A TRUST-BASED ROUTING FRAMEWORK FOR THE INTERNET OF THINGS

David Osemeojie Airehrour

Department of Information Technology and Software Engineering

School of Engineering Computer and Mathematical Sciences

Auckland University of Technology

A thesis submitted to Auckland University of Technology in fulfilment of the requirements for the degree of

Doctor of Philosophy

28.11.2017

Dedication

To Joy, my wonderful wife

Whose immeasurable sacrifice and care for me

and our children made the completion of this thesis a possibility

To our lovely children, Sonia and Matthew

Who are indeed a treasure and a bundle of joy from the Good Lord

And my mother, Victoria

A strong and gentle soul, whose love, affection

and ceaseless prayers made the success of this work a reality

"Light shines on the righteous and joy on the upright in heart"

- *Psalms 97:11.*

DECLARATION

I, David Osemeojie Airehrour, hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person (except where explicitly defined in the acknowledgements), nor material which to a substantial extent has been submitted for the award of any other degree or diploma of a university or other institution of higher learning.

[illegible]

Signed:____

Date: 27.11.2017

David Osemeojie Airehrour

Abstract

The intelligent connectivity of smart sensor devices commonly referred to as the Internet of Things (IoT) — is swiftly progressing productivity and communication levels and providing many functionalities throughout many organizations globally. The benefits heralded by the IoT's revolution is threatened, however, by the general lack of understanding of IoT's specific security demands thus, limiting its swift adoption and potential growth.

Two distinguishing features of IoT that makes it unique are the interconnection of billions of smart devices, and the resource-constrained nature of smart "things". However, most IoT devices and applications operate with either no security; limited or insufficient security to protect the data they transmit during operation due to their limiting properties like CPU, memory capacity, battery life and mobility. This issue is further compounded for IoT system designers, as a global security framework has not been well defined, and most IoT system designers lack the knowledge or expertise to design or define secure IoT systems since this is a new and emerging technology.

The routing of data in the IoT network is a specific security area of concern. With the massive scale of data exchange between these devices, and no adequate security to protect the communication of data, compromising data routes becomes easy for attackers. This thesis therefore, proposes a secure routing communication framework called *SecTrust*, which scales on IoT size and provides acceptable network performances while not depleting the resource availability of these smart "things". The proposed *SecTrust* is a secure Trust-based framework for IoT that provides a platform for trust computation, trust evaluation and trust formation among nodes. This framework provides a secure communication among the connected nodes. The framework further provides a system for the identification and isolation of malicious nodes operating within the network. In this system, every node computes the trustworthiness of its direct neighbours based on the computed direct trust value and the recommended trust value. While neighbours with high trust values are chosen for secure routing, nodes with lower trust values are categorised either as malicious, compromised, or perhaps selfish nodes that seek to preserve their resources like battery power. *SecTrust* consists of five main processes: trust calculation process, trust monitoring process, detection and isolation of malicious nodes, trust rating process and trust backup/recuperation process.

The development of this system provides insight into the use of modelling and analytical tools in building effective designs for P2P networks, through the design and development of trust computation, trust creation and trust propagation mechanisms, which are embedded, tested and validated using an IoT platform. The utility of *SecTrust* as a promising framework for IoT systems is demonstrated via its practical applications comprising: detection and isolation of malicious actors, management and sustenance of trust and recommendation systems in IoT networks and secure routing in IoT using a trust-based mechanism. Through the framework proposed, this thesis demonstrates that the *SecTrust* system showed promising performance results over other trust-based systems while simulations and testbed experiments offer proof-of-concept of the practicality of the proposed framework solution regardless of the operations of unreliable nodes, malicious nodes, selfish nodes, and even trust related attacks in the network.
Furthermore, this study is supported by proposing, implementing, and evaluating the trust-based system for large-scale IoT networks, and it constitutes three main parts. In the first part, the design and evaluation of *SecTrust* is reported. The effectiveness and transaction validity metrics are measured under purely naïve (attacking nodes working

independently) and purely collective (attacking nodes colluding together) scenarios while scaling the network size from small size to a large-sized network.

The second part covered the actualization of the *SecTrust* framework into an IoT routing protocol (*SecTrust-RPL*). The *SecTrust* framework was embedded into the RPL routing protocol and simulated using an IoT platform. The simulation was conducted to demonstrate the performance of the trust framework in mitigating known IoT attacks while providing acceptable levels of network performance. The performance of *SecTrust*-RPL protocol was compared with the RPL routing protocol.

The third part was a testbed experiment, which served as a proof-of-concept to validate the simulation results presented, and to show the practicality and efficacy of the *SecTrust* framework in mitigating IoT attacks in a real-world environment with minimal impact on network performance.

ACKNOWLEDGEMENTS

This research study has been successful due to the immense support of many people whom I respect, and will like to acknowledge for their various roles while pursuing this research study.

My profound gratitude goes to my supervisor, Associate Professor Jairo Gutierrez. Thank you very much for your unflinching support and persistent guidance throughout my PhD journey. You have provided me with more support than I deserved, and I cannot thank you enough. Through the journey, I have learnt many things from you, and I believe it will carry me further in my career.

My special thanks to my second supervisor, Associate Professor Nurul Sarkar for his valuable advice throughout my research studies. Your seasoned advice during our Network Security Research Group (NSRG) meetings is very much appreciated.

Special thanks to my third supervisor, Sayan Kumar Ray, for all your efforts in ensuring that my research work was in line with the standards required.

My thanks also go to Emmanuel Ndashimye, Aminu Bello Usman and Sotharith Tauch. You all have provided good feedbacks during my research journey, and we have had many valuable scientific discussions together, which have sharpened our minds and improved the quality of our research works.

To all my colleagues at NSRG, it has been fun being in the company of you all, getting your kind support and receiving valuable feedbacks during my presentations. I say a big thank you to all of you.

I want to use this opportunity to thank the management of the Auckland University of Technology for the Vice Chancellor's Doctoral Scholarship. This has in no little way assisted me to focus on my studies while performing quality research work. My thanks also to the management and staff of the School of Engineering, Computer and Mathematical Sciences.

Finally, to my wonderful families both immediate and extended that have stood by me all through the journey. I am grateful for all your affections, support and prayers. This thesis is dedicated to you all, as without your support, my coming to New Zealand to undertake this study would not have been possible. I love you all.

CONTENTS

1 INTRODUCTION ... 1

1.1 MOTIVATION ... 3

1.2 RESEARCH QUESTIONS ... 4

1.3 SIGNIFICANCE OF THE STUDY ... 5

1.4 SCOPE OF THE STUDY ... 6

1.5 CONTRIBUTIONS ... 6

1.6 PUBLICATIONS ... 8

1.7 THESIS OUTLINE ... 10

2 THE INTERNET OF THINGS ... 13

2.1 IOT ARCHITECTURE ... 16

2.2 ENABLING TECHNOLOGIES OF IOT ... 24

2.2.1 Internet Protocol Version 6 (IPv6) ... 24

2.2.2 Radio Frequency Identification (RFID) ... 24

2.2.3 Wireless Sensor Network (WSN) ... 24

2.2.4 Wi-Fi ... 25

2.2.5 Bluetooth Low Energy (Bluetooth LE, BLE) ... 25

2.2.6 ZigBee ... 25

2.3 GLOBAL APPLICATION OF IOT ... 26

2.3.1 Smart Cities ... 26

2.3.2 Smart Environment ... 27

2.3.3 Smart Water ... 28

2.3.4 Smart Metering ... 28

2.3.5 Security and Emergencies ... 29

2.3.6 Retail ... 29

2.3.7 Logistics ... 30

2.3.8 eHealth ... 31

2.3.9 Smart Agriculture ... 32

2.4 SUMMARY ... 33

3 SECURE ROUTING IN INTERNET OF THINGS ... 35

3.1 IOT ROUTING PROTOCOLS ... 36

3.1.1 IPv6 over Low Power Personal Area Networks (6LoWPAN) ... 36

3.1.2 Routing Protocol for Low Power and Lossy Networks (RPL) ... 37

3.1.3 IPv6 over the Time Slotted Channel Hopping mode of IEEE 802.15.4e (6TiSCH) .. 40

3.2 IOT APPLICATION PROTOCOLS ... 40

3.2.1 EXtensible Messaging and Presence Protocol (XMPP) ... 42

3.2.2 Constrained Application Protocol (CoAP) ... 42

3.2.3 Message Queue Telemetry Transport (MQTT) *42*
3.3 SECURITY IN IOT: WHERE THE NEED LIES 43
3.3.1 Confidentiality *43*
3.3.2 Integrity *44*
3.3.3 Availability *44*
3.3.4 Authenticity *44*
3.3.5 Non-repudiation *44*
3.4 ROUTING VULNERABILITIES IN IOT 48
3.5 THREATS ASSOCIATED WITH ROUTING PROTOCOLS 48
3.6 ATTACK TYPES IN IOT NETWORKS 48
3.6.1 Active Attacks *48*
3.6.2 Passive Attacks *49*
3.6.3 External Attacks *49*
3.6.4 Internal Attacks *49*
3.7 ROUTING ATTACKS IN IOT 49
3.7.1 Blackhole Attacks *50*
3.7.2 Wormhole Attacks *50*
3.7.3 Sybil Attacks *51*
3.7.4 Greyhole Attacks *51*
3.7.5 Sinkhole Attacks *51*
3.7.6 Selective Forwarding Attacks *51*
3.7.7 Hello Flood Attacks *52*
3.7.8 Neighbour Attacks *52*
3.7.9 Local Repair Attacks *53*
3.7.10 DODAG Information Solicitation (DIS) Attacks *53*
3.7.11 Version Number Attacks *53*
3.7.12 Rank Attacks *54*
3.7.13 Modification Attacks *55*
3.7.14 Byzantine Attacks *55*
3.7.15 Route Cache Poisoning Attacks *56*
3.7.16 Fabrication Attacks *56*
3.7.17 Location Spoofing Attacks *57*
3.7.18 Rushing Attacks *57*
3.8 SECURE ROUTING IN IOT: LIMITING FACTORS 58
3.8.1 Energy Level of IoT Nodes *59*
3.8.2 Scalable Nature of IoT Networks *59*
3.8.3 Heterogeneity of IoT Networks *59*
3.8.4 Unavailability of Intermediate Nodes *59*

3.8.5 Mobility of IoT Networks ... *60*
3.8.6 Open Wireless Medium ... *60*
3.8.7 Memory and CPU Capacity ... *60*
3.9 SECURE ROUTING PROTOCOLS IN IOT ... 60
3.9.1 Secure Multi-Hop Routing for IoT Communication ... *60*
3.9.2 TSRF: A Trust-Aware Secure Routing Framework in Wireless Sensor Networks ... *61*
3.9.3 Two-Way Acknowledgment-based Trust (2-ACKT) ... *61*
3.9.4 The Group-based Trust Management Scheme (GTMS) ... *61*
3.9.5 Collaborative Lightweight Trust-based (CLT) Routing Protocol ... *62*
3.10 SECURE ROUTING IN IOT: ISSUES AND CHALLENGES ... 64
3.10.1 6LowPAN: Research Challenges ... *65*
3.10.2 RPL: Research Challenges ... *67*
3.11 SOFTWARE DEFINED PERIMETER ... 71
3.12 TRUST AND TRUST MODELS ... 73
3.12.1 Trust Models ... *74*
3.12.2 Quantifying Trust in IoT ... *76*
3.13 SUMMARY ... 81
4 SECTRUST: DESIGN AND METHODOLOGY ... 83
4.1 METHODOLOGY FOR INVESTIGATION ... 84
4.2 SECTRUST: FRAMEWORK DESCRIPTION ... 88
4.2.1 Quantifying Trust in SecTrust ... *90*
4.2.2 SecTrust Process ... *90*
4.3 ENERGY LEVEL OF NODES ... 97
4.4 THE RESILIENCY OF SECTRUST ... 97
4.5 EVALUATION OF SECTRUST ... 98
4.6 TOPOLOGY AND DEPLOYMENT SCENARIO FOR EVALUATION ... 100
4.7 SUMMARY ... 103
5 SECTRUST VS GLOBAL TRUST AND RECOMMENDATION SYSTEMS: DESIGN AND PERFORMANCE COMPARISON ... 104
5.1 GLOBAL TRUST AND RECOMMENDATION SYSTEMS ... 105
5.1.1 EigenTrust ... *106*
5.1.2 EigenTrust Incremental ... *107*
5.1.3 Trust Network Analysis with Subjective Logic (TNA-SL) ... *108*
5.1.4 No Trust and Recommendation System (None) ... *110*
5.2 FRAMEWORK DESCRIPTION AND JUSTIFICATION FOR A BASELINE EVALUATION ... 110
5.3 QTM P2P TRUST SIMULATOR ... 112
5.3.1 Trace Generation ... *113*
5.3.2 QTM Trace Simulation ... *118*

5.4 EMBEDDING SECTRUST IN THE QTM P2P TRACE SIMULATOR 120
5.5 METRICS FOR THE EVALUATION OF TRUST SYSTEMS 123
5.6 SIMULATION SETUP AND MEASUREMENTS 124
5.6.1 The Case of Trust Network Analysis with Subjective Logic(TNA-SL) 125
5.6.2 Purely Naïve and Purely Collective Scenarios 125
5.6.3 Transaction Validity under Purely Naïve and Purely Collective Scenarios 132
5.6.4 Sybil Naïve 136
5.6.5 Sybil Collective... 140
5.7 EIGENTRUST, EIGENTRUST INCREMENTAL, NO TRUST AND SECTRUST: A COMPARATIVE FRAMEWORK ANALYSIS 144
5.8 SUMMARY 144
6 SECTRUST: SIMULATION STUDY 146
6.1 SIMULATION AND ITS VALIDITY 148
6.2 RPL: IPV6 ROUTING PROTOCOL FOR LOW-POWER AND LOSSY NETWORKS (LLN) 149
6.2.1 Design Principles and Topology Formation in RPL.. 150
6.2.2 Routing in RPL 151
6.2.3 Routing Metrics in RPL 156
6.2.4 Defining the RPL Objective Function 156
6.2.5 The RPL Trickle Algorithm 157
6.3 CONTIKI 158
6.4 COOJA 159
6.5 RADIO PROPAGATION MODELS IN COOJA 160
6.6 CONTIKIRPL 162
6.7 EMBEDDING SECTRUST IN CONTIKIRPL 165
6.7.1 Rank and Trust Computation in SecTrust-RPL.. 165
6.8 NETWORK SIMULATION SETUP 171
6.9 SIMULATION RESULTS 174
6.9.1 SecTrust-RPL vs MRHOF- RPL under Blackhole Attacks.................................. 176
6.9.2 SecTrust-RPL vs MRHOF-RPL under Rank Attacks 180
6.9.3 SecTrust-RPL vs RPL (MRHOF) Under Selective Forwarding Attacks................ 184
6.9.4 SecTrust-RPL vs RPL (MRHOF) Under Sybil Attacks 188
6.10 SUMMARY 192
7 SECTRUST: TESTBED EXPERIMENTS 194
7.1 ADVANTAGES OF TESTBED EXPERIMENTATION 195
7.2 CASE STUDY OF A SMART BUILDING 196
7.3 IOT HARDWARE AND SOFTWARE PLATFORMS 197
7.4 CONTIKI AND TESTBED DEPLOYMENT 199
7.5 TESTBED SETUP 200

7.5.1 Embedding ContikiRPL (MRHOF) and SecTrust-RPL in AS-XM1000 .... *200*

7.5.2 Link failure Model *207*

7.5.3 Data Logging and Capture *210*

7.6 EXPERIMENTAL RESULTS 211

7.6.1 SecTrust-RPL vs MRHOF-RPL under Blackhole Attacks *212*

7.6.2 SecTrust-RPL vs MRHOF-RPL under Rank Attacks *217*

7.6.3 SecTrust-RPL vs MRHOF-RPL under Selective Forwarding Attacks *222*

7.6.4 SecTrust-RPL vs MRHOF-RPL under Sybil Attacks *227*

7.7 SUMMARY 232

8 STATISTICAL EVALUATION AND THE IMPLICATIONS OF A TRUST-BASED DESIGN FRAMEWORK FOR IOT **233**

8.1 STATISTICAL EVALUATION OF THE VALIDITY OF THE SIMULATION STUDY AND THE TESTBED EXPERIMENTS 234

8.1.1 Z-Confidence Intervals for Means and Margin of Error *235*

8.1.2 Confidence Intervals and Margin of Error under Blackhole Attacks *237*

8.1.3 Confidence Intervals and Margin of Error under Rank Attacks *240*

8.1.4 Confidence Intervals and Margin of Error under Selective Forwarding Attacks ... *243*

8.1.5 Confidence Intervals and Margin of Error under Sybil Attacks *246*

8.2 FRAMING THE THESIS' CONTRIBUTION: SECURITY IN EMBEDDED IOT DEVICES 249

8.3 TRUST-BASED FRAMEWORK FOR IOT: AN EVOLUTIONARY PATH FOR SECURE IOT COMMUNICATION 253

8.4 IMPLICATIONS AND CHALLENGES FOR TRUST-BASED FRAMEWORKS IN SECURE IOT COMMUNICATIONS 256

8.5 CLOUD SECURITY ALLIANCE'S SOFTWARE DEFINED PERIMETER VS SECTRUST 258

8.6 SUMMARY 259

9 CONCLUSIONS AND FUTURE WORK **261**

9.1 FUTURE WORK 263

9.2 CONCLUSIONS 265

10 REFERENCES **267**

11 APPENDICES **279**

LIST OF TABLES

TABLE 3.1: IOT APPLICATION PROTOCOL FEATURES 41
TABLE 3.2: A SUMMARY OF SECURE ROUTING PROTOCOLS FOR IOT 63
TABLE 3.3: SUMMARY OF RPL ATTACKS AND COUNTERMEASURES 71
TABLE 3.4: A SUMMARY OF TRUST MODELS APPLIED IN SECURE NETWORK ROUTING IN SENSOR NETWORKS 75
TABLE 3.5: A SUMMARY OF TRUST SYSTEM MODELS WITH THE METRICS USED AND ATTACKS ADDRESSED 80
TABLE 4.1: DESIGN SCIENCE METHOD APPLIED TO SECURE IOT ROUTING 87
TABLE 4.2: A FIVE-TUPLE TRUST LEVEL BAND 96
TABLE 5.1: OPINION COMPUTATION FROM PAST AND DIRECT INTERACTION BETWEEN NODES 108
TABLE 5.2: DISCOUNT AND CONSENSUS TRUST OPERATORS IN TNA-SL [152, 153] 109
TABLE 5.3: USER INITIALIZATION PARAMETERS 114
TABLE 5.4: A USER-REQUESTER ADOPTION MODEL OF SOURCE SELECTION PROVIDER 119
TABLE 5.5: TRUST SIMULATION PARAMETER SETTINGS 126
TABLE 5.6: COMPLETION TIME FOR 50 NODES IN SECONDS 126
TABLE 5.7: COMPLETION TIME FOR 100 NODES IN SECONDS 128
TABLE 5.8: COMPLETION TIME FOR 500 NODES IN SECONDS 129
TABLE 5.9: COMPLETION TIME FOR 1000 NODES IN SECONDS 131
TABLE 5.10: TRUST SIMULATION PARAMETER SETTINGS 132
TABLE 5.11: TRUST SYSTEMS COMPLETION TIMES UNDER PURELY NAÏVE SCENARIO (SECONDS) 134
TABLE 5.12: TRUST SYSTEMS COMPLETION TIMES UNDER PURELY COLLECTIVE SCENARIO (SECONDS) 135
TABLE 5.13: TRUST SIMULATION PARAMETER SETTINGS 137
TABLE 5.14: TRUST SYSTEMS COMPLETION TIMES UNDER THE SYBIL NAÏVE SCENARIO (SECONDS) 137
TABLE 5.15: SYBIL EFFECT METRIC RATIO IN A SYBIL NAÏVE SCENARIO 140
TABLE 5.16: TRUST SYSTEMS COMPLETION TIMES UNDER SYBIL COLLECTIVE SCENARIO (SECONDS) 142
TABLE 5.17: SYBIL EFFECT METRIC RATIO IN A COLLECTIVE SCENARIO 144
TABLE 5.18: SECTRUST EFFECTIVENESS COMPARISON 144
TABLE 6.1: NETWORK SIMULATION SETTINGS AND PARAMETERS 174
TABLE 7.1: SUMMARY OF THE FEATURES OF THE AS-XM1000 MOTE [209] 198
TABLE 7.2: NETWORK TESTBED SETTINGS AND PARAMETERS 202

TABLE 8.1: STATISTICAL SUMMARY OF THE FREQUENCY OF NODE RANK CHANGES IN MRHOF-RPL AND SECTRUST-RPL DURING BLACKHOLE ATTACKS.. 238

TABLE 8.2: STATISTICAL SUMMARY OF ATTACKS DETECTED BY SECTRUST-RPL DURING BLACKHOLE ATTACKS.. 238

TABLE 8.3: STATISTICAL SUMMARY OF THROUGHPUT RATE OF MRHOF-RPL AND SECTRUST-RPL DURING BLACKHOLE ATTACKS 239

TABLE 8.4: STATISTICAL SUMMARY OF PACKET LOSS RATES OF MRHOF-RPL AND SECTRUST-RPL DURING BLACKHOLE ATTACKS 240

TABLE 8.5: STATISTICAL SUMMARY OF THE FREQUENCY OF NODE RANK CHANGES IN MRHOF-RPL AND SECTRUST-RPL DURING RANK ATTACKS 241

TABLE 8.6: STATISTICAL SUMMARY OF ATTACKS DETECTED BY SECTRUST-RPL DURING RANK ATTACKS.. 241

TABLE 8.7: STATISTICAL SUMMARY OF THROUGHPUT RATE OF MRHOF-RPL AND SECTRUST-RPL DURING RANK ATTACKS 242

TABLE 8.8: STATISTICAL SUMMARY OF PACKET LOSS RATES OF MRHOF-RPL AND SECTRUST-RPL DURING RANK ATTACKS 243

TABLE 8.9: STATISTICAL SUMMARY OF THE FREQUENCY OF NODE RANK CHANGES IN MRHOF-RPL AND SECTRUST-RPL DURING SELECTIVE FORWARDING ATTACKS......................... 244

TABLE 8.10: STATISTICAL SUMMARY OF ATTACKS DETECTED BY SECTRUST-RPL DURING SELECTIVE FORWARDING ATTACKS... 244

TABLE 8.11: STATISTICAL SUMMARY OF THROUGHPUT RATE OF MRHOF-RPL AND SECTRUST-RPL DURING SELECTIVE FORWARDING ATTACKS 245

TABLE 8.12: STATISTICAL SUMMARY OF PACKET LOSS RATES OF MRHOF-RPL AND SECTRUST-RPL DURING SELECTIVE FORWARDING ATTACKS 246

TABLE 8.13: STATISTICAL SUMMARY OF THE FREQUENCY OF NODE RANK CHANGES IN MRHOF-RPL AND SECTRUST-RPL DURING SYBIL ATTACKS.. 247

TABLE 8.14: STATISTICAL SUMMARY OF ATTACKS DETECTED BY SECTRUST-RPL DURING SYBIL ATTACKS.. 247

TABLE 8.15: STATISTICAL SUMMARY OF THROUGHPUT RATE OF MRHOF-RPL AND SECTRUST-RPL DURING SYBIL ATTACKS 248

TABLE 8.16: STATISTICAL SUMMARY OF PACKET LOSS RATES OF MRHOF-RPL AND SECTRUST-RPL DURING SYBIL ATTACKS 249

TABLE 11.1: JAVA IMPLEMENTATION METHOD SUMMARY FOR SECTRUST............................. 282

LIST OF FIGURES

FIGURE 1.1: INTERCONNECTIVITY OF IOT NODES COMPRISING OF EDGE ROUTERS (GATEWAY TO THE CLOUD), ROUTING NODES, WHICH ALSO SERVE AS CONTROL NODES, AND MOBILE SENSORY/ ACTUATOR NODES [9] 2

FIGURE 1.2: A FORECAST OF 50 BILLION CONNECTED ONLINE DEVICES [16] 3

FIGURE 1.3: THESIS STRUCTURE 12

FIGURE 2.1: ARCHITECTURAL VIEW OF A SENSOR NODE 16

FIGURE 2.2: INTERNET OF THINGS ARCHITECTURE LAYER AS PROPOSED BY [27] 17

FIGURE 2.3: ETSI M2M TOP-LEVEL ARCHITECTURE [33] 18

FIGURE 2.4: FI-WARE IOT ARCHITECTURE [34] 20

FIGURE 2.5: FUNCTIONAL-DECOMPOSITION VIEWPOINT OF THE IOT REFERENCE ARCHITECTURE [35] 22

FIGURE 2.6: IOT6 ARCHITECTURE [38] 23

FIGURE 2.7: A WIRELESS SENSOR NETWORK 25

FIGURE 2.8: SMART PARKING IN AN URBAN CITY CENTRE [44] 27

FIGURE 2.9: A SAMPLE IOT SENSOR NODE FOR DETECTING AVAILABLE SPACE FOR SMART PARKING [44] 27

FIGURE 2.10: THE FRESHWATER AQUARIUM IS HOME TO MORE THAN 20 PIRANHAS [47] 28

FIGURE 2.11: SMART FLEET TRACKING [48] 30

FIGURE 2.12: SENSOR BASED TRACKING OF GOODS THROUGHOUT THE TRANSPORTATION CHAIN [50] 31

FIGURE 2.13: REAL-TIME EHEALTH MONITORING OF PATIENT [52] 32

FIGURE 2.14: IOT SENSOR DEVICES FOR REMOTE CROP MANAGEMENT MONITORING [53] 33

FIGURE 2.15: IOT ENABLED WEATHER STATION MONITORING SENSORS [53] 33

FIGURE 3.1: A LAYERED IPV6 ARCHITECTURE SHOWING END-TO-END CONNECTIVITY COVERING [58] 36

FIGURE 3.2: A RPL NETWORK SHOWING THE FLOW OF PACKETS IN A POINT-TO-POINT TRAFFIC BETWEEN TWO NODES 39

FIGURE 3.3: ARCHITECTURAL VIEW OF THE INTERPLAY OF IOT APPLICATION PROTOCOLS [71] 41

FIGURE 3.4: AN IETF LAYERED REPRESENTATION OF COAP [69] 42

FIGURE 3.5: AN MQTT PUBLISH/SUBSCRIBE TOPOLOGY 43

FIGURE 3.6: SECURITY VULNERABILITY OF IOT DEVICES [78] 46

FIGURE 3.7: ATTACKER WITH STRONG SIGNAL BROADCASTING HELLO PACKETS 52

FIGURE 3.8: SENSOR NODES ENTERING INFINITE LOOP WHILE TRYING TO TAKE ROGUE NODE AS NEIGHBOUR 52

FIGURE 3.9: A SUMMARY OF IOT ATTACKS ACROSS THE OSI LAYERS 58

FIGURE 3.10: ARCHITECTURAL COMMUNICATION PROCESS OF THE SOFTWARE DEFINED PERIMETER [129] 73

FIGURE 4.1: DESIGN SCIENCE METHOD APPLIED TO SECURE IOT ROUTING ADAPTED FROM [193] 86

FIGURE 4.2: SECTRUST SYSTEM 89

FIGURE 4.3: DIRECT AND RECOMMENDED TRUST RELATIONS 91

FIGURE 4.4: THE FIRST SECTRUST TOPOLOGY FRAMEWORK AS PRESENTED IN [191] 99

FIGURE 4.5: EVALUATION PROCESS OF SECTRUST 100

FIGURE 4.6: A 30-NODE TOPOLOGY PLAN OF A SMART HOME WITH 1 SINK NODE, 26 SENSOR NODES AND 3 MALICIOUS NODES 102

FIGURE 5.1: DESIGN EVALUATION TESTING OF SECTRUST AGAINST GLOBAL TRUST MODELS 105

FIGURE 5.2: TRUST COMPUTATION IN EIGENTRUST MODEL 107

FIGURE 5.3: INITIALIZATION AND NORMALIZATION OF TRUST TUPLE COMPUTATION 107

FIGURE 5.4: A WORKFLOW FOR THE EVALUATION OF TRUST SYSTEMS AS ILLUSTRATED IN [1] 112

FIGURE 5.5: BASIC LOOP PROCEDURE OF QTM TRACE SIMULATOR 118

FIGURE 5.6: TRUST SYSTEM EFFECTIVENESS OF A 50-NODE NETWORK IN A PURELY NAIVE SCENARIO 127

FIGURE 5.7: TRUST SYSTEM EFFECTIVENESS OF A 50-NODE NETWORK IN A PURELY COLLECTIVE SCENARIO 127

FIGURE 5.8: TRUST SYSTEM EFFECTIVENESS OF A 100-NODE NETWORK IN A PURELY NAIVE SCENARIO 128

FIGURE 5.9: TRUST SYSTEM EFFECTIVENESS OF A 100-NODE NETWORK IN A PURELY COLLECTIVE SCENARIO 128

Figure 5.10: Trust system effectiveness of a 500-node network in a purely naive scenario 130
Figure 5.11: Trust system effectiveness of a 500-node network in a purely collective scenario 130
Figure 5.12: Trust system effectiveness of a 1000-node network in a purely naive scenario 131
Figure 5.13: Trust system effectiveness of a 1000-node network in a purely collective scenario 132
Figure 5.14: Transaction validity metric under a purely naïve scenario 134
Figure 5.15: Transaction validity metric under a purely collective scenario 135
Figure 5.16: Effectiveness metric in a Sybil naïve scenario 138
Figure 5.17: Enlarged effectiveness metric of EigenTrust, EigenTrust Incremental and SecTrust in a Sybil naïve scenario 138
Figure 5.18: Sybil effect metric ratio in a Sybil naïve scenario 139
Figure 5.19: Enlarged Sybil effect metric ratio of EigenTrust, EigenTrust Incremental and SecTrust in a Sybil naïve scenario 140
Figure 5.20: Effectiveness metric in a Sybil collective scenario 141
Figure 5.21: Effectiveness metric in a Sybil of EigenTrust, EigenTrust Incremental and SecTrust in a Sybil collective scenario 141
Figure 5.22: Sybil effect metric ratio in a Sybil naïve scenario 143
Figure 5.23: Enlarged Sybil effect metric ratio of EigenTrust, EigenTrust Incremental and SecTrust in a Sybil naïve scenario 143
Figure 6.1: Simulation testing and validation stage of SecTrust 147
Figure 6.2: RPL Network Topology 151
Figure 6.3: A RPL routing protocol operation flowchart [205] 152
Figure 6.4: A RPL instance showing DIO broadcast 153
Figure 6.5: RPL DODAG Version - A RPL global repair due to Rank inconsistencies in the DODAG 154
Figure 6.6: Layout for IPv6 control and route message types 155
Figure 6.7: Layout for the DIO base object 155
Figure 6.8: A layout of the DAO base object 155

FIGURE 6.9: PROTOCOL STACK IN CONTIKI OS 159

FIGURE 6.10: A UDGM DISTANCE LOSS ILLUSTRATION IN COOJA 162

FIGURE 6.11: A RPL PROCESS INITIALIZATION 163

FIGURE 6.12: DIO BROADCAST BY NODE 4 TO SURROUNDING NODES FOR WHO MAY POTENTIALLY CONSIDER IT AS THEIR PREFERRED PARENT NODE 164

FIGURE 6.13: DAO MESSAGE SENT BY NODE 3 TO NODE 1 FOR DOWNWARD ROUTE FORMATION 164

FIGURE 6.14: DIS MESSAGE SENT BY NODE 2 TO NODES 8 AND 14 FOR DIO SOLICITATION 164

FIGURE 6.15: SECTRUST-RPL ADAPTATION OF RANK COMPUTATION AS ADAPTED FROM RFC6550 166

FIGURE 6.16A: SECTRUST-RPL ALGORITHM FOR TRUST COMPUTATION, DETECTION AND ISOLATION OF MALICIOUS NODES DURING PARENT SELECTION 167

