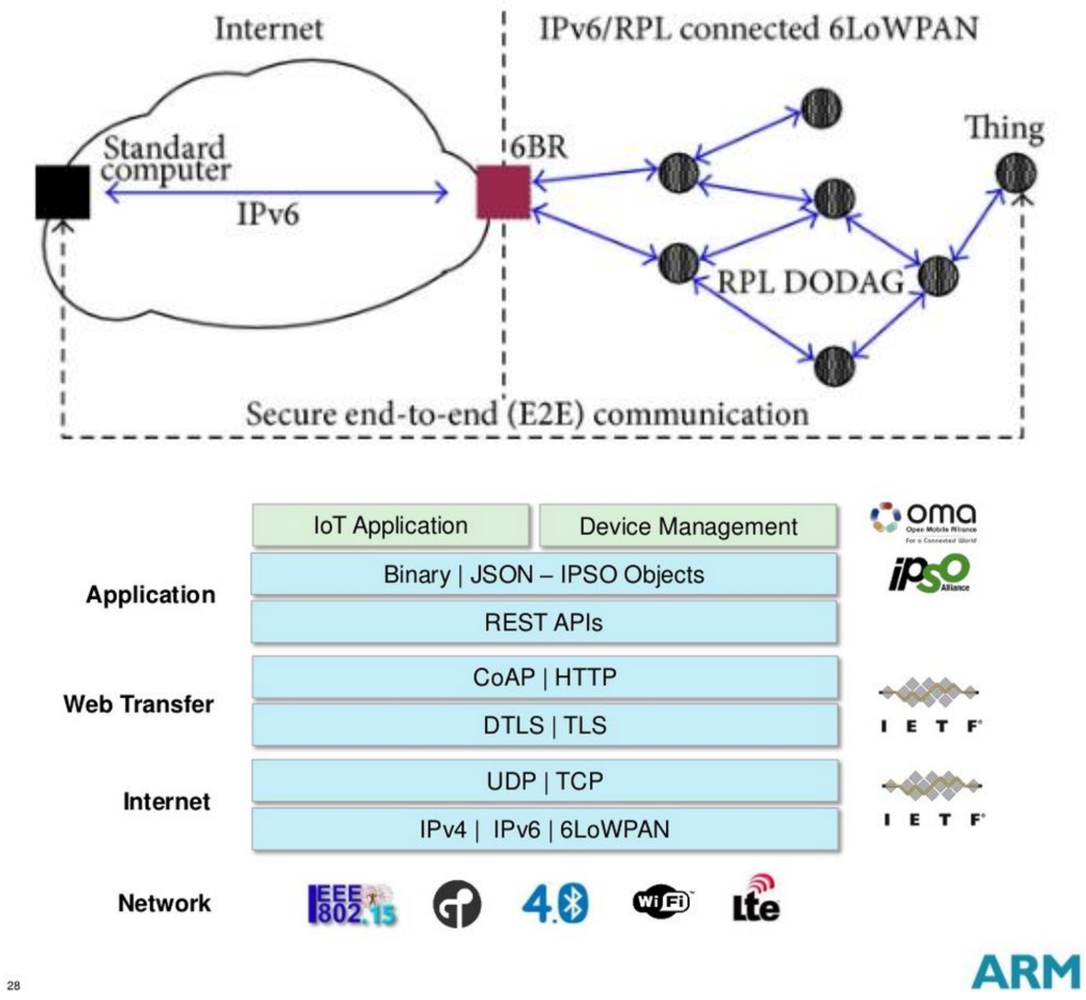
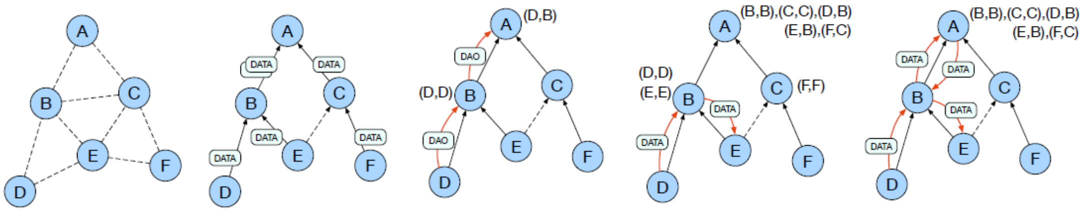


Figure 1.2: RPL 6LowPAN Environment [2]

There are number of aspects and dimensions with number of attacks and vulnerable issues which must be taken care while deployment of the Internet of Things (IoT) based environment so that overall transmission and routing can be made secured. Here, the transmission channels and the routing aspects are presented which may be susceptible to the sniffers and assaults and therefore needs higher degree of integrity and security and the figure1.3 is below of DODAG nodes.



(a) A sample wireless network.
(b) Multipoint-to-point communication.
(c) Point-to-multipoint route construction: storing mode.
(d) Point-to-point communication: storing mode.
(e) Point-to-point communication: non storing mode.

Figure 1.3: Routing in RPL [2]

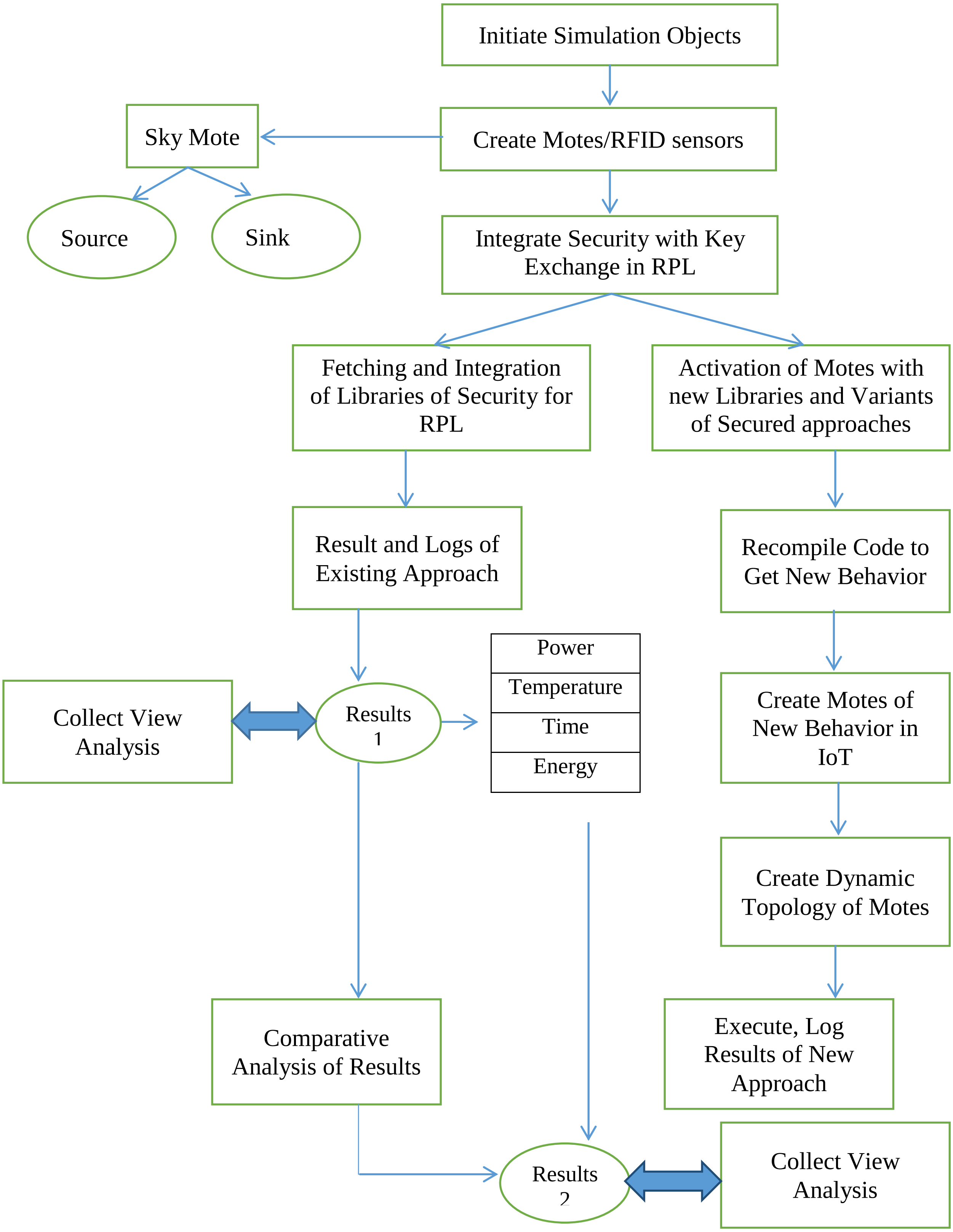
1.2 Problem Statement

With the increasing assaults, there is need to implement the security based approach on RPL in IoT environment so that overall integrity and performance can be retained in the Internet of Things.

1.3 Objectives

1. To identify assorted attacks and security protocols
2. To perform the detailed survey of literature associated with IoT security
3. To implement the proposed approach on Contiki Cooja environment

1.4 Methodology



1.5 Organization

Chapter 1 : Introduction. This chapter gives information about Internet of Things, Network Attacks and RPL.

Chapter 2 : Literature Review. This chapter presents the extracts from assorted research papers and articles.

Chapter 3 : System Development is presenting the research perspectives, objectives and projected findings with the methodology factors of the work.

Chapter 4 : Performance Analysis gives the data interpretation results and analysis. In this the results are presented in form of assorted graphs and charts to the results in presentable aspects.

Chapter 5: Conclusion and Future Scope summarizes and concludes the work with the scope of future work.

Chapter-2

LITERATURE SURVEY

This segment is present the extracts of literature with the analysis of similar domain and the suggestive remarks on security of IoT. Enormous multi-sources based manuscripts, research papers and articles are analyzed from the time span up to recent time so that the latest trends in security and IoT can be evaluated.

2.1 Literature Review

After analyzing and extraction of contents with the technical mechanism, the inferences are drawn so that the further research can be initiative towards implementation and fetching the results.

The brief descriptions of the papers which are useful for drawing inference are mentioned as follows.

Table 2.1. Extracts from the Literature and Related Work

Author	**Technique and Algorithm used**	**Key Issues and Limitations**	**Year**
V. Gampala [3]	ECC	Complexity	2012
H. Ning [4]	Two layer intrusion detection approach	Overheads	2012
M. Agrawal [5]	Symmetric key cryptogra	Latency and Time Factors	2012

	phy mechanisms for network scenarios.		
D. Kozlov [6]	Secured Architecture	Execution Time and Latency	2012
X. Qian [7]	IDS Based Security	Generalization	2012
R. Roman [8]	Use of higher degree of security required in the distributed environment.	Execution Time and Complexity	2013
T. Kothmayr [9]	Datagram Transport Layer Security (DTLS) security	Resource Consumption	2013
J. Gubbi [10]	Cloud Centric Vision	Integrity and Privacy	2013
Mahalle [11]	Identity Authentication and Capability Based Access Control (IACAC)	Complexity	2013
R. Hummen et al. [12]	Integration of DTLS based handshake which are based on	Integrity and Privacy	2013

Q. Jing et al [13]	RFID based secured transmission	Complexity	2014
Z. Yan [14]	Novel Trust Evaluation and Integrity Protocol	Latency	2014
D. Lake et al. [15]	Telemedicine with IoT	Latency	2014
Y. Ning [16]	Network layer based Security	Time	2014
M. Turkanovic [17]	Hybrid Authentication	Time and latency factors	2014

S. Sicari [18]	Privacy Aware Approach	Complexity	2015
J. Granjal [19]	IoT Architecture	Overheads	2015
K. T. Nguyen [20]	IPv6 enabled Secured Architecture	Additional resources	2015
M. Vucinic et al. [21]	OSCAR Approach	Privacy	2015
W. Trappe [22]	Multilayered Architecture for	Integrity	2015

	Security		

F. Li et al. [23]	Security with Multiple Keys	Integrity	2016
S. R. Moosavi [24]	Multi-Level Approach and Algorithm	Resource optimization	2016
K. A. Rehiman [25]	Secured Key Based Approach	Resource factors	2016
D. Airehrour [26]	IoT security with multiple layers	Latency	2016
E. Bertino [27]	Trust Management	Overheads	2016

M. Usman et al. [28]	SIT	Complexity	2017
M. B. Mollah et al. [29]	Cloud Technologies	Latency	2017
P. P. Jayaraman et al. [30]	Multilayered Architecture for Security	Generalization	2017
C. Schmitt et al. [31]	Two way solution for the authentication and overall security in	Generalization	2017

	the Low Power Wireless Networks.		
S. Prabhakar [32]	Cloud Environment	Generalization	2017

The following inferences can be drawn from literature survey

Security and integrity are the key issues in Internet of Things (IoT) which is still the domain of research as number of interconnecting devices are increasing.