FIGURE 6.17: SECTRUST ENGINE SHOWING ALL FIVE PROCESSES FOR TRUST COMPUTATION AND PROCESSING 170

FIGURE 6.18: SECTRUST-RPL FLOWCHART WITH THE TRUST ENGINE USED FOR ROUTING DECISIONS AND RANK CONSISTENCY 171

FIGURE 6.19: A TOPOLOGICAL VIEW OF A SMART HOME WITH 27 NODES AND 3 ATTACKING NODES 172

FIGURE 6.20: A RPL NETWORK OF 30 NODES INCLUDING 3 ATTACKER NODES IN COOJA SIMULATOR 173

FIGURE 6.21: A REPRESENTATION OF SMART BUILDING TOPOLOGY ARRANGEMENT OF 30 SKY MOTE SENSORS IN CONTIKI/COOJA 175

FIGURE 6.22: BLACKHOLE ATTACK ACTIVATION ON NODE 28 IN A RPL SIMULATION NETWORK 176

FIGURE 6.23: DETECTION OF BLACKHOLE ATTACKING NODES DURING RPL OPERATION 177

FIGURE 6.24: SECTRUST-RPL DETECTION AND ISOLATION OF BLACKHOLE ATTACKS IN RPL 177

FIGURE 6.25: NODE FREQUENCY RANK CHANGES BETWEEN MRHOF-RPL AND SECTRUST-RPL 178

FIGURE 6.26: THROUGHPUT PERFORMANCE MEASUREMENT BETWEEN MRHOF-RPL AND SECTRUST-RPL 179

FIGURE 6.27: PACKET LOSS RATE COMPARISON BETWEEN RPL (MRHOF) AND SECTRUST-RPL 179

FIGURE 6.28: COMPARISON OF PACKET TRANSMISSION DELAY BETWEEN RPL (MRHOF) AND SECTRUST 180

FIGURE 6.29: RANK ATTACK ACTIVATION IN A RPL SIMULATION NETWORK 180

FIGURE 6.30: RANK ATTACK ACTIVATION IN A RPL SIMULATION NETWORK 181

FIGURE 6.31: COMPARISON OF FREQUENCY OF NODE RANK CHANGES BETWEEN MRHOF-RPL AND SECTRUST-RPL DURING SIMULATION 182

FIGURE 6.32: NETWORK THROUGHPUT PERFORMANCE COMPARISON BETWEEN SECTRUST-RPL AND MRHOF-RPL DURING RANK ATTACKS 183

FIGURE 6.33: PACKET LOSS PERCENTAGE BETWEEN SECTRUST-RPL AND MRHOF-RPL DURING RANK ATTACKS 183

FIGURE 6.34: PACKET TRANSMISSION DELAY OF SECTRUST-RPL AND MRHOF-RPL UNDER RANK ATTACKS 184

FIGURE 6.35: SELECTIVE FORWARDING ATTACKS ACTIVATION IN A RPL SIMULATION NETWORK 184

FIGURE 6.36: SELECTIVE FORWARDING ATTACKS DETECTION AND ISOLATION A RPL SIMULATION NETWORK 185

FIGURE 6.37: COMPARISON OF FREQUENCY OF NODE RANK CHANGES DURING SELECTIVE FORWARDING ATTACKS IN RPL NETWORK SIMULATION 186

FIGURE 6.38: NETWORK THROUGHPUT PERFORMANCE COMPARISON BETWEEN SECTRUST-RPL AND MRHOF-RPL DURING SELECTIVE FORWARDING ATTACKS 187

FIGURE 6.39: PERCENTAGE OF PACKET LOSS COMPARISON BETWEEN SECTRUST-RPL AND MRHOF-RPL DURING SELECTIVE FORWARDING ATTACKS 187

FIGURE 6.40: PACKET TRANSMISSION DELAY OF SECTRUST-RPL AND MRHOF-RPL UNDER SELECTIVE FORWARDING ATTACKS 188

FIGURE 6.41: SYBIL ATTACKS ACTIVATION IN A RPL SIMULATION NETWORK 188

FIGURE 6.42: SYBIL ATTACKS DETECTION AND ISOLATION IN A RPL SIMULATION NETWORK 189

FIGURE 6.43: COMPARISON OF FREQUENCY OF NODE RANK CHANGES DURING SYBIL ATTACKS IN RPL NETWORK SIMULATION 190

FIGURE 6.44: NETWORK THROUGHPUT PERFORMANCE COMPARISON BETWEEN SECTRUST-RPL AND MRHOF-RPL DURING SYBIL ATTACKS 191

FIGURE 6.45: PERCENTAGE OF PACKET LOSS COMPARISON BETWEEN SECTRUST-RPL AND MRHOF-RPL DURING SYBIL ATTACKS 191

FIGURE 6.46: PACKET TRANSMISSION DELAY OF SECTRUST-RPL AND MRHOF-RPL UNDER SELECTIVE FORWARDING ATTACKS 192

FIGURE 7.1: TESTBED TESTING AND VALIDATION STAGE OF SECTRUST 195

FIGURE 7.2: A TOPOLOGY VIEW OF 14 XM1000 MOTES WITH THE SINK MOTE ATTACHED TO A DESKTOP PC, AND 2 DEPLOYED ATTACKING MOTES 197

FIGURE 7.3: THE AS-XM1000 MOTE [209] 198

FIGURE 7.4: AN ARCHITECTURE OF THE CONTIKI OS [204] 199

FIGURE 7.5: UDP SINK CODE EMBEDDING AND COMPILATION ON XM1000 MOTE 203

FIGURE 7.6: UDP SENDER CODE EMBEDDING AND COMPILATION ON XM1000 MOTE 204

FIGURE 7.7: XM1000 UDP SENDER MOTE SENDING DATA PACKET TO THE SINK MOTE........ 205

FIGURE 7.8: BLACKHOLE MALICIOUS CODE EMBEDDING AND COMPILATION ON XM1000 MOTE 206

FIGURE 7.9: TESTBED SETUP USING CONTIKI AND XM1000 MOTE AS SINK 207

FIGURE 7.10: XM1000 MOTE AS SINK MOTE CONNECTED TO CONTIKI VIA A DESKTOP PC 207

FIGURE 7.11: AN XM1000 MOTE DEPLOYED IN THE PHD RESEARCH LAB COMMUNICATING WITH THE SINK MOTE ATTACHED TO A DESKTOP PC 208

FIGURE 7.12: XM1000 MOTES DEPLOYED IN THE PHD RESEARCH LAB OF THE AUT WT BUILDING 208

FIGURE 7.13: XM1000 MOTES DEPLOYED IN THE PHD RESEARCH LAB OF THE AUT WT BUILDING 209

FIGURE 7.14: XM1000 MOTES DEPLOYED IN THE PHD RESEARCH LAB OF THE AUT WT BUILDING 209

FIGURE 7.15: XM1000 MOTES DEPLOYED IN THE PHD RESEARCH LAB OF THE AUT WT BUILDING 210

FIGURE 7.16: A SENSOR DATA COLLECTOR IN CONTIKI/COOJA 211

FIGURE 7.17: TESTBED IMPLEMENTATION OF ROUTE AND DATA COLLECTION SCHEME MECHANISM........ 211

FIGURE 7.18: SECTRUST-RPL ROUTE TOPOLOGY SETUP AND ISOLATION OF BLACKHOLE ATTACKS DURING THE TESTBED EXPERIMENT WITH XM1000 MOTES 213

FIGURE 7.19: MRHOF-RPL ROUTE TOPOLOGY SEGMENTATION UNDER BLACKHOLE ATTACKS DURING THE TESTBED EXPERIMENT WITH XM1000 MOTES 213

FIGURE 7.20: BLACKHOLE ATTACKS DETECTION AND ISOLATION DURING THE TESTBED EXPERIMENT 215

FIGURE 7.21: COMPARISON OF FREQUENCY OF NODE RANK CHANGES UNDER BLACKHOLE ATTACKS DURING THE TESTBED EXPERIMENT 215

FIGURE 7.22: NETWORK THROUGHPUT PERFORMANCE COMPARISON BETWEEN SECTRUST-RPL AND MRHOF-RPL DURING THE TESTBED EXPERIMENT 216

FIGURE 7.23: PERCENTAGE OF PACKET LOSS COMPARISON BETWEEN SECTRUST-RPL AND MRHOF-RPL UNDER BLACKHOLE ATTACKS DURING THE TESTBED EXPERIMENT 216

FIGURE 7.24: PACKET TRANSMISSION DELAY OF SECTRUST-RPL AND MRHOF-RPL UNDER BLACKHOLE ATTACKS DURING THE TESTBED EXPERIMENT 217

FIGURE 7.25: SECTRUST-RPL ROUTE TOPOLOGY SETUP AND ISOLATION OF RANK ATTACKS DURING THE TESTBED EXPERIMENT WITH XM1000 MOTES 218

FIGURE 7.26: MRHOF-RPL ROUTE TOPOLOGY SEGMENTATION UNDER RANK ATTACKS DURING THE TESTBED EXPERIMENT WITH XM1000 MOTES. 218

FIGURE 7.27: RANK ATTACKS DETECTION AND ISOLATION DURING THE TESTBED EXPERIMENT 219

FIGURE 7.28: COMPARISON OF FREQUENCY OF NODE RANK CHANGES UNDER RANK ATTACKS DURING THE TESTBED EXPERIMENT 220

FIGURE 7.29: NETWORK THROUGHPUT PERFORMANCE COMPARISON BETWEEN SECTRUST-RPL AND MRHOF-RPL DURING THE TESTBED EXPERIMENT 220

FIGURE 7.30: PERCENTAGE OF PACKET LOSS COMPARISON BETWEEN SECTRUST-RPL AND MRHOF-RPL UNDER RANK ATTACKS DURING THE TESTBED EXPERIMENT 221

FIGURE 7.31: PACKET TRANSMISSION DELAY OF SECTRUST-RPL AND MRHOF-RPL UNDER RANK ATTACKS DURING THE TESTBED EXPERIMENT 222

FIGURE 7.32: SECTRUST-RPL ROUTE TOPOLOGY SETUP AND ISOLATION OF SELECTIVE FORWARDING ATTACKS DURING THE TESTBED EXPERIMENT WITH XM1000 MOTES 223

FIGURE 7.33: MRHOF-RPL ROUTE TOPOLOGY SEGMENTATION UNDER SELECTIVE FORWARDING ATTACKS DURING THE TESTBED EXPERIMENT WITH XM1000 MOTES 223

FIGURE 7.34: SELECTIVE FORWARDING ATTACKS DETECTION AND ISOLATION DURING THE TESTBED EXPERIMENT 224

FIGURE 7.35: COMPARISON OF FREQUENCY OF NODE RANK CHANGES UNDER SELECTIVE FORWARDING ATTACKS DURING THE TESTBED EXPERIMENT 225

FIGURE 7.36: NETWORK THROUGHPUT PERFORMANCE COMPARISON BETWEEN SECTRUST-RPL AND MRHOF-RPL DURING THE TESTBED EXPERIMENT 225

FIGURE 7.37: PERCENTAGE OF PACKET LOSS COMPARISON BETWEEN SECTRUST-RPL AND MRHOF-RPL UNDER SELECTIVE FORWARDING ATTACKS DURING THE TESTBED EXPERIMENT 226

FIGURE 7.38: PACKET TRANSMISSION DELAY OF SECTRUST-RPL AND MRHOF-RPL UNDER SELECTIVE FORWARDING ATTACKS DURING THE TESTBED EXPERIMENT 227

FIGURE 7.39: SECTRUST-RPL ROUTE TOPOLOGY SETUP AND ISOLATION OF SYBIL ATTACKS DURING THE TESTBED EXPERIMENT WITH XM1000 MOTES 228

FIGURE 7.40: MRHOF-RPL ROUTE TOPOLOGY SEGMENTATION UNDER SYBIL ATTACKS DURING THE TESTBED EXPERIMENT WITH XM1000 MOTES 228

FIGURE 7.41: SYBIL ATTACKS DETECTION AND MITIGATION BY SECTRUST-RPL DURING TESTBED EXPERIMENT 229

FIGURE 7.42: COMPARISON OF FREQUENCY OF NODE RANK CHANGES UNDER SYBIL ATTACKS DURING THE TESTBED EXPERIMENT 230

FIGURE 7.43: NETWORK THROUGHPUT PERFORMANCE COMPARISON BETWEEN SECTRUST-RPL AND MRHOF-RPL DURING THE TESTBED EXPERIMENT 230

FIGURE 7.44: PERCENTAGE OF PACKET LOSS COMPARISON BETWEEN SECTRUST-RPL AND MRHOF-RPL UNDER SYBIL ATTACKS DURING THE TESTBED EXPERIMENT 231

FIGURE 7.45: PACKET TRANSMISSION DELAY OF SECTRUST-RPL AND MRHOF-RPL UNDER SYBIL ATTACKS DURING THE TESTBED EXPERIMENT 232

FIGURE 8.1: CONFIDENCE INTERVAL AS A NORMAL DISTRIBUTION OF A MEAN BASED POPULATION ESTIMATE 236

FIGURE 8.2: PROPOSED SECURE TRUST-BASED IOT ROUTING FRAMEWORK 256

1 Introduction

With the advancement in mobile computing and wireless communications, a new paradigm known as the *Internet of Things* (IoT) is swiftly generating a lot of research interest and industrial revolution. The *Internet of Things* could be described as the pervasive and global network which aids and provides monitoring and control of the physical world through the collection, processing and analysis of generated data by IoT sensor devices [2]. These devices have built-in sensing and communication interfaces such as sensors, radio frequency identification devices (RFID), global positioning system (GPS), infrared sensors, laser scanners, actuators, wireless local area networks (WLANs) and even local area networks (LANs) interfaces [3].

These *things* can be connected to the internet and hence could be controlled and managed remotely. These devices could interact among themselves (machine-to-machine communication) by way of sending and receiving information, sensing the environmental temperature, pressure etc. while transmitting same to other devices for further processing or other actions such as activating an actuator [4].

Furthermore, IoT could be viewed as a melding of heterogeneous networks, which brings not only the same security challenges present in sensor networks, mobile telecommunications and the internet, but with some peculiar and accentuated issues such as network privacy problems, authentication on a heterogeneous network, access control issues and secure routing in heterogeneous devices [3]. The driving aim of IoT however, is to connect machine to machine (M2M), human to human (H2H), human to machine (H2M) while providing ease of communication, identification, management and control among the devices [3]. There are numerous opportunities and benefits of IoT to mankind and these include: wildlife monitoring, environmental monitoring (pollution, water reservoir observation), e-health systems and monitoring, ITS (Intelligent Transportation System) and smart grids among others [5].

In essence, IoT promises to bring about a wide range of smart services and applications beneficial to individuals and organisations while bringing about great comfort and ease in our everyday lives, through the connection of machine-to-machine (M2M), human-to-machine (H2M) and human-to-human (H2H) in diverse ways, at any place, and at any time. [5, 6] Figure 1.1 illustrates the interconnectedness of different IoT devices across a heterogeneous network.

However, IoT currently, is not without a number of interesting research challenges and some of which are the unique identification of objects on the network, the representation and storage of exchanged messages and communication protocols, and the security of the information (both data and control messages) exchanged [4, 7, 8]. Security in IoT is quite different from Internet security and in particular, routing security - for it is far more complicated due to the need to provide safety for the routing information and the information payload that may traverse heterogeneous networks of billions of devices. It is therefore necessary that concentrated research work and proper emphasis be given to each aspect of the security problems mentioned to ensure stable and sustainable IoT [3, 8].

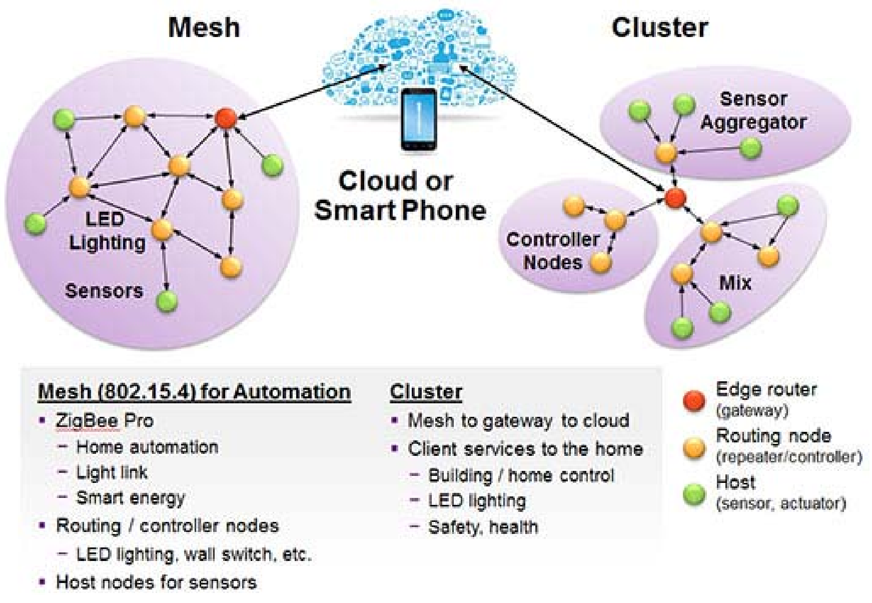


Figure 1.1: Interconnectivity of IoT nodes comprising of edge routers (gateway to the cloud), routing nodes, which also serve as control nodes, and mobile sensory/ actuator nodes [9]

1.1 Motivation

The Internet of Things (IoT) has in the last few years, become a topical issue in the academia and the technology industry. Although it is becoming increasingly ubiquitous, IoT supports a comprehensive representation of the physical environment and a good level of interaction with the physical world [7, 10]. Some typical areas such as logistics, intelligent transportation systems (ITS), business/process management and e-health are but a few instances of conceivable application fields where this novel paradigm is gaining attention.

The realization of IoT will greatly hinge on various criteria such as the system's architecture, networks and communications, data processing, and ubiquitous computing technologies, which support efficient, reliable, physical and cyber interconnectivity [7, 11, 12]. Again, for swift public adoption of IoT, social and technological issues emanating from the introduction of IoT will also have to be addressed.

One of the fundamental driving forces of IoT is networking, which drives and facilitates the interconnection of devices [12]. Of particular importance is routing in the network; it involves how traffic routes are built, transmitted and controlled within the network so that a routed packet could travel from source to its final destination. Furthermore, with the interconnectivity among the billions of devices, the big issue is how secure is the network from various forms of attacks. A realisation at this point is that users will not feel secure if they know their private data could be accessed and compromised by unauthorised individuals or machines over the network, especially from the perspective of the billions of devices that will be communicating online as projected in Figure 1.2 [13-15].

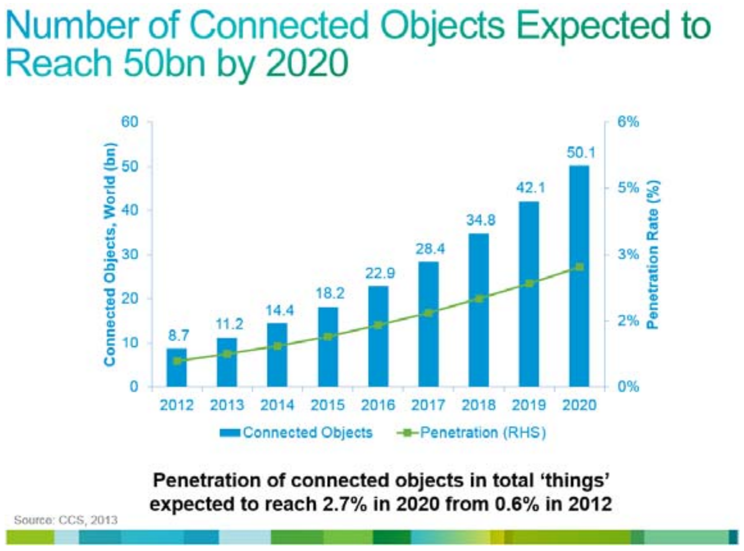


Figure 1.2: A forecast of 50 billion connected online devices [16]

Networking and related issues including the security of networks are of great importance in IoT [8, 12], especially when packets have to be routed through heterogeneous networks of smart nodes to a server on the internet. As noted by [8], *routing* and *addressing* are two important issues in IoT, which need to be dealt with since network topologies across several networks vary and the need for common understanding and uniformity is important for proper routing of a packet originating or arriving at an IoT enabled device. Undeniably, the roadmap to achieving ubiquitous IoT brings various challenges ranging from device integration, heterogeneity, scalability to mobility, routing, security and other specific challenges [7]. It is clear from the above that IoT security is an aspect that requires in-depth research work as the need to secure networks today have become imperative. Recent IT security threats reports for 2015 and 2016 reveal that threats like botnets, malware, Distributed Denial of Service (DDoS) attacks, web-based malware, android malware and Spams, are on the increase and the evolution of IoT further provides a rife platform for the propagation of more attacks [17, 18]. The Cisco annual security report of 2015 [19] submits that as IoT continues its evolution, and the number of IoT devices with “people-less” contacts multiply, attacks of great consequences will unavoidably happen. The report concludes therefore, that a good and embedded IoT security design of products will help stop or at least mitigate the impact of any attacks.

1.2 Research Questions

This thesis seeks to develop a trust-based IoT routing system that will embed integrity while providing availability during the routing process of an IoT network. Specifically, this work proposes to address the research questions (RQ) outlined below:

RQ1. What are the routing attacks that compromise availability and integrity of routing information amongst IoT sensor nodes?

RQ2. What system can be implemented to maintain secure network routing amongst IoT sensor nodes in an efficient manner?

RQ3. What system can be built such that the level of routing security enforcement is dependent on the trust level of the collaborating nodes (i.e. a trust-based system)? How can the proposed system address the attacks identified in RQ1?

RQ4: How can the new system minimize the impact on network performance as it addresses the routing attacks?

1.3 Significance of the Study

The significance of this study is outlined below based on the research questions proposed in Section 1.2.

RQ1

IoT being an emerging disruptive technology requires the swift identification and understanding of the present and potential attack types during routing of data packets. The early identification of the attacks could help IoT designers, manufacturers and the research community address the attacks that could potentially limit the growth and adoption of IoT on a global scale. RQ1 addresses this research gap.

RQ2

IoT networks perform mission-critical tasks therefore, the security of routing data in the network layer becomes imperative. The significance of RQ2 is the development of a system that could be embedded in the IoT sensor devices, which could defend or mitigate the identified attacks. This will provide a framework that researchers and IoT designers could study and further advance. It is hoped that with improved security in IoT systems, it will enhance the swift adoption of IoT in the global market place.

RQ3

The development of a suitable system, which mitigates the routing attacks in IoT networks, will no doubt help facilitate IoT adoption. However, such a system must be able to scale from a small to a large network without depleting the limited resources of the IoT sensor nodes. The significance of RQ3 is the development of a system that can fulfil this demand. In view of this, a trust-based system as proposed in RQ3 provides a promising solution that fulfils this unique requirement as opposed to other defence or mitigation systems.

RQ4

A good IoT network defence system must not significantly impact the performance of the network. The significance of RQ4 is to ensure that the design of the proposed system does not degrade network performance while mitigating attacks. This is validated with experimental measurements against widely used systems.

1.4 Scope of the Study

Designing a trust-based IoT routing system with a view to enshrining integrity and providing availability in IoT protocols in order to eliminate the threats of these attacks is the focus of this research thesis. Confidentiality is another key aspect of security that is not examined in this thesis.

1.5 Contributions

The contribution of this research work involves the development of a system for securing IoT routing protocols. The system is designed in line with the recommendations of [20] while validation and testing of the system developed is undertaken using advanced simulators as proposed in [21]. This is to have a proof-of-concept of the efficacy and performance of the proposed solution. In addition, a small-scale testbed was developed as validation of the simulation studies.

In conclusion, the thesis contributions are outlined below:

i. The investigation, identification and analyses of threats and vulnerabilities in IoT protocols, which until now had not being addressed or properly addressed. Security frameworks proposed by the research fraternity are examined and available standards are analysed from a security perspective.

ii. The design of a trust-based framework (*SecTrust*) for distributed systems like peer-2-peer networks. In this trust-based framework, security enforcement is dependent on the trust level of the collaborating nodes. The proposed system though based on trust enforces security and collaboration among nodes to make the network secure. This trust-based framework is extended and adapted to IoT networks.

iii. A performance comparison between the proposed Trust-based system with other global trust-based research studies in isolating studied attacks.

iv. The development of *SecTrust-RPL* protocol – an enhanced RPL routing protocol, which embeds the *SecTrust* framework used for the detection and isolation of RPL routing protocol attacks. Simulation study is used to verify this solution. In addition, a small-scale testbed is implemented to validate the new trust-based protocol system proposed. This involves the deployment of sensor motes and observing their peculiar behaviour during routing attacks.

v. Performance evaluation and comparison of proposed enhanced RPL protocol (*SecTrust-RPL*) with RPL routing protocol to ensure an acceptable level of network performance. This involves both simulation and testbed experiment of sensor motes and observing the result in comparison to other existing research studies in IoT routing protocols.

1.6 Publications

Journal Papers

i. Airehrour, D., Gutierrez, J., & Ray, S. (2017). A Trust-Aware RPL Routing Protocol to Detect Blackhole and Selective Forwarding Attacks. *Australian Journal of Telecommunications and the Digital Economy*, 5(1), 50-69. doi:http://dx.doi.org/10.18080/ajtde.v5n1.88

ii. D. Airehrour, J. Gutierrez, S.K. Ray (2016), Secure routing for Internet of Things: A survey, *Journal of Network and Computer Applications*. ISSN 1084-8045, http://dx.doi.org/10.1016/j.jnca.2016.03.006.

Conference Papers

iii. D. Airehrour, J. Gutierrez and S. K. Ray (2017), A Testbed Implementation of a Trust-Aware RPL Routing Protocol, *27th IEEE International Telecommunication Networks and Applications Conference (ITNAC)*, Melbourne, Australia, November 22 – 24, 2017.

iv. D. Airehrour, J. Gutierrez and S. K. Ray (2016), Securing RPL Routing Protocol from Blackhole Attacks Using a Trust-based Mechanism, *26th IEEE International Telecommunication Networks and Applications Conference (ITNAC)*, Dunedin, New Zealand, December 7-9, 2016.

v. D. Airehrour, J. Gutierrez and S. K. Ray (2016), "A Lightweight Trust Design for IoT Routing", 14th *International Conference on Pervasive Intelligence and Computing (PICom)*, Auckland, New Zealand, August 8-10, 2016.

vi. D. Airehrour, J. Gutierrez and S. K. Ray (2016), "Greening and Optimizing Energy Consumption of Sensor Nodes in the Internet of Things through Energy Harvesting: Challenges and Approaches", *International Conference on Information Resources Management (Conf-IRM)*, Cape Town, South Africa, May 18-20, 2016.

vii. D. Airehrour, J. Gutierrez, W. Liu and J. Wu (2015), "When Internet Raised to the Things Power: Are Energy Efficiency Standards Sufficient to Curb Carbon Footprints?", *IEEE Global Communications Conference (GLOBECOM)*, San Diego, USA, 6-10 Dec 2015.

viii. D. Airehrour, J. Gutierrez and S.K. Ray (2015), "GradeTrust: A Secure Trust Based Routing Protocol for MANETs", *25th International Telecommunication Networks and Applications Conference (ITNAC)*, Sydney, Australia, Nov., 18-20, 2015. http://dx.doi.org/10.1109/ATNAC.2015.7366790

ix. D. Airehrour and J. Gutierrez (2015), "An analysis of secure MANET routing features to maintain confidentiality and integrity in IoT routing", *International Conference on Information Resources Management (Conf-IRM)*, Ottawa, Ontario, Canada, May 18-20, 2015. http://aisel.aisnet.org/cgi/viewcontent.cgi?article=1030&context=confirm2015

x. D. Airehrour, "A study of secure routing features of MANETs and their potential for IoT routing", *13th New Zealand Computer Science Research Student Conference (NZCSRSC)*, Auckland, New Zealand, April 9-10, 2015.

Symposia

xi. D. Airehrour, J. Gutierrez and S. K. Ray (2016), “A novel trust system for the detection and isolation of sinkhole attacks in Internet of Things (IoT) networks”, 10th Annual AUT Postgraduate Research Symposium, August 18, 2016. https://www.aut.ac.nz/__data/assets/pdf_file/0010/673534/David-Airehrour.pdf

xii. D. Airehrour, J. Gutierrez and S.K. Ray (2015), “RankTrust: Isolating Sybil attack using secure trust level routing in wireless sensor networks”, 9th Annual AUT Postgraduate Research Symposium, August 21, 2015. http://www.aut.ac.nz/__data/assets/pdf_file/0007/578203/David-Airehour-edited-for-publication.pdf

xiii. D. Airehrour (2014), “Developing a Secure Routing Protocol for MANETs”, *2014 IEEE ComSoc Distinguished Lecturer Talks*, Auckland, NZ, November 13, 2014.

1.7 Thesis Outline

A secure trust management system for secure routing in IoT is proposed and specific research questions have been outlined with a proposed research method. To actualize the result of this study, a trust frame work was developed and tested using simulation study. In addition, the simulation study was validated using a small-scale testbed. The nature of this proposed research is constructive starting with problem formulation, trust solution modelling, evaluation and validation of the research study.

This research uses both simulation study and testbed experimentation to actualize the outcome of this thesis because: simulation can be used to explore the effects of node interaction in different scenario settings including network topology change over a protracted period of time. Similarly, simulation provides the study of complex node-network systems that would otherwise be difficult to explore. On the other hand, testbed provides an accurate hardware-software setting where live IoT sensor nodes embedded with the proposed framework from this thesis can be deployed to test and observe system performance without having the final product. The proposed testbed gives a means to advance the understanding of the practical requirements and node operational performance in *situ*. The testbed provides the measurements from which results on the performance of the proposed system can be derived while making a comparison with the simulation results and deriving the standard deviation between both results to ascertain how far from reality is the simulation model. As a justification for using testbed, the testbed offers an environment in which, design choices can be founded using theoretical and practical studies.

Chapters 2 and 3 explore IoT applications and a literature study of secure communication in IoT. Furthermore, a study of trust and management systems is also presented.

Chapter 4 presents Secure Trust (*SecTrust*) framework definition and description. A secure trust-based framework for peer-to-peer (P2P) distributed networks. In this framework, the trustworthiness of a node is computed and evaluated by overhearing and observing the packet forwarding behaviour of nodes. The trustworthiness reflects the level of reliability or dependability (trust) that other peer nodes in the network have on a given node. The trustworthiness of a node is characterized by the evaluation from its neighbouring nodes during service provision (*recommendations* trust values) to other nodes. This framework predicates trustworthiness of a node as a time-based successful packet exchange between nodes and positive packet acknowledgements with a continuous observation of the characteristic behaviour of linked nodes in the network.

In Chapter 5, a validation of *SecTrust* model is presented against other global trust models covered in the survey of literature. The models intended for validation against *SecTrust* include: The *EigenTrust* model, *EigenTrust* with speed-up functionalities and *No Trust* model (the lack of trust management system). These models were tested under various malicious attacks with a scaling range of 50 to 1000 sensor nodes to test the scalability of the *SecTrust* system against the aforementioned systems.

Chapter 6 quantitatively demonstrates with simulation the performance of *SecTrust* while being implemented in RPL routing protocol. A comparison is made with the standard RPL routing protocol to provide a proof-of-concept of the effectiveness of the *SecTrust* system in providing optimal routes in addition to making the routes secure.

A testbed experiment is presented in Chapter 7 that validates and demonstrates the efficacy of *SecTrust* in RPL (*SectTrust-RPL*) against the standard RPL. Results obtained during experimentation were observed, analysed and discussed.

Chapter 8 provides a statistical evaluation of the correlation of the simulation and testbed data gathered. The chapter concludes with a framing of the thesis' contribution. In addition, areas of improvements are identified in the advent of the adoption of a trust-based IoT framework.

Finally, in Chapter 9, conclusions are provided, which discuss the limitations of the study and some areas of future study. The structure of this thesis is summarised in Figure 1.3.

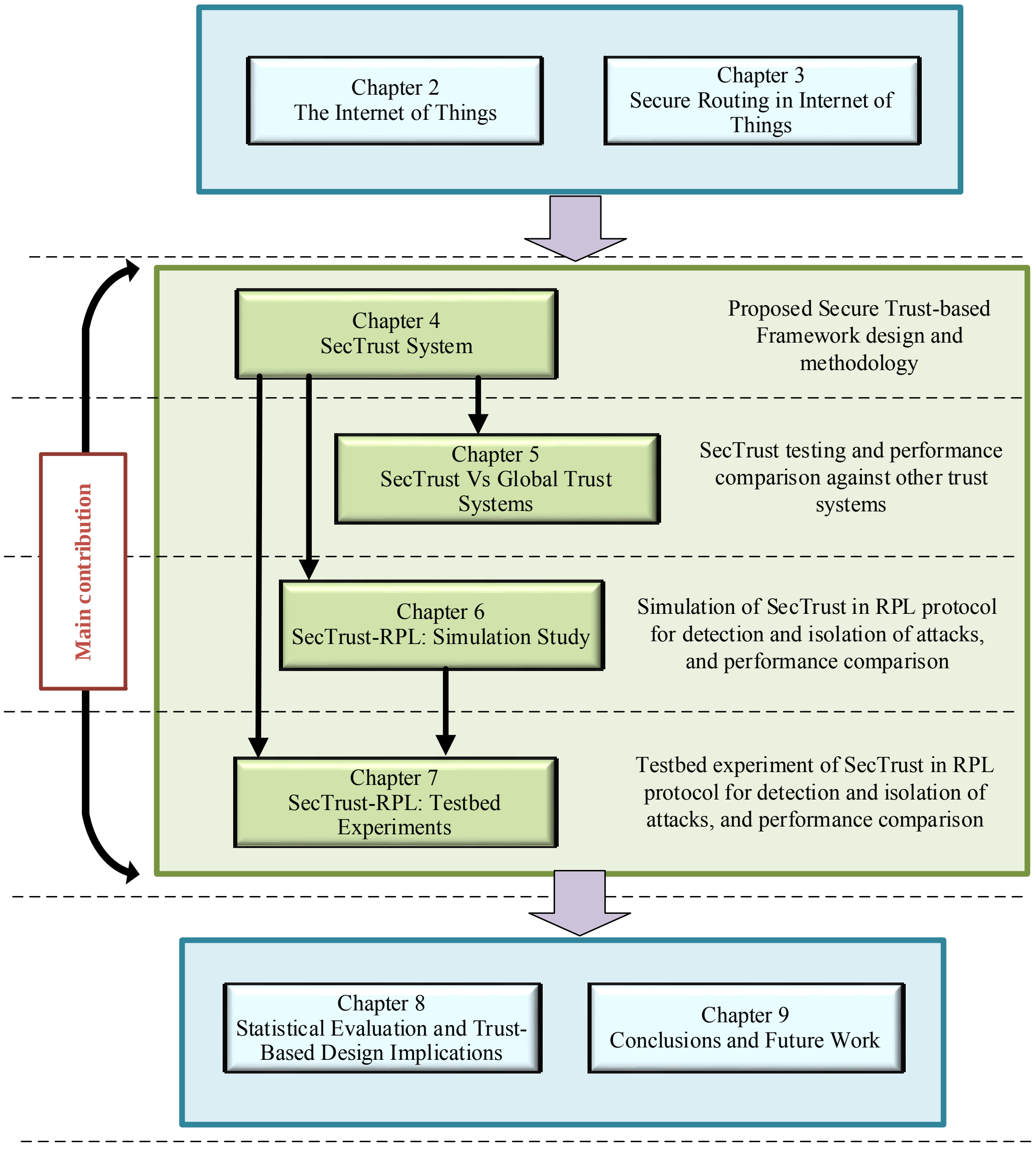

Chapter 2
The Internet of Things
Chapter 3
Secure Routing in Internet of Things
Chapter 4
SecTrust System
Proposed Secure Trust-based Framework design and methodology
Chapter 5
SecTrust Vs Global Trust Systems
SecTrust testing and performance comparison against other trust systems
Main contribution
Chapter 6
SecTrust-RPL: Simulation Study
Simulation of SecTrust in RPL protocol for detection and isolation of attacks, and performance comparison
Chapter 7
SecTrust-RPL: Testbed Experiments
Testbed experiment of SecTrust in RPL protocol for detection and isolation of attacks, and performance comparison
Chapter 8
Statistical Evaluation and Trust-Based Design Implications
Chapter 9
Conclusions and Future Work

Figure 1.3: Thesis structure

2 The Internet of Things

The Internet of Things is a technological revolution that has emerged from previous technologies including Mobile Ad hoc NETworks (MANETs) and Wireless Sensor Networks (WSN) [22]. A sensor is a device contained within a personal area network (PAN) containing sensing (measuring), computing, and communication elements that gives the observer the opportunity to observe and respond to events and occurrences within a location.

MANET is a collection of mobile sensor devices (called nodes) that communicate with each other without the use of infrastructure such as access points or base stations. These networks are self-configuring, capable of self-directed operations and are easily deployable hence, they are referred to as a Self-Organizing Networks (SONs). Nodes cooperate to provide connectivity and operate without centralized administration [23]. MANET nodes require no fixed infrastructure thus, they communicate in a peer-to-peer manner with directly connected neighbours. Due to their mobility and limited battery life, they are limited in their transmission power and bandwidth availability. Mobile nodes in ad hoc networks often cooperate to transmit data and route information to other nodes that they are not directly connected to, hence they act as routers where they compute routes and create route tables of destination to other nodes.

A WSN consists of densely distributed sensor nodes which support sensing, signal processing, embedded computing, and connectivity; they are interconnected and self-organizing. Sensors are deployed in a short-hop point-to-point and in a master–slave pair order. They transmit information to monitoring nodes (sink node) which aggregate data collected to a central station for further analysis. WSNs just like MANETs have peculiar features, including limited battery power, redundant data acquisition, and low duty cycle. They could communicate in a multipoint-to-point fashion (i.e. sensor nodes transmit their data to a central node referred to as the sink node). It is noteworthy that, early on in the development of MANETs and WSNs their potential benefits were quickly recognized and this led to their deployment in many key application areas like agriculture, manufacturing, healthcare systems [24]. Today the Internet of Things has emerged from the ashes of these two technology pillars.

The Internet of Things (IoT) refers to the widespread use of systems, heterogeneous technologies and the evolving paradigm of the interconnectedness of devices, using TCP/IP protocols, around physical environments. From an initial perspective, IoT looks like a M2M communication (the communication between system entities in a wired or wireless system that does not necessarily require direct human intervention) however, IoT encompasses not only M2M but also humans, home appliances, vehicles, machinery, pets, cattle in the field, animals in the wild, habitats, habitat occupants, even corporate organisations and how they interact with one another. IoT has become a buzz word today and promises to change the way internet is seen and used. In a recent survey report by the McKinsey Global Institute, IoT could have an US$ 11 trillion global market impact by 2025 [25]. As noted also in [26], the Internet of Things is not just a futuristic technology rather it is here with us. IoT includes the new wave of sensor devices and it operates with the growing cloud network infrastructure.

On the long run, it is envisioned that an IoT ecosystem will evolve, which will not be too different from the internet. It will facilitate the interaction of devices (mobile or fixed), smart objects and other real-world devices just as humans interact nowadays using internet-based applications. It is anticipated that IoT will be an extension of the internet as it will not only involve the interaction of people, content and media, but also the interaction of smart objects with humans and/or other machines in supporting and enhancing the quality of life of mankind [27]. In fact, interactions will be achieved using diverse networks of computerized devices having various functionalities, sizes, and capabilities like RFID tags, android smartphones, iPads, monitoring nodes, sensors, and tags with a host of assorted application servers serving different purposes.

In general, IoT is considered to have a three-tier spectrum of communication. At the “low end” of the spectrum of communication will be the unique identification (UID) and/or electronic product code (EPC) where information is stored in an RFID tag and the information is transmitted in the network via a non-contact reading using an RFID reader. The “mid-range” of the spectrum will include nodes with implanted intelligence (microchips) with active wireless capabilities, which can perform a variety of processing tasks such as data gathering, routing and even control functions. Examples of such devices are, home appliances, power management systems, wearable biomedical sensors and industrial control sensors. At the far side of the spectrum are more enhanced sensors, which could be integrated into the IoT for diverse operational uses [28]. Some of these sophisticated sensors utilize distributed wireless sensor network(WSN) systems, which can gather environmental data such as temperature, pressure, environmental chemical content and ambient video images from geographically dispersed locations; These sensors also have capabilities to pre-process data and forward information processed to a centralized site for further processing [28]. Figure 2.1 illustrates the architectural perspective of a wireless sensor node. Examples of typical applications of IoT in real life scenarios include their use in agriculture where sensors and actuators could monitor and control the climatic environment of plant and adapt the conditions according to the specific needs of the plant without any human intervention. Through IoT technology, smart metering solutions are implemented while the introduction of automatic meter reading applications assists customers achieve better energy monitoring, usage and spending.

Furthermore, in mining where the process involves blasting, crushing, grinding and ore processing, IoT technology could be introduced in the automation and interconnection of all systems and processes. The machinery utilized can be controlled and monitored remotely while the mine sites are linked, and mining shafts observed in relation to the air and gaseous levels. Since the energy consumption from a ventilation is 50% of the total energy required, energy savings could be achieved through a very accurate ventilation reading where the diesel vehicles are operating and sensors in the mine could give information about the location of the machines. Also, sensors and actuators will control the mining sites, the drilling and haulage equipment. All these devices will be communicating with each other in a synchronous manner achieving results while eliminating accidents to humans, which is one of the main hazards of the mining industry. With the above expectations about the IoT, it is generally expected that this technology will be unique in its ability to self-organise and thus, the need to have an efficient and distributed algorithmic protocol in determining network routing and link establishment and network organization.

In summary, the study presented in this Chapter constitutes part of the findings that have been published in peer reviewed conferences [29, 30]. This Chapter provides reviews with respect to the Internet of Things architecture and security.

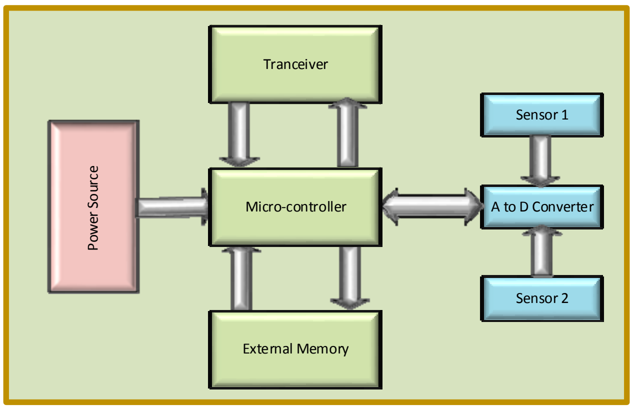


Figure 2.1: Architectural view of a sensor node

2.1 IoT Architecture

The importance of having a standardized IoT architectural reference model cannot be over emphasised as IoT reference models and architectures are necessary because they provide the guidelines, best practices, views, implementations, applications and perspectives, which could be utilized for developing interoperable IoT systems. From the software perspective, the reference architecture provides a basis from which software development firms can capitalize on the advantages of developing consumer-oriented platforms including hardware, software and allied services. Today, there is no generally acceptable IoT architecture. However various IoT architectural standards have been proposed by several researchers and research bodies. Some typical IoT architectures proposed to date are examined below.

The authors in [27] proposed a three-tier layer hierarchical structure model for IoT defined by three namely; perception, network and application layer functions. The perception layer, which is at the bottom represents the sense organ of IoT. It aims at recognizing objects and gathering information. This layer includes RFID tags, 2-D barcode labels and readers, terminals, GPS units, cameras, sensors and sensor networks. The second layer is the network layer. This layer represents the nucleus of IoT. It processes and transmits information received from the perception layer to the application layer. The network layer comprises of the following: information centre, intelligent processing centre, internet network systems and network management centre. The third layer is the application layer which is a fusion of IoT's socio-business requirements, which defines the in-depth task associated with the node. This layer represents the confluence of IoT and industrial technology with a mix of industrial needs and system processes. Although IoT is still emerging, many researchers still consider it a "cloudcastle" as it is still in its formative stage and does not yet have a definite form [27]. As advised by [27], for a proper understanding of IoT, the two system structure of IoT namely; *the Internet and communications network* should be analysed to gain better understanding of the technology and hence create a better architecture for the Internet of Things. Figure 2.2 below represents a topological representation of the architecture proposed by [27].

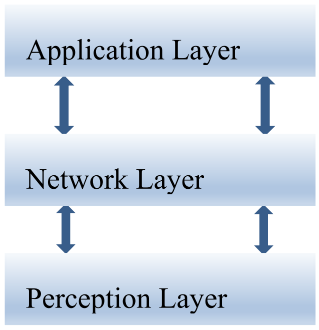


Figure 2.2: Internet of Things architecture layer as proposed by [27]

The IEEE is currently working on a standard for the IoT architecture [31]. The working group revealed that the standard defines an architectural framework for the Internet of Things (IoT), which includes succinct descriptions of several IoT fields, descriptions of IoT domain concepts, and identification of similarities between dissimilar IoT domains. It also provides a standard for the reference architecture, which builds upon the reference model. The reference architecture will define basic IoT architectural building blocks and how they could be seamlessly combined into multi-tiered systems. The model addresses how to document and limit architectural deviations [31]. The European Telecommunications Standards Institute (ETSI) delivers standards for Information and Communications Technologies (ICT) with a global focus. Technologies covered include fixed, mobile, radio, converged, broadcast and internet technologies. The ETSI M2M technical committee developed and maintained an end-to-end high level architecture for M2M. Their architecture has two different domains which include the *device and gateway* domain and the *network* domain. In their architecture, the IoT/M2M *gateways* enable communication of M2M devices with other parts of the system via an access network (wide area network). Multiple M2M devices can be connected to an M2M gateway although M2M devices connected through a gateway do not necessarily implement M2M applications and service functionality capabilities. But should an M2M device implement the M2M applications and service capability functionality, then this device can be connected directly to the access network, which can communicate with other systems in the network. The *network* domain comprises of the wide area communication networks (access and core networks), M2M service capabilities and M2M applications functions. Other modules of the network domain include M2M management and network management functions [32, 33]. Figure 2.3 illustrates the ETSI architecture. The ETSI latest release was made available in September, 2014 [33] and the corresponding architecture is shown in Figure 2.3.

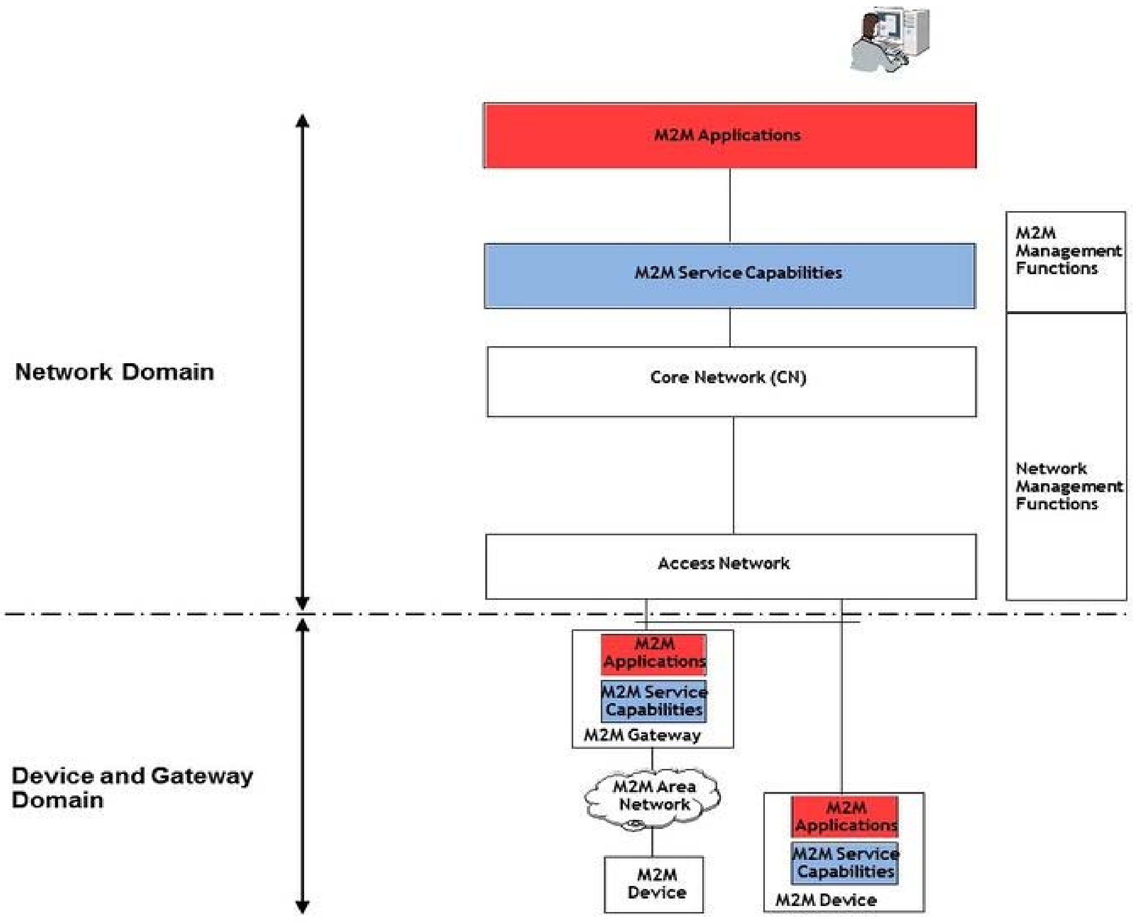


Figure 2.3: ETSI M2M top-level architecture [33]

FI-WARE, a non-profit making organization, and partially funded by the European Union, seeks to provide an open, public and royalty-free architecture for the future internet. Their proposed architecture is based on open components referred to as Generic Enablers (GE). These GEs give reusable and commonly shared functions which serve many purposes across various sectors [34]. The generic enablers are categorized into six main groups providing an architectural reference model for the specific features addressed in the architecture and they include:

i. *Cloud Hosting*: This involves the computation, storage and network resources, upon which services are based and managed.

ii. *Data/Context Management*: Involves the accessing, processing, and analysing of the big data streams while transforming them into valuable information base available to applications.

iii. *Applications/Services Ecosystem and Delivery Framework*: This is the infrastructure to build, distribute, manage and consume FI (Future Internet) services across their life cycle and handling technical and business concerns.

iv. *Internet of Things (IoT) Services Enablement*: The link (bridge) where FI services interface and leverage the ubiquity of heterogeneous and resource-constrained nodes in the Internet of Things.

v. *Interface to Networks and Devices (I2ND)*: This layer provides the open interfaces to networks and devices, providing connectivity needs of services delivered across the platform.

vi. *Security*: This layer ensures that the delivery and usage of services is trustworthy and meets security and privacy requirements.

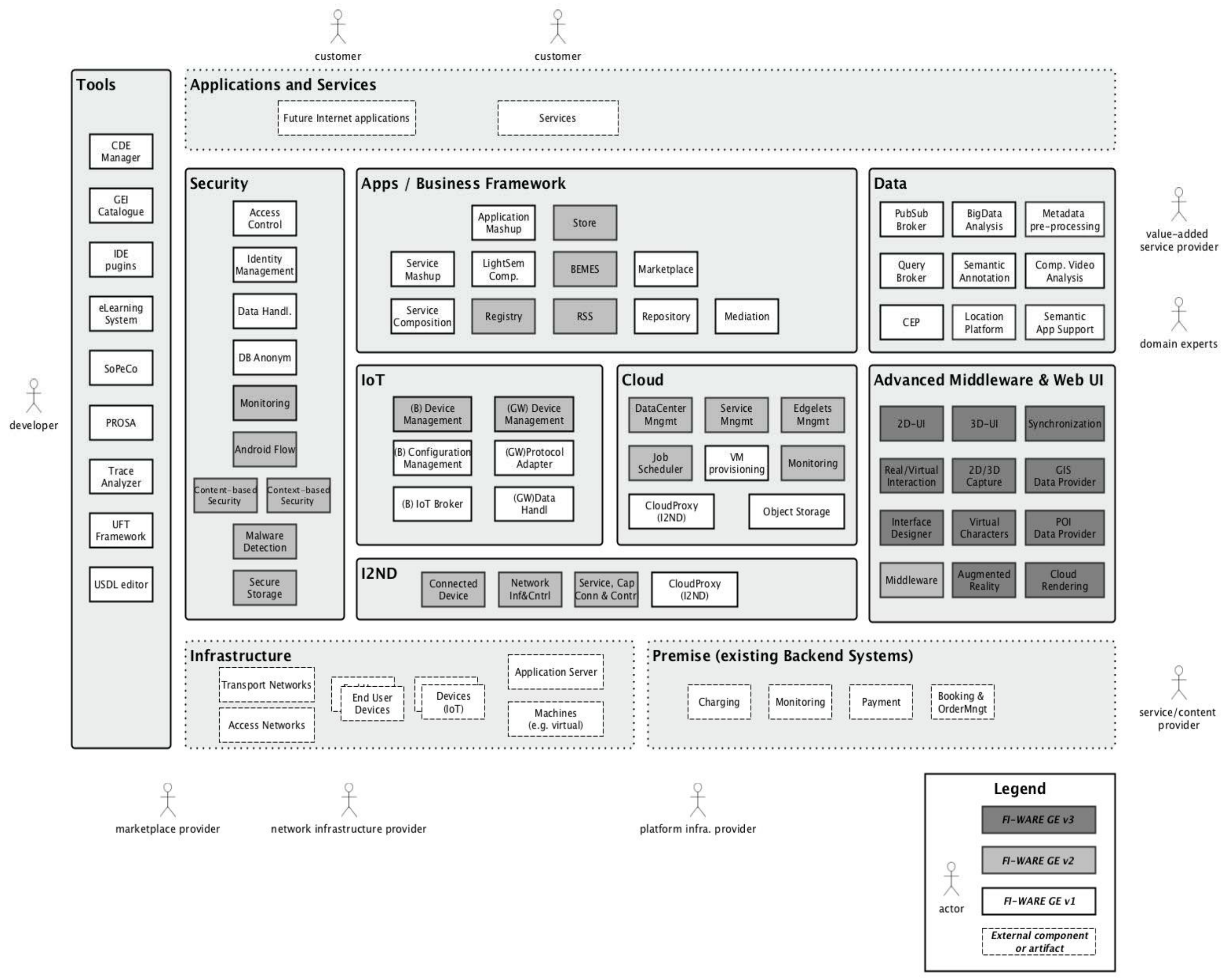


Figure 2.4: FI-WARE IoT architecture [34]

The IoT-A project was funded by the European Union (EU) under the FP7 Research Framework. The project was embarked upon to create a standard for the Internet of Things, and it was code named Architectural Reference Model (ARM) [35]. For this project researchers, proposed an abstract reference architecture. The objective of the IoT-A project was to provide an abstract Architectural Reference Model (ARM) that could be used to develop concrete IoT architectures. Simply put, IoT-A is not primarily concerned with prescribing an architecture for the Internet of Things, but rather with providing various means (models, views, perspectives, best practices, etc.) of developing an IoT architecture. With this view, two architects concentrating on two precise IoT applications could use and share the same reference architecture as a tool, but they would ultimately end up with two different architectures at the end of the architecting progression [35]. The ARM has basically three interconnected parts while drawing some vital aspects from the best practices in software engineering as presented by [36]. The constituent parts included the IoT Reference Model (RM), The IoT Reference Architecture (RA) and a Set of Guidance (referred to as best practices). They are discussed further below.

i. Reference Model (RM): The RM provides the models used in defining aspects of the architectural views. One such model is the IoT Domain Model [37], which describes the classification of IoT concepts such as the physical, virtual and augmented things, devices, resources and services and the relationships between the concepts. Furthermore, the RM provides the following:
    a. Information Model (IM) which is a meta-model used to describe information as handled within the system.
    b. Communication Model (CM), which deals with the communication within the system.
    c. Functional Model (FM) used as the foundation of the Functional View, which is used for modelling Security, Trust and Privacy [35].

ii. Reference Architecture (RA): With regards to the RM model, the RA entails a set of views (for representing certain structural aspects of the system) and perspectives (which focus on quality of the system covering different views such as, resilience and security). The Functional View (see Figure 2.5) gives the topological description in the form of a layered approach of the Domain Model (DM) in relation to the associated interfaces available in an IoT system [35].

iii. Set of Guidance: This defines the process to be followed based on the RA and RM for generating concrete IoT architectures. Specifically, guidance describes the prerequisites, incorporates more views (i.e. Physical View and Context View), which are not part of the ARM as they are very much application dependent and describe how and in what order the set of architectural views should be developed. Guidance also gives recommendations for achieving certain system qualities by giving a fairly large list of design choices [35].

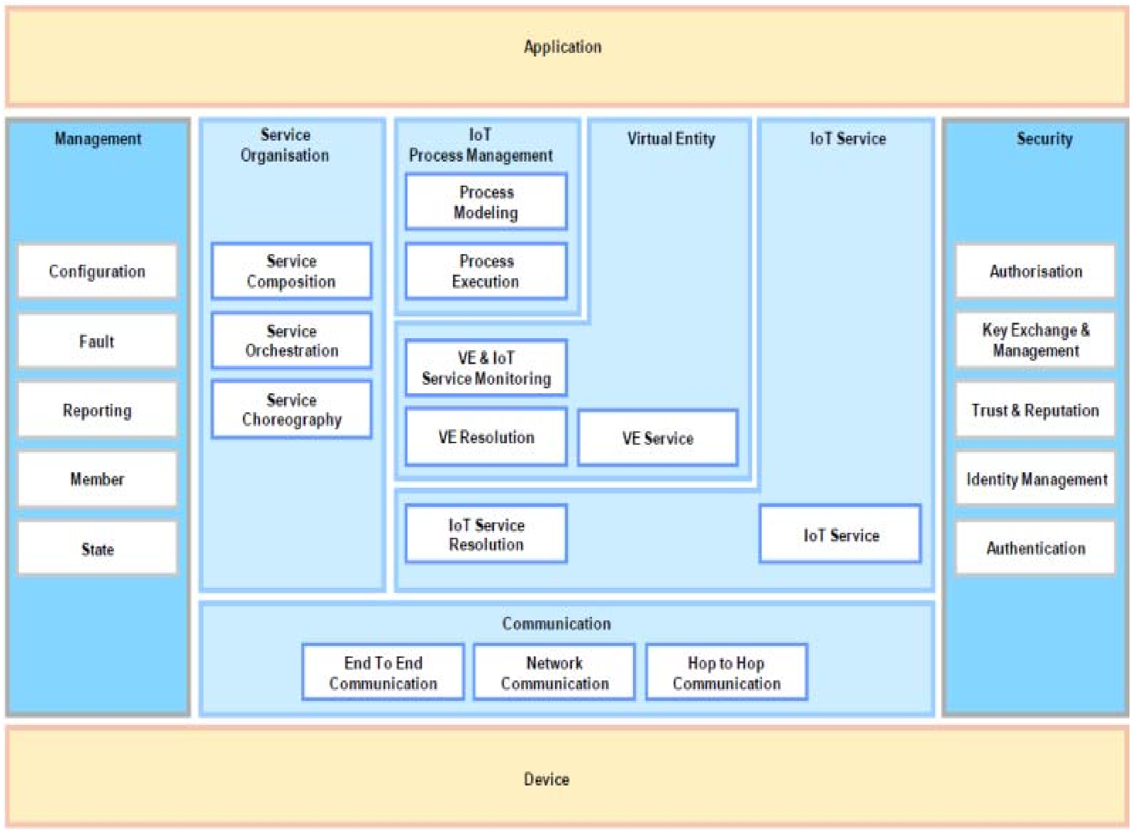


Figure 2.5: Functional-decomposition viewpoint of the IoT Reference Architecture [35]

The approach taken by the IoT6 architectural reference design group was to reuse, to any extent possible, the results of other projects [38], particularly IoT-A, ETSI M2M and FI-WARE, and to integrate and enhance them with IoT6's precise features and components (mainly IPv6 functionalities). The aim is to utilise the properties of this protocol and re-use them within the architecture model, possibly replacing some of the standard components. For example, parts of the service and resource discovery functionality are replaced with domain name system service-discovery (DNS-SD) [39] and DNS [40] based approaches. Looking at the ARM functional model (refer to Figure 2.6), the main focus of IoT6's contribution is on the Communication, Service organization, IoT service and Security components [38, 41].

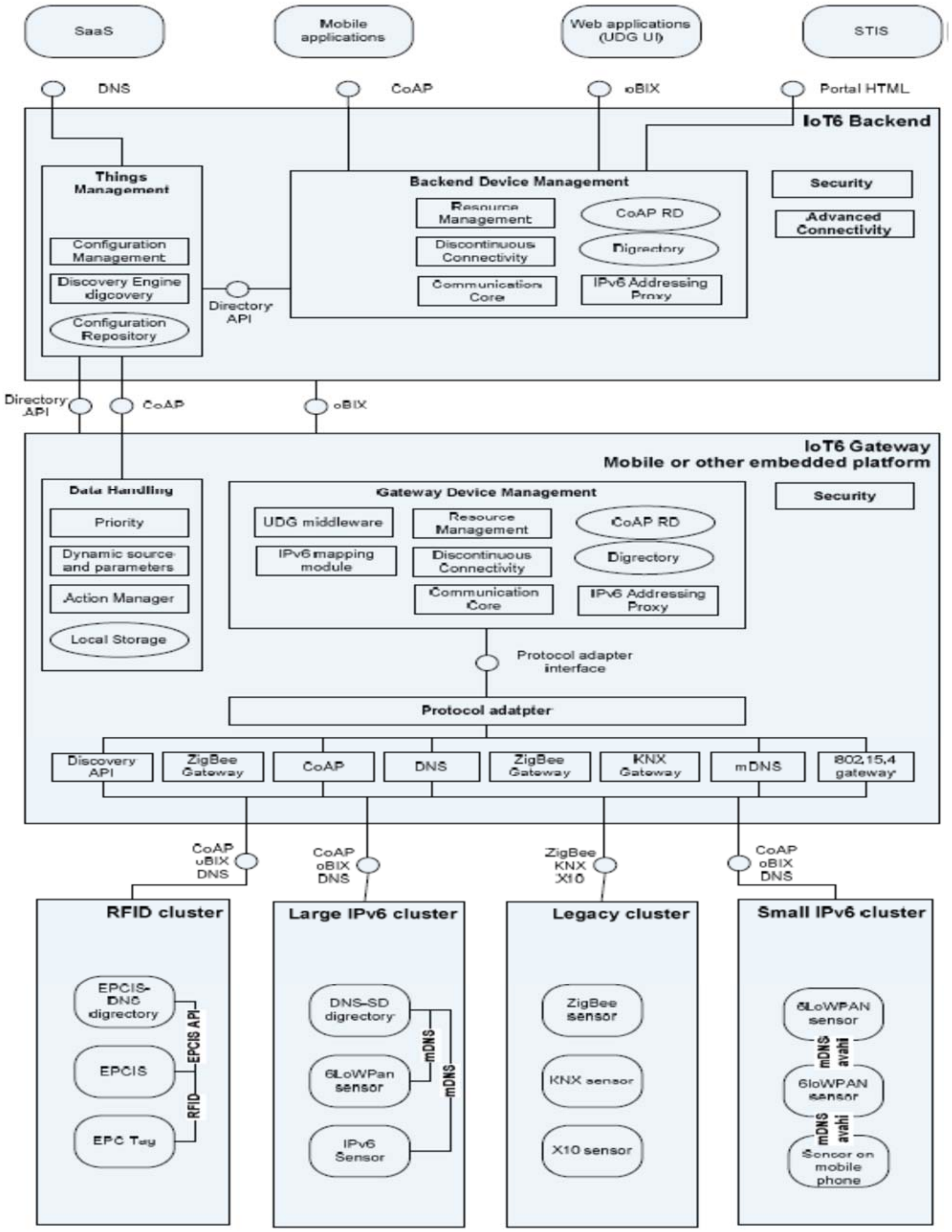


Figure 2.6: IoT6 architecture [38]

2.2 Enabling Technologies of IoT

This section explores the technologies that have accelerated the rapid evolution of the Internet of Things. Since the technology of IoT consists of large numbers of devices, many technologies will evolve, which will either help to deploy, manage or even identify things. Various technologies which have evolved and are perceived to promote the evolution of the Internet of Things are discussed below.

2.2.1 Internet Protocol Version 6 (IPv6)

The deployment of IPv6 over IPv4 has been a great enabler of the Internet of Things. This protocol has the capacity to give every object on the network an IP address and most networking equipment now support IPv6. IPv4 supports 32-bit addresses, which gives about 4.3 billion addresses; this address space has been exhausted even before the full take-off of the Internet of Things. In contrast, IPv6 supports 128-bit addressing, translating to about 3.4 x 10^{38} IP addresses – an almost limitless number that can sufficiently handle all conceivable IoT devices.

2.2.2 Radio Frequency Identification (RFID)

This is a technology used for the identification of items, entities and even people. RFID technology helps in explicitly labelling objects to facilitate their identity with the aid of computer devices. A typical RFID system consist of tags and readers, software drivers, a hardware system and middleware applications [42].

2.2.3 Wireless Sensor Network (WSN)

This is a collection of transducers fitted with communication devices such as antenna, microcontroller and a battery and are hence referred to as "sensor nodes". These sensing nodes communicate using a wireless multi-hop system among themselves. Sensors could be used for monitoring and measuring various conditions including: temperature, humidity, pressure, wind direction and speed, illumination intensity, vibration intensity, sound intensity, power-line voltage, chemical concentrations, pollutant levels and vital body statistics. The features of WSN have made it extremely important and relevant for use in IoT [4].