- There is need to implement different types of hash algorithms in the IoT Simulated Environment for evaluation of different parameters.
- As the work on dynamic hash security is not implemented in the Cooja Platform, the implementation in Cooja provides the scenarios and performance of hash approaches.
- The implementation of dynamic hash security is quite novel in Contiki and Cooja and therefore the simulation in this segment with multiple motes can be done.
- As Cooja is comparatively performance aware IoT platform, there is need to evaluate the performance of hash approach using this library.

Chapter-3

SYSTEM DEVELOPMENT

Problem Identification and Statement is one of the key tasks in any research work. This segment of the research report presents the need of research work and the key points which motivated to work in this domain. The problem formulation and related research methodology is presented in this chapter along with the research objectives and proposed work.

3.1 Problem Formulation and Proposed Work

The implementation of Dynamic Hash Based Security on Cooja based environment is required as this library makes use of sensor nodes or motes. In addition, the Cooja based framework not equipped with Dynamic Hash Based Security in traditional approach. In this research work, the multiple variants of Dynamic Hash Based Security are implemented to evaluate the performance of each Dynamic Hash Based Security variant in multiple sensor notes in IoT environment. Secured Hash Approach is one of the key mechanisms for security and integrity. The implementation of Dynamic Hash Based Security on Cooja is not performed so far in the research implementations and that is the key motive to work in this segment. In the segment of security and dynamic encryption, Dynamic Hash Based Security is a family of cryptography algorithms having various flavors in which the keys of different size and hash are generated and these can be used for security and overall integrity of network transmission.

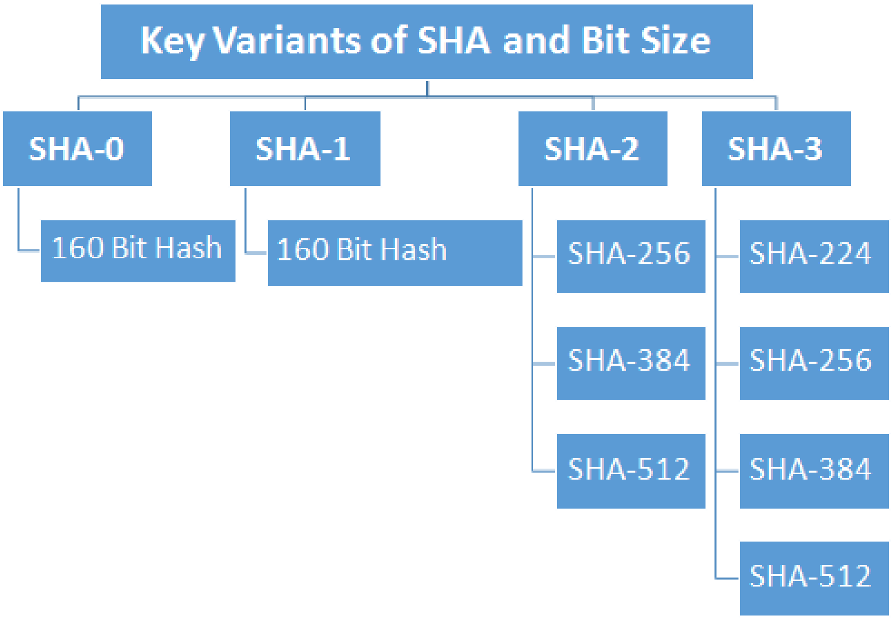


Fig. 3.1. Key Variants and Bit Size of Dynamic Hash Based Security

Figure 3.1 depicts the different versions and inherent properties of Dynamic Hash Based Security. This research work is having focus on the evaluation of these variants for analytics in IoT environment using Cooja.

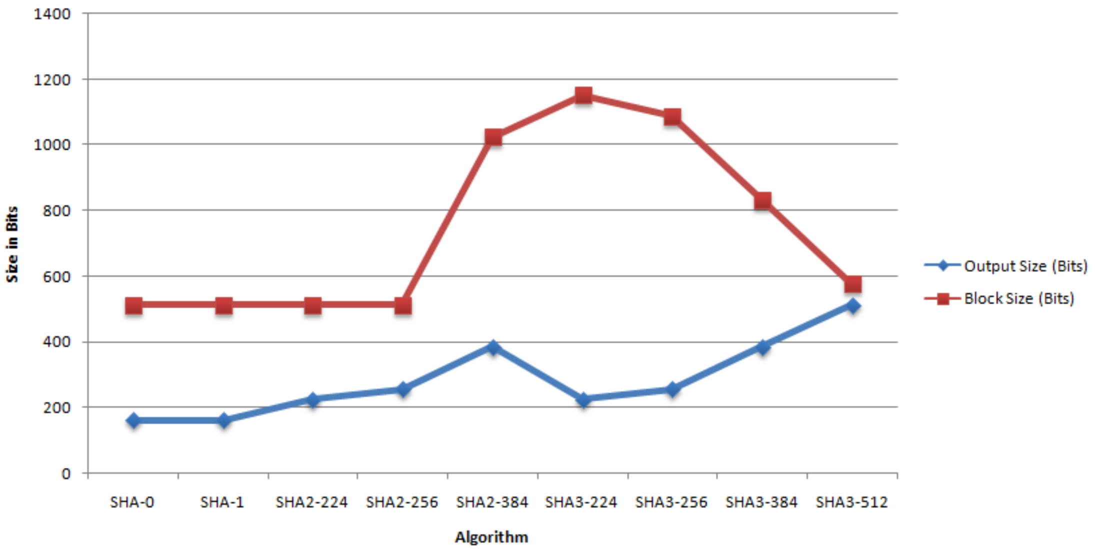


Fig. 3.2. Line Graph of Key Variants and Bit Size of Dynamic Hash Based Security

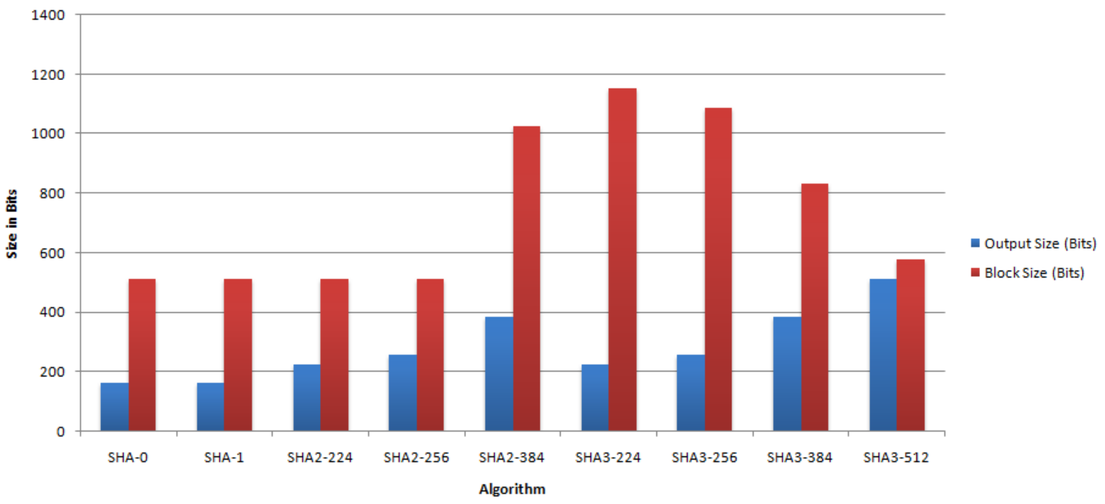


Fig. 3.3. Bar Graph Comparison of Key Variants and Bit Size of Dynamic Hash Based Security

3.2 Proposed Work

The present research work implements the variants of Dynamic Hash Based Security on IoT scenario. The key variants implemented in this research work are DynamicHashAlgorithm-1, DynamicHashAlgorithm-2-256 and DynamicHashAlgorithm-3-256 and their impact on multiple parameters and resource optimization factors in the Internet of Things Scenario.

The following steps are taken

1. Simulation objects are initiated.
2. Sky motes/RFIDs are created by defining source and sink of data.
3. IPV6 based algorithms in Cooja are activated.
4. With existing algorithms Sky motes of default behavior are created.
5. Results in terms of Time, Energy etc. are collected and analyzed.
6. Dynamic Key Exchange (DynamicHashAlgorithm-1, DynamicHashAlgorithm-2-256 and DynamicHashAlgorithm-3-256) is integrated in C language code of algorithms in Cooja.
7. Integrated code is recompiled to get new behavior and motes of new behavior are created.
8. Dynamic topology is created.
9. Result and Log of new approach are collected and analyzed.
10. The results of existing and new approach are compared.

Advantages of the Research Work

- The valuation of variants on secured hash approaches gives clear view of the performance factors in each flavor of Dynamic Hash Based Security.
- Dynamic Hash Based Security is one of the widely used approaches for generation of keys still its performance in distributed and IoT based environment gives another dimension to evaluate the performance.
- The key advantage to work on this domain is the assessment of different parameters which are paramount in Internet of Things.
- Evaluation of variants of Dynamic Hash Based Security using Cooja gives the working scenario on security in Internet of Things (IoT).
- IoT based implementation of Dynamic Hash Based Security on Cooja provides the performance in distribute environment.

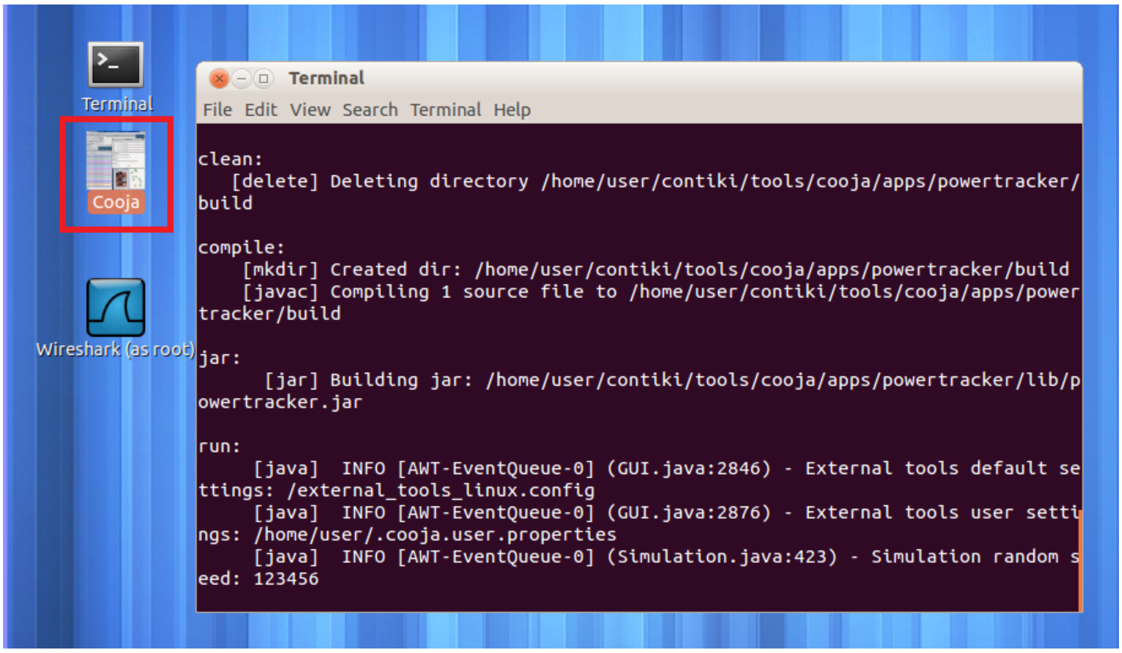
Chapter-4

PERFORMANCE ANALYSIS

Implementation is a mandatory point to justify, defend and prove the research work based on simulation or data analytics. In this segment, the implementation technologies, strategies and parameters are specified which are required to validate and rationalize the proposed research work.

4.1 Implementation Strategy

- Fetching the Libraries of Contiki and Cooja
- Creation of RFID Nodes
- Creation and Activation of Motes
- Generation of IPv6 and IPv4 based transmission channel
- Evaluation of motes using collect view
- Generation of working environment for Cooja



```
clean:
   [delete] Deleting directory /home/user/contiki/tools/cooja/apps/powertracker/
build

compile:
    [mkdir] Created dir: /home/user/contiki/tools/cooja/apps/powertracker/build
    [javac] Compiling 1 source file to /home/user/contiki/tools/cooja/apps/power
tracker/build

jar:
      [jar] Building jar: /home/user/contiki/tools/cooja/apps/powertracker/lib/p
owertracker.jar

run:
     [java]  INFO [AWT-EventQueue-0] (GUI.java:2846) - External tools default se
ttings: /external_tools_linux.config
     [java]  INFO [AWT-EventQueue-0] (GUI.java:2876) - External tools user setti
ngs: /home/user/.cooja.user.properties
     [java]  INFO [AWT-EventQueue-0] (Simulation.java:423) - Simulation random s
eed: 123456
```

Figure 4.1: Cooja Platform for IoT Implementation

Figure 4.1 presents the loading screen of Cooja Simulator in Contiki. This implementation is done using VMWare in Windows Environment.

Creation of IoT Environment in Cooja

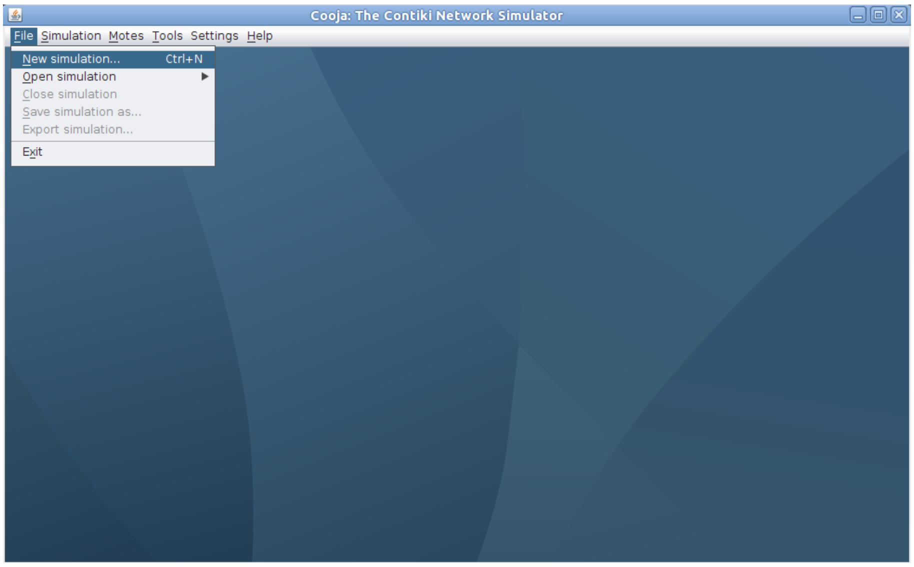


Figure 4.2: Creation of New Network Environment

From the dialog box, the network environment is set with the inherent parameters and associated modules. This environment load the components required for simulating the IPv6 environment with RPL security as per the objectives.

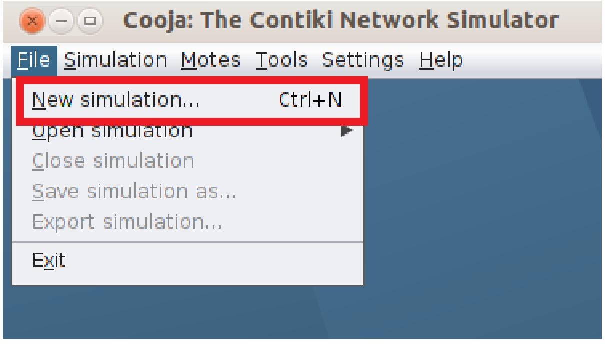
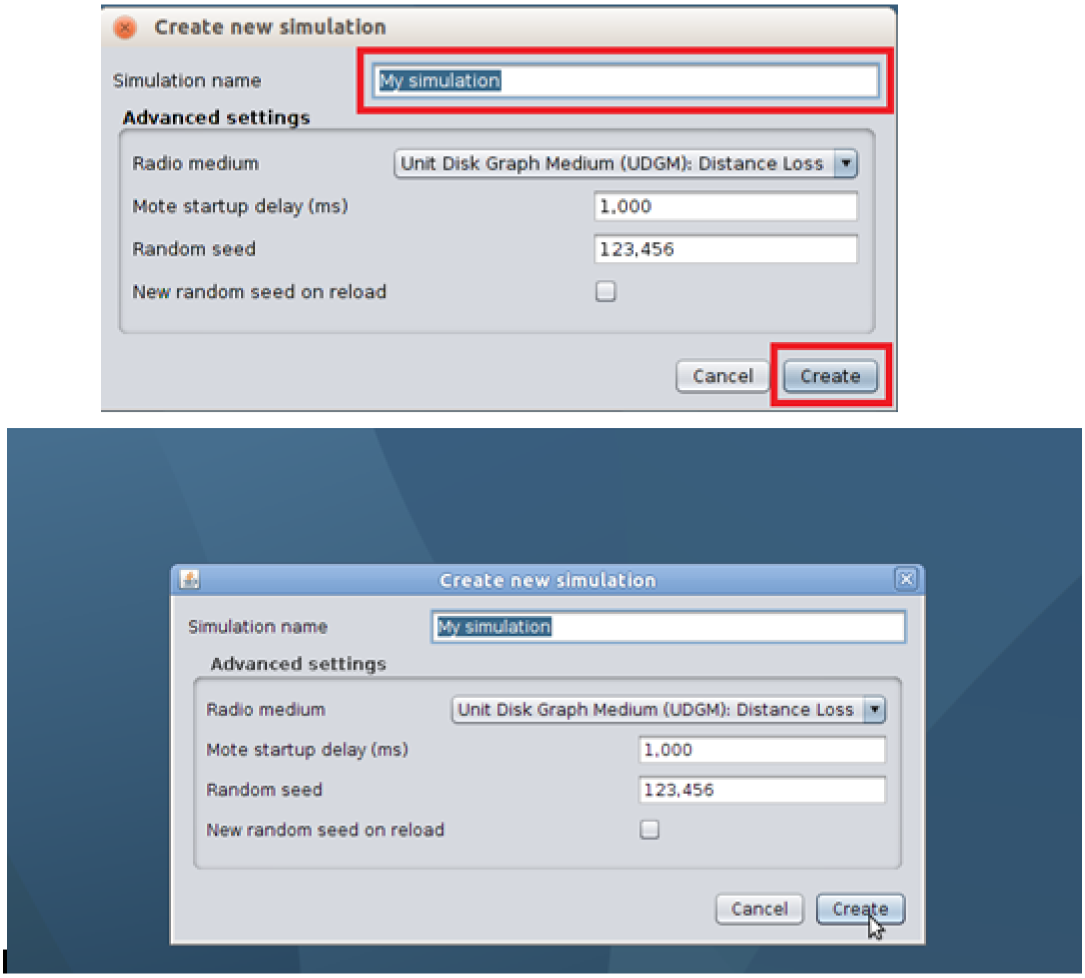


Figure 4.3: Selection of Cooja Network



Simulation name: My simulation

Advanced settings

Radio medium: Unit Disk Graph Medium (UDGM): Distance Loss

Mote startup delay (ms): 1,000

Random seed: 123,456

New random seed on reload: ☐

Simulation name: My simulation

Advanced settings

Radio medium: Unit Disk Graph Medium (UDGM): Distance Loss

Mote startup delay (ms): 1,000

Random seed: 123,456

New random seed on reload: ☐

Figure 4.4: Parameters Setting in IoT Cooja Simulator

Figure 4.4 presents the radio medium, random seed and startup parameters so that the IoT environment can be setup. The installation and deployment with logging of real time sensor can be done in Cooja.

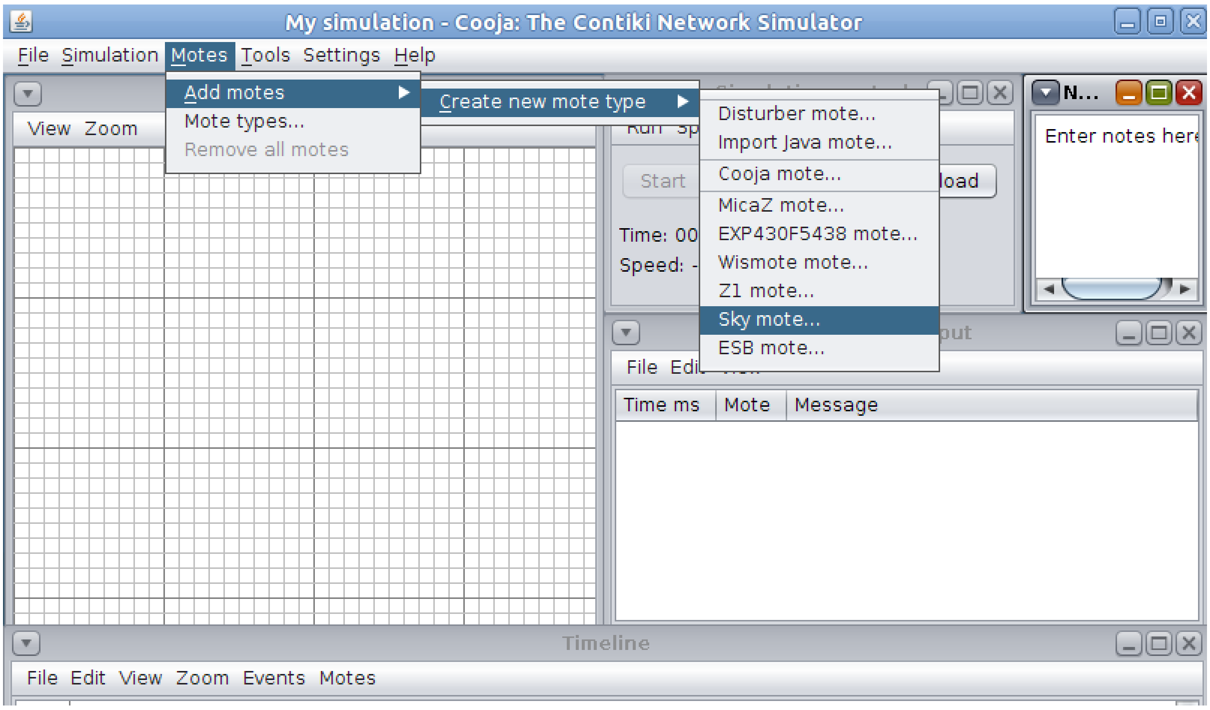


Figure 4.5: Creating Sky Mote as RFID Sensor in Cooja

The different types of motes are presented in the network environment of IoT including Sky mote, Z1 mote, EZB mote and many others. These are real time sensors which can be used in real time analysis as in Figure 4.5.

Tools and Technologies Used

- Ubuntu is Open Source Linux Operating System for PC, tablets and smartphones.
- 64 Bit or 32 Bit Architecture based Workstation
- Contiki is an Operating System for memory constrained systems with focus on low power wireless IoT devices.
- Cooja IoT is the Contiki Network Simulator which allows faster simulation of Contiki motes and large networks as compared to the hardware implementation for the same.
- C Programming Language.

The implementation of proposed implementation with secured hash based key algorithm result in higher degree of Security and Performance in terms of turnaround time, power consumption, energy in IoT Network. The implemented work is quite effective in terms of multiple parameters including minimum overheads and complexity with higher degree of performance for multiple and increasing number of nodes. SHA based security escalate performance and integrity with overall security in RPL.