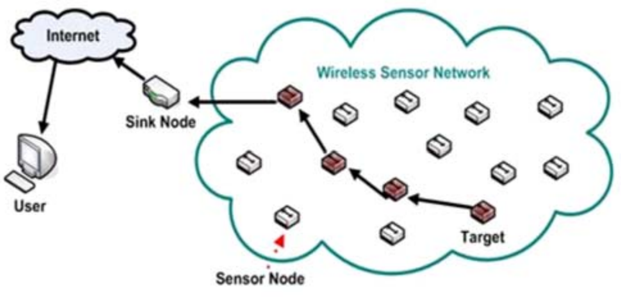


Figure 2.7: A wireless sensor network

2.2.4 Wi-Fi

Today's wireless technology uses radio waves, and is referred to by various names including: Wi-Fi, WLAN and Wireless LAN. Wi-Fi delivers wireless connection for both the internet and Local Area Networks (LANs). Many devices in the market today are Wi-Fi enabled; examples include: smartphones, video games consoles, personal computers digital cameras. They connect to the internet and network points to access resources. Wi-Fi is considered an IoT enabling technology since it provides how devices communicate.

2.2.5 Bluetooth Low Energy (Bluetooth LE, BLE)

A wireless personal area network technology already embedded in various mobile operating systems such iOS, Android, Windows Phone, BlackBerry, as well as OS X, Linux, and Windows 8 and 10. It is favoured over the traditional Bluetooth because of its energy efficiency usage and delivers equal data throughput as traditional Bluetooth. It is thus, a good technology driver for the Internet of Things.

2.2.6 ZigBee

A suite of high level communication protocols developed for wireless personal area networks targeted at low powered devices. It is based on the IEEE 802.15.4 standard with a transmission range of 10 to 100 metres. It is perceived as the global wireless standard which provides the foundation for the Internet of Things. It is embedded with the following features: self-configuring, self-healing system of redundant, low-power, low-cost and, in some instances, battery-free nodes.

2.3 Global Application of IoT

The Internet of Things provides for and enables many novel applications targeted at giving mankind better life style. From smart cities and home automation to wearables; IoT touches every aspect of human lives with possibilities for tremendous growth. A review of some notable IoT enabled applications currently being used in different sectors of the global economy is discussed below.

2.3.1 Smart Cities

IoT sensor nodes and applications are been deployed in cities around the world to help improve the quality of life style of the citizens. There are now urban and regional control systems based on wireless sensor networks, which help maintain and avert disruption of vital city services in these smart cities. By connecting the physical world to the Internet, IoT technology can improve resilience in areas like telecommunications systems, electricity systems, water supply systems, urban maintenance systems and traffic management systems. An example could be seen at La Garrotxa, Spain where Waspmote sensor nodes have been deployed, and these power three key application configurations that help measure parameters for forest fire prevention, river flood monitoring, air pollution and greenhouse gases [43].

Smart parking is another feature of a smart city. The smart parking system gives system integrators the opportunity to offer full parking management solutions to town management planners. The smart parking system provides precise information on vacant parking spaces while motorists save time and fuel. As a result, atmospheric pollution and congestion is reduced [44]. The smart parking sensor is designed to be concealed in parking spaces while sensing the arrival and exit of automobiles. Figure 2.8 illustrates smart parking in an urban city, and Figure 2.9 shows a smart sensor for detecting available space for parking.

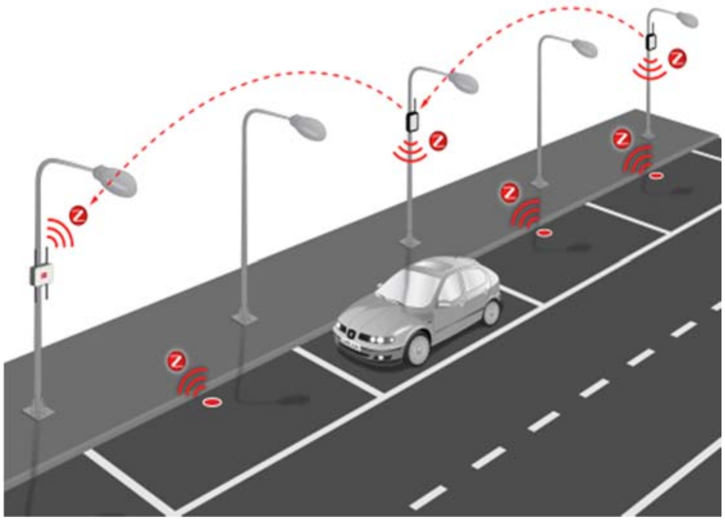


Figure 2.8: Smart parking in an urban city centre [44]

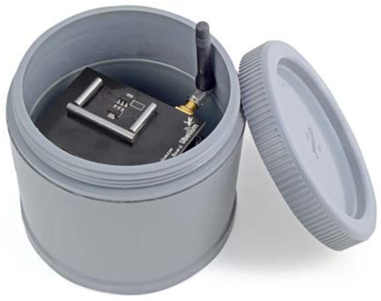


Figure 2.9: A sample IoT sensor node for detecting available space for smart parking [44]

2.3.2 Smart Environment

The World Health Organisation's (WHO) report [45] revealed that:

i. About 40 million people in the 115 biggest cities in the European Union (EU) are exposed to air pollution exceeding WHO's air quality guidelines. Children living close to roads with high vehicular traffic are at twice the risk of respiratory problems compared to those living in less congested streets.
ii. Transport is currently the fastest growing source of fossil-fuel emissions (Carbon Dioxide (CO_2)), the largest contributor to climate change, which contributes to air pollution and climate change.

In consequence to the above, some EU cities have undertaken to make their environments smarter to respond to the health hazards this portends. Some cities in Serbia have deployed Waspmotes (wireless sensor nodes) to control public transportation and observe environmental conditions [46].

2.3.3 Smart Water

Environmental pollution and degradation has placed numerous marine species in peril and annihilation, and this has brought about the destruction to earth's biodiversity. To conserve and replicate this nature's endowment a freshwater aquarium system was developed for the Amazon River [47]. A recreation of the Amazon River filled with over 20 piranhas (*Pygocentrus nattereri*) and 3 false piranhas or pacús (*Colossoma macropomum*) live in 13,300 litres of water. A smart water system was incorporated into it to measure the physical and chemical properties of the tank containing soft and acid waters, typical of the river type. Changes in the conditions of the aquatic environment can be detected in minutes, ensuring the quickest reaction of the aquarium team [47]. Figure 2.10 shows a freshwater aquarium with the piranhas inhabiting it while IoT sensor nodes monitor the vital physio-chemical components.

Figure 2.10: The freshwater aquarium is home to more than 20 piranhas [47]

2.3.4 Smart Metering

Modern electronic meters (smart meters) are now fitted with sensor nodes, which record more regular and more accurate electricity consumption information and have a two-way remote communication capability. Electricity consumption can be read remotely without giving bill estimates. Most homes in New Zealand are currently fitted with smart meters [48].

2.3.5 Security and Emergencies

Some countries with nuclear power plants have created radiation sensor boards to help measure the levels of radiation of any affected zones during nuclear spillages like the Fukushima nuclear disaster episode. This system helps in the early notification and evacuation of personnel without compromising the life of citizens and workers. An autonomous battery powered Geiger counter is deployed, which could read-off radiation levels automatically and periodically. Readings are instantly sent in real time using IoT enabling technologies such as ZigBee and GPRS.
The IoT enabled radiation sensor motes sleep often to save battery power. However, at specific intervals they wake up and perform readings of pulses, which are being generated in the Geiger tube while calculating the counts per minute. The values read are compared with established threshold limits. If the values are within acceptable limits they are sent using the ZigBee radio interface to the Gateway of the larger network and the values are stored in an Internet database.
However, with values higher than the threshold set, the values including the location coordinates (longitude and latitude) are immediately sent to the security control centre for action through the available interfaces such as the ZigBee network and the GPRS radio network [49].

2.3.6 Retail

IoT enabled sensors have been employed in various retail services to help monitor the condition and position of goods and vehicles. For example, bakery and pastry companies with fleets of vehicles could monitor the temperature and position of every delivery vehicle in real-time over the internet with a computer, tablet or smartphone. An illustration is shown in Figure 2.11.

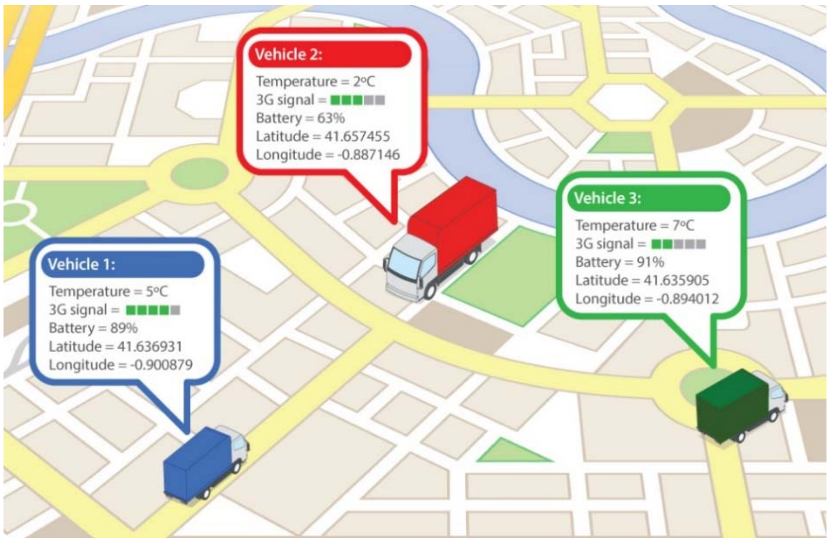


Figure 2.11: Smart fleet tracking [48]

2.3.7 Logistics

Every year, over 100 million shipping vessels circulate the world, moving goods across the seven seas of the world. Shipping vessels convey nearly 90% of global non-bulk cargo. Other modes of transportation such as the road, rail and air freight networks then deliver these goods to their specific locations. Management of cargo freight such as sea containers, train cargos are an essential element of world economic systems. The transport and distribution of goods across the globe enable business growth, and this in turn enable people access to required goods and consumer items. However, mishandling and mishaps with transport vessels cause major impact to economy, trades, workforces, societies and the environment [50].

IoT enabled sensor nodes are now being incorporated into cargo containers to allow cargo owners, transportation companies, and end-receiving customers determine where goods are located at every stage of the vessel’s movement. For sensitive goods transportation, such as food items, medicines, precious metals and humanitarian aids, this information is critical. GPS sensors allow for positional monitoring while GPRS/3G sends alerts thus, demonstrating the application and importance of IoT in cargo and vehicle tracking. Travel can be programmed to define the routes they ought to follow into the devices and comparison made with actual path being travelled. In this way, SMS alerts can be sent to alert the vessel control centre of any diversions or emergencies. The use of the IoT enabled sensing tracking devices helps with the consistent tracking of goods throughout the transport chain, discovery of unanticipated container openings, tracking of transport conditions and the identification of storage inconsistencies. Figure 2.12 demonstrates a sensor based tracking of goods in a transportation based system.

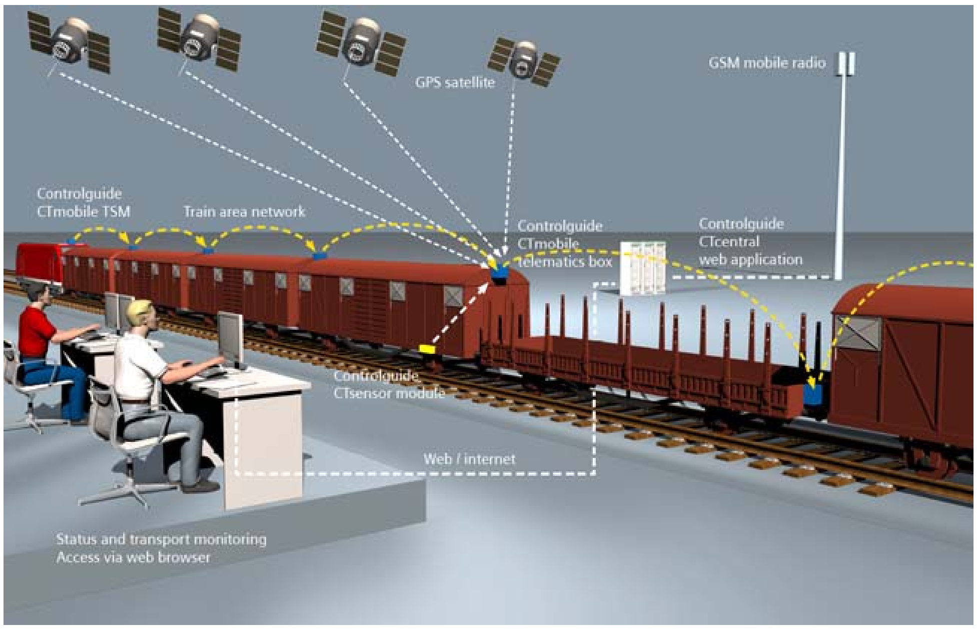

GPS satellite
GSM mobile radio
Controlguide CTmobile TSM
Train area network
Controlguide CTmobile telematics box
Controlguide CTcentral web application
Controlguide CTsensor module
Web / internet
Status and transport monitoring Access via web browser

Figure 2.12: Sensor based tracking of goods throughout the transportation chain [50]

2.3.8 eHealth

The design, development and advancement of wearable eHealth technologies for health monitoring has created a lot of buzz in the research fraternity and the health sector recently. This was propelled by the rising healthcare costs and advances in sensor technologies. The increasing advancement of wearable sensor-based technologies will eventually revolutionize the future of healthcare by eventually creating a real-time health management and pervasive monitoring of a patient's health condition. An example can be seen with the e-Health Sensor platform developed by Cooking Hacks [51], which allows Arduino and Raspberry Pi users to implement biometric and medical applications where body monitoring is needed by utilizing different sensors including: airflow (breathing), pulse, body temperature, oxygen in blood (SPO2), electrocardiogram (ECG), glucometer, galvanic skin response (GSR-sweating), blood pressure (sphygmomanometer) and patient position (accelerometer). Figure 2.13 shows a typical implementation of an IoT enabled eHealth monitoring system for a patient.

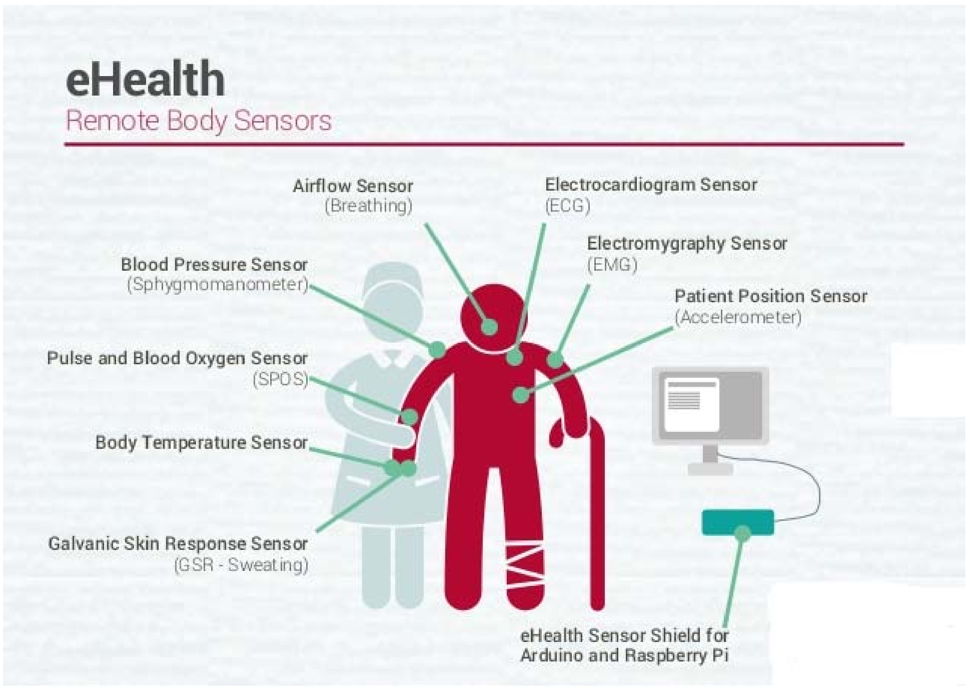

eHealth
Remote Body Sensors
Airflow Sensor (Breathing)
Electrocardiogram Sensor (ECG)
Electromygraphy Sensor (EMG)
Blood Pressure Sensor (Sphygmomanometer)
Patient Position Sensor (Accelerometer)
Pulse and Blood Oxygen Sensor (SPO2)
Body Temperature Sensor
Galvanic Skin Response Sensor (GSR - Sweating)
eHealth Sensor Shield for Arduino and Raspberry Pi

Figure 2.13: Real-time eHealth monitoring of patient [52]

2.3.9 Smart Agriculture

Researchers in Indonesia discovered factors causing the dwindling cocoa production which included effects of climatic changes, aging cocoa trees exposed to pests and diseases, and insufficient expertise on cocoa management during planting. In addition, the Indonesian cocoa farms and research stations are sparsely located which required researchers to travel for days for data collection in such fields. This was quite tasking and daunting. Researchers, plant scientists and agronomists converged to define a set of improved breeding and agronomic practices. They sought for a way to work together through laboratory and field-based tests in the cocoa fields in Indonesia [53].

However, IoT technology was employed to solve the fundamental challenges of access and remote data collection via remote monitoring systems. With the IoT enabled system, parameters were measured and transmitted to the field research station. Data collected include temperature, humidity, photo-synthetically active radiation (PAR), soil water potential (Figures 2.14 and 2.15 show the pictorial view of the IoT sensor node deployment for the cocoa field). Data collected was used in the creation of pest-resistant cocoa clones, and triggered a knowledge exchange, which helped in the rehabilitation of aged and unproductive cocoa trees and the prevention of deforestation for ecological conservation [53].

Figure 2.14: IoT sensor devices for remote crop management monitoring [53]

Figure 2.15: IoT enabled weather station monitoring sensors [53]

2.4 Summary

The Internet of Things, although an evolving paradigm, it however, has many numerous industrial and life applications. This Chapter has discussed important IoT application concepts. In particular, this study explored IoT architectures put forward by the research fraternity and various IoT applications. A review of the various enabling IoT technologies were also presented.

A unique characteristic of IoT sensor nodes is there resource-constrained nature. Nevertheless, they can communicate using an open wireless medium, and operate in a dynamic and mobile network topology. They also operate in infrastructure-less distributed network system that traverse many heterogeneous networks. This however, reveals some fundamental security issues during network communication among the sensor nodes. In Chapter 3, the study further explores how these nodes communicate within the network and understudies the security of the network communication among the IoT devices. Specifically, the security issues during routing among IoT sensor nodes are explored.

3 SECURE ROUTING IN INTERNET OF THINGS

A fundamental aspect of the Internet of Things is the manner low powered devices self-organize and share information (route and data information) among themselves. These sensor devices are constrained in terms of power and have limited processing capabilities hence, the need for them to be energy-efficient while routing data within the network.
IoT has found its application in many areas of the economy ranging from agriculture, building and management automation, VANETs, urban networks, industrial smart grids systems, water grids and smart cities. They perform storage and computational functions while communicating over lossy channels. These nodes work in unison though they can join and leave the network at any time. It is of importance that the wireless routing solution for these sensor networks should be scalable, autonomous in addition to being energy-efficient. The devices utilized in these low power lossy networks (LLN) are basically sensors and actuators with routing capabilities. Some of these sensor nodes act as border routers and hence, connect the LLNs to the internet or to a closely located Local Area Network (LAN). Such routers are commonly referred to as LLN border routers (LBR) [54, 55]. Figure 3.1 shows a layered IPv6 architecture of the end-to-end connectivity covering a field area network.
The findings reported in this Chapter (Chapters 1 and 2 inclusive) have been presented in four peer reviewed journal and conference papers [2, 29, 56, 57]

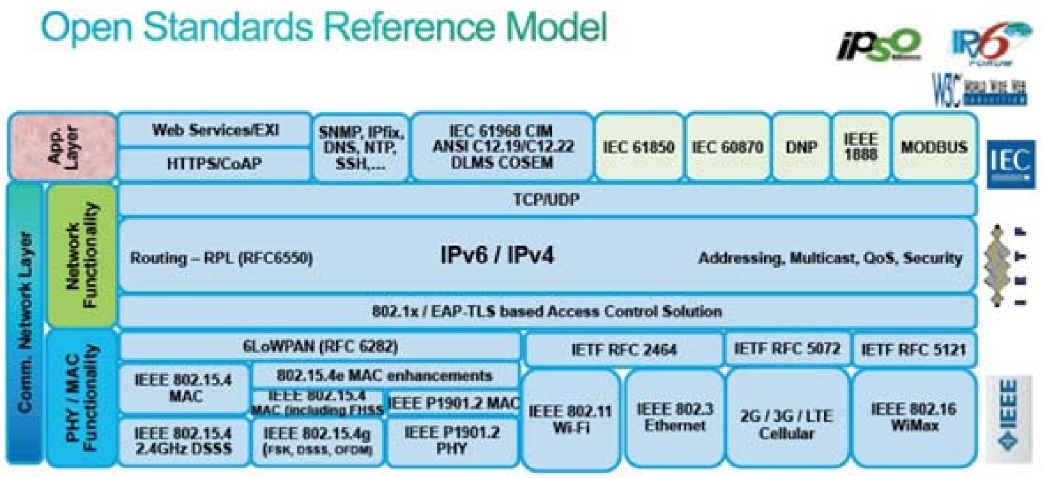


Figure 3.1: A Layered IPv6 architecture showing end-to-end connectivity covering [58]

3.1 IoT Routing Protocols

The Internet Engineering Task Force (IETF) created working groups (WGs) which developed various IoT protocols for IoT devices. A discussion is presented below of the IETF protocols which have been developed for the Internet of Things (IoT) and a review of the weaknesses inherent in these protocols.

3.1.1 IPv6 over Low Power Personal Area Networks (6LoWPAN)

6LoWPAN is an IETF-standardized IPv6 adaptation layer (data link and cross-layer protocol) that enables IP connectivity over low power and lossy networks [23, 32]. This is the foundation for the network build up for the Internet of Things such as smart homes, smart cities and industrial control systems [27]. Many applications utilize 6LoWPAN for IP-based communication through an upper layer protocol such as the RPL routing protocol. 6LoWPAN essentially adjusts IPv6 packets into frames of 127 bytes – a frame size requirement that low power sensor devices can utilize among themselves. Also, 6LoWPAN supports the transmission of large-sized IPv6 packets on the data link layer of the IEEE 802.15.4. 6LoWPAN provides fragmentation support at the adaptation layer although, the system of fragmentation makes processes such as buffering, forwarding and processing of fragmented packets resource expensive on these already resource-constrained devices. Rogue nodes can send duplicate, overlapping or stale fragments to disrupt the network. A security breach can be seen in this layer as there is no authentication at the 6LoWPAN layer, hence receiving nodes are unable to differentiate between legitimate and spurious packets during fragment re-assembly. Receiving nodes during re-assembly normally store up the fragments received so they could re-assemble them. If the entire set of frames making up the packet are received after a certain timeout they are discarded. This system could also be exploited by malicious nodes which could send fake fragments to fill up the node's store, so it does not receive the legitimate fragments for re-assembly. This is indeed a challenging security issue in IoT networks [59]. However, some protocols which have adopted 6LoWPAN [60-62] hinge on the security sublayer of the IEEE 802.15.4 to prevent 802.15.4 frames being introduced by malicious nodes. Indeed, the 802.15.4 security sublayer actively achieves this aim by adding to every frame a Message Integrity Code (MIC) and a frame counter.

The design of an 802.15.4 frame includes an unspecified key for security purposes though the purpose of the key is uncertain thus, utilising this key to preload each node with a shared-key for the network exposes the nodes to network attacks. An attacker could physically interfere with a node to decipher its cryptographic details. Tamper-resistant hardware could be used to forestall this type of attack [63], but it comes at a high cost even though it does not guarantee absolute security [64]. Once a node has been compromised the attacker could easily inject spurious frames into the network and thus, add other non-authorised nodes to the victim's network. This error and security loophole is carried even to the upper layer protocols since they rely on the 802.15.4 security sublayer for the security of frames [61, 62].

3.1.2 Routing Protocol for Low Power and Lossy Networks (RPL)

RPL was developed by the IETF working group [ROLL WG] as routing functionalities in 6LoWPAN were very challenging due to the unique characteristics of IoT nodes. RPL operates at the network layer hence, it permits routing across multiple types of link layers. Operating at the network layer, RPL quickly builds up the routes and efficiently distributes route information among other nodes in the IoT network [65]. RPL is a proactive Distance Vector IPv6 routing protocol for LLNs, which organises its network path information using a set of Directed Acyclic Graphs (DAGs). When RPL is initiated, it forms an inverted tree-like topology known as Destination Oriented Direct Acyclic Graph (DODAG). A DODAG typically consist of sending sensor node(s) and receiving sink node(s) as shown in Figure 3.2. Every DODAG is distinguished by four factors which include: DODAG ID, DODAG version number, RPL instance ID and Rank. DODAG sink nodes are linked to each other [62]. Route selection in RPL depends on the DODAG link, cost of information to a node such as workload, throughput, node power, latency or reliability. To produce a route topology, every node selects a set of parents that comprises nodes with equal or better paths towards the sink. The node with the best route link is chosen as the parent. RPL employs three types of control messages to form and manage routing of information in the network and these are:

i. DODAG Information Object (DIO), used for setting up and updating the network topology.

ii. DODAG Advertisement Object (DAO) used for spreading and advertising destination information upwards during network route updates.

iii. DODAG Information Solicitation (DIS) used when a new node seeks topology information while waiting to join the network.

DAO and DIS are involved during a topology change process while the DIO message is broadcast and mainly used for the purpose of starting a topology change process. DIO is commonly used to distribute its routing state to other nodes using its Rank (Rank specifies the link quality to a sink node) and objective function (OF) [62, 65]. Every node computes its Rank according to the Rank of its selected parent and the objective function. A DIO message is sent to all nodes every time a node updates its Rank or preferred parent. To prevent the formation of loops, RPL utilizes the Rank rule whereby a node in a parent should always have lower Rank than its children. Also, to limit the amount of data broadcast, RPL uses the trickle algorithm for scheduling DIO messages to be sent. It does this by setting a counter, which observes the network topology, and thereby decides when a node should send a DIO message. For every DIO message received without comparing it with the previous DIO message this will cause the DIO counter to increase and if the DIO counter reaches a threshold value (redundancy value) the node will reset its DIO counter and double the trickle time. This is done to stabilize the network topology over a period of time and avoid the unnecessary frequent route updates which could consume the limited battery power and bandwidth. This further helps to limit the number of DIOs produced so as to preserve scarce network resources. For incoming traffic, the node resets its DIO to zero and reduces its trigger time. This gives the opportunity for quick network route update through a rapid DIO generation [62].

The RPL routing protocol has capacity to incorporate different types of traffic and signalling information swapped among nodes although this depends on the requirements of the considered data flows. RPL supports the Multipoint-to-Point (MP2P), Point-to-Multipoint (P2MP) and Point-to-Point (P2P) traffic types [13, 66].

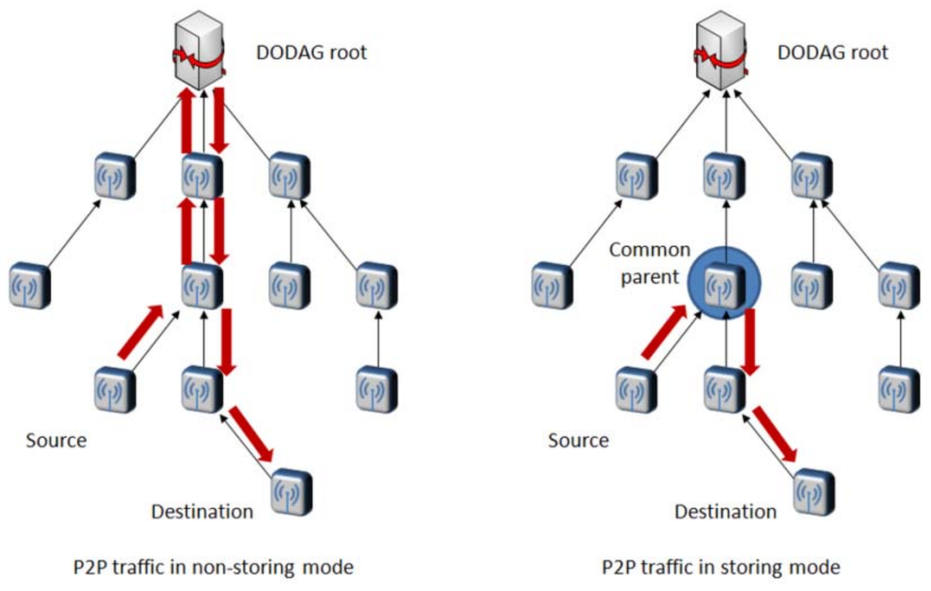


Figure 3.2: A RPL network showing the flow of packets in a point-to-point traffic between two nodes

The Rank property is central to the efficient routing operations of the RPL routing protocol. The Rank property helps to manage control overhead, prevent loop formation and create optimal network topologies. So, any attack at the Rank property will severely disrupt the proper functioning of the RPL protocol. RPL presumes nodes in the network are consistent and follow the protocol rules thus, it does not provide a system for examining consistent node behaviour and this creates the opportunity for malicious nodes to attack the Rank property [65]. RPL is also susceptible to the HELLO message attack. This is a broadcast message a node sends out when trying to join a network. These malicious nodes usually have strong broadcast signal to reach other nodes, but they are distant from their victims. A malicious node advertises itself with a perceived good routing metric, but when the victim node seeks to align with it, its request messages get lost because it is out of range from the malicious node. This process continues until the victim node exhausts its battery power while trying to connect with the malicious node. DIO messages used to advertise DODAGs information could be used by these malicious nodes to succinctly launch HELLO flood attacks. However, with link-layer security turned on this could be averted. Nevertheless, malicious nodes realising this, try to compromise one of the legitimate nodes in the network in order to launch the hello flood attack regardless [67].

Another attack RPL is susceptible to is the Rank attack. The RPL protocol gives numerous details to nodes in the DODAG so it could determine the node that will act as the default route. One such detail is the Rank, this is computed and sent out to the neighbouring nodes from the DODAG root. A malicious node simply advertises a better Rank to other nodes, which attracts other unsuspecting nodes so as to become their parent in the DODAG. Although RPL uses the link-layer quality to compute routes this makes a Rank attack less effective in RPL but the attacks are still possible [65, 67].

3.1.3 IPv6 over the Time Slotted Channel Hopping mode of IEEE 802.15.4e (6TiSCH)

The development of this IoT protocol is currently ongoing and has not been deployed yet. It will be based on IPv6's multi-link subnet spanning over high speed IEEE 802.15.4e TiSCH wireless mesh networks linked to the backbone via synchronized backbone routers [13, 66]. The new protocol will include details about how packets, belonging to a deterministic IPv6 flow, may be treated while issues such as classification, routing and forwarding of packets over the mesh network can be addressed. Other areas to be addressed will include security, link management for the IPv6 network layer, neighbour discovery and routing (6TiSCHesIoT) [66, 68].

3.2 IoT Application Protocols

Today, IoT enabled devices have embedded application protocols that handle communication between the IoT gateways, the final application and the internet. These application protocols, update connected servers with the recent values of the end-devices. They in addition, transmit commands from the applications to end-device actuators which perform specific actions. Figure 3.3 depicts an architectural view of the interplay of IoT application protocols. A profile overview of the application layer protocols driving such communication include EXtensible Messaging and Presence Protocol (XMPP), Constrained Application Protocol (CoAP), Message Queue Telemetry Transport (MQTT), Data Distribution Service (DDS), HyperText Transfer Protocol/REpresentational State Transfer ((HTTP/REST), Universal Plug and Play (UPnP), ZeroMQ, Continua (Home Health Devices), Device Profile for Web Services (DPWS) Schema) [13, 66, 69, 70]. Although these protocols overlap in functionality in a minimal way, they however, target specific functions in IoT. MQTT performs device data collection. XMPP is used for consumer dependent applications since it deals with security, scalability and addressing. CoAP delivers a request/response communication model between application endpoints and provides discovery of services required by clients. In the next section a brief description is given of three main application protocols used by IoT devices while table 3.1 provides a summary of various IoT application protocols, security level and architecture.