The cumulative performance of DynamicHashAlgorithm-3 is quite effective and complexity aware as compared to DynamicHashAlgorithm-1 and DynamicHashAlgorithm-2. As DynamicHashAlgorithm-3 is having minimum number of rounds, the comparative resource optimization is achieved along with the higher degree of security and minimum power consumption.

Key Advantages

- Fast to compute
- Resistant to pre-image and second-preimage attacks
- Collision resistant
- Widespread use in security certificates
- Provides One Way Hash with less complexities
- Longer hash as compared to traditional message digest
- Stronger protection against attacks
- Resistance to Hashing Collisions
- Consistency Check

To enforce and integrate the higher degree of security, there is need to implement IPv6 for IoT scenarios with Secured Hash Based Cryptography SHA in the keys generation and authentication. The IPv6 based approach can be enabled with fully secured algorithms and non vulnerable towards the interceptions.

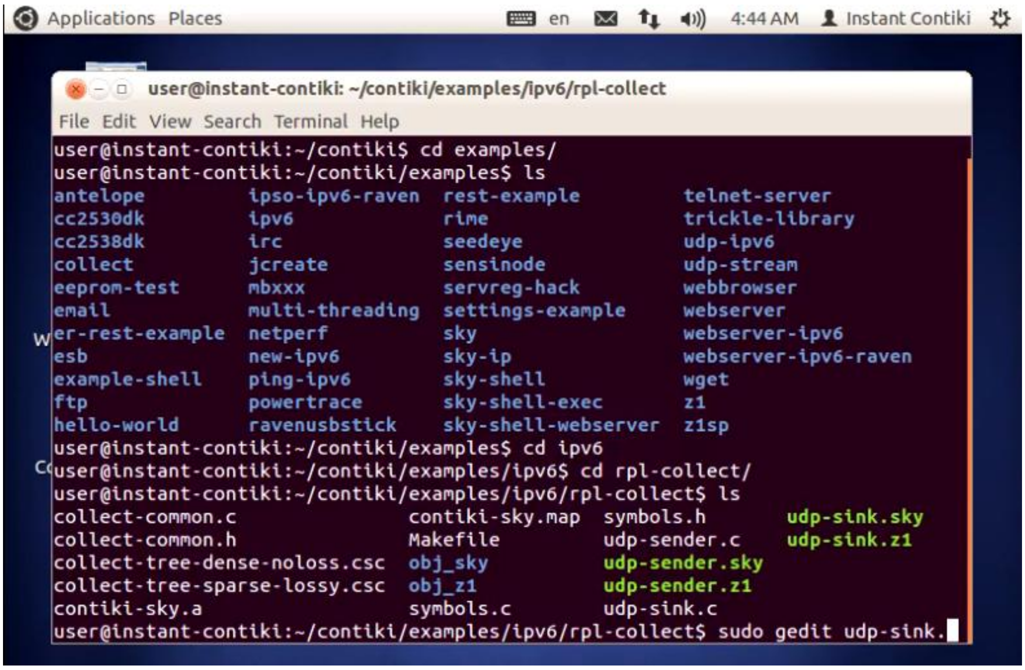
RPL is the IPv6 Based Protocol for IoT. It is primarily integrated for IPv6 over Low power Wireless Personal Area Networks (6LowPAN). It works with the dynamic creation of Destination-Oriented Directed Acyclic Graph (DODAG) having unidirectional as well as bi-directional communication. It is having multiple instances with the localized behavior for higher optimization. Using RPL based security, the multidimensional security can be enforced in IoT environment.

Contiki IoT Platform

Contiki is the key platform or operating system with free and open source distribution available on (http://www.contiki-os.org). Contiki is equipped with Cooja Simulator that is used for the simulation as well as programming for sensor devices having enormous options to program the IoT nodes for real life implementations. Contiki is having an excellent and powerful IoT simulator Cooja which enable the programmer to import and program enormous types of IoT motes and get the results from different algorithms.

Following integrations can be done in Cooja for the programming of wireless networks including smart devices based IoT

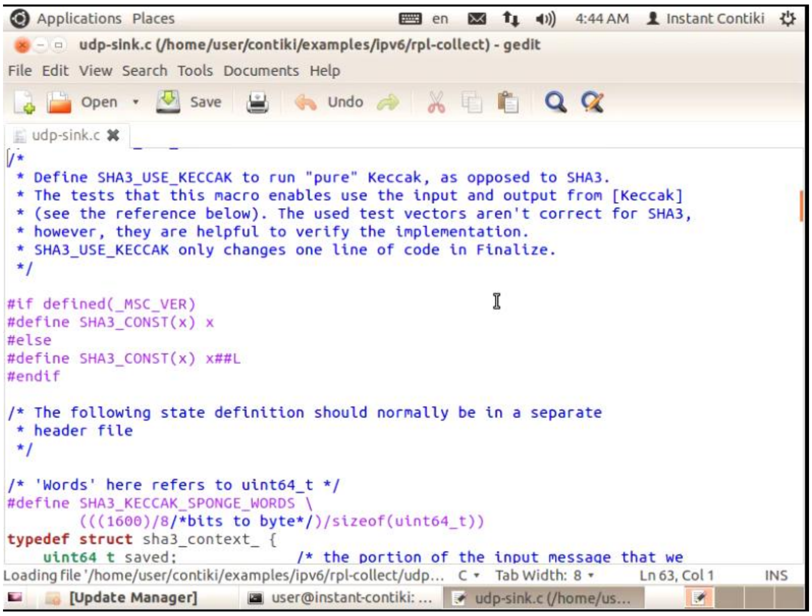
- Proto-Thread Programming for avoidance of Complexity and Overheads associated with Low Memory.
- Fully Flexible and Open Source
- Excellent TCP/IP Support
- Event Based Kernel Programming



```
user@instant-contiki:~/contiki$ cd examples/
user@instant-contiki:~/contiki/examples$ ls
antelope         ipso-ipv6-raven  rest-example         telnet-server
cc2530dk         ipv6             rime                 trickle-library
cc2538dk         irc              seedeye              udp-ipv6
collect          jcreate          sensinode            udp-stream
eeprom-test      mbxxx            servreg-hack         webbrowser
email            multi-threading  settings-example     webserver
er-rest-example  netperf          sky                  webserver-ipv6
esb              new-ipv6         sky-ip               webserver-ipv6-raven
example-shell    ping-ipv6        sky-shell            wget
ftp              powertrace       sky-shell-exec       z1
hello-world      ravenusbstick    sky-shell-webserver  z1sp
user@instant-contiki:~/contiki/examples$ cd ipv6
user@instant-contiki:~/contiki/examples/ipv6$ cd rpl-collect/
user@instant-contiki:~/contiki/examples/ipv6/rpl-collect$ ls
collect-common.c               contiki-sky.map  symbols.h       udp-sink.sky
collect-common.h               Makefile         udp-sender.c    udp-sink.z1
collect-tree-dense-noloss.csc  obj_sky          udp-sender.sky
collect-tree-sparse-lossy.csc  obj_z1           udp-sender.z1
contiki-sky.a                  symbols.c        udp-sink.c
user@instant-contiki:~/contiki/examples/ipv6/rpl-collect$ sudo gedit udp-sink.
```

Figure 4.6: Location of File System for Code Customization

Figure 4.6 shows the location of file system at the back end which is addressed and customized to embed the new code so that new protocols and security paradigms can be programmed.



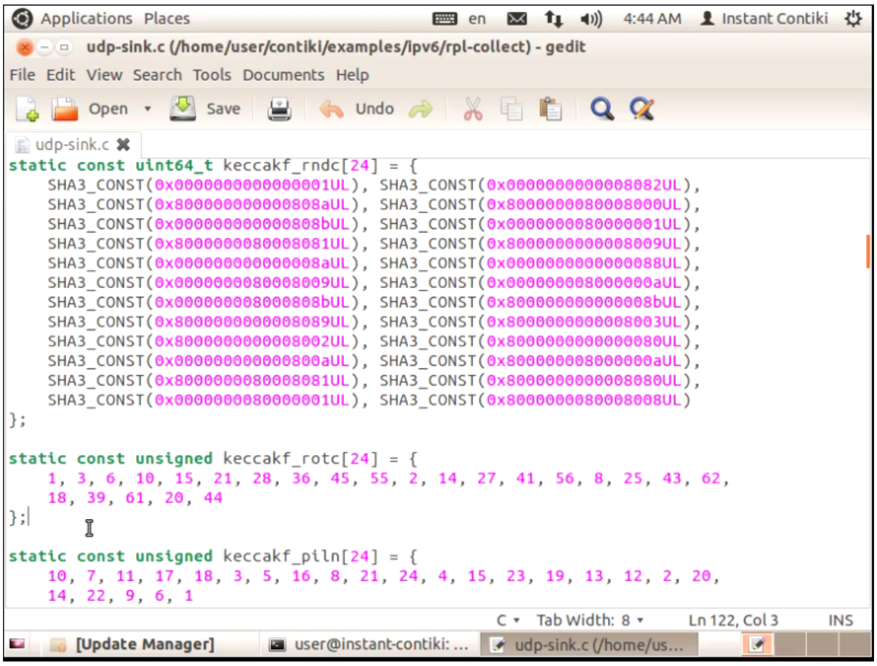
```c
/*
 * Define SHA3_USE_KECCAK to run "pure" Keccak, as opposed to SHA3.
 * The tests that this macro enables use the input and output from [Keccak]
 * (see the reference below). The used test vectors aren't correct for SHA3,
 * however, they are helpful to verify the implementation.
 * SHA3_USE_KECCAK only changes one line of code in Finalize.
 */

#if defined(_MSC_VER)
#define SHA3_CONST(x) x
#else
#define SHA3_CONST(x) x##L
#endif

/* The following state definition should normally be in a separate
 * header file
 */

/* 'Words' here refers to uint64_t */
#define SHA3_KECCAK_SPONGE_WORDS \
        (((1600)/8/*bits to byte*/)/sizeof(uint64_t))
typedef struct sha3_context_ {
    uint64_t saved;             /* the portion of the input message that we
```

Figure 4.7: Editing and Inserting C Code with SHA



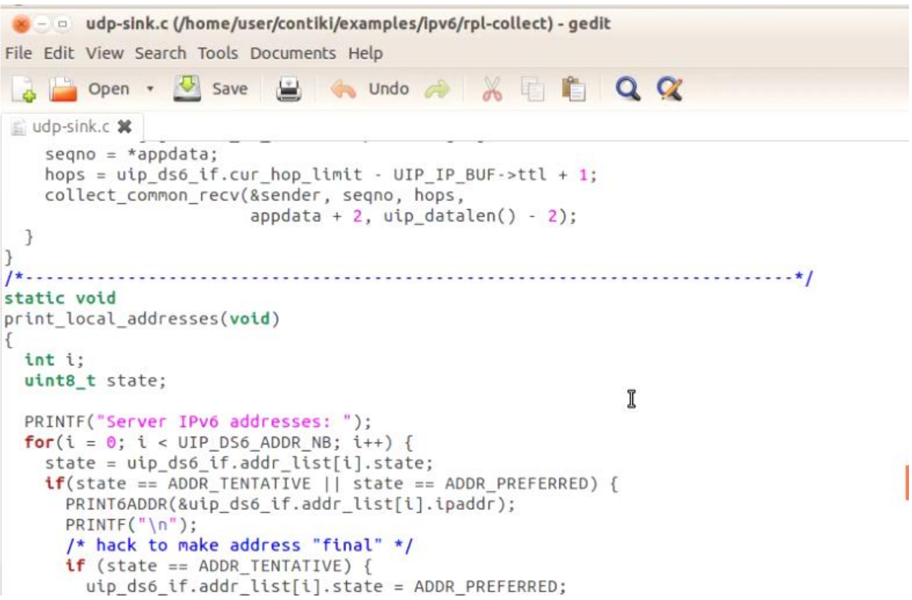
```c
static const uint64_t keccakf_rndc[24] = {
    SHA3_CONST(0x0000000000000001UL), SHA3_CONST(0x0000000000008082UL),
    SHA3_CONST(0x800000000000808aUL), SHA3_CONST(0x8000000080008000UL),
    SHA3_CONST(0x000000000000808bUL), SHA3_CONST(0x0000000080000001UL),
    SHA3_CONST(0x8000000080008081UL), SHA3_CONST(0x8000000000008009UL),
    SHA3_CONST(0x000000000000008aUL), SHA3_CONST(0x0000000000000088UL),
    SHA3_CONST(0x0000000080008009UL), SHA3_CONST(0x000000008000000aUL),
    SHA3_CONST(0x000000008000808bUL), SHA3_CONST(0x800000000000008bUL),
    SHA3_CONST(0x8000000000008089UL), SHA3_CONST(0x8000000000008003UL),
    SHA3_CONST(0x8000000000008002UL), SHA3_CONST(0x8000000000000080UL),
    SHA3_CONST(0x000000000000800aUL), SHA3_CONST(0x800000008000000aUL),
    SHA3_CONST(0x8000000080008081UL), SHA3_CONST(0x8000000000008080UL),
    SHA3_CONST(0x0000000080000001UL), SHA3_CONST(0x8000000080008008UL)
};

static const unsigned keccakf_rotc[24] = {
    1, 3, 6, 10, 15, 21, 28, 36, 45, 55, 2, 14, 27, 41, 56, 8, 25, 43, 62,
    18, 39, 61, 20, 44
};

static const unsigned keccakf_piln[24] = {
    10, 7, 11, 17, 18, 3, 5, 16, 8, 21, 24, 4, 15, 23, 19, 13, 12, 2, 20,
    14, 22, 9, 6, 1
```

Figure 4.8: Setting Up the Algorithmic Features of SHA in Code

Figure 4.8 presents the view of text editor which shows the C code embedded to program the IoT with SHA. The code of SHA is inserted and code is compiled for the new results.

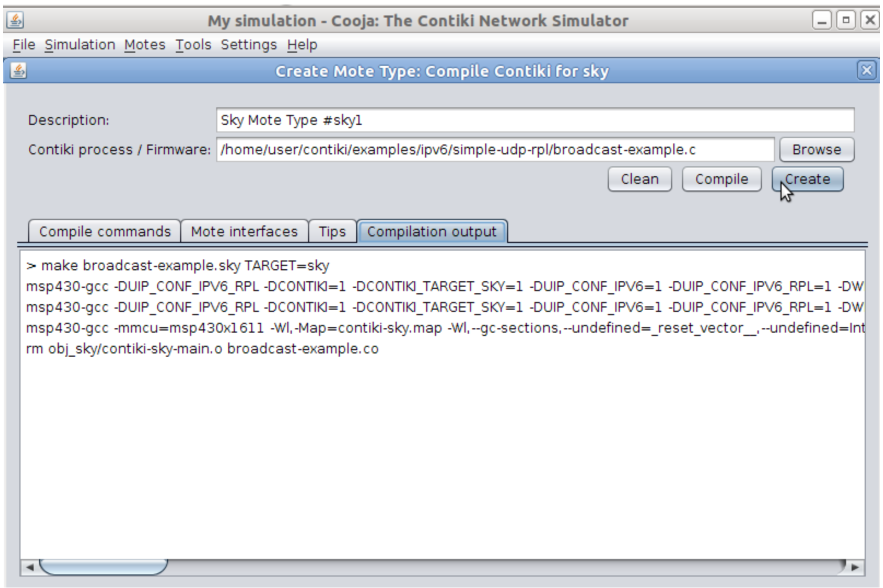


```
    seqno = *appdata;
    hops = uip_ds6_if.cur_hop_limit - UIP_IP_BUF->ttl + 1;
    collect_common_recv(&sender, seqno, hops,
                        appdata + 2, uip_datalen() - 2);
  }
}
/*---------------------------------------------------------------------------*/
static void
print_local_addresses(void)
{
  int i;
  uint8_t state;

  PRINTF("Server IPv6 addresses: ");
  for(i = 0; i < UIP_DS6_ADDR_NB; i++) {
    state = uip_ds6_if.addr_list[i].state;
    if(state == ADDR_TENTATIVE || state == ADDR_PREFERRED) {
      PRINT6ADDR(&uip_ds6_if.addr_list[i].ipaddr);
      PRINTF("\n");
      /* hack to make address "final" */
      if (state == ADDR_TENTATIVE) {
        uip_ds6_if.addr_list[i].state = ADDR_PREFERRED;
```

Figure 4.9: IP Logging Aspects in Cooja

Figure 4.9 gives the screenshot of text editor in which the code for logging the IP addresses is given. Here, the required code to sniff the IP addresses and related transmission are analyzed.



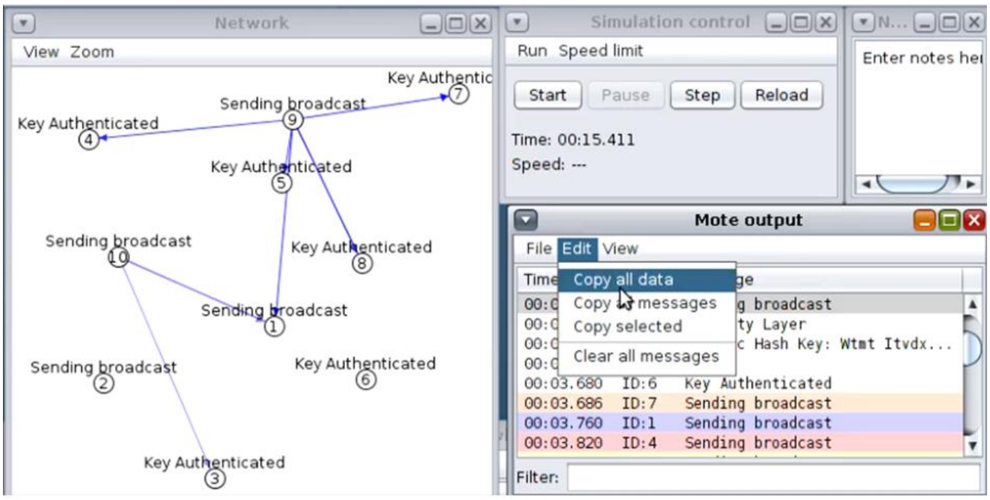
Description: Sky Mote Type #sky1

Contiki process / Firmware: /home/user/contiki/examples/ipv6/simple-udp-rpl/broadcast-example.c

```
> make broadcast-example.sky TARGET=sky
msp430-gcc -DUIP_CONF_IPV6_RPL -DCONTIKI=1 -DCONTIKI_TARGET_SKY=1 -DUIP_CONF_IPV6=1 -DUIP_CONF_IPV6_RPL=1 -DW
msp430-gcc -DUIP_CONF_IPV6_RPL -DCONTIKI=1 -DCONTIKI_TARGET_SKY=1 -DUIP_CONF_IPV6=1 -DUIP_CONF_IPV6_RPL=1 -DW
msp430-gcc -mmcu=msp430x1611 -Wl,-Map=contiki-sky.map -Wl,--gc-sections,--undefined=_reset_vector__,--undefined=Int
rm obj_sky/contiki-sky-main.o broadcast-example.co
```

Figure 4.10: Recompilation of C Code for Results and Outcome

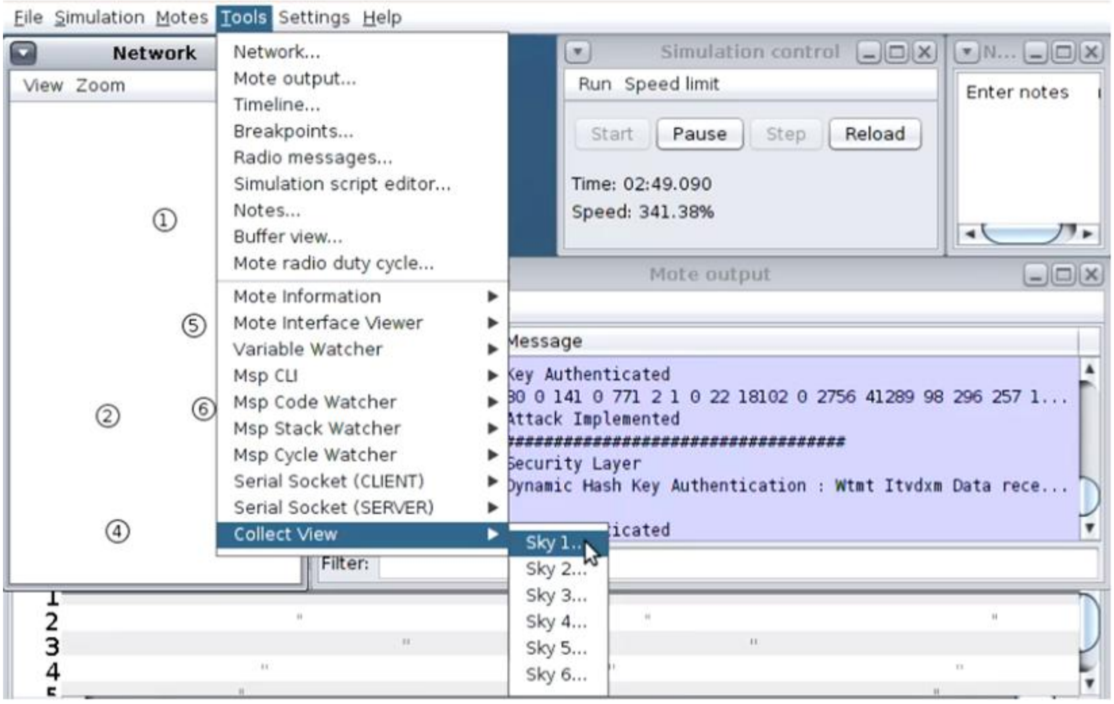
Figure 4.10 is the screenshot of Cooja which presents the recompilation module of Cooja which is required to have new binaries to be integrated in the Cooja Motes.



```
00:0          g broadcast
00:0          ty Layer
00:0          c Hash Key: Wtmt Itvdx...
00:0
00:03.680  ID:6   Key Authenticated
00:03.686  ID:7   Sending broadcast
00:03.760  ID:1   Sending broadcast
00:03.820  ID:4   Sending broadcast
```

Figure 4.11: Implementation of Security using Cooja

Figure 4.11 shows the implementation of secured hash algorithm in the simulated environment. The wireless sky motes can be viewed here with the key authentication process and the option to copy the network logs.

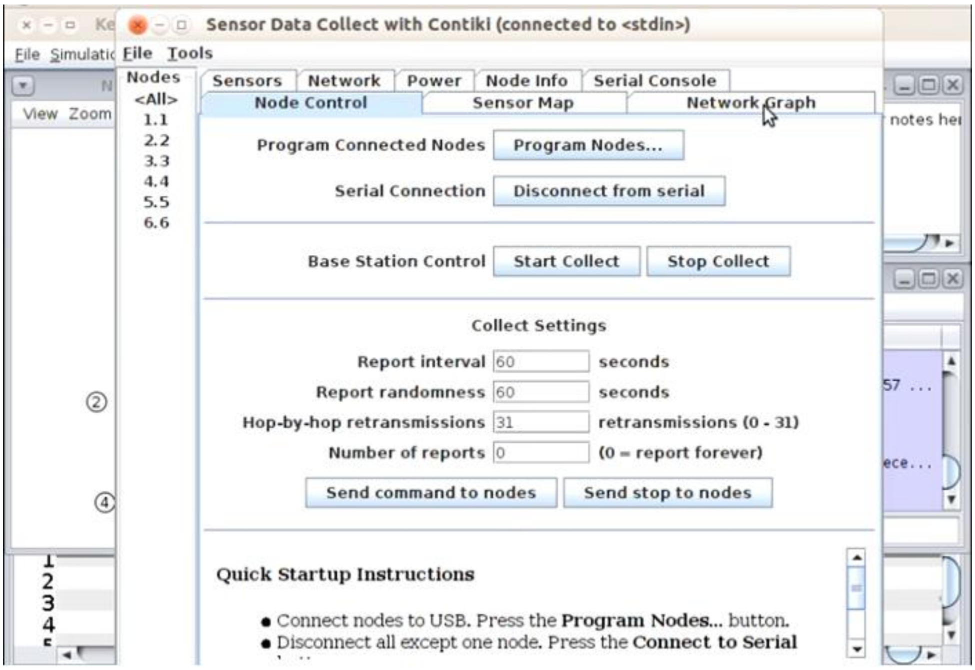


```
Key Authenticated
30 0 141 0 771 2 1 0 22 18102 0 2756 41289 98 296 257 1...
Attack Implemented
##################################
Security Layer
Dynamic Hash Key Authentication : Wtmt Itvdxm Data rece...

      icated
```

Figure 4.12: Evaluation of Mote Based Results in Collect View

Figure 4.12 depicts the analytics of results based on individual motes in the IoT environment. By this perspective, the sky motes with their packet transmission results can be analyzed.