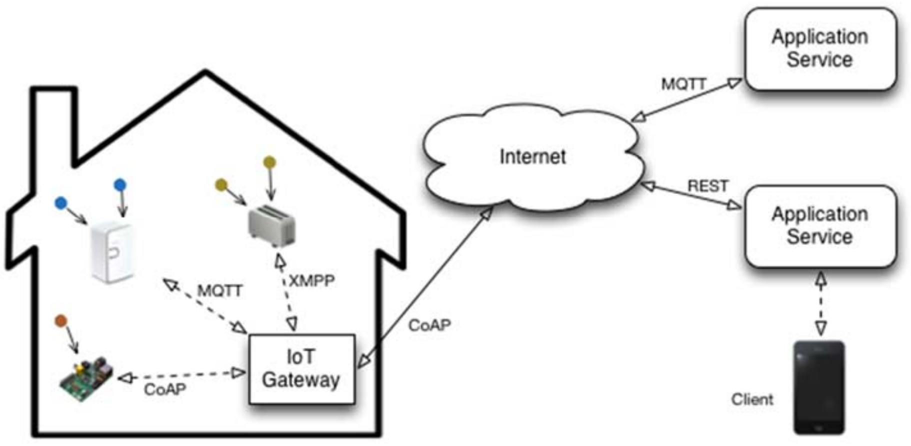


Figure 3.3: Architectural view of the interplay of IoT application protocols [71]

Table 3.1: IoT application protocol features

Protocol	Transport	Messaging	2G, 3G,4G (1000's)	Low Power and Lossy (1000's)	Compute Resources	Security	Success Stories	Arch.
CoAP	UDP	Request/Response	Excellent	Excellent	10Ks/RAM Flash	Medium - Optional	Utility field networks	Tree
Continua HDP	UDP	Publish/Subscribe	Fair	Fair	10Ks/RAM Flash	None	Medical	Star
DDS	UDP	Publish/Subscribe Request/Response	Fair	Poor	10Ks/RAM Flash +++	High - Optional	Military	Bus
DPWS	TCP		Good	Fair	10Ks/RAM Flash ++	High - Optional	Web servers	Client server
HTTP/REST	TCP	Request/Response	Excellent	Fair	10Ks/RAM Flash	Low - Optional	Smart energy phase 2	Client server
MQTT	TCP	Publish/Subscribe Request/Response	Excellent	Good	10Ks/RAM Flash	Medium - Optional	IoT messaging	Tree
SNMP	UDP		Excellent	Fair	10Ks/RAM Flash	High - Optional	Network monitoring	Client server
UPnP		Publish/Subscribe Request/Response	Excellent	Good	10Ks/RAM Flash	None	Consumer	P2P client server
XMPP	TCP	Publish/Subscribe Request/Response	Excellent	Fair	10Ks/RAM Flash	Medium – Mandatory	Remote Management	Client server
ZeroMQ	UDP	Publish/Subscribe Request/Response	Fair	Fair	10Ks/RAM Flash	High - Optional	CERN	P2P

3.2.1 EXtensible Messaging and Presence Protocol (XMPP)

This is a TCP communication protocol grounded on XML. It permits a very close real-time exchange of controlled data among connected devices. A good feature of XMPP include management of contact list and presence information. Although the two features were initially intended for instant messaging, it is quite clear they could be applied to IoT. Furthermore, since XMPP is open-source, it is now being adapted in publish-subscribe systems, which makes it ideal for use in IoT.

3.2.2 Constrained Application Protocol (CoAP)

CoAP is an application layer or software protocol developed by the "Constrained RESTful Environments" (CoRE) working group of the IETF to let resource-constrained devices communicate over the Internet using User Datagram Protocol (UDP) rather than TCP. It is a simple request/response protocol (again, very similar to REST) which resembles the client/server model. Clients could perform GET, PUT, POST, and DELETE requests to resources. CoAP packets make wide use of mappings of strings and integers to minimize the size of packets transmitted and displayed on devices. In addition, it utilizes bit-fields to optimize memory efficiency. Figure 3.4 shows the topological view of the CoAP layers as proposed by IETF.

```
+----------------------+
|      Application     |
+----------------------+
+----------------------+
|  Requests/Responses  |
|----------------------|  CoAP
|       Messages       |
+----------------------+
+----------------------+
|          UDP         |
+----------------------+
```

Figure 3.4: An IETF layered representation of CoAP [69]

3.2.3 Message Queue Telemetry Transport (MQTT)

This is an application layer protocol developed specifically for resource constrained IoT devices. MQTT was developed as a purely lightweight publish/subscribe messaging transport. The idea behind MQTT is simple. Physical sensor devices could exchange raw data amongst one another or they could be remotely controlled by another device. An example of such a scenario could be the interconnectivity of a temperature sensor, mobile phone and a laptop which have been fitted with various sensor devices. A common use in this scenario could be the assembly, broadcast, merging and displaying of sensor data while an alarm or an actuator could be triggered when a certain temperature threshold value is reached as shown in Figure 3.5. MQTT is an application layer protocol based on 6LoWPAN [13, 66].

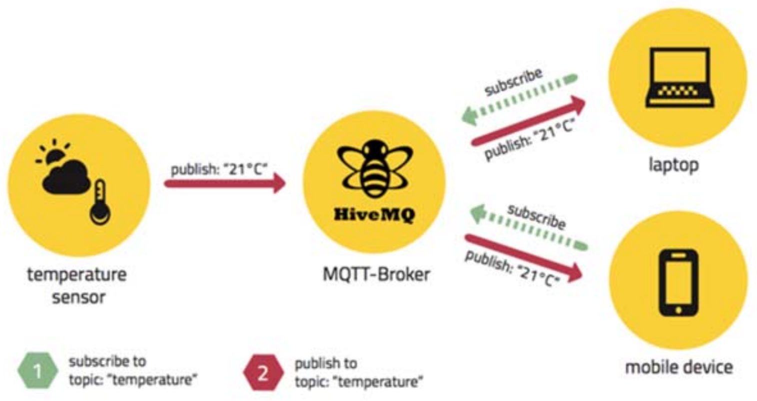


Figure 3.5: An MQTT publish/subscribe topology

3.3 Security in IoT: Where the Need Lies

IoT has many promising areas of application including commercial (oil well sensing, intelligent vehicular transportation system, gaming, and agriculture), smart homes, wearables, healthcare, automotive industries and the power smart grid system. IoT exhibit unique characteristics, which requires security while in operation to protect the network from various attacks. The fundamental requirements ensuring the security of any IoT network remains a challenge. To achieve the goal of having a secure IoT network, the CIA triad (Confidentiality, Integrity and Availability) in addition to some other properties must be considered and are discussed below [72].

3.3.1 Confidentiality

Confidentiality guarantees information does not get divulged to the wrong source. In ad hoc networks, it ensures malicious nodes do not gain unauthorized access to vital routing or data information either from any legitimate node or while such information is in transit. Confidentiality imposes a prohibition on untrusted nodes from comprehending and accessing the content of vital data being communicated. In IoT, to protect the confidentiality of information transmission between nodes, the encryption of routing data is important to provide stronger safety measures during network communication.

3.3.2 Integrity

This is the assurance that data received by a destination node has not been changed in transit, either through collision or via a deliberate tampering by an untrusted node, and that the data received was as originally sent. In some instances, data packets could suffer from collision due to radio wave propagation; equally, data packets could still be modified by untrusted nodes to disrupt the network. In IoT networks, data integrity should be embedded in the design since an IoT device collects, stores, sends, and shares data according to a given protocol standard.

3.3.3 Availability

Availability is the provisioning of network services at all layers of a network to all nodes while ensuring the survivability of all network services even in the presence of malicious nodes. Since IoT will be employed in crucial and important areas of global economy, security from the perspective of availability will be of top priority.

3.3.4 Authenticity

A process whereby nodes are required to identify themselves and prove their identities on the network. This is required to protect the security of the network from impersonating nodes who could disrupt the network or gain access to vital information and hence, disrupt the network system. Since many nodes will be communicating in a heterogeneous fashion, node authentication is necessary to avert illegal node access to the network.

3.3.5 Non-repudiation

Non-repudiation involves a source node owning up to data it has sent while a receiving node acknowledging the receipt of the same. Neither party can deny knowledge of either sending or receiving the information. Non-repudiation is essential in detecting and isolating untrusted IoT nodes that may seek to send false network information while seeking to deny they ever sent such information. A node sending erroneous or false information could prove the source of such message and thus, detect and avoid the node perpetrating such an attack. In addition, trust associated with the broadcast of updates, which are initiated from distant nodes, creates a cause for concern; this however, can be addressed through developing a secure trust system and enforcing non-repudiation among distant IoT node routers.

IoT uncovers new aspects of security challenges in the underlying network topology, and the heterogeneity of IoT networks makes them even more susceptible to malicious attacks than in wired networks. The vulnerability of communication channels, mobility, and the resource-constrained properties of nodes make IoT security a daunting task to deal with [73]. Issues like, eavesdropping on wireless broadcast of messages, and injection of false information into the network greatly compromise the integrity of IoT communication. Moreover, due to the constrained nature and self-organizing attribute of IoT sensor nodes, the use of a solution centred on certification authorities (CA) via connected servers poses extreme difficulty for secure routing among IoT nodes [74-76].
According to Gartner, by 2020 the number of interconnected IoT devices is expected to reach 25 billion [77] and further research from HP highlights that on an average there are approximately 25 vulnerabilities per IoT device, which supports the requirement for better IoT security [70]. A summary of HP's findings is presented below:

i. Privacy Issues: Huge number of IoT devices gather sensitive and private information, like name, address, and insurance policy number etc., of users. An example is in the health sector where IoT nodes collect and transmit some form of personal information such as name, address, date of birth and health statistics. These concerns become even more accentuated when details are transferred and deployed unto the cloud using mobile applications, which work with these IoT devices. Transmission of this ultra-sensitive information across the IoT networks, without adequate security measures, is a huge concern, as it is possible for unauthorized personnel to access the information.
ii. Inadequate Authentication/Authorization: HP surveyed multiple IoT devices (webcams, TVs, home thermostats, remote power outlets, home alarms, door locks, garage door openers, and scales) present in the market and found out that they either do not require strong passwords for accessing them or may have poor password recovery systems. In addition, several such devices utilize similar insecure passwords on their websites and/or mobile applications, enabling a possibility for potential malicious software to remotely gain control of them.
iii. Absence of Transport Encryption/Standard: Since these IoT devices collect and transmit confidential data it is imperative for a Transport encryption system to be

in place (an encryption system in place while transmitting data over IoT networks). However, the HP research showed that most of the devices did not encrypt network data transmission for both local network data and the Internet. This is due largely to the lack of standardization in the IoT framework.

iv. Web Interface Vulnerability: This is a security vulnerability in web applications used by hackers to circumvent access controls. HP, in its report, identified recurrent cross-site scripting, vulnerable weak sessions and poor credentials management as severe security issues. Bearing in mind that most of these devices provide access through the cloud, these become notable security issues.

v. Software and Firmware Vulnerability: As identified by HP, more than 60% of the IoT devices have software and firmware vulnerabilities present in them resulting from lack of encryption standards while upgrading the software and firmware. This proves that malicious software and firmware could gain remote access to these devices through system updates. Figure 3.6 summarise the above-mentioned findings by HP.

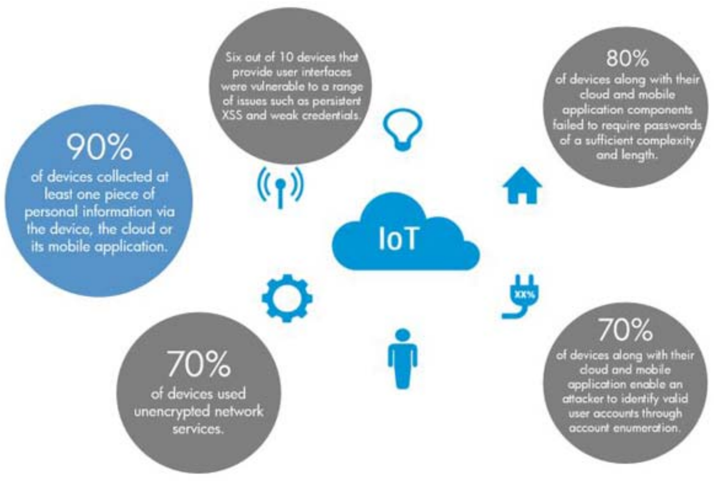

Six out of 10 devices that provide user interfaces were vulnerable to a range of issues such as persistent XSS and weak credentials.
80% of devices along with their cloud and mobile application components failed to require passwords of a sufficient complexity and length.
90% of devices collected at least one piece of personal information via the device, the cloud or its mobile application.
IoT
XX%
70% of devices used unencrypted network services.
70% of devices along with their cloud and mobile application enable an attacker to identify valid user accounts through account enumeration.

Figure 3.6: Security Vulnerability of IoT devices [78]

Privacy preservation for IoT devices and users is another key issue in IoT. Even with the existing authentication approaches and cryptographic mechanisms in place to safeguard the user's privacy in IoT networks, issues like heterogeneity of IoT networks, limited battery capacity of devices/nodes involved and resource constraints in terms of available memory, inhibit effective communication in IoT. Thus, multiple devices in IoT networks end up not utilizing in an optimal manner the available authentication and cryptographic mechanisms. This clearly shows the need for better security in IoT systems. The US

Federal Trade Commission (FTC) has already identified this and have announced the need to secure the IoT ecosystem after a security violation was reported for the TRENDNet IP camera in 2012, wherein live footage from thousands of TRENDNet security cameras where penetrated, permitting web users to access live video footage without requiring any password [79]. The European Union Data Protection WP29 committee, which was saddled with the responsibility of developing a working party on the protection of individuals with regard to the processing of personal data, also captured in their report the need for privacy preservation among IoT devices and users [80].
Generally speaking, security threats in IoT networks could be classified into the following two groups:

a) *General security threats in IoT networks*: such threats are comparable to those occurring in traditional network systems due to issues like confidentiality, integrity, and availability (CIA) and they include: DoS attacks, and man-in-the-middle attacks (MITM). However, in view of the massive size, complexity, and magnitude of IoT networks along with the heterogeneity of the underlying communication networks, IoT threats are bigger challenges compared to traditional network systems.

b) *IoT specific security threats*: this is largely because of the massive interconnectivity of different types of IoT devices and the heterogeneity of the underlying networks. These threats are specific to the ways IoT systems interact with our daily lives. For example, when measuring and exchanging sensitive private data, like a patient's medical readings (e.g., heart beat rate, blood pressure, temperature) or smart meter readings or eco-forest readings, over the IoT communication networks, the security of the data and the exchanging IoT devices may be compromised. The data may be hacked or it may be wrongly transmitted to rogue IoT devices.

IoT is still evolving and having appropriate security measures, perhaps in the form of a framework. It is imperative to address the conglomerate of security challenges that may come with it. The framework should define proper information gathering mechanisms from the associated IoT devices/nodes, a proper data privacy definition system, and further consider the type and nature of the underlying communication between the connected devices for which limited energy is a constraint. Albeit, such a framework can be useful for IoT networks, as it could help in protecting private data from being compromised by rogue and malicious nodes while giving users the assurance that their information is not being divulged to the wrong party.

3.4 Routing Vulnerabilities in IoT

IoT is firmly based on the use of the IPv6 addressing scheme. This makes it exposed to the same attack threats as IPv4. Some attacks include: Blackhole attacks, reconnaissance, Sybil attacks, spoofing, fragmentation attacks, smurfing, eavesdropping, neighbour discovery attacks, man-in-the-middle attacks, rogue devices etc. This clearly demands the same security measures being used today for IPv4. In addition, since IoT is envisioned as the intersection of where the Internet meets with the physical world, it further unlocks a whole new vista of security concerns. In view of this, attack threats will shift from the manipulation of information to the actual control of actuating devices, in other words, moving threats from the cyber world to the physical world. Accordingly, this radically creates a wide and fertile attack surface from well-known threats and devices, to the added security threats of new devices, protocols, and workflows (the porting of electronic devices from closed systems such as Modbus, SCADA into IP-based systems). This will further increase the risk of more attacks.

3.5 Threats Associated with Routing Protocols

Data communication among IoT devices could be achieved based on an end-to-end or on a hop-by-hop basis. To enshrine global acceptance and boost public confidence that their data and transactions will not be compromised, IoT networks require adequate security. IoT networks, just like WSN and MANET have similar attributes and thus, face similar routing attacks such as the Blackhole attacks, and sinkhole attacks amongst others. A comprehensive study on these attacks have been covered by [81-89].

3.6 Attack Types in IoT Networks

Attacks in IoT networks could be viewed or categorised from different perspectives depending on whether the attacks are passive or active attacks. Also, they can be viewed from the perspective of whether the attack is from within the network (internal attack) or from outside the network (external attack).

3.6.1 Active Attacks

This form of attack involves specific actions performed by the adversary node(s) such as the replication, modification and/or deletion of exchanged data among the nodes. The essential aim here is to compromise and destabilize the network.

3.6.2 Passive Attacks

The form of attack involves only eavesdropping on the data that is being communicated over the network. Actions such as covert channels, traffic analysis, sniffing to compromise keys could be regarded as passive attacks [75, 76].The aim of some passive attacks is to save the battery life of some nodes hence, these nodes do not cooperate in routing activities. These types of nodes are referred to as misbehaving nodes [75].

3.6.3 External Attacks

Network attacks launched from outside the network or from a remote location from the network are referred to as an external attack. These attacks are essentially active attacks that subtly cause network congestion, stale route propagation, deny services in the network to legitimate users and cause the eventual network shutdown. External attacks are normally prevented by using standard security mechanisms, such as firewalls, encryption, and other cryptography based algorithms. For example an attack launched by a remote adversary will be classified as external [75, 76].

3.6.4 Internal Attacks

This is an attack from within a network from unsuspecting malicious nodes. Normally, these malicious nodes are part of the network. Internal attacks are typically more severe attacks, since the malicious nodes are within the network and are regarded as legitimate nodes hence, they enjoy the self-defending mechanism of the internal network which they could use to perpetrate their malicious activities while going undetected [75, 76].

3.7 Routing Attacks in IoT

For a route to be established in a wireless mobile network, route information is transmitted from node to node (multi-hopping) until the desired destination is found. Throughout the route maintenance phase nodes can add, delete or needlessly delay the transmission of control information (selfish or misbehaving nodes). It is during this route discovery or route forwarding that malicious nodes propagate their activities; thus, several types of attacks are possible in the routing of information. As an example, a node can introduce a *routing table overflow attack* by transmitting a large amount of false route information to its neighbours in a manner that will cause its neighbours routing table to overflow. This action causes the neighbour’s routing table to be occupied by spurious routes thus, denying the real routes from been captured in the routing table. Also, malicious nodes can advertise fabricated routes for neighbour nodes to update their routes in order to poison their routing cache. In AODV, which is an ad hoc routing protocol, a malicious node can advertise a false route with the smallest hop count and with the latest sequence number, hence, other nodes seeing this as a route update quickly invalidate their old route to accept the new false route. Furthermore, in the route maintenance phase, a malicious node can transmit false route error messages, which can trigger the start of a costly route maintenance process [81]. In the network layer, several advanced routing attacks are possible, and a discussion of these attacks is presented below [81].

3.7.1 Blackhole Attacks

In Blackhole attacks, which normally occurs in the route discovery phase a node expresses its willingness to transmit a packet towards its destination; however, this is when a malicious node quickly intervenes to guarantee the route is established through it. Therefore, during the forwarding phase the malicious nodes instead of forwarding packets received to the next hop address, it rather drops them. A more serious form of Blackhole attack is, when the malicious node begins to alter the packets before forwarding them. The effect of this kind of attack is a low packet delivery ratio and false route advertisements.

3.7.2 Wormhole Attacks

This form of attack involves more than one malicious node. In a wormhole attack, the first attacking node creates a path with a second colluding node. Packets received by the first attacker are channelled to the colluding node which then sends the packet through the normal path but not without tinkering with the packets. The tunnel between the two conspiring attacking nodes is referred to as a wormhole. Wormhole attacks are considered severe threats to routing protocols in sensor networks because they hinder the discovery of any routes other than through the wormhole, and thereby compromise the routing information of the protocols involved as in Ad Hoc On-Demand Distance Vector (AODV) and Dynamic Source Routing (DSR) routing protocols [82]. This attack is extensible to IoT routing protocols such as 6LoWPAN and RPL.

3.7.3 Sybil Attacks

In sybil attacks, a node assumes multiple identities to give an impression that there are many malicious nodes colluding together to disrupt the network. The essence of this kind of attack is to create a state of confusion in the network and destabilizing the routing system thus, creating the opportunity for other malicious nodes to operate.

3.7.4 Greyhole Attacks

This is a modified form of the Blackhole attacks however, the attacking node drops data packets but transmits the routing control packets. This attack is difficult to detect as a route test will always show that there is an end-to-end connectivity but data packets are lost during transmission. A promiscuous network mode procedure within the routing protocol will help reveal the attack. Nevertheless, variants of this attack exist.

3.7.5 Sinkhole Attacks

In sinkhole attacks, a malicious or compromised node advertises false routing messages to make other unsuspecting nodes believe that optimal routes to a particular destination is via it. This causes the neighbour nodes to route their traffic through the malicious node. When the malicious node receives the traffic it immediately modifies various routing information including network secure data to complicate the network topological structure. Sinkhole attacks affect the efficient running of sensor network routing protocols such as AODV and DSR by manipulating and changing the sequence hop-count of these protocols thus, reporting incorrect hop-count of advertised routes. Through this manipulation and modification, the route presented via the malicious node appears to be the optimal route [83]. A variant of sinkhole attack is the Rank attack, which destabilizes the RPL protocol by advertising a wrong Rank value to unsuspecting neighbour nodes.

3.7.6 Selective Forwarding Attacks

This form of attack involves nodes selectively dropping packets meant to be transmitted. In a way, it looks like the Blackhole attack however, in a selective forwarding attack, the rogue node in a bid to avoid detection, it selectively sends and drops packets. This form of attack degrades the performance of networks [84].

3.7.7 Hello Flood Attacks

During route discovery, some sensor network routing protocols broadcast their links to other nodes. Nodes which get these broadcasts assume that such nodes are close by hence, are regarded as neighbour nodes. Malicious nodes with high transmission power, such as laptops could sway nodes in the network to believe it is near and when these unsuspecting nodes update their route cache they begin to send route packets. But a node sending its packets to the adversary node will be wasting its packets and loosing battery life as the adversary node is far away. This leaves the network in a routing loop. Routing protocols which rely on the exchange of information locally between nodes for route and topology maintenance are primarily affected [84]. Figure 3.7 shows an attacker broadcasting hello packets with a higher transmission power than the base station while Figure 3.8 shows the unsuspecting nodes taking the rogue node as a neighbour hence, sending traffic to it [85].

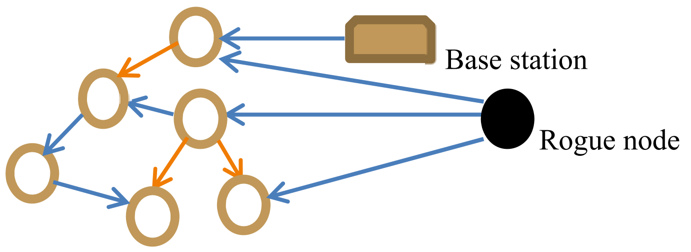


Figure 3.7: Attacker with strong signal broadcasting hello packets

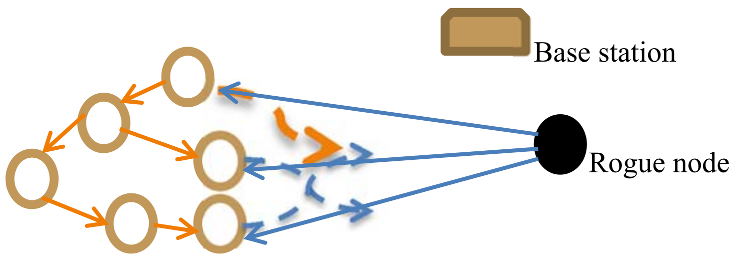


Figure 3.8: Sensor nodes entering infinite loop while trying to take rogue node as neighbour

3.7.8 Neighbour Attacks

This is a RPL attack in which the attacking node with strong transmission power broadcasts DIO messages while omitting DIO details (DODAG Preference, version Number, Rank, RPLInstanceID, Destination Advertisement Trigger Sequence Number (DTSN)) in the broadcast message. A neighbour which receives this broadcast assumes a new node has joined the network which could possibly be a parent node; thus, when it tries to select it as a parent node, the node is out of reach. This is also considered a variant of flooding attacks.

3.7.9 Local Repair Attacks

This attack occurs mostly on RPL where an attacker periodically sends local repair messages to other nodes even though the link quality is good. The incessant local repair messages force the neighbour nodes to initiate a route repair on the network thus, destabilizing the network. This form of attack impacts the packet delivery ratio in the network, produces more end-to-end delay and significant packet drops during route topology formation. The incessant local repair messages generated by the attacker also leads to energy depletion of neighbouring sensor nodes using battery.

3.7.10 DODAG Information Solicitation (DIS) Attacks

When a new node seeks to join the RPL network it sends a DIS message to get the topology information from neighbour nodes who could become its potential parent. In DIS attack, an attacking node sends DIS messages to its neighbours, and this in turn makes the neighbour nodes reset their DIO timer with the assumption that there is a new node desiring to join the network. Thus, the neighbour nodes send DIO messages to the new node (attacking) indicating their willingness to accept it into the network. However, the malicious node continues broadcasting DIS messages, which eventually leads to the resource exhaustion of neighbour nodes. Although, this kind of attack does not impact on the packet delivery ratio, it does increase end-to-end delay and eventual battery exhaustion of neighbour nodes around the attacker.

3.7.11 Version Number Attacks

In RPL, the version number is employed by the sink node (root) to regulate the global repair process of a RPL topology besides guaranteeing that the routing states of all nodes in the DODAG are current. Each DIO message conveys the version number, so that receiving nodes which were part of an outdated DODAG topology could re-join the new network by re-computing and updating their stored version number. During RPL global rebuild process two or more DODAGs could temporarily coexist. To ensure a loop-free topology, packets from the old DODAG are permitted in the new DODAG topology. In the new DODAG, the version number is propagated unchanged throughout the DODAG to avoid irregularities in the network. However, in RPL there is no mechanism to confirm if the integrity of the version number is preserved in the received DIO messages.

An attacking node could alter the version number field of its own DIO messages and transmit same to neighbour nodes to disrupt the network. Neighbour nodes receiving this DIO embedded with the new, but malicious version number now reset their trickle timer and update the version number contained in their records. They further broadcast this new and malicious version number through their DIO messages to their neighbour nodes. As a result of this, the spurious version number contained in the DIO packet is propagated throughout the network. The manipulation of the version number in the DIO packets initiates an inefficient rebuild of the DODAG with routing loops. Additionally, since the rebuild of the DODAG topology was not initiated by the root node (sink) the topology becomes non-acyclic thus, giving way for loops to occur. A significant effect of this attack is the disruption of the network topology, node energy depletion, routing loops and node communication channel availability.

3.7.12 Rank Attacks

In RPL protocol, the Rank of a node is a determining factor during the selection of parents and routes. Each node in RPL creates and uses Rank as its peculiar parameter for selecting and maintaining the best route. The Rank of a node increases in a downward direction of the network topology (from the root node to the child nodes) and decreases in an upward fashion from the bottom up (from the child nodes to the root node). The Rank mechanism is employed in RPL to eliminate communication loops and to constrain nodes from creating non-optimal routes.

A malicious node perpetrating Rank attack falsifies its Rank value to attract neighbouring unsuspecting nodes as a better parent route for their traffic destination. RPL has no mechanism to protect against the alteration of a node’s Rank value. The Rank attack does not destabilize normal network operations, but when combined with other attacks, it could prove very devastating. Some of the impacts of Rank attacks include: the creation of non-optimal routes even though an optimized route already exists in the topology, creation of routing loops without discovery, decrease in packet delivery ratio due to un-optimized paths, increased control overheads with every DIO update since updates are synchronized around the malicious node.

3.7.13 Modification Attacks

In a modification attack, malicious nodes exploit the idea that trust levels in normal ad-hoc networks are not measured nor enforced, and that control packets and protocol messages contain important routing information, which guides the behaviour of their routing. These rogue nodes participate in route discovery, intercept and disrupt the routing operation of the network. Malicious nodes cause redirection of network traffic and DoS attack by simply changing fields in the protocol messages. Modification attacks can further be classified into two types [86].

1. Redirection by Modifying Route Sequence Number

Routing protocols maintain fresh routes by increasing the sequence number for each next hop destination and the higher the destination sequence number the more recent and reliable the route. A malicious node with the intent of disrupting network operations could advertise its route as having the shortest path to a node, with a destination sequence number greater than the real value, and this will cause packet interception for the traffic of interest for the malicious node.

2. Redirection by Modifying Hop Count

This type of attack is mostly prevalent in routing protocols utilizing hop-count such as RPL and DSR. A malicious node advertises itself as having the shortest path to a destination by advertising a low hop-count than what is obtainable. This gives opportunity to various attacks such as DoS and routing loop attacks. In the DSR protocol particularly, route in data packets are explicitly stated as source route and while there are no integrity checks on the source route, a malicious node can change this piece of key control information.

3.7.14 Byzantine Attacks

This attack involves a node working alone or colluding with other nodes to selectively drop route packets, create routing loops and forward route packets via sub-optimal paths with the eventual aim of creating a declining network service or the total disruption of the network service [87].

3.7.15 Route Cache Poisoning Attacks

In sensor networks, a promiscuous mode of routing is a process whereby a node overhears other packet route information and then, update its route cache even when it does not need the route information. Nodes use the promiscuous operation mode to build their routing table (cache). However, malicious nodes exploit this technique to poison the route caches of other nodes. To illustrate this, consider a malicious node M, and an unsuspecting node X. The malicious node M broadcasts spurious route packets with source route to X through itself. Other neighbouring nodes overhearing the packet transmission quickly update their route by adding the route to their routing tables thereby poisoning their route caches. This is common with reactive routing protocols such as DSR and AODV [82].

3.7.16 Fabrication Attacks

This is the creation of spurious routing information and there are three types of such attacks.

1. Routing Table Overflow Attacks

In this type of attack, the malicious node publicizes routes that lead to nowhere in the network. The rogue node creates so much route discoveries, which tend to overwhelm the routing tables of neighbour nodes and beyond. This is to hinder real route entries from being made. Proactive routing algorithms are more susceptible to this type of table overflow attacks since proactive routing protocols try to discover routing information before they are actually needed. Examples include RPL and OLSR [82].

2. Resource Consumption Attacks

In a resource consumption attack, a malicious or compromised node attempts to consume battery life of a node or other nodes by making excessive route discovery requests or by forwarding superfluous packets to the target nodes in the network. This causes unnecessary activity among nodes, depleting their battery life. This type of attack is also known as the sleep deprivation attack [82].

3. Falsifying Route Error Messages

Reactive routing protocols like AODV and DSR have procedures for handling broken routes when integral nodes leave the network. When a destination node or a transitional node along any active path moves, or leaves the network, the node which precedes the broken link sends out a route error message to all active neighbours about the topology or broken link. These nodes update their links by invalidating the route for this destination in their routing tables. However, a malicious node can mimic this situation by sending out fake route error messages to all neighbouring nodes to create a DoS attack across the network.

3.7.17 Location Spoofing Attacks

Spoofing is the art of deception. In a spoofing attack, the aim of the malicious node is to deceive and disrupt the normal operations of the protocol by transmitting its position information in favour of itself. It achieves this, by announcing a spurious position such as, having the closest link to the destination node to attract traffic to itself, and thereby launch its attack to disrupt the network service. A combination of Sybil and a position spoofing attack can create a perimeter around a node, which has the capacity to control all traffic from that node.

3.7.18 Rushing Attacks

This form of attack capitalizes on one of the features of reactive routing protocols which is flooding. In reactive routing, when a node seeks to transmit or discover a destination node it does this by sending a route request through controlled flooding to all neighbour nodes. Any node with a first positive route reply sends back a route reply to the source or initiator node. The initiator node accepts this as the best and optimal route, being the first to send a route reply while discarding subsequent replies from other nodes. In a rushing attack, a rogue node "rushes" to quickly give a route reply to the initiator node, hence the initiator node (source node) accepts the reply from the rogue node. The initiator node therefore, discards subsequent route replies it may receive, even if they are legitimate replies. The resulting route will now include the rogue node's detail, which gives the rogue node the advantage of exploiting the network [82]. Furthermore, two colluding nodes which have formed a wormhole could further perpetrate a rushing attack by using the tunnel they have created as a fast route propagation channel that may seem better than the normal multi-hop route channel. Rushing attacks prove to be effective DoS attack against most reactive routing protocols and some secure routing protocols like ARAN and Ariadne [82, 88, 89]. To sum up, Figure 3.9 shows the classification of attacks in sensor networks, which also applies to IoT networks since they are essentially sensor nodes. The Figure is based on the Open System Interconnection (OSI) reference model.