Program Connected Nodes

Serial Connection

Base Station Control

Collect Settings

Report interval 60 seconds

Report randomness 60 seconds

Hop-by-hop retransmissions 31 retransmissions (0 - 31)

Number of reports 0 (0 = report forever)

Quick Startup Instructions

- Connect nodes to USB. Press the **Program Nodes...** button.
- Disconnect all except one node. Press the **Connect to Serial**

Figure 4.13: Fetching Live Sensor Data in Cooja

Figure 4.13 presents the fetching of live sensor data from simulation in Cooja. The results from mobility of wireless nodes can be evaluated here with the deep analytics.

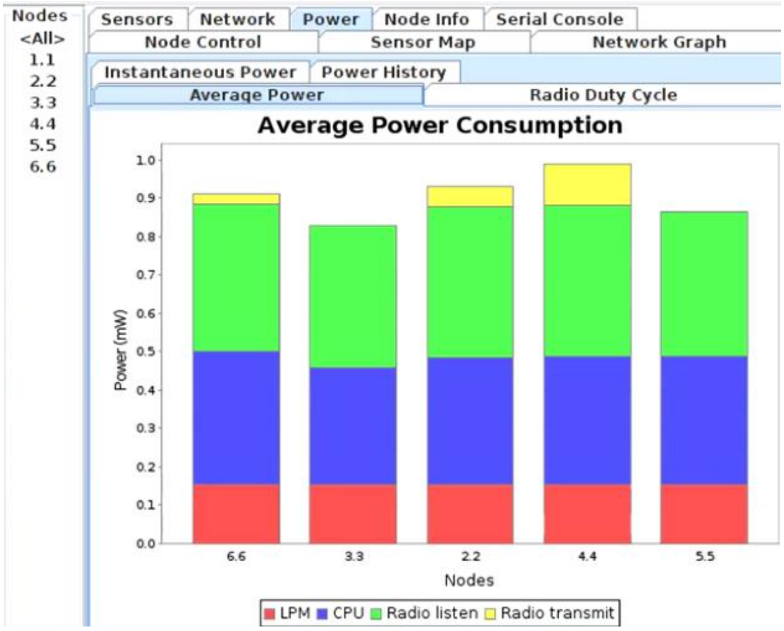


Figure 4.14: Evaluation of Parameters including Power Consumption

Figure 4.14 depicts the power consumption parameter during the simulation. This parameter is dynamic and keeps changing during the mobility of sensor nodes.

Execution Scenario: 1

Number of Motes: 10

Source Mote: 2

Sink Motes: 8

Table 4.1: Evaluation of Security Approaches

Algorithm	Power Consumption (mW)	Latency (Seconds)
Traditional (Without SHA)	0.82	0.48
DynamicHashAlgorithm-1	0.83	0.49
DynamicHashAlgorithm-2	0.95	0.41
DynamicHashAlgorithm-3	0.75	0.19

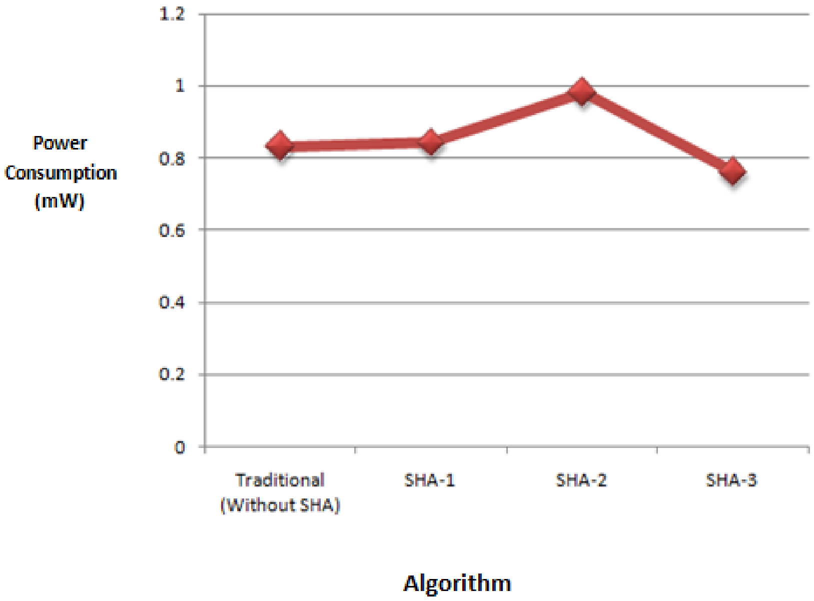


Figure 4.15: Evaluation of Power and Energy Factor

The Figure 4.15 presents the results that the approach-3 is having effectual result as compared to the traditional approaches of security in IoT.

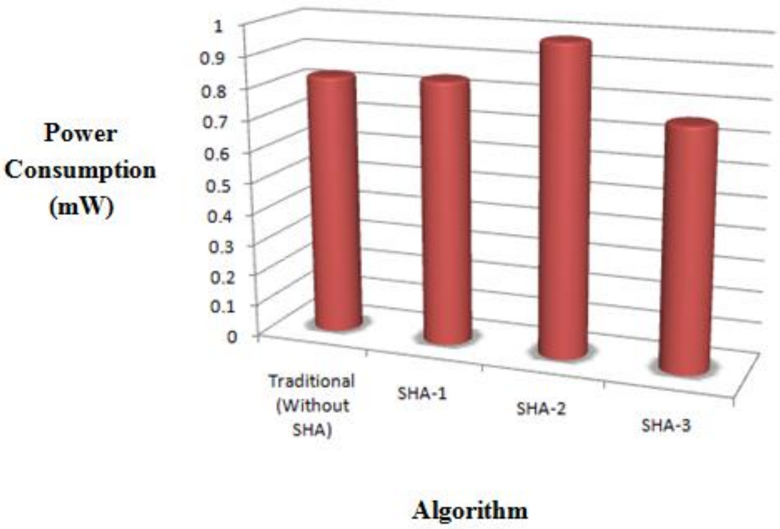


Figure 4.16: Evaluation of Power Factor in Cooja

The results show that the effective performance with DynamicHashAlgorithm-3 as related to similar techniques with context to less latency which is an important aspect of overhead and resource optimization factor.