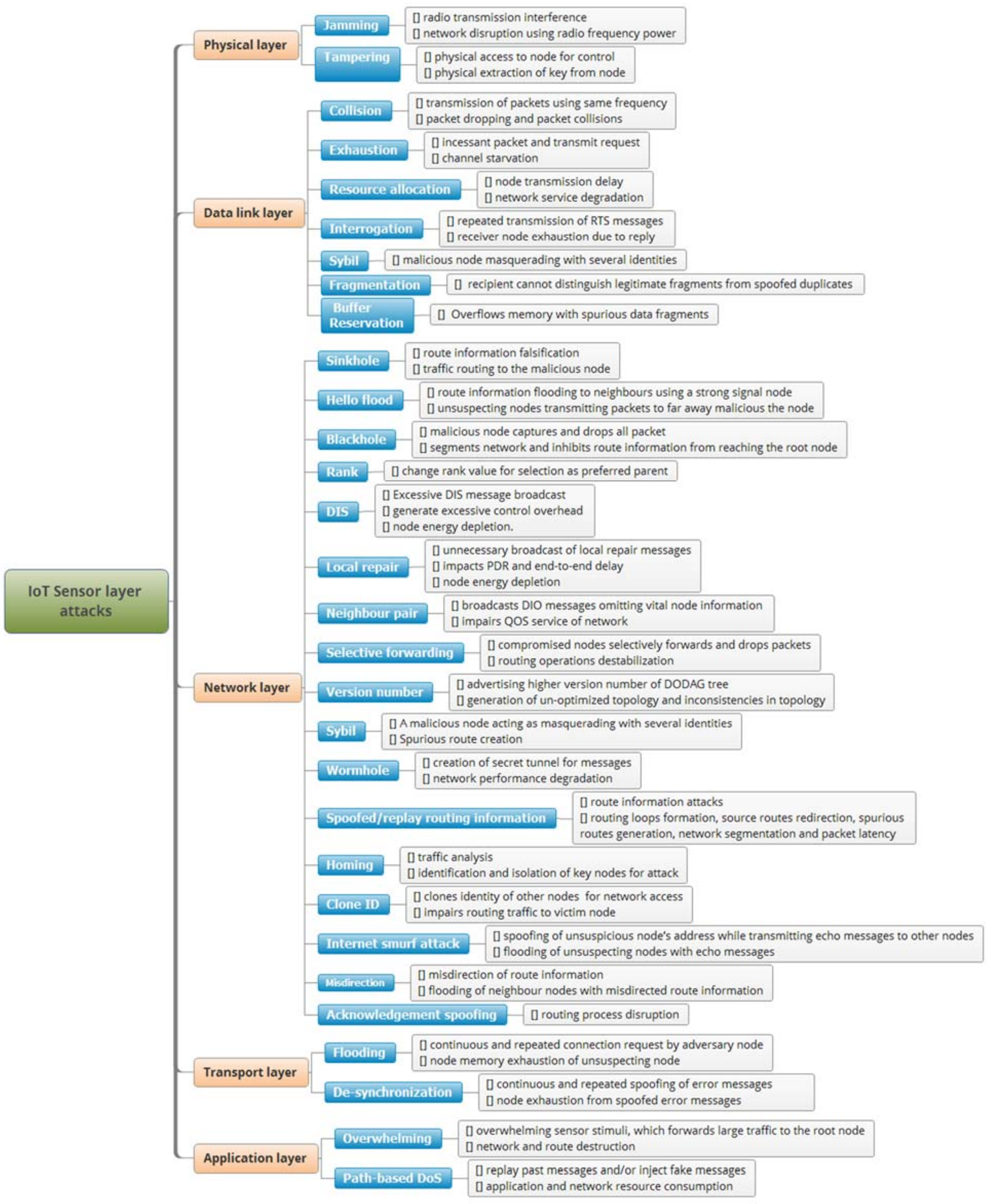

IoT Sensor layer attacks
Physical layer
Jamming
▯ radio transmission interference
▯ network disruption using radio frequency power
Tampering
▯ physical access to node for control
▯ physical extraction of key from node
Data link layer
Collision
▯ transmission of packets using same frequency
▯ packet dropping and packet collisions
Exhaustion
▯ incessant packet and transmit request
▯ channel starvation
Resource allocation
▯ node transmission delay
▯ network service degradation
Interrogation
▯ repeated transmission of RTS messages
▯ receiver node exhaustion due to reply
Sybil
▯ malicious node masquerading with several identities
Fragmentation
▯ recipient cannot distinguish legitimate fragments from spoofed duplicates
Buffer Reservation
▯ Overflows memory with spurious data fragments
Network layer
Sinkhole
▯ route information falsification
▯ traffic routing to the malicious node
Hello flood
▯ route information flooding to neighbours using a strong signal node
▯ unsuspecting nodes transmitting packets to far away malicious the node
Blackhole
▯ malicious node captures and drops all packet
▯ segments network and inhibits route information from reaching the root node
Rank
▯ change rank value for selection as preferred parent
DIS
▯ Excessive DIS message broadcast
▯ generate excessive control overhead
▯ node energy depletion.
Local repair
▯ unnecessary broadcast of local repair messages
▯ impacts PDR and end-to-end delay
▯ node energy depletion
Neighbour pair
▯ broadcasts DIO messages omitting vital node information
▯ impairs QOS service of network
Selective forwarding
▯ compromised nodes selectively forwards and drops packets
▯ routing operations destabilization
Version number
▯ advertising higher version number of DODAG tree
▯ generation of un-optimized topology and inconsistencies in topology
Sybil
▯ A malicious node acting as masquerading with several identities
▯ Spurious route creation
Wormhole
▯ creation of secret tunnel for messages
▯ network performance degradation
Spoofed/replay routing information
▯ route information attacks
▯ routing loops formation, source routes redirection, spurious routes generation, network segmentation and packet latency
Homing
▯ traffic analysis
▯ identification and isolation of key nodes for attack
Clone ID
▯ clones identity of other nodes for network access
▯ impairs routing traffic to victim node
Internet smurf attack
▯ spoofing of unsuspicious node's address while transmitting echo messages to other nodes
▯ flooding of unsuspecting nodes with echo messages
Misdirection
▯ misdirection of route information
▯ flooding of neighbour nodes with misdirected route information
Acknowledgement spoofing
▯ routing process disruption
Transport layer
Flooding
▯ continuous and repeated connection request by adversary node
▯ node memory exhaustion of unsuspecting node
De-synchronization
▯ continuous and repeated spoofing of error messages
▯ node exhaustion from spoofed error messages
Application layer
Overwhelming
▯ overwhelming sensor stimuli, which forwards large traffic to the root node
▯ network and route destruction
Path-based DoS
▯ replay past messages and/or inject fake messages
▯ application and network resource consumption

Figure 3.9: A summary of IoT attacks across the OSI layers

3.8 Secure Routing in IoT: Limiting Factors

Routing protocols enable nodes acting as routers to exchange route details to create routes between nodes. These route details could become a target for malicious nodes who intend to cause harm to the network. 6Lowpan and RPL are two common IETF standard IoT protocols. However, none of them, has effective mechanisms to protect against malicious attacks. This research work aims to study the common security threats and create a framework for secure routing among IoT devices. Implementing secure routing mechanisms in IoT networks is a big challenge and the following limiting factors need to be taken into account while working towards achieving a secure routing system for IoT networks.

3.8.1 Energy Level of IoT Nodes

Most IoT devices or nodes in the networks are small, and have limited battery life. However, they need to be energy-efficient to function appropriately over a considerable time. For example, running a power hungry secure routing mechanism that incurs high computational overhead in terms of energy-efficiency, could be a limiting factor in IoT nodes.

3.8.2 Scalable Nature of IoT Networks

IoT devices in the networks are randomly deployed in very large numbers and can largely be left unattended throughout their lifetime (based on their usage and purpose). Moreover, depending on the nature of the applications running, more number of nodes may need to be added to the networks. This requires the IoT networks to be highly scalable, which can be a constraint not only when it comes to designing routing mechanisms between the thousands of nodes, but also imposing appropriate security mechanisms [90].

3.8.3 Heterogeneity of IoT Networks

It is expected that the IoT sensor nodes will be able to communicate with other devices connected to the internet through wireless and wired medium and vice versa. However, this poses a challenge as to how the larger devices like a computer server and laptop with full IP packets could send details to a remotely located sensor node considering the sensor node’s limited capability.

3.8.4 Unavailability of Intermediate Nodes

Most battery powered IoT sensor nodes have a finite battery life. These nodes while acting as intermediate devices to other sensor nodes, could in the process of time exhaust their battery power or could be compromised by an attacker and thus, become unavailable. This creates a communication gap between these sensor nodes in the network.

3.8.5 Mobility of IoT Networks

Some IoT nodes are by default mobile thus, they create a disruption of network service when they move and are unavailable in the routing sequence, which was already formed for communication and routing.

3.8.6 Open Wireless Medium

Most IoT nodes communicate using the IEEE 802.15.4 wireless medium which is open and largely unsecure and hence, could be prone to eavesdropping by malicious nodes.

3.8.7 Memory and CPU Capacity

The memory and computational resources of sensor nodes are constrained and hence, typical hardware platforms comprise of a simple micro-controller with few millions of instructions per second (MIPS) and 32 kilobytes of RAM with no support for complex operations.

3.9 Secure Routing Protocols in IoT

In preventing routing attacks, several secure routing strategies have been proposed. In this section, a review of some secure routing protocols proposed by the research fraternity is discussed below.

3.9.1 Secure Multi-Hop Routing for IoT Communication

The work described in [91] introduces a secure multi-hop routing protocol (SMRP) which allows IoT devices to communicate in a secure manner. It achieves this, by making sure that IoT devices authenticate before they join or create a new network. The routing protocol proposed incorporates a multi-layer parameter into the routing algorithm and hence, when nodes want to join the network, they must authenticate.

The authors claim this protocol comes with no additional overhead on the routing process as the multi-layer parameters contain the permissible applications on the network, a unique User-Controllable Identification, and a summary of devices allowed on the network. However, a closer observation reveals much overhead is incurred in creating a multi-layer parameter that will host even as few as 100,000 IoT nodes in this type of network. This makes the protocol unusable on a large-scale network.

3.9.2 TSRF: A Trust-Aware Secure Routing Framework in Wireless Sensor Networks

The Trust-aware secure routing framework (TSRF) [92] designed for WSNs was based on trust derivation which consists of direct and indirect observations of behavioural patterns of sensor nodes with trust values among nodes represented in a range from zero to one. A zero signifies no trust exists between nodes, and a one shows a good level of trust for the corresponding node. The authors opined that their system addressed the following attacks: on-off attack, conflicting behaviour attack, selfish attack, bad mouthing attack and collusion attack. However, TSRF expended significant amount of memory due largely to the complex trust computations among the nodes. Also, rogue nodes were identified based on previous trusts among one another, which reveals that a new rogue could join the network and behave well for a while and earn a good reputation history. After earning this good reputation of trust, it begins to carry out its malicious behaviour within the network.

3.9.3 Two-Way Acknowledgment-based Trust (2-ACKT)

This system operates in a non-promiscuous mode and is contingent only on direct trust between nodes. The scheme is based on a dual acknowledgement system in developing trust among neighbour nodes. The scheme further develops a route to the sink node as well as introduced a new node (regarded as the sponsor and third party node), which creates a two-hop acknowledgement in the network. One basic assumption the protocol makes is that all malicious nodes drop data packets and not the acknowledgments hence, it cannot deal with greyhole attacks. Also, since the neighbouring nodes were not the source of the recommendations, it follows that the conclusions on trust relationships might not be in consonance with the state of the network [93].

3.9.4 The Group-based Trust Management Scheme (GTMS)

The Group-based trust management scheme (GTMS) was proposed by [94], and it is a trust-based scheme involving the computation of trust via a direct observation among nodes i.e., the number of successful and unsuccessful interactions among nodes. The authors defined successful interaction as positive collaboration among nodes and indirect observations (recommendation of trusted peers concerning a node in the network) among nodes. Cluster Heads (CH) were created at the intragroup level, and a distributed trust management scheme was used for gathering recommendations from all its group members, and about other CHs directly from the sink. The trust level was defined using unsigned integers from 0 to 100 to decrease memory usage. Although, the system addressed Blackhole attacks, the cluster heads at the intragroup level have a high-energy requirement for them to communicate with the sink node (central node), and this could easily drain the sensor batteries of the CH nodes. In addition, the CHs were a point of failure. A failed or compromised CH will imply that nodes connected to that CH will be isolated or compromised as well.

3.9.5 Collaborative Lightweight Trust-based (CLT) Routing Protocol

This protocol focuses on a collaborative trust effort among nodes while minimizing memory overhead and battery dissipation in nodes. The novelty of this system is the employment of a trust counsellor, which monitors, warns and improves any node whose trust level is diminishing. It achieves this by utilizing a sliding window system to develop a trust history of all neighbours' nodes. It further uses an aging mechanism to determine misbehaving nodes within the network, and thus, uses this to prevent various attacks. The authors claim that the protocol could prevent Blackhole, on-off, bad mouthing and good mouthing attacks. The system however, fails to prove the outcomes for autonomous nodes as may be needed in some application areas. It assumes that all nodes have a unique identity.

A summary of secure routing protocols in IoT investigated is presented in Table 3.2.

Table 3.2: A Summary of Secure Routing Protocols for IoT

Protocol/References	Techniques	Attacks addressed	Brief description	Weaknesses
Secure Multi-Hop Routing for IoT Communication[91]	Multi-layer parameter authentication	Grey hole, Blackhole, Sinkhole and spoofing attacks	System authenticates IoT devices before they could join or create a new network. It also uses a multi-layer parameter into the routing algorithm and hence, when nodes want to join the network, they must authenticate.	Excessive overhead in creating a multi-layer parameter that will host IoT nodes in the network making the protocol unsuitable large scale deployment.
TSRF: A Trust-Aware Secure Routing Framework in Wireless Sensor Networks [92]	Direct and Indirect Trust metric system	On-Off attack, conflicting behaviour attack, selfish attack, bad mouthing attack and collusion attack.	A system designed for WSNs and based on trust derivation, which is a direct and an indirect observation of behavioural patterns of sensor nodes with trust values among nodes represented in a range from 0 (no trust) to 1 (absolute trust).	The system expended too much memory due largely to the complex trust computations among the nodes. Also, rogue nodes were identified based on previous trust history which implies that new rogue nodes behaving well for a while will evade detection.
Two-way acknowledgment-based trust (2-ACKT) [93]	Direct trust metric between nodes	Blackhole, spoofing and selfish behaviour attacks	The scheme is based on a dual acknowledgement system in developing trust among neighbouring nodes while creating a route to the sink node with a third-party sponsor that creates the two-hop acknowledgement in the network.	Does not detect greyhole attacks and the trust relationships is not in consonance with the state of the network since neighbouring nodes are not the source of the recommendations.
The Group-based trust management scheme (GTMS) [94]	Trust computation using direct observation of nodes	Addressed black hole attacks	A trust management scheme involving the computation of trust using the number of successful and unsuccessful interactions among nodes and indirect observations among nodes while using Cluster Heads (CH) at intragroup level for gathering recommendations from all its group members.	The cluster heads at the intragroup level had a high-energy requirement for them to communicate with the sink node and this drains the sensor batteries of the cluster head nodes.
Collaborative lightweight trust-based (CLT) routing protocol [95]	Collaborative trust effort among nodes	black hole, on-off, bad mouthing and good-mouthing attacks	Protocol which uses a trust counsellor in monitoring and warning nodes with diminishing trust levels through the use of a sliding window system to develop a trust history of all neighbours' nodes. It also employs an aging mechanism to determine misbehaving nodes within the network and thus prevent network attacks.	The system fails to prove the outcome for autonomous nodes as may be needed in some application areas and assumes that all nodes have unique identity.
Lithe: Lightweight Secure CoAP for the Internet of Things [96]	DTLS compression Mechanisms for CoAP	Fragmentation attacks, end-to-end secure delivery of data in CoAP.	A 6LoWPAN datagram transport layer security (DTLS) compression protocol for CoAPs which extended the 6LoWPAN standard and introduced an integration module for header compression and end-end delivery of data packets in COAP.	System involves use of cryptographic processing of Record and handshake protocols which are computationally expensive and the system is still susceptible to attacks like Grey hole, Black hole Sinkhole and spoofing attacks
Security Access Protocols in IoT Networks with Heterogeneous Non-IP Terminals [97]	Time-based key-generating server system	Prevents replay attacks	A time-based system which generates keys for secure transaction between short range non-IP devices. A security procedure is used for both uni- and bi-directional devices, contingent on the devices' capabilities. The security algorithms are based on a local key renewal while considering the local clock time.	A potential weakness is with the mediator server being compromised de-synchronization, replay and reader impersonation attacks will be very possible. Also, the system assumes IoT devices have GPS system which is rarely the case.
Secure communication for the Internet of Things—	IPsec	Secure end to end transmission	This system explores the use IPsec as a security mechanism for secure end-to-end transmission in IoT. An IPsec extension was designed based on 6LoWPAN through the extension various header in the	Complex protocol design as protocol does not accomplish a trade-off between simplicity and compatibility – The approach seeks to apply

a comparison of link-layer security and IPsec for 6LoWPAN [98]			6LoWPAN frame header format while also taking advantage of the cryptographic system within the IEEE 802.15.4 transceivers for 6LoWPAN/IPsec.	IPsec to resource constrained devices by harmonizing link-layer security and IPsec security
Energy-Efficient Probabilistic Routing Algorithm for Internet of Things [99]	Node residual energy and expected transmission (ETX) count	None.	A protocol which controls the broadcast of the routing request packets stochastically so as to boost network lifetime while reducing packet loss due to flooding. Using the residual energy of a node and the expected transmission (ETX) count as the routing metrics, the system stochastically controls the number of route requests hence gaining an improved energy-efficient route setup.	Susceptible to all forms of attacks.
An Energy-Aware Trust Derivation Scheme With Game Theoretic Approach in Wireless Sensor Networks for IoT Applications [100]	Trust Derivation Dilemma Game system	Bad mouthing, DoS and Selfish attacks	A game theoretic energy-aware secure protocol for IoT which proposes a risk approach model in finding the best number of recommendations which fulfils the network security requirements. The trust derivation dilemma game (TDDG) is introduced into the trust derivation system based on the optimal recommendations received while the mixed strategy Nash equilibrium is used to compute the probability of the selected strategy.	Excessive overhead produced by trust request which degrades the performance of the network. The network is also susceptible to attacks such as greyhole, black hole.
A standard compliant security framework for IEEE 802.15.4 networks [101]	Encryption and authentication.	Replay attack	A security compliant framework developed for setting up and managing secure IEEE 802.15.4 networks. The framework envisions some likely secure configurations in a low-power and lossy network while describing how each could be used in defending against layer 2 attacks (MAC) through a key exchange.	The framework does not extend to the layer 3 (routing layer) which makes it vulnerable to layer 3 attacks such spoofing, bad mouthing, Greyhole and black hole attacks.
6LoWPAN: a study on QoS security threats and countermeasures using intrusion detection system approach [102]	Statistical-based intrusion detection system (IDS) and Cryptography	Grey hole, Black hole Sinkhole, spoofing attacks, selfish attack, bad mouthing attack and collusion attacks.	A 6LoWPAN IDS framework for securing network operations at the link layer. The paper proposes the use of an RPL system based IDS for fortifying network topology while utilizing a statistical anomaly method in guaranteeing performance of nodes.	A framework yet to be implemented and tested.
Optimal and Secure Protocols in the IETF 6TiSCH communication stack [103]	6TiSCH	Addressing security issues at the MAC layer as found in 6LoWPAN and RPL.	Presents a work-in-progress of the standardization effort of the new routing protocol which hopes to address the optimal distributed scheduling technique that can assign resources between network nodes in an efficient manner and providing a scalable system which supports the setting up and management of secured domains for the industrial sector.	This is yet to be seen as 6TiSCH is still a work-in-progress.

3.10 Secure Routing in IoT: Issues and Challenges

The lack of standardization in secure routing among IoT devices raises many concerns related to the current security level of routing practiced in IoT networks. While the security consequences for IoT remains imminent and near perhaps the introduction of a secure routing framework could become the foundation to the adoption and

implementation of security standards in the IoT ecosystem. Today, the growth of IP-based sensors implies a further increase of possible attacks in IoT. This highlights the need for a new or improved security protocols and identification techniques in IoT. It is without any doubt that IoT presents fresh challenges to network and security designers. The design of IoT devices will need to evolve to cope with these fresh challenges.
The further exploration of the IETF standard routing protocols is presented while pointing possible research challenges.

3.10.1 6LowPAN: Research Challenges

The IETF RFC 4944 [104, 105] identified the idea of adopting various security mechanisms within the concept of the 6LoWPAN adaptation layer, it however addressed only the general security threats and requirements, and there is still no security implementation. Furthermore, several proposed solutions to the open research challenges in guaranteeing the IoT network-layer for secure route communications using 6LoWPAN are discussed.
The Internet Protocol Security (IPsec) [106-108] design facilitates the authentication and encryption of IP packets operating at the network layer during a communication session. It further provides support for Virtual Private Networks (VPN) while in different operation modes. As indicated earlier end-to-end network-layer security may find their usefulness, in future IoT deployments and these IoT devices will be required to be in sync with other internet devices that are more resource endowed than them. Notwithstanding the benefits of end-to-end network layer security and the proposal in the RFC 4944 of the IETF standard, no precise security model has been defined for adoption regarding the 6LoWPAN adaptation layer.
A major challenge to the adoption of IPsec and IKE in 6LoWPAN as a network layer security is predicated on the resource constraints of the sensing nodes and a comprehensive study to buttress this have been presented by [109, 110]. Furthermore, a look at other means of securing network routes like frame header compression and embedding the concept of efficient trust models to work in consonance with the 6LoWPAN adaptation layer will facilitate secure end-to-end communications at the network layer while providing guarantees regarding the confidentiality, integrity, authentication and non-repudiation of network data. Furthermore, some proposals have been put forward, which emphasized on the implementation of compressed security headers for the adaptation layer of 6LoWPAN while achieving the same goal as the

existing Authentication Header (AH) and Encapsulating Security Payload (ESP) headers of the Internet Protocol Security (IPsec) as proposed in [106-108]. This approach was strengthened by [111], where the authors submitted that the introduction of compressed security headers within the adaptation layer was promising so long as a careful design pattern is followed and the various technology platforms could support a seamless hardware security optimization. In another submission, the same authors further advanced and performed a trial evaluation of the usage of AH and ESP compression header security for 6LoWPAN in tunnel and transport modes using AES/CCM encryption at the hardware layer and a presumed application security profile [112, 113].

More recently [114] considered the design of header compression security for 6LoWPAN, in this case using a context sharing LOWPAN_IPHC header compression. A detailed review of this proposal and evaluation against IEEE 802.15.4 link layer security has been presented in [115]. A basic advantage of the header compression proposals is in the usage of the more recent IPHC header compression scheme since it supports the use of IPv6 for global and multicast usage. With the proposals of these research authors presented above for the support of 6LoWPAN network security layer, this will obviously require the support of industry and technology players that will either support the compression security header philosophy or support the end-to-end network security implementation via a security gateway. Both aspects represent opportunities for research, like creating a design of mechanisms to support the conversion between IPsec and 6LoWPAN security, or the mediation of gateways during key management, and key mapping operations. This of course is in addition to decision by industry players as to what is practicable and beneficial in the interest of the industry, sustainability and future development of the technology.

Addressing the security concerns of 6LoWPAN, the authors in [59] substantiated the consequences of packet fragmentation attacks which is a major issue affecting 6LoWPAN. The occurrence of these attacks make buffering, forwarding and handling of fragmented packets challenging for these devices running 6LoWPAN since a malicious node could send spurious, identical or overlapping fragments, which could alter the normal network flow. All these occur because of the lack of verification at the 6LoWPAN adaptation layer, since the receiving nodes have no way of differentiating fake fragments from the authentic frames during fragment reassembly. The consequences of fragmentation attacks range from getting buffer overflows to the mismanagement of the available computational capability on the sensor devices. The authors suggested the introduction of new field fragmentation headers of 6LoWPAN to address the fragmentation attacks like using a timestamp, which protects against unidirectional fragment replays and a one-time protection mechanism against bidirectional fragment replays.

Also, [59] proposed the use of an authentication mechanism which performs a per-fragment sender authentication and removal of messages from the receiver's buffer, for nodes considered suspicious. The authors employed a hash chain system, which grants a legitimate sender the authority to add an authentication value to each fragment during the 6LoWPAN fragmentation. In the event of an overflow the receiver has the option of deciding which fragments to discard. This decision is based on the size of frames captured and processed and the sending behaviour of the source node. Although this scheme does not necessitate any adjustments to 6LoWPAN's frame format, it is rather obvious that the proposed security mechanisms must be co-opted into the adaptation-layer of 6LoWPAN.

In the formal specification of 6LowPAN standard, key management was a vital security functionality considered in the 6LoWPAN. This aspect could be considered a cross layer security that is interconnected with authentication since keys need to be negotiated and intermittently refreshed to guarantee effective and lasting security regardless of the layer at which the communication may ensue. Although the authors did not put forward any definite key management solution, however, RFC 6568 [116] shows the likelihood of adopting a simple but effective Internet key management solutions. For example, minimal IKEv2 [117] co-opts Internet key management to resource-constrained sensing in environmental locations while preserving its compatibility with the current Internet standard.

In [118] the authors proposed that public-key management strategies could necessitate the use of nodes which are more powerful than the state-of-the-art sensing platforms, especially if they require supporting services. Other proposed methods that could be explored include the introduction of a new lightweight key management technique suitable for the specific IoT device environment.

3.10.2 RPL: Research Challenges

Although the IETF RPL standard includes versions that attempt to secure route control messages using simple security procedures, it however suffers from having a basic system for supporting important secure routing operations. To illustrate this point, a discourse on the current state of research in RPL while focusing on the security of RPL is presented below.

This research study notes that although the secure versions of RPL which attempt to secure the route control messages, there are no extra security mechanisms implemented in the present version of RPL protocol standard [62]. A further investigation reveals that RPL suffers from the following attacks including: falsification attacks, routing information replay, byzantine attacks, physical device compromise or remote device access attacks, selective-forwarding attacks, sinkhole attacks, Blackhole attacks and greyhole attacks and version number manipulation attacks [67, 119, 120]. The RPL IETF ROLL report further discussed the general security requirements and goals, but did not give specific security models for RPL. It would be worthwhile investigating into the various security threat models specific to RPL and to its application areas and to eventually develop systems to protect RPL routing protocol from threats identified. Part of the focus of this thesis is to develop such a system.

Furthermore, the present RPL standard [62] mainly addresses the management of keys with applications using device pre-configuration and how such devices could join a network using a preconfigured common group key or a key learned from a trusted DIS configuration message. It does not describe how authentication and secure network connection mechanisms could be designed to facilitate other devices that are dynamic and run security critical applications. A research possibility for routing profiles in RPL could be the definition of routing profile for specific application areas. A further investigation and standardization could be to survey the design of security policies describing how security could be used in protecting routing operations regarding an application area. The policies could further establish the requirements of applications with respect to confidentiality, integrity, authenticity, non-repudiation and the ability to replay control messages within the network.

The authors in [119] have presented open issues with respect to the security of RPL. They presented various threat analysis against Routing Over Low power and Lossy networks (ROLL) routing systems while making contributions on how to address these threat challenges. The authors in their study, identified threats by using the ISO 7498-2 security reference model as specified in [121] which listed the various attributes of a good and secure communication and these include access control, authentication, confidentiality, integrity and non-repudiation, and availability. The model defines what to protect while also identifying possible vulnerable points needing protection that could be undermined in the network. The model supports the classification of the threats and the precise attacks relating to confidentiality, integrity and availability of routing and control message exchanges in the perspective of routing protocols in ROLL. The paper further advanced a security framework for ROLL protocols, which is based on earlier work on security for routing while adjusting the parameters to suit the constraints peculiar to the 6LoWPAN environments. Within this framework, security features are enumerated that could be adapted and fitted within the RPL perspective along with some general system security features which could affect the routing protocol, however, this requires serious attention as the method to be adopted and the impact goes beyond the routing protocol itself. The evaluation presented in this study could give a promising pathway to good security recommendations for integration into the ROLL protocols. It is noteworthy that the implications of the various security recommendations presented for the ROLL protocols presents possible issues for future research consideration.

A look at the aspect of RPL security, as proposed in [62] provides security only against external attacks. An internal attacker who has compromised a node within the network could selectively inject routing messages with malicious purposes. The authors of [65] gave a comprehensive analysis of the internal attacks on RPL, with specific attention on the Rank property as specified in the RPL protocol. Rank in RPL is used for the prevention of loops, route optimization, and for the minimization of route control overhead. The paper further presented the attacks against the Rank property of RPL and the impact on network performance. They identified that the limitations in RPL was largely because a child node does not have access to the control messages of its parent(s) and hence, unable to determine what services its parents are providing thus, for a compromised parent node, a child node will certainly follow the unstable and compromised route. Although they did not put forward a new scheme for detecting node parental activities, the paper however, recommended the integration of techniques in RPL that could aid the child node to observe the behaviour of its parents to defend against internal attacks coming from the parent node.

Reference [122] discussed internal attacks and submitted that an internal attacker is capable of undermining a node so as to mimic a gateway (like the DODAG root) or a node within the circumference of a gateway. The authors in their submission proposed the use of a version number with a Rank authentication scheme centred on one-way hash chains, which links the version numbers with the authentication information (MAC codes) and signatures. This system provides a good defense against internal attackers capable of sending DIO messages using higher version numbers or attackers with capabilities of issuing higher Rank values. The purpose of an attacker sending higher version number is to impersonate the DODAG root node and commence the creation of the routing topology. The higher Rank value issued by the attacker will force a larger part of the nodes in the network to connect to the DODAG root via the attacker hence, giving the attacker the advantage of eavesdropping and manipulating a part of the network traffic. The paper gave an evaluation of performance against the impact of these mechanisms on computational time, but did not address the energy impact and memory requirements, which are a constraint on these IoT devices.

The authors in [123] focused on various internal attacks against RPL. In their submission, the authors reflected on the impact of sinkhole attacks on RPL networks. They scrutinized the end-to-end data delivery performance in the presence of sinkhole attacks. Sinkhole attacks undermine a node by capturing and discarding the routing and control messages. The authors recommended the blend of a parent fail-over system with a Rank authentication scheme. They further illustrated their idea using simulation results to prove that the blend of the two approaches gave promising results and that by populating the network the permeation of sinkholes could be mitigated without requiring to know the specific location of the sinkhole nodes within the network. Their Rank-authentication system was centred on one-way hash chains provided in [122], whereas the parent fail-over scheme uses an end-to-end acknowledgment system which the DODAG root controls.

To recap, various research proposals have been presented to address security research challenges in RPL, specifically referencing the threat models and internal attacks to 6LoWPAN and RPL. These proposals could help in evolving and delivering useful contributions for addressing the security loopholes of these protocols and their subsequent adoption as possible standards in the future. A leaf could be taken from other similar areas to IoT routing such as WSNs and MANETs where extensive research knowledge has been built over time and various approaches have been proposed. This may guide in the final adoption of a truly secure routing framework for IoT devices so long as, with a proper internalization and the efficient design it fits the requirements of IoT routing protocols.

Finally, while not restricting the study to key management only, other secure routing techniques such as the concept of trust systems, which have also been researched and applied in WSNs and MANETs and have even found practical application in various computing fields could also be explored with the intent of adapting them in developing a secure routing framework for IoT. Table 3.3 gives a summary of some RPL attacks on confidentiality, integrity, availability with countermeasures.

Table 3.3: Summary of RPL Attacks and Countermeasures

Attacks	Classification of attacks	Effect on network performance	Protocols addressing attacks
Rank	Confidentiality & Integrity	Low packet delivery ration and packet delay; Generation of non-optimal path and loop	Use of IDS based solutions [124], [125], VeRA [122], TRAIL [126]
Selective forwarding	Confidentiality & Integrity attack	Disruption of route path	Heartbeat protocol [67]
Sinkhole	Confidentiality & Integrity attack	Compromising huge traffic passing through attacker node	IDS solution [124], Parent fail-over, Rank authentication technique [123]
Hello flooding	Availability attack	Dissipation of sensor battery power	RPL's Global and Local repair mechanism removes attack
Wormhole	Confidentiality & Integrity	Disruption of route topology and traffic flow	Merkle tree authentication [127]
Sybil and Clone ID	Confidentiality & Integrity attack	Route compromise and traffic unreachable to victim's node	Routing attacks and countermeasures in RPL-Based IoT [67]
Denial of Service	Availability attack	Resources unavailable to nodes	Intended user IDS based solution [128]
Blackhole	Availability, Confidentiality & Integrity	Dropped packets and increased route traffic and control overhead	SVELTE [124] Monitoring of counters [73], Parent fail-over [123],
Version number	Confidentiality & Integrity	Increased control overhead and low packet delivery ratio, high end to end delay	VeRA [126]
Local Repair Control overhead	Confidentiality & Integrity	Control and routing traffic disruption	IDS based solution [102]
Neighbour attack	Confidentiality, Integrity & Availability	False route, route disruption and resource consumption	TRAIL [126]
DIS Attack	Availability	Resource consumption	TRAIL [126]

3.11 Software Defined Perimeter

Software Defined Perimeter is a security framework proposed by the Cloud Security Alliance (CSA) to protect IT assets from application and network attacks [129]. According to [129], the traditional network architecture employs the use of static security perimeter, which involves the deployment of firewalls, Intrusion Prevention and Intrusion Detection Systems amongst others. The traditional network architecture provides security for the internal devices and its services in the network, which defines the perimeter. This is achieved by making resources unavailable to unauthenticated external devices to the network. However, with the increasing number of unsecured BYOD's in a network, attacks on the traditional network infrastructure could easily be breached. In addition, the migration of an organisation's network and service infrastructure to the Cloud further accentuates the vulnerability of the traditional network architecture. This according to the

CSA group has necessitated the introduction of a new security architecture called the Software Defined Perimeter (SDP).