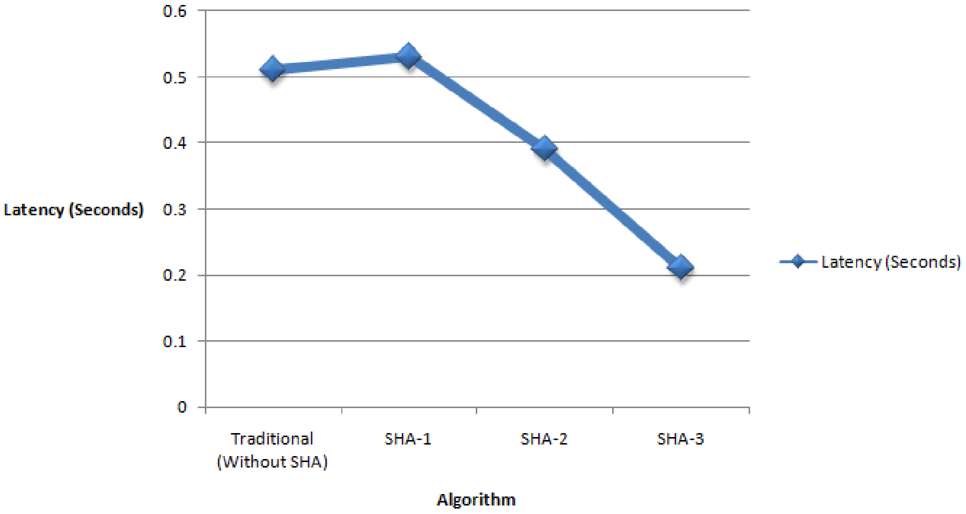


Figure 4.17: Evaluation of Latency

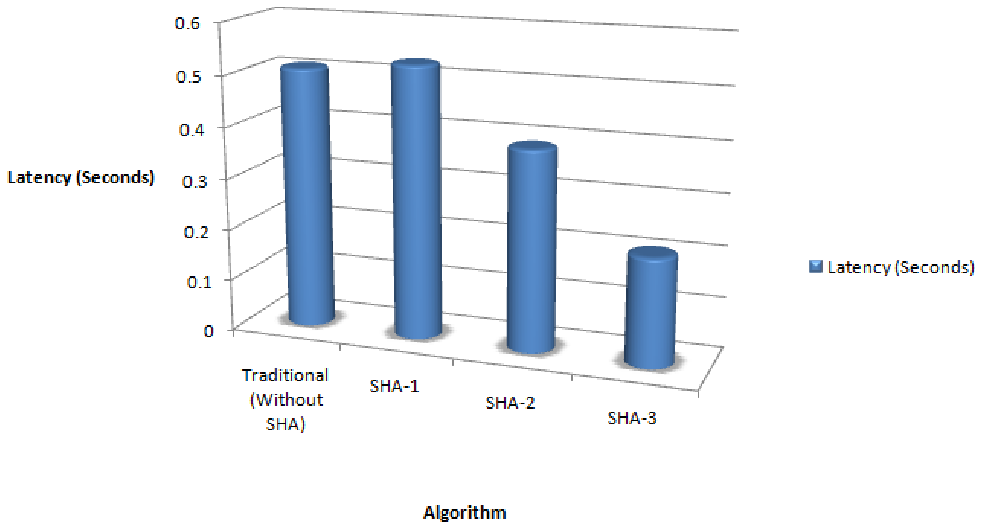


Figure 4.18: Evaluation of Latency

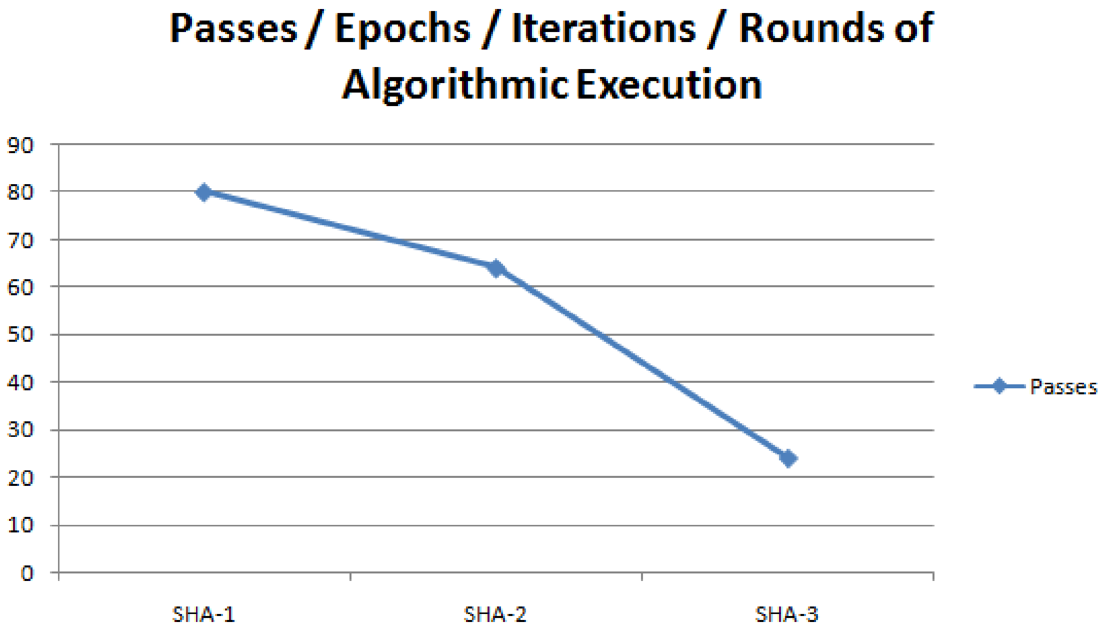


Figure 4.19: Line Graph Evaluation of Algorithmic Rounds

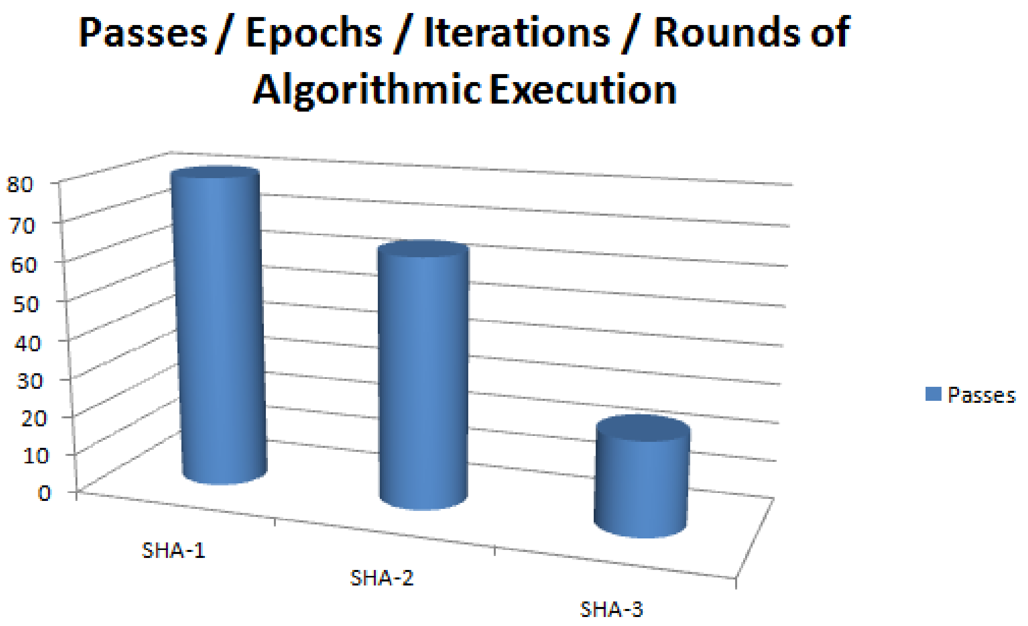


Figure 4.20: Bar Graph Evaluation of Rounds or Epochs

Figure 4.20 shows the effective performance of DynamicHashAlgorithm-3 as compared to other approaches in terms of less number of rounds in the generation of hash. The minimum number of rounds lead to less complexity and time.

Execution Scenario - 2

Number of Motes: 12

Source Mote: 2

Sink Motes: 10

Table 4.2: Evaluation on Multiple Perspectives

Algorithm	Power Consumption (mW)	Latency (Seconds)	Passes
Traditional (Without SHA)	0.99	0.32	-
DynamicHashAlgorithm-1	0.98	0.43	80
DynamicHashAlgorithm-2	0.88	0.33	64
DynamicHashAlgorithm-3	0.67	0.21	24

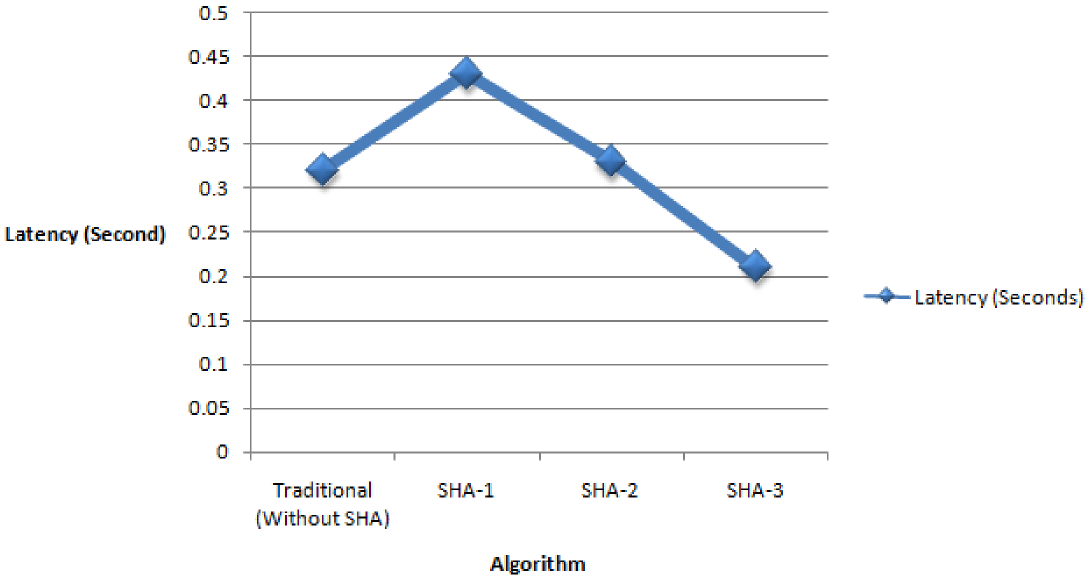


Figure 4.21: Evaluation of Latency

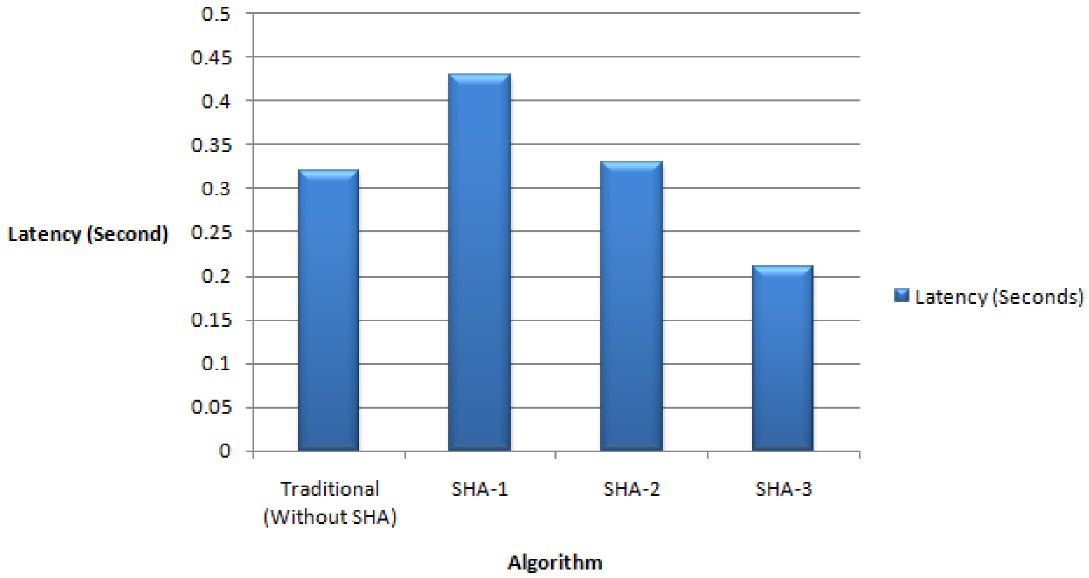


Figure 4.22: Evaluation of Latency

Figure 4.22 shows the effective performance of DynamicHashAlgorithm-3 having less latency or diversion factors which is an important aspect of overhead and resource optimization factor.

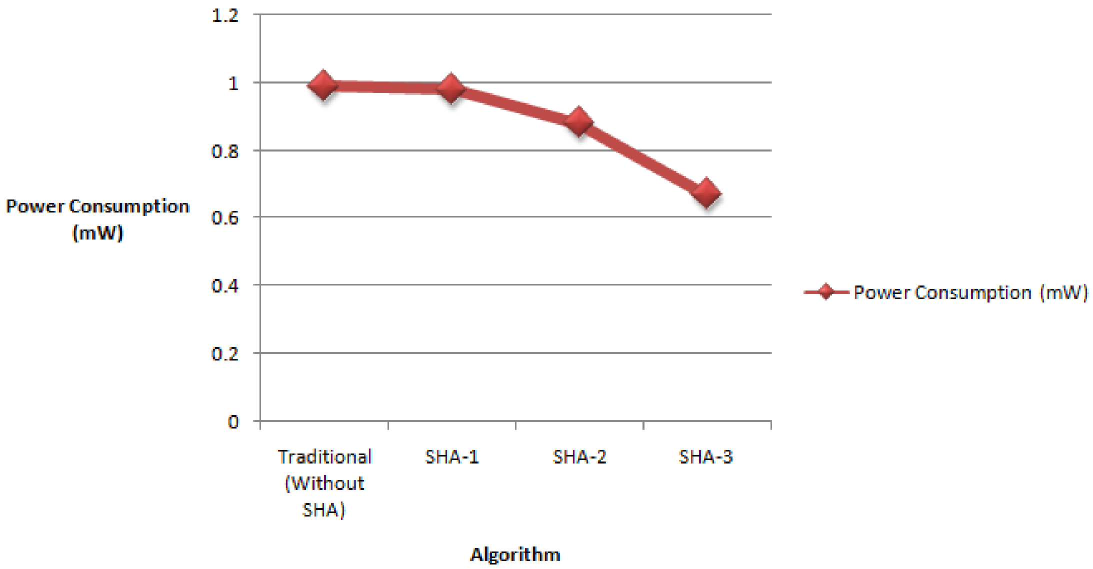


Figure 4.23: Analysis of Power Factor

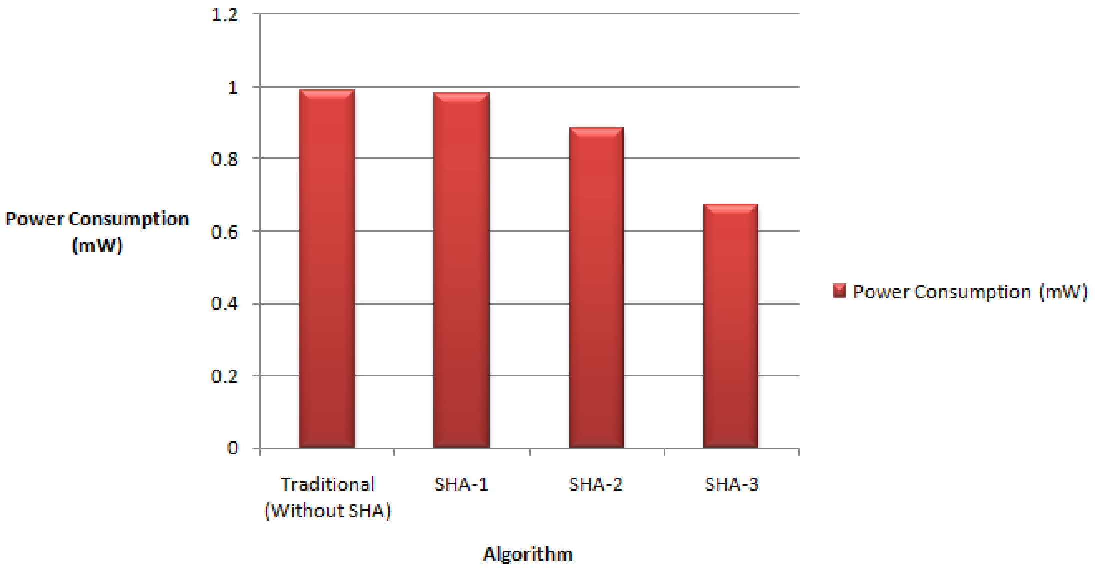


Figure 4.24: Analysis of Power Factor

Security and Integrity

Table 4.3: Evaluation of SHA with Multiple Parameters

Algorithm	**Power Consumption (mW) (P)**	**Latency (Seconds) (L)**	**Block Size (Bits) (B)**	**Output Size (Bits) (O)**	**Rounds (R)**
Traditional (Without SHA)	0.83	0.51	512	128	4
DynamicHashAlgorithm-1	0.84	0.53	512	160	80
DynamicHashAlgorithm-2	0.98	0.39	512	256	64
DynamicHashAlgorithm-3	0.76	0.21	1088	256	24