The concept of SDP is to narrow the perimeter to the server(s) that deliver network and application services to user hosts. In this approach, all devices including legitimates hosts are considered untrusted and hence, are not able to view nor access network and application services. SDP consists of three main components which include:

i. The SDP Controller; this is the authenticating platform that determines which host can communicate with another host. Without the SDP Controller, there is no access to any resource in the network.

ii. The Initiating SDP host; a terminal host like laptop, PC, that requires a certain network or application service from another host like a file server, database server or printer services.

iii. The Accepting SDP host; typically, these could be servers that deliver services to other initiating SDP hosts. However, Accepting SDP hosts accept requests only at the behests of the SDP Controller thus, enshrining the model of trusting no devices even in its physical perimeter.

SDP follows the need-to-know model, which requires the authentication of a host via a controller, but without a remote gateway before it can be granted access to any resource or service. In addition, SDP necessitates the initial authentication and authorization of terminal hosts before these terminal hosts can have network access to secure servers while connections to these servers are encrypted in real time lasting the duration of the communication process. Figure 3.10 shows a summary of the SDP architecture and its communication process between an initiating SDP host with an Accepting SDP host. The communication process is summarised according to the steps given below.

1. SDP Controllers are initiated and connected to optional authentication features.
2. Accepting SDP hosts (servers) are also initiated, authenticated and connected to the SDP Controllers. But, communication with other hosts that are not authorised via the SDP Controllers' approval is not accepted.
3. Every Initiating SDP host first communicates and authenticates with the SDP Controllers to request access to an Accepting SDP host.
4. Upon a successful validation of an Initiating SDP host, the SDP Controllers assign the appropriate Accepting SDP hosts that the Initiating SDP host could communicate with.

5. The SDP Controllers further direct the Accepting SDP hosts to now accept and transact with a specified Initiating SDP host using the policies and procedures specified for secure communications.
6. The Initiating SDP host receives from the SDP Controllers a summary of approved Accepting SDP hosts along with any obligatory, but optional policies for secure communications.
7. Finally, the Initiating SDP host triggers a VPN connection to Accepting SDP hosts it is permitted to.

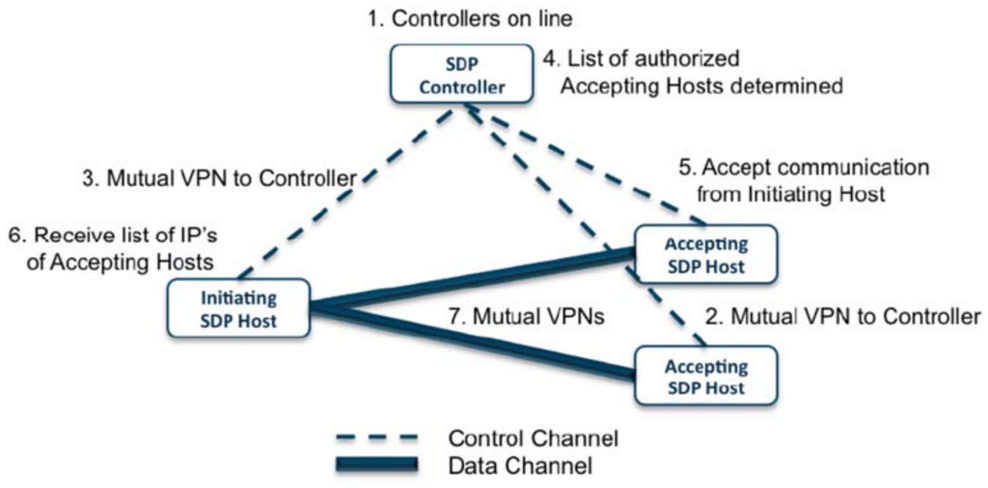

1. Controllers on line
SDP Controller
4. List of authorized Accepting Hosts determined
3. Mutual VPN to Controller
5. Accept communication from Initiating Host
6. Receive list of IP's of Accepting Hosts
Accepting SDP Host
Initiating SDP Host
7. Mutual VPNs
2. Mutual VPN to Controller
Accepting SDP Host
Control Channel
Data Channel

Figure 3.10: Architectural communication process of the Software Defined Perimeter [129]

3.12 Trust and Trust Models

Trust can be defined as the affiliation between two parties, where one party (trustor) is ready to count on the (expected) actions performed by the second party (trustee). In other words, the trustor is the evaluator while the trustee is being evaluated to determine its trust level. In the social sciences, trust can be attributed to relationships between individuals and an object or action; or within and between social groups (family, companies, countries etc.). In the domain of computer science, and particularly in sensor networks, trust is a complex term, which refers to the confidence, belief and expectation about the reliability, integrity, security, dependability and character of a sensor node. The cumulative trust value of a node is used in defining the reputation of a node, which is a quantifiable limit of the observable experience a node has with its neighbour either directly or indirectly. This is used in the decision of the limit of trust that a node will have towards its neighbour [130].

The authors in [131] have defined trust as the certain subjective possibility where an agent (node) examines a fellow agent (node) or group of agents (nodes) and believing they will perform a particular action as expected before it has the opportunity to observe the action within the context as it affects its own action. They classified trust into three categories and opined that trust could happen before a node's behaviour is observed (*apriori*) and that trust be viewed from the perspective of the trustor (the evaluating node).

The authors of [132] defined trust from a computational perspective, as the level of doubt and optimism regarding an outcome with the consideration and perspective of the individual. This is the aggregation of positive direct experiences among nodes and this count in the trust build process.

Trust can be classified further according to time and conditions. *General trust* is the trust an agent or node has in another without any bias to any particular condition. *Situational trust* is the trust an agent or a node has in another due to a peculiar experience or situation [133].

In the field of communications and networking the concept of trust is an attractive topic as trust can be embedded in communication and network protocol designs. Cooperation and collaboration are considered critical in the development of trust relationships among participating nodes as these determine the scalability, survivability, dependability and secure operations of the network. Trust models under different titles have been proposed in various literatures and a summary of trust models is presented in Table 3.4 based on their categories, features and weaknesses.

3.12.1 Trust Models

Trust modelling is the practice of using trust in the evaluation of a system. It pinpoints the concerns that could affect the trust of a system while helping to identify areas where a low value of trust could degrade a system's operational efficiency and usability. Trust modelling assists in developing functional measures, which can be adopted in a system to make it more trustworthy for users. Various trust models have been proposed and covered in several literatures [134, 135].

Table 3.4: A summary of trust models applied in secure network routing in sensor networks

Trust model type	Description	References
Bayesian trust model	This model utilizes the Bayes theorem in arriving at the truth of a value using probability distributions. The theorem is utilized as part of a precise method for statistical inference. The Bayesian theorem expresses how a subjective degree of trust should realistically change to be considered as an evidence.	[136-140]
Game theory trust model	This model relies on strategic decision making which normally involves two or more players wanting to reach an optimal solution in this case the best trust value that can be obtained in making a decision. It involves the use of mathematical models to make intelligent and rational decisions.	[141-144]
Entropy trust model	This model takes precisely given data among the nodes and with some probabilistic distribution, it considers the set of all trust values of all the nodes and computes their values using probability distributions. From those values, it considers the one with the maximal information entropy (trust) and this value is used as the trust for making a decision of the best route to follow.	[145-147]
Fuzzy trust model	This trust model relies on a form of multi-valued logic, which involves giving varying levels of values to a certain truth because of their variableness. This is used in comparison with the traditional binary logic where variables (trust) could assume either true or false (0 or 1) and not a variation of values.	[148-153]
Probability trust model	This model is concerned with probability distribution of values (trust) in even (normal) manner using the analysis of random phenomena. The essential entities of probability theory are random variables, and events (observable behaviour of nodes).	[154-157]
Neural network model	A concept using artificial intelligence to determine the behaviour of nodes in a sensor network. It is inspired by biological neural networks. It is a model that seeks to simulate the behaviour of a rational person while being applied to sensor nodes.	[158-161]
Swarm intelligence model	A model based on the communal behaviour of distributed, self-organized systems. This system employs a collection of agents (nodes) relating locally with each other within their neighbourhood. The idea is from the biological ecosystem of living things within their environment. Although there is no particular way the nodes are required to behave but with the random interaction among the agents a common trust behaviour is developed and adopted.	[162-165]
Directed and undirected graph model	A model derived from graph theory in mathematics, which represents nodes as a set of vertices and the network, or communication links as a set of edges connecting the vertices. Through some mathematical computation, the nodes are assigned values regarded as the trust levels between each node while the direction of the links determine the direction of communication between the nodes.	[166-168]
Arithmetic/Weighting trust model	This model considers product of trust as reputation. The product of past actions and observations are considered as direct reputation, an aggregated weight is assigned to determine the trust value of each node, and this is used to assess the authenticity of any information from a node.	[169, 170]
Markov chain model	Essentially a key management trust model. This approach evaluates the trust value and distributes trust certificates for key management. Trust values are evaluated based on Markov chain analysis whereby each one-hop neighbour's trust value is examined with respect to their past trust performance. The trust value is estimated and sent across all nodes. From the trust estimates, the node with the highest trust value is selected as the key management certification authority.	[171-173]

3.12.2 Quantifying Trust in IoT

Trust in IoT networks plays an important role in the development of a stable and secure configuration and it facilitates maintenance of such a network especially when the number of IoT nodes expands to such a number that it becomes challenging to manage or for the system to be informed about the replacement of failing nodes within the network. With the vast number of IoT nodes coming online, it is only pertinent that the formation, operation, administration and subsistence of an IoT network will be dependent upon the interrelationship and trusting capabilities of these nodes. This underscores the fact that trust formation among these nodes becomes imperative.

The use of various tools and technologies have been proposed. The use of cryptography to create trust confirmation and form trust and traditional protocols to exchange and distribute keys has been shown to be a computational expense on these IoT nodes, which are resource constrained [174, 175]. Therefore, the need for novel and innovative ways of securing route information among IoT nodes becomes very necessary.

Trust as a system for securing IoT networks has not been researched upon thoroughly and this highlights a very important and challenging area for further research. The authors in [134, 135] emphasized on trust level valuation of IoT objects. The authors assume that the smartest objects are human-aided or human-related objects, which are visible to the public and this makes them vulnerable to hackers. They remarked that smart objects are diverse in features and need to collaborate to work together. They defined trust using three communal relationships, which include *friendship*, *ownership* and *community. Friendship* is the relationship that exists in a social network system, the people who own the gadgets (*ownership*), and the gadgets that are part for some organisation (*community*). Malicious nodes target the basic functionality of IoT using some trust related attacks including: good-mouthing and bad-mouthing. The trust system for IoT discussed in [134] is a distributed, encounter-based and transaction-based system. Two IoT nodes in direct communication can evaluate one another and exchange trust values of their neighbour nodes. Thus, they perform an indirect assessment of one another (recommendation based trust). The authors specified the reference parameters for trust evaluation as collaboration, honesty and communal interest.

A fuzzy approach is presented in [176] which looks at trust evaluation founded on three layers: sensor layer, core layer, and application layer. The components of the sensor layer are physical devices such as WSN, RFID and base stations. The core layer is made of access network and Internet while the application layer comprises the distributed networks such as Cloud Computing, P2P, Grid systems, application systems and interfaces. The authors opined that users view IoT systems as a Service Provider (SP) in which the trust management system aims at delivering a supplementary service, which assists the IoT system in providing a more robust service to any Service Requester (SR). The relationship here is mutual as the trust system has an impact on both the SR (for privacy protection) and the SP. This trust management system has three steps including trust extraction, trust transmission, and trust decision making. The authors in their model showed that the requested information service and the trust-based service are integrated as one entity. They further opined that trust management should be a self-organizing component to provide efficient information flow and prevent any privacy information from finding its way to any untrusted SR. The authors further made use of fuzzy logic theory and a formal semantics based language in developing a layered trust mechanism that is assessed by using some defined characteristics (e.g. history, efficiency, risk). In this general framework, a user has access to the IoT system if the security particulars satisfy the security procedures, which are set via a decision-making system based on the user's trust value. The research work did not put forward any solid trust models, but it did establish a general framework, which could be utilised and integrated by other trust models.

The research in [177] recommends a trust management model for IoT which evaluates the trust level of an IoT node judging from the node's historical behaviour from various specific collaborative services. The purpose of this system is to maintain collaborative relationships in a heterogeneous IoT system while considering the capabilities of the nodes within the network using a decentralized method. This model assesses the direct observations and personal interactions with the nodes and indirect experiences and recommendations testified by neighbour nodes. In this trust management system, diverse stages are involved including: (i) the collection of information regarding the trust level of available nodes; (ii) Formation of a symbiotic service with the entreating nodes; (iii) Learning from its previous performances through self-updates for a better operation; and (iv) The assignment of a good recommendation point to the nodes after each positive interaction during the learning stage.

The authors of [178] considered the traditional access control models as been unsuitable for a decentralized and dynamic IoT setting in which node identities are unknown beforehand. They further emphasised that trust relationship among devices could help in driving the positive behaviour and performance of nodes. Services and resources could be shared among nodes when trust exists which is basically the same opinion expressed in [134, 176]. The paper also put forward a fuzzy approach to the Trust Based Access Control (FTBAC). Trust scores are computed by the FTBAC framework using features like recommendation, knowledge and experience. The scores are then linked to permissions while access requests are attached with a set of authorisations, which together establish a proof for grant of access to any node. The FTBAC framework comprises three layers, which are:

i. Device Layer: This consists of all the IoT nodes and devices and the intercommunications between them;
ii. Request Layer: This is the layer in charge of collecting knowledge, experiences, recommendations among the nodes and the computation of the trust values;
iii. Access Control Layer: This is the decision-making layer which maps the computed trust values to the access authorizations using the principle of least privilege.

The authors in their simulation results showed that the system ensured flexibility, scalability and energy efficiency. In addition, they submitted that a cryptographic model will achieve the same level of security, but will generate excessive overheads in terms of time and energy consumption.

In the work presented in [179], *recent trust* and *reputation management* models offered stiff and strict mechanisms in calculating reputation scores, and this has rendered their adaptation to the situation where they are implemented practically unusable. Although these *recent trust* and reputation models provided some features which are configurable they however, fall short of the diverse and dynamic IoT context requirements. They further proposed in their design a system, which selects and adapts to the most appropriate trust and reputation model in a heterogeneous system. They proposed their trust system could be applied to any process from a collection of other predefined systems while considering the current system's situation such as the allocated resources, and the number of users.

A proposal for a three-tiered layered IoT design for a trust management control system was put forward by authors in [180]. The IoT organisation is categorised into three layers including: sensor layer, core layer and application layer. Every layer is guided by a definite trust system using the policies of self-organization, multi-service and routing. The ultimate decision-making is made by the service requester (user) based on the aggregated trust information collected and using the requester policy. Semantics and fuzzy theory were used in computing the trust values of the system.

In [181], the authors proposed a system based on node behaviour and anomaly detection. The system routinely evaluates the nodes based on recommendation, history and statistical trust. The Bayes theorem is then applied to generate the trust value that is applied to the node for the determination of its trust level.

In conclusion, most research work on trust management embarked upon have been based on successful and unsuccessful node transactions, feedback and historical observation of nodes and this has led to the modelling of trust in sensor nodes (including IoT nodes) from a communication perspective. These distinct features (successful and unsuccessful node transactions, feedback and historical observation of nodes) have been utilized as key metrics in this thesis. Indeed, they form the basis of modelling, calculation and estimation of trust between IoT network nodes. Table 3.5 gives a summary of various trust management models, metrics employed and the attacks addressed.

Table 3.5: A summary of trust system models with the metrics used and attacks addressed

Trust model	Metrics employed	Attacks addressed
Dynamic Trust Management for Internet of Things Applications **[134]**	• Honesty • Cooperativeness • Community interest • Node recommendation	Self-promoting attacks, good-mouthing and bad-mouthing attacks.
Trust Management for the Internet of Things and Its Application to Service Composition **[135]**	• Honesty • Cooperativeness • Community-interest	Self-promoting attacks, good-mouthing and bad-mouthing attacks.
Distributed Trust Management Mechanism for the Internet of Things **[176]**	• Service • Decision-making • Self-organising	A general framework mechanism
Trust management system design for the Internet of Things: A context-aware and multi-service approach **[177]**	• Service: Service for which the node provides assistance; • Capability: Capability of node when assisting the Service; • Note: Node capability, Score given by the requester node to for evaluating the offered service; • Time: Time at which the service was obtained.	On-off attack, bad-mouthing attack, selective behaviour attack
A Fuzzy Approach to Trust Based Access Control in Internet of Things **[178]**	• Node recommendation • Knowledge about neighbour nodes • Node experience	No attacks specified but energy consumption was efficient.
Dynamic and flexible selection of a reputation mechanism for heterogeneous environments **[179]**	• Node behavioural information • Node scoring and Ranking entities • Node transaction levels with other selected entity • Reward/Penalty on nodes	A model proposed for adaptation into other systems to build trust in systems.
Trust management mechanism for Internet of Things **[180]**	• Self-organization policy • Multi-service • Routing across the IoT layers	A universal framework in which mature trust models could be used for integration.
TRM-IoT: A Trust Management Model Based on Fuzzy Reputation for Internet of Things **[182]**	• Successful node interaction with neighbour nodes • Energy level of node • Packet forwarding behaviour of node	Node selfish behaviour Routing information attack
A trust model based on fuzzy recommendation for mobile ad-hoc networks **[183]**	• Historical behaviour • Recent node behaviour • Node recommendation • Transitivity relationship • Node consensus opinion	Node selfish behaviour and better network performance
A dynamic trust model exploiting the time slice in WSNs **[184]**	• Success rate of exchanging • The probability of successful interactions • Packet loss rate • Energy	Selfish node attack and improved network performance
PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities **[185]**	• The feedback from neighbour nodes • The total number of transactions a node has with other peers • The credibility factor source of feedback received • Transaction context factor for discerning critical transactions from noncritical ones • The community context factor (community related features and susceptibilities)	Bad mouthing attack, good mouthing attack and Man-in-the-middle attack
LDTS: A Lightweight and Dependable Trust System for Clustered Wireless Sensor Networks **[186]**	• The number of successful and • unsuccessful interactions between nodes • Feedback reported by the cluster head nodes about specific nodes	Detects and prevent selfish and faulty cluster heads and malicious nodes. Consumes less memory and communication overhead

Hierarchical Trust Management for Wireless Sensor Networks and its Applications to Trust-Based Routing and Intrusion Detection **[187]**	Social trust: • Intimacy (for measuring closeness based on interaction experiences between nodes) • Honesty (for measuring regularity/anomaly) QoS trust: • Energy (for measuring competence) and • Unselfishness (for measuring cooperativeness)	Good-mouthing and bad-mouthing attacks
Statistical trust establishment in wireless sensor networks **[188]**	• Observation of neighbour's behaviour over the time • Accumulation of past behaviours over time.	Simple link node failures and Misbehaviour and energy conservation
Trust Management in Mobile Ad Hoc Networks Using a Scalable Maturity-Based Model **[189]**	• Node behaviour towards a specific neighbour node • Recommendation appraisal of the trust level of a neighbour node based on age and relationship maturity	Bad-mouthing attacks and node selfish behaviours efficient energy consumption

3.13 Summary

The desirability of IoT emanates from the pervasiveness of the enormous number of IoT sensor devices that are embedded into and continuously transmit information about the physical world, such that users can interact with the physical world much like is being done with the virtual world of the World Wide Web (WWW). Disappointingly, this pervasiveness also brings with it grave security challenges that requires swift and efficient solution for the IoT revolution to be a global success. In addition, the unique characteristics of IoT sensor nodes such as: open wireless medium, dynamic and mobile network topology, infrastructure-less network system, distributed cooperation and limited device resources exhibit higher security problems than the traditional wired networks. This becomes a worrisome issue especially with global growth of IoT and its application and the open platform for rogue attacks especially in incapacitating the routing functionality of IoT networks. A variety of attacks has been identified as reviewed above, various secure routing systems have been discussed, and weaknesses identified while there is also a continual threat of future attacks on IoT networks.

Secure routing plays an essential role in guaranteeing the security of a network and this makes it a nontrivial matter in the stabilization of IoT networks. Thus, several research gaps are yet to be addressed in the security of IoT routing. As [75, 190] noted, more research work is needed in the area of trust-based routing to effectively detect and defend against previous and future threats while providing secure and efficient routing in sensor networks. In the light of the consequent reviews made, and gaps identified, this research aims at answering questions as highlighted in section 1.2. From the studies conducted and survey of metrics presented in Table 3.5, important metrics are identified and explored further in Chapter 4, were these metrics were refined and used as building blocks for the formation of trust relationship among IoT sensor nodes, which provide availability and integrity of secure routing data.

In summary, this Chapter presented a background review of routing in IoT and the problem of secure routing in IoT networks. The Chapter further explored the security issues that have come with the IoT revolution while some countermeasures based on trust models were presented. The trust models presented include models linked to the exchange of data, and of particular importance for the work presented here is, work on trust models that can be used during the exchange of control information used to create and secure the routing path of IoT network protocols.

In Chapter 4, a secure trust-based framework is proposed, which explores the successful packet interactions between the IoT sensor nodes - a reflection of IoT sensor nodes' trustworthy behaviour.

4 *SecTrust*: Design And Methodology

IoT incorporates various technologies including information technology, low-power electronics, cognitive sciences and communication technology. The advent of IoT has brought about the emergence of a newer information society and knowledge economy and hence, making it an applied research discipline. As a new and emerging area, not much research has been embarked upon especially in secure routing as designers, manufacturers and investors are rather keen on getting IoT up and running before worrying about the security aspects particularly in secure communication among the IoT devices.

This Chapter is devoted to the design and development of a trust-based framework that can be applied to secure IoT routing protocols. Specifically, this study proposes a design which implements and addresses the research questions presented in Chapter 1. In addition, a general methodology is presented upon which the system proposed is predicated upon.

The novelty of this framework is five-fold:

(i) Its simple design makes it appropriate for resource-constrained IoT nodes in a peer-2-peer distributed network.

(ii) It utilizes node neighbour information for operation hence, supports the concept of a distributed system.

(iii) The system offers scalability for a large sized IoT network through the efficient management and transmission of control/route data, and scalable algorithms for trust computation and isolation of malicious nodes.

(iv) It provides secure communication among the sensor nodes

(v) It is intended for networks demanding availability and integrity of secure control, and route packet exchange.

The concept and parts of the findings in this Chapter have been published in a peer-reviewed conference and journal [191, 192]. This Chapter however, provides a revised and extensive version of the papers presented.

4.1 Methodology for Investigation

This research explores a design science research method (DSRM) approach [193] for the implementation of a secure routing system for IoT. Design science has been recognized as a particularly appropriate methodology for this type of study as it offers a relevant and rigorous process for developing and evaluating artefacts, which solve systemic deficiencies, while communicating results to relevant audiences, and contributing to knowledge. Furthermore, design science involves the discovery of a solution to a hitherto unsolved problem or the discovery of an even more efficient solution to existing unanswered problems [194].

To demonstrate the application of DSRM to this thesis, an objective-centred approach to the Peffers *et. al.* DSRM is adopted since this thesis is stimulated by a need that can be addressed by developing artefacts to address the challenges. Figure 4.1 and Table 4.1 present a description of the various phases of the proposed research work. From Figure 4.1, a discussion of the lack of secure routing in IoT protocols is based on literature review presented in Chapter 3, which indicates the problem identification and motivation for the study. Secondly, the thesis proposes the main objective of providing a secure routing system to improve security in IoT routing at the network layer. Thirdly, the research study develops a trust and recommendation system that can be mapped to IoT routing protocols to provide secure routing while minimizing the impact on network traffic. The proposed system is code named *SecTrust* system. The fourth activity is a demonstration of the artefact created (*SecTrust* system) in the design and development phase that maintains availability and provides integrity of communication routes. A key and important phase is the evaluation of the artefact, which describes how well the artefact performs when compared with global trust systems in defending against malicious attacks. Likewise, the artefact is embedded in standard RPL protocol to observe and analyse its performance during routing and assess its potency against malicious attacks. This is demonstrated using simulation study and a live testbed.

The evaluation and communication phases provides the opportunity to refine further the objectives of the solution and the design and development of the artefacts. Iteration is a key feature in DSRM and in the design of this thesis, as it gives the needed supports for the *rigor* and *relevance* of the artefacts developed.

Within this scope, this research aims at designing and experimenting a dynamic trust system for securing IoT network nodes. The sections following describe the design using DSRM, basic assumptions, trust metrics employed and a schema for the trust system.

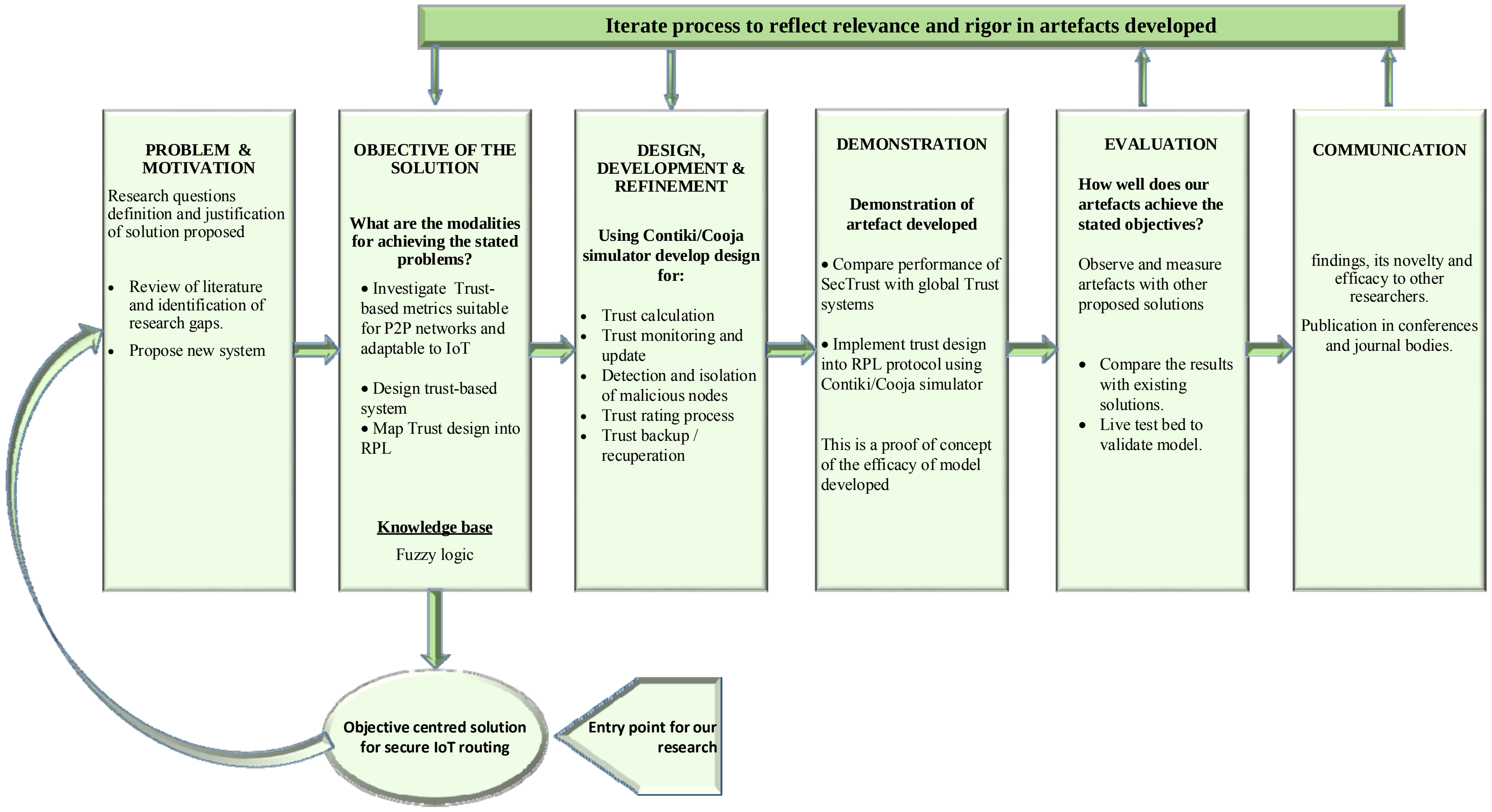

Iterate process to reflect relevance and rigor in artefacts developed
PROBLEM & MOTIVATION
Research questions definition and justification of solution proposed
Review of literature and identification of research gaps.
Propose new system
OBJECTIVE OF THE SOLUTION
What are the modalities for achieving the stated problems?
Investigate Trust-based metrics suitable for P2P networks and adaptable to IoT
Design trust-based system
Map Trust design into RPL
Knowledge base
Fuzzy logic
DESIGN, DEVELOPMENT & REFINEMENT
Using Contiki/Cooja simulator develop design for:
Trust calculation
Trust monitoring and update
Detection and isolation of malicious nodes
Trust rating process
Trust backup / recuperation
DEMONSTRATION
Demonstration of artefact developed
Compare performance of SecTrust with global Trust systems
Implement trust design into RPL protocol using Contiki/Cooja simulator
This is a proof of concept of the efficacy of model developed
EVALUATION
How well does our artefacts achieve the stated objectives?
Observe and measure artefacts with other proposed solutions
Compare the results with existing solutions.
Live test bed to validate model.
COMMUNICATION
findings, its novelty and efficacy to other researchers.
Publication in conferences and journal bodies.
Objective centred solution for secure IoT routing
Entry point for our research

Figure 4.1: Design science method applied to secure IoT routing adapted from [193]

Table 4.1: Design science method applied to secure IoT routing

DSRM ACTIVITIES	DESCRIPTION OF ACTIVITIES	KNOWLEDGE BASE
PROBLEM AND MOTIVATION	• Research question definition and justification of solution proposed • A thorough review of the literature as a foundation for a study of current trends in IoT and identification of research gaps. • Propose a system that will enshrine availability and integrity for secure routing in IoT.	• Analysis of current problems with IoT routing, current solutions and limitations • Link layer security
OBJECTIVE OF THE SOLUTION	• What are the modalities for achieving the stated solution to the identified problems? In addition to general objectives such as feasibility and performance, what are the specific criteria that a solution for the problem defined in step one should meet? • Design a trust based system capable of being embedded in IoT routing protocols such as: ○ RPL	• A consideration of computational trust theories as the knowledge base for achieving the goals will include: • Fuzzy logic
DESIGN AND DEVELOPMENT	• Design artefacts, which addresses questions, outlined. • Create constructs, models, methods that makes a novel contribution to the body of knowledge. • Design a trust calculation system • Design trust monitoring system • Design trust rating system • Design trust backup/ recuperation • Design system for the detection and Isolation of malicious nodes • Adopt default energy maintenance scheme	• Design and implementation of procedures, knowledge base and models to create an artefact that addresses the problem.
DEMONSTRATION (PROOF OF CONCEPT)	Demonstrate artefact developed addresses the problems outlined. This is a proof of concept of the efficacy of the model developed in guaranteeing secure routing in the IoT network. Comparison of *SecTrust* with renowned trust-based systems on varying scale of network size including: • EigenTrust • EigenTrust Incremental • No trust • TNA-SL	• Extensive know-how of how to use artefacts in solving the problems.
EVALUATION	• How well does the artefacts achieve the stated objectives? Observe and measure artefacts with other proposed solutions and comparing the objectives with observed results. • Embed *SecTrust* model to RPL protocol in Contiki/Cooja • Compare results with existing standard RPL protocol • Conduct live testbeds to validate model developed.	• Knowledge of trust metrics, motes, and evaluation techniques. Simulation study using Contiki/Cooja Testbed experimentation Sensor nodes (motes)
COMMUNICATION	• Communication. • Communicate findings, its novelty and efficacy to other researchers and relevant audiences. • Publication quality Ranked conferences and journal bodies.	Knowledge of the disciplinary culture.