Table 4.4: Evaluation of Protocols

Algorithm	Security Bits (SB)	Integrity (I)	Security (S)
Traditional (Without SHA)	128	63.9166	19.1917
DynamicHashAlgorithm-1	63	59.5077	12.2508
DynamicHashAlgorithm-2	128	70.7585	19.8758
DynamicHashAlgorithm-3	128	132.608	26.0608

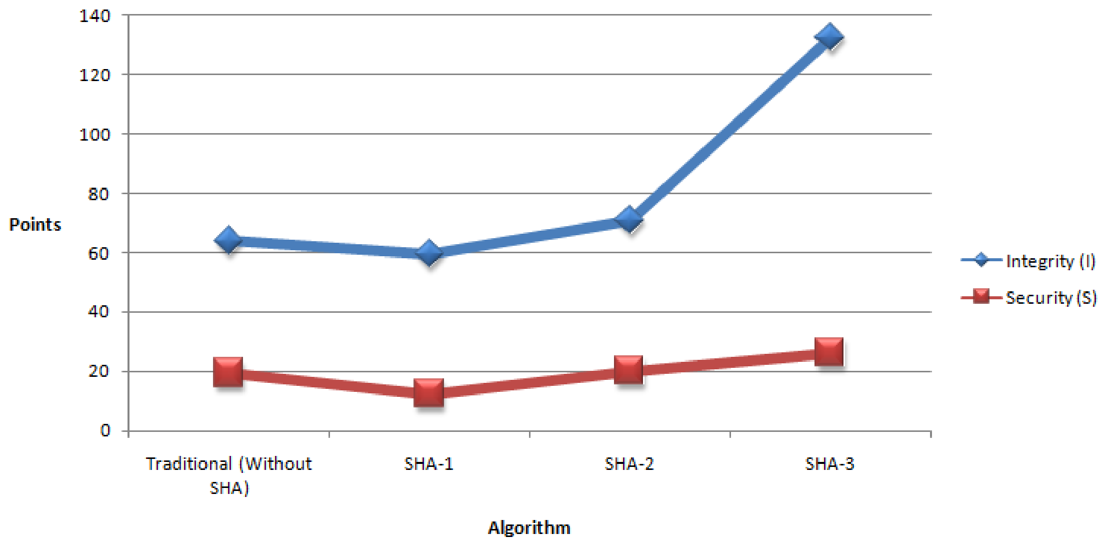


Figure 4.25: Evaluation of Integrity and Security

Figure 4.25 presents the integrity and security of the variants of SHA and it is found that DynamicHashAlgorithm-3 is comparatively better than other approaches.

Integrity

(Block Size+Output Size-Rounds+1/Power Consumption+1/Latency)/10

Security

Security Bits + Integrity

Following Network Metrics Related Plots can be generated in Cooja

- Mote Neighbors
- Beacon Interval: Beacon refers to the IoT transmitter which transmits and receives the signals or light waves which other smart objects can listen and react upon.
- Network Hops
  - Over Time
  - Per Node
- Router Metric (Over Time): The platform of Cooja is equipped with the features to customize the code with objective to use routing metrics to minimize the delay and achieving higher degree of performance
  - Instantaneous
  - Average
- ETX (Over Time): ETX refers to the default objective function which is minimized by default and in addition can be programmed to minimize in code.
- Next Hop (Over Time)
- Latency: Latency refers to the delay in the transmission of data
- Lost Packets (Over Time)
- Received Packets
  - Over Time
  - Per Node
  - Every 5 min

Chapter-5

CONCLUSIONS

This research concludes with the presentation and validation of the fact from simulation that DynamicHashAlgorithm-3 is quite effective in multiple parameters even in the scenarios of Internet of Things (IoT). The implementation done in Cooja depicts the valuable results and the approach is superior as compared to the other variants of security approaches for RPL.

5.1 Conclusions

Now days, huge work is going on in the segment of IoT that is the advance wireless network environment but lots of vulnerability factors are there which are required to the addressed and that is the key goal in this work. A number of hash approaches and security protocols are implemented in this work with the evaluation of multiple parameters. From the results, it is found that DynamicHashAlgorithm-3 is having maximum block size with the minimum rounds and effective in collisions. This variant of SHA is giving the results which can be adopted in the different protocols of IoT and varied implementations for key generations.

From the results and inherent aspects of SHA, the following key points related to the efficiency of DynamicHashAlgorithm-3 are presented

- Less Overheads and Complexity
- Higher Degree of Security
- Rich Block Size and Output Size
- Higher Degree of Integrity
- Less Latency

5.2 Scope of Future Work

The future work associated with this work can be the implementation of soft computing approaches with the devising of new algorithm which can impose the higher degree of security as compared to the existing work. The soft computing approaches can provide the higher degree of accuracy and security with other related parameters for overall performance and in generalized network environment.

The key implementation components associated with soft computing includes Machine Learning, Evolutionary computation (EC), Probability and many others.

REFERENCES

[1] Vasseur J, Agarwal N, Hui J, Shelby Z, Bertrand P, Chauvenet C. "RPL: The IP routing protocol designed for low power and lossy networks", *Internet Protocol for Smart Objects (IPSO) Alliance.* pp.201-218, 2011.

[2] J. Ko, A. Terzis, Dawson-Haggerty and Culler DE. "Connecting low-power and lossy networks to the internet", *IEEE Communications Magazine.* pp.11-19, 2011

[3] V. Gampala, S. Inuganti, and S. Muppidi, "Data Security in Cloud Computing with Elliptic Curve Cryptography," *Int. J. Soft Comput. Eng.*, vol. 2, no. 3, pp. 138–141, 2012.

[4] H Ning and Liu H. Cyber-physical-social based security architecture for future internet of things. *Advances in Internet of Things*. pp. 11-14, 2012.

[5] M. Agrawal and P. Mishra, "A comparative survey on symmetric key encryption techniques," *Intern. J. Comput. Sci. Eng.*, vol. 4, no. 5, pp. 877–882, 2012.

[6] J. Veijalainen, D. Kozlov, and Y. Ali, "Security and Privacy Threats in IoT Architectures," *Proc. 7th Int. Conf. Body Area Networks*, no. September, pp. 256-262, 2012.

[7] X. Qian, "Security-enhanced Search Engine Design in Internet of Things," *Journal of Universal Computer Science*, vol. 18, no. 9, pp. 1218–1235, 2012.

[8] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed internet of things," *Computer Networks*, vol. 57, no. 10, pp. 2266–2279, 2013.

[9] T. Kothmayr, C. Schmitt, W. Hu, M. Brnig, and G. Carle, "DTLS based security and two-way authentication for the Internet of Things," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2710–2723, 2013.

[10] J. Gubbi, R. Buyya, and S. Marusic, *Future Generation Computer Systems* "1207.0203," no. 1, pp. 1–19., 2013

[11] P. N. Mahalle, B. Anggorojati, N. R. P. Prasad, and R. Prasad, "Identity Authentication and Capability Based Access Control (IACAC) for the Internet of Things," *J. Cyber Secur. Mobil.*, vol. 1, no. 4, pp. 309–348, 2013.

[12] R. Hummen, J. H. Ziegeldorf, H. Shafagh, S. Raza, and K. Wehrle, "Towards viable certificate-based authentication for the internet of things," Proc. 2nd ACM Work. *Hot Top. Wirel. Netw. Secur. Priv. - HotWiSec* '13, pp. 37, 2013.

[13] Q. Jing, A. V. Vasilakos, J. Wan, J. Lu, and D. Qiu, "Security of the Internet of Things: perspectives and challenges," *Wireless Networks*, vol. 20, no. 8, pp. 2481–2501, 2014.

[14] L. Chen, Z. Yan, W. Zhang, and R. Kantola, “TruSMS: A trustworthy SMS spam control system based on trust management,” *Future Generation Computer Systems*, vol. 49, no. October, pp. 77–93, 2015.

[15] D. Lake, R. Milito, M. Morrow, and R. Vargheese, “Internet of Things: Architectural Framework for eHealth Security,” *Journal of ICT*, vol. 1, no. 3, pp. 301–328, 2014.

[16] N. Ye, Y. Zhu, R. C. Wang, R. Malekian, and Q. M. Lin, “An efficient authentication and access control scheme for perception layer of internet of things,” *Appl. Math. Inf. Sci.,* vol. 8, no. 4, pp. 1617–1624, 2014.

[17] M. Turkanović, B. Brumen, and M. Hölbl, “A novel user authentication and key agreement scheme for heterogeneous ad hoc wireless sensor networks, based on the Internet of Things notion,” *Ad Hoc Networks*, vol. 20, no. April, pp. 96–112, 2014.

[18] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, “Security, privacy and trust in Internet of Things: The road ahead,” *Computer Networks,* vol. 76, pp. 146–164, 2015.

[19] J. Granjal, E. Monteiro, and J. Sa Silva, “Security for the internet of things: A survey of existing protocols and open research issues,” *IEEE Commun. Surv. Tutorials*, vol. 17, no. 3, pp. 1294–1312, 2015.

[20] K. T. Nguyen, M. Laurent, and N. Oualha, “Survey on secure communication protocols for the Internet of Things,” *Ad Hoc Networks,* vol. 32, no. February, pp. 17–31, 2015.

[21] M. Vucinic, B. Tourancheau, F. Rousseau, A. Duda, L. Damon, and R. Guizzetti, “OSCAR: Object Security Architecture for the Internet of Things,” *Science Direct Ad Hoc Networks,* pp. 3-16, 2014.

[22] W. Trappe, R. Howard, and R. S. Moore, “Low-energy security: Limits and opportunities in the internet of things,” *IEEE Security and Privacy,* vol. 13, no. 1, pp. 14–21, 2015.

[23] F. Li, Z. Zheng, and C. Jin, “Secure and efficient data transmission in the Internet of Things,” *Telecommunication Systems,* vol. 62, no. 1, pp. 111–122, 2016.

[24] S. R. Moosavi, T. N. Gia, E. Nigussie, A. M. Rahmani, S. Virtanen, H. Tenhunen, and J. Isoaho, “End-to-end security scheme for mobility enabled healthcare Internet of Things,” *Futur. Gener. Comput. Syst*., vol. 64, no. May, pp. 108–124, 2016.

[25] K. A. Rafidha Rehiman and S. Veni, “A secure authentication infrastructure for IoT enabled smart mobile devices - an initial prototype,” *Indian J. Sci. Technol.,* vol. 9, no. 9, pp. 1-6 2016.

[26] D. Airehrour, J. Gutierrez, and S. K. Ray, “Secure routing for internet of things: A survey,” *Journal of Network and Computer Applications*., vol. 66, pp. 198–213, 2016.

[27] E. Bertino, “Data Security and Privacy in the IoT,” Proc. 19th Int. Conf. Extending Database Technol., *Open Proceedings,* pp. 1–3, 2016.

[28] M. Usman, I. Ahmed, M. I. Aslam, S. Khan, and U. A. Shah, “SIT : A Lightweight Encryption Algorithm for Secure Internet of Things,” *International Journal of Advanced Computer Science and Applications,* vol. 8, no. 1, pp. 1–10, 2017.

[29] M. B. Mollah, A. K. Azad, and A. Vasilakos, “Secure data sharing and searching at the edge of cloud-assisted internet of things,” *IEEE Cloud Computing,* vol. 4, no. 1, pp. 34–42, 2017.

[30] P. P. Jayaraman, X. Yang, A. Yavari, D. Georgakopoulos, and X. Yi, “Privacy preserving Internet of Things: From privacy techniques to a blueprint architecture and efficient implementation,” *Future Generation Computer Systems*, pp. 1–10, 2017.

[31] C. Schmitt, M. Noack, W. Hu, T. Kothmayr, and B. Stiller, “*Two-way Authentication for the Internet-of-Things*,” Securing Internet Things through Progress. *IGI Global Journals*, pp. 27-56, 2017.

[32] S. Prabhakar. *International Journal of Research in Computer Applications and Robotics,* vol. 3, no. 6, pp. 93-101, 2017.