4.2 *SecTrust*: Framework Description

SecTrust is a secure trust-based distributed peer-2-peer framework that can be applied to IoT in secure routing. Although it is being applied in secure IoT routing, *SecTrust* can however, be applied in other spheres that use the recommendation system like e-commerce, online shopping, and the social media. *SecTrust* identifies and isolates attacks and in this case, routing attacks. It essentially computes and evaluates the trustworthy behaviour of a node. A node's trustworthiness in *SecTrust* is characterized by the evaluation from its neighbouring nodes while providing service (recommendations) to other nodes. This trustworthiness reflects the level of reliability or dependability (trust) that a peer node has towards its neighbour. The trustworthiness of a node is evaluated as a time-based successful packet exchange between neighbour nodes and the positive packet acknowledgements, with a continuous observation between linked neighbour nodes. In this system, every node computes the trustworthiness of its direct neighbours based on the computed direct trust value and the recommended trust value. While neighbours with greater trust values are chosen for secure routing, nodes with lower trust values are categorised either as malicious, compromised, or perhaps selfish nodes that seek to preserve their resources like battery power. The proposed system is made up of five major processes: *Trust calculation process*, T*rust monitoring process*, *Detection and isolation of malicious nodes*, *Trust rating process*, and *Trust backup/recuperation process*. Figure 4.2 provides a flowchart representation of the *SecTrust* system highlighting all five processes.

From review of literature [134, 176-180, 182-184, 186-189] and critical examination of the research methods and design, key components were identified that formed the building blocks of the *SecTrust* system and these components have being organised and modelled into five identifiable processes as mentioned above with each connected to one or more processes. A modular concept is adopted in this framework. The concept behind the use of a modular process is to consolidate a complex trust-based system as a set of distinct and identifiable components that can be autonomous, while having minimal communications and transmission overheads between processes. The modular process employed in SecTrust facilitates manageability and ease of interoperability. This is in addition, to the fact that the distinct processes can be integrated together to create the SecTrust system.

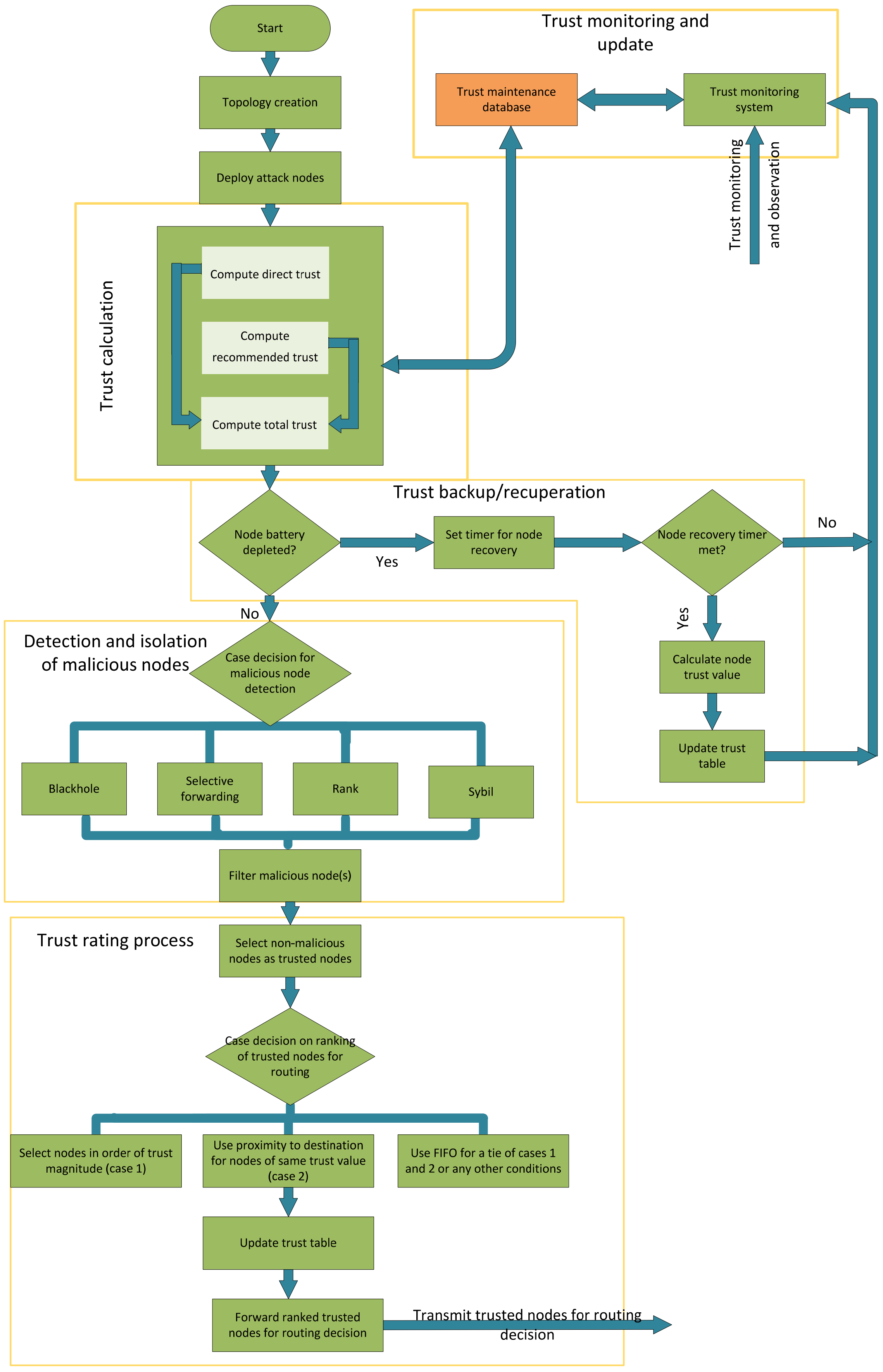

Start
Topology creation
Deploy attack nodes
Trust calculation
Compute direct trust
Compute recommended trust
Compute total trust
Trust monitoring and update
Trust maintenance database
Trust monitoring system
Trust monitoring and observation
Trust backup/recuperation
Node battery depleted?
Yes
Set timer for node recovery
Node recovery timer met?
No
Yes
Calculate node trust value
Update trust table
No
Detection and isolation of malicious nodes
Case decision for malicious node detection
Blackhole
Selective forwarding
Rank
Sybil
Filter malicious node(s)
Trust rating process
Select non-malicious nodes as trusted nodes
Case decision on ranking of trusted nodes for routing
Select nodes in order of trust magnitude (case 1)
Use proximity to destination for nodes of same trust value (case 2)
Use FIFO for a tie of cases 1 and 2 or any other conditions
Update trust table
Forward ranked trusted nodes for routing decision
Transmit trusted nodes for routing decision

Figure 4.2: *SecTrust* system

4.2.1 Quantifying Trust in *SecTrust*

Many trust properties have been employed for trust evaluation in Wireless Sensor Networks and these metrics find relevance for adoption among IoT sensor nodes. Some of these key performance metrics have been introduced in Chapter 3 (Section 3.12) and a summary was presented. Consequently, *SecTrust* adopts and refines some of these trust properties identified in Section 3.12 of Chapter 3 for computation, evaluation and trust formation among IoT sensor nodes.

Moreover, *SecTrust* uses the concept of fuzzy logic as a threshold-based trust broadcast [195]. The concept is to broadcast trustworthy nodes to neighbour nodes in the network by, i) maintaining effective communication only among trusted nodes and, ii) ensuring the broadcast of only trustworthy information to neighbour nodes in the network. *SecTrust* ensures this by validating the trustworthy nature of every forwarding node in the network and by securing the source of route information transmitted.

Furthermore, a method is provided to rank the best trust formation to optimize the performance of secure trust-based communication among sensor nodes. The *SecTrust* system observes various trust properties while assigning different weights to them. It is particularly important that more weight is assigned to the evaluation of a node at the current time rather than its observable history. This is necessary to defeat malicious nodes seeking to build a good reputation over time and later commence the perpetration of their malicious activities. The *SecTrust* system demonstrates that trust converges to the extremes (i.e. highly trusted nodes tend to maintain high trust values while malicious nodes quickly tend to low trust values).

Finally, a repository of nodes with high trust values are formed while the trust value between any two adjoining nodes that do not optimally trust each other tends to zero. A penalty value is introduced for misbehaving nodes that seek to subvert the system. A detailed description of the *SecTrust* process and its components are presented in Section 4.2.2.

4.2.2 *SecTrust* Process

SecTrust is a composition of five systemic processes that operate in unison to provide trustworthy nodes that are ranked in the order of trust magnitude to provide secure communication and isolate untrusted nodes. Nodes classified as trustworthy are selected and used for secure communication. When applied to routing in IoT, selected nodes are incorporated for secure routing. A description of the inner workings of each process is presented below.

1. Trust Calculation Process

In the design of *SecTrust*, a trustor node evaluates a trustee node. It uses the trust value derived to judge if, the trustee node is sufficiently reliable to fulfil an assigned mission as assessed by the fuzzy based five-tuple trust level band specified in Table 4.2. More specifically, the trust computation represents reliability, dependability, competence, and successful positive interactions or positive recommendations on performance of tasks sent by nodes from direct or indirect interactions with other nodes. *Direct* and *recommended* trust relationships could exist between two or more nodes. Each node gathers the *direct* and *recommended* trust values of directly and indirectly linked neighbours. Here, the two trust types are described. Figure 4.3 illustrates the relationship between *direct* and *recommended* trusts. In *direct trust*, node *i* computes the trust value of node *m* since it has a direct relationship with node *m*. Thus, when there is a positive direct interaction between node *i* and node *m* trust is established and a value is computed (i.e., node *m* successfully forwards packets sent by node *i* to either the destination or the next trusted hop node). In the context of Figure 4.3, *recommended trust* is described as follows. Since, node *i* has a direct relationship with node *m,* and node *m* has a direct relationship with node *j*, it follows that node *m* will make available its direct trust value of node *j* to node *i*, so that node *i* could use it to deduce the final recommended trust value of node *j* as specified in Equation (4.2).

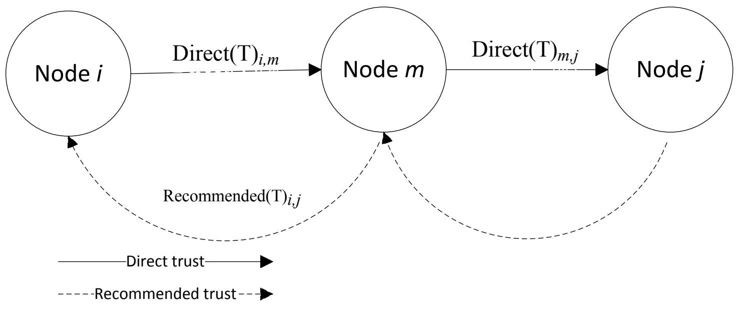


Figure 4.3: Direct and recommended trust relations

The *direct trust* between two nodes is therefore, defined as the positive interaction between nodes. This positive interaction is described as the determinable packet forwarding behaviour of a node towards its neighbour(s) in real time with a penalty for any misbehaviour [77]. This relationship is expressed in Equation (1) below.

$$DT(N_i, N_j) = \frac{PF_{ji}(t)}{PF_{ji}(t) + \beta[PS_{ij}(t) - PF_{ji}(t)]} \qquad (4.1)$$

From Equation 4.1, the direct trust relationship between two nodes N_i and N_j, is computed as DT(N_i,N_j), where node N_i keeps three records of its neighbour nodes. These records include: *i)* Time *t* is defined as the time during the evaluation of node *j*; *ii)* $PS_{ij}(t)$, the total number of packets sent to N_j by N_i; *iii)* $PF_{ji}(t)$, the total number of packets forwarded by N_j on behalf of N_i. Therefore, the likelihood of positive interactions or feedback between N_i and N_j, is given by DT(N_i,N_j), which evaluates whether N_j is sufficiently truthful to forward packets on behalf of N_i. To ensure nodes, behave as expected, a penalty (β) is introduced to penalize any misbehaving nodes. Higher β values are attached to nodes with lower trust values while nodes with higher trust values receive little or no penalty. β is a constant, which defines the penalty weight applied to any misbehaving node. During the first evaluation of a node, the initial β value is 1, which gives every node the benefit of assuming they are well behaved. However, the subsequent evaluation of nodes is based on the trust threshold boundary that a node was previously at. Table 4.2 shows the trust thresholds and the β values. In Table 4.2, a higher penalty value is attached to misbehaving nodes with lower trust values.

In the context of Figure 4.3, the recommended trust value of a node is as shown in Equation (2) below.

$$RT(N_i, N_j) = DT(N_i, N_m) * DT(T_{m,j}) \qquad (4.2)$$

$RT(N_i, N_j)$ indicates the recommended trust value and is computed as the *product* of the direct trust value forwarded by node *m* ($DT(T_{m,j})$) , which exist between m and j, and the direct trust value of node *m* as evaluated by node *i* ($DT(N_i, N_m)$). In the design of *SecTrust*, a differentiation is made between recommended trust and direct trust. For example, with reference to Figure 4.3, node *i* computes the recommended trust value of *j* as the product of the direct trust value between node *i* and node m and the direct trust value between node m and node *j*. Computing the recommended trust value in this way accounts for a possible trust decline in space due to time.

In general, a node can have a fairly precise trust judgement regarding its next hop neighbours using observation, eavesdropping and snooping methods. For nodes beyond

1-hop, a node must depend on trustworthy recommender nodes for accurate trust evaluation toward the distant node of interest. *SecTrust*'s aggregation of recommendations is based on a weighted calculation of the trustworthiness of the recommender node. Additionally, a trust aggregation system is used by the trustor node to compute the trust value of its neighbour node at time *t*, with the objective of measuring at runtime the computed trust value of its neighbour node against the threshold of the fuzzy trust tuple provided. In the *SecTrust* system, in addition to using both direct observations and recommendations for trust computation and aggregation, a novelty incorporated in the design is its ability to make adaptive changes relative to the prevailing environmental conditions. As an example, with nodes moving at high speeds in a network, computing direct trust may become very ineffective due to nodes joining and leaving the network in a very rapid fashion. Under this condition, the trust system can subjectively reduce the weight attributed to direct trust while increasing the weight associated with recommended trust to have a more balanced assessment of the trust values of nodes.

2. Trust Monitoring Update Process

This process is an observatory and update phase where every node gathers the trust information of their immediate and distant neighbours based on (a) direct relationships and (b) the recommendations between the nodes. It also updates the trust values of nodes. In updating the trust values of a node, *SecTrust* employs two methods namely: periodic trust update and reactive trust update.

i. The periodic trust update is based on a given set time when trust values are re-computed to have an up-to-date trust value. This thesis adopted the RPL protocol implementation of the trickle timer for sending DIO messages (Chapter 6 explains more on the DIO trickle timer).
ii. The reactive trust update method is intended for the spatially connected and distributed IoT nodes that trigger route updates and are hence, reactive based on their sparse communication links.

The frequent update of the trust database can increase communications overhead and hence, impact a protocol's performance. To mitigate this, *SecTrust* assumes an existing protocol's implementation of its routing table update as the basis for its trust update. A low frequent trust update rate will mean that some node actions will not be captured and as result, a low trust convergence rate will result in the network. Contrariwise, if the trust update rate becomes too frequent, it may take up excessive network resources like the energy, memory and CPU cycles of node and thus, culminate in a shortened network lifetime for the network.

3. Detection and Isolation of Attacks Process

The most important task of any secure information system is to guarantee the confidentiality, integrity, availability and authenticity of the information transmitted. This issue becomes challenging especially in IoT networks for reasons presented in Chapter 3. This research focuses on the detection and isolation of insider attacks that are rather more difficult to detect because the malicious nodes form part of the network and are privy to every detail of the network process. Specifically, this thesis focused on the following attack models in IoT networks:

i. *Packet drop attacks*: In an IoT network, a malicious node can broadcast itself as having a short path to a proposed destination during a route discovery, but as soon as it receives the packets, it begins to drop the same. Malicious nodes could adopt various techniques in dropping these control/route packets. In some cases, they could plainly drop all packets (Blackhole attack), they could selectively drop the packets (Selective Forwarding attack), or still they could randomly drop the route packets (Greyhole attack). This research study uses overhearing (a node listens to its neighbour to determine if the neighbour could fulfil its forwarding requests) and monitoring methods to determine if packet have been dropped by neighbour nodes.
ii. *Rank and Sybil attacks*: In this attack model, a malicious node advertises a spuriously beneficial routing path by changing its Rank thereby attracting neighbour nodes to route their traffic through it. Packets routed through this malicious node are either dropped, selectively forwarded to their destination, sent to other colluding nodes, or are used to divulge confidential information of the network. The success of a Rank attack depends largely on the collaborative nature of the attacks (e.g., a Rank attack node colluding with a Blackhole or a Sybil attack carried out to drop or falsify the route information and thus, disrupt the network route topology). This study uses node overhearing and monitoring methods to determine anomalous route traffic toward a node as possible indication of a Rank attack or colluding attack.

The study considers that *SecTrust* being a Trust and Recommendation system, it could be subjected to trust related attacks hence, various mechanisms have been implemented in

SecTrust to defend against such trust related attacks. A brief discussion of the trust related attacks is presented below.

a. *Sybil attacks*: A malicious node perpetrating this type of attack masquerades as several entities. The node could further launch Byzantine attacks on the network. This creates an illusion that there are many malicious nodes operating within the network thus, overwhelming the network and thereby disrupting the topology of the network. In *SecTrust*, the high weight attached to the time-based evaluation of every node will defend against this type of attack. Since, its neighbour will, indeed consider any node that masquerades as a new node, and hence, upon evaluation at current time, the node will have a low trust value since its forwarding behaviour will tend to zero, as a result of being new in the network. This way a Sybil node masquerading will never be considered for secure communication.
b. *Good-mouthing/ bad-mouthing attacks*: In a colluding attack, a malicious node could endorse another misbehaving node as a good node by attributing a high trust value as its recommended trust value (good-mouthing) while recommending a good node as being bad and malicious hence, reporting it as having a low trust value (bad-mouthing). Moreover, a malicious node could carry out random attacks to avoid being detected. *SecTrust* defends against this type of attacks by, i) giving a higher weight (80%) to the current trust value of the recommender node rather than the recommended node, ii) if the current trust value of the recommender node falls short of the trust threshold required, both the recommending and the recommended nodes are discarded and alternative nodes are sought. However, if the recommender node's trust value is above the trust threshold then, the computed trust value of the *recommender node* is multiplied with the computed direct trust value of the *recommended node* to get the final weighted computed trust value of the *recommended node*.
c. *Self-endorsing/white-washing attacks*: A malicious node may seek to recommend itself to gain a high trust value however, the *SecTrust* system defends against this, since it does not incorporate, as part of its metric, a *self-recommendation* system in the computation of trust values. In addition, every node is evaluated by its neighbour at current time to know its trustworthiness in the network.

4. Trust Rating Process

The *SecTrust* design employs the concept of a fuzzy threshold-based trust rating system. The idea is to use high-quality trusted nodes only during routing decisions. After the trust values, have been determined and for proper judgement, a trust rating system is further adopted to Rank the nodes per their trust magnitude during the trust calculation process. This helps in delivering only highly rated (trusted) nodes for the purpose of secure communication. This further helps in detecting and eliminating misbehaving nodes, which seek to adjust their trust values maliciously (e.g. Blackhole, greyhole, and man-in-the-middle (MITM) attacks).

To estimate the trust values, this research study advances a five-tuple trust level system and deduce its trust degree by using fuzzy judgment. A five tuple trust level is assigned to each node, and it is defined thus: $V = [v_1, v_2, v_3, v_4, v_5]$.This is expressive of the various trust levels as: *["no trust","poor trust","fair trust","good trust","complete trust"]* respectively and this is represented in table 4.2.

Table 4.2: A five-tuple trust level band

Five tuple (*V*)	**Trust level**	**Range of positive relations (%)**	**β**
v_1	No trust	≥ 0 and ≤ 25	0.03
v_2	Poor trust	≥ 26 and ≤ 50	0.02
v_3	Fair trust	≥ 51 and ≤ 75	0.01
v_4	Good trust	≥ 76 and ≤ 90	1
v_5	Complete trust	≥ 91 and ≤ 100	1

SecTrust considers nodes in the v_4, and v_5 tuple as perfect nodes for secure routing while nodes in the v_3 tuple, have some mixed-element behaviour. *SecTrust* does not rely on nodes in the v_2, and v_1 tuple and hence, avoids using nodes under these classifications for secure communication.

5. Trust Backup/Recuperation Process

It is possible that some nodes could have the following peculiar problems such as battery depletion, collision and signal interferences, temporary network link loss and thus, the trust system could list them as malicious and/or insecure even though they are trustworthy nodes. This could lead to the nodes having low trust values and hence, being isolated from the secure communication process. This is expected since these nodes will still behave selfishly due to lossy links or they will seek to conserve their battery power. The *trust backup/recuperation* process addresses this by placing the affected nodes under observation for a period of time so they could recoup their energy.

In the recuperation routine, a node, which has depleted its energy is kept under observation, and its details are stored. Once the node is recovered, the node is assigned a basic trust value in the v_4 category (80%) for re-integration into the network. From then on, trust computation and evaluation rules apply. They could continue to boost their trust values over time otherwise, if their trust values decline below the threshold they are considered insecure and are isolated.

4.3 Energy Level of Nodes

Most nodes in an IoT network are small devices constrained in terms of buffer size, energy, bandwidth, and computational ability. A consideration of their limited resources is essential when studying the performance of IoT networks. In battery powered IoT devices, energy is viewed as a vital metric in ensuring that devices can operate for an optimal period. *SecTrust*, through the estimation of the battery energy level of a node, determines if a node could accomplish a given task such as forwarding a packet. To underscore the importance of the scarce resources of these nodes, a node could act contrarily in a routing decision if its battery level is low or its buffer size is almost occupied thus, it declines offers except critical requests for urgent packets. Also, a node with a high lossy link, very low bandwidth or a limited processing speed could experience extended delays in packet delivery. A node without sufficient resources is isolated and allowed a period to recoup its exhausted resources, especially battery power, before it can be reconsidered for re-integration into the network. Since most IoT nodes are energy constrained thus, using an energy-conserving scheme will be beneficial to the operations of the IoT nodes. However, this study does not adopt any specific energy model but uses the default energy scheme provided on the platform for simulation.

4.4 The Resiliency of *SecTrust*

Malicious nodes do carry out diverse attacks to disrupt the proper functioning of a network. The goal of the *SecTrust* system is to achieve resiliency when confronted with the aforementioned attacks. In investigating the resiliency of *SecTrust*, an evaluation of the trust computation, trust recommendation and detection/isolation of malicious nodes are examined.

In the design of *SecTrust*, direct trust is computed based on the packet forwarding behaviour of nodes and this is observed through monitoring and observation. Moreover, a penalty (β) is also applied to every misbehaving node in the network. The penalty increases in a progressive fashion for a node as its trust value plummets - an indication of a node's continuous misbehaviour in the network. Furthermore, in *SecTrust*, recommendations are weighted by a recommender's direct trust value. Thus, if a malicious node gives a good recommendation about another malicious node, the good recommendation forwarded by the malicious node are discarded due to the recommender's low trust value. *SecTrust* uses purely trustworthy nodes as recommenders. Even at that, the recommended trust value is computed by multiplying the recommender's trust value with the trust value of the recommended node to derive the final trust value of the recommended node. This further tightens the gap for malicious nodes to filter through undetected.

SecTrust analyzes traffic patterns and node behaviour to pre-empt if specific nodes are misbehaving in the network. Every node under suspicion is observed under the *trust update and monitoring* process. Even if the nodes are not malicious and are probably trying to conserve their depleted resources they are still being observed. A continuous observation of these nodes reveals if they are malicious or just plain depleted of their resources and a course of action is taken as specified in the *SecTrust* process in Figure 4.2.

4.5 Evaluation of *SecTrust*

In demonstrating the application of the principles of DSRM's *Rigor* and *Relevance* to the SecTrust system, the first design of SecTrust as shown in Figure 4.4 was presented at the 14th IEEE International Conference on Pervasive, Intelligence and Computing conference where positive feedback was given that led to the improved version presented in Figure 4.2.

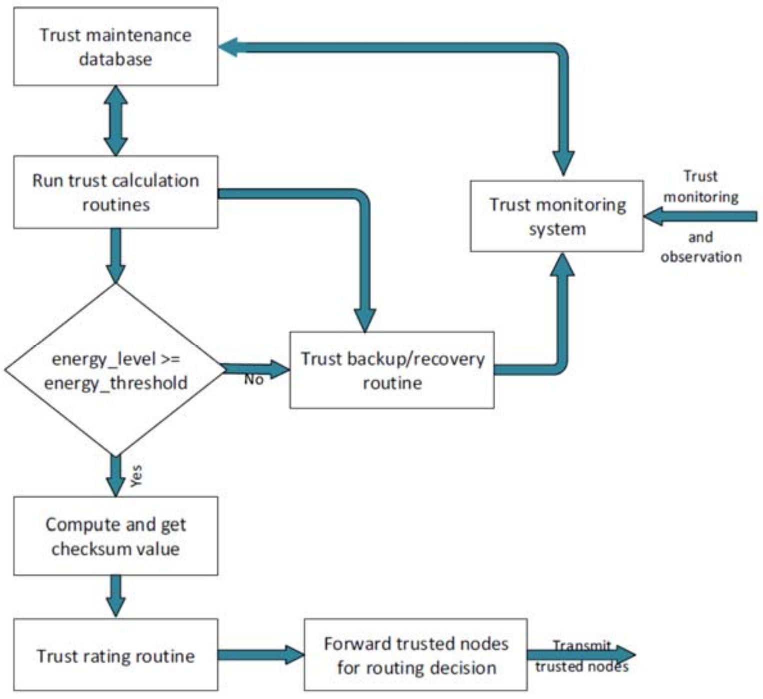

Trust maintenance database
Run trust calculation routines
energy_level >= energy_threshold
No
Yes
Trust backup/recovery routine
Trust monitoring system
Trust monitoring and observation
Compute and get checksum value
Trust rating routine
Forward trusted nodes for routing decision
Transmit trusted nodes

Figure 4.4: The First SecTrust Topology Framework as presented in [191]

In addition, three evaluation processes are presented that brings together the elements of the formal methodology as depicted in Figure 4.1. Each evaluation task, given its unique environment, measures the performance of the various components for every performance metric under study.

The first evaluation process, which is a design evaluation of *SecTrust* system against other global trust systems using a P2P simulator, a global trust, and a reputation management algorithm. The simulator has been designed to facilitate the deployment of other trust systems like *SecTrust* for a comparison with other trust and reputation systems. Figure 4.4 illustrates the evaluation process for testing *SecTrust*'s design. In the Figure, *SecTrust* is first evaluated using the P2P Trust simulator; results are observed, analysed and presented. The analysis and presentation of results is covered in Chapter 5.

The second process simulates the implementation of *SecTrust* in RPL using the Contiki/Cooja platform to observe its performance as also illustrated in Figure 4.4. In this evaluation process, *SecTrust* is embedded into the RPL protocol to create a new version of RPL tagged *SecTrust-RPL*. The simulation is observed and compared with the standard RPL protocol while the results are collated and discussed in Chapter 6.

The third process is the design of a testbed which embeds the *SecTrust*-RPL system into the physical IoT nodes using the AS-XM1000 sensor motes. The testbed is an experiment that models the real-world scenario of a smart home. A key reason for using a testbed for further evaluation is that, it will model and react according to the natural environment - a key factor that most simulators may not be able to model accurately. The results are presented in Chapter 7.

In all three evaluation platforms presented in Figure 4.4, there is a feedback continuum for the refinement of *SecTrust* through the application of *rigor* (optimizing the *SecTrust* system to meet up with the three rigorous evaluation systems proposed) and *relevance* (improving the *SecTrust* system to the relevant IoT case study proposed in this research) to optimize the *SecTrust* system. The observed results across the three evaluation platforms have been presented in scholarly publications.

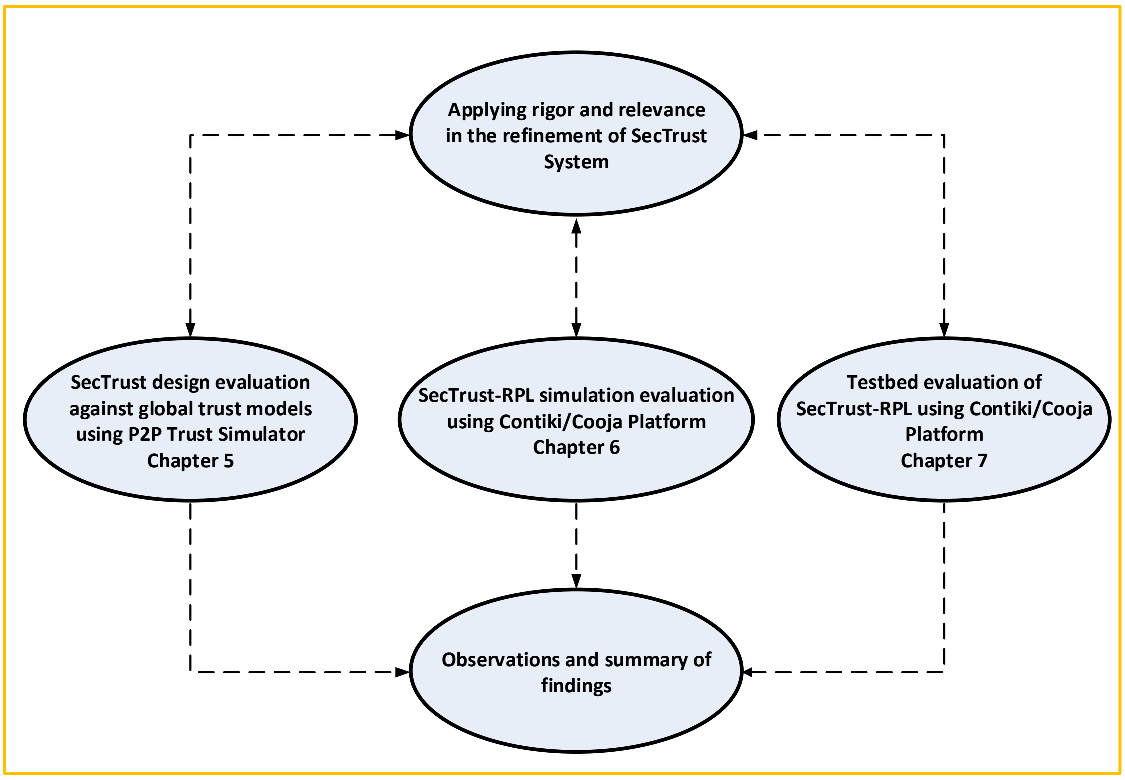

Applying rigor and relevance in the refinement of SecTrust System
SecTrust design evaluation against global trust models using P2P Trust Simulator Chapter 5
SecTrust-RPL simulation evaluation using Contiki/Cooja Platform Chapter 6
Testbed evaluation of SecTrust-RPL using Contiki/Cooja Platform Chapter 7
Observations and summary of findings

Figure 4.5: Evaluation process of *SecTrust*

4.6 Topology and Deployment Scenario for Evaluation

Home and smart building automation are a few fundamental application areas of the future Internet of Things. It is expected that sensor nodes operating in such networks will be constrained in terms of network and computing resources. These sensor nodes are engaged in various roles while performing different automation tasks. Some of these sensor nodes are deployed today in light dimmers, HVAC systems, window shades, motion sensors, refrigerators and remote-control units.

In every sensor network setup, there is always a controlling unit referred to as the sink node or root node or controller node. This node communicates with the other local sensor devices via the wireless medium. In general, two types of network architectures could be used in the case of the smart building network proposed. They are:

i. Centralized architecture where each device communicates with the controller node, which makes the decision in the network.
ii. Distributed architecture where each device is capable of communicating with other devices and make requests for a process execution [196].

This research study adopts the centralized architecture since every device is expected to communicate with the controlling unit (root or sink node). This research study acknowledges that an exhaustive evaluation of various testbed scenarios would be too daunting to undertake thus, a simplified testbed design scenario of a smart home is presented that will allow for the evaluation of the secure communication details among sensor nodes. To better provide a realistic evaluation process of a practical environment, in the presented model of a smart home/building, a measurement of 70m^2 and 30m^2 in the simulation and the testbed experimentation were used respectively. This will give a fair evaluation when computing the standard deviation or variance of results observed during both simulation and testbed experiments. Figure 4.5 shows a topological arrangement of nodes consisting of 1 controller node (sink), 26 sending nodes and 3 attacking nodes in the proposed smart building/home configuration. This thesis adopts the smart building home plan as a background for evaluating *SecTrust* on a fixed sized low power and lossy network (LLN). Today, networks of 20 nodes or more exist although the number of sensor driven home appliances are expected to grow to more than 500 nodes per home [196].

Topologies used for the study are generated with the supposition that a device is placed uniformly in a home, on the walls, on the ceiling and at some stationary positions representing some typical immovable devices like a refrigerator, microwave oven, coffee maker, TV set and lights. The number of IoT sensor devices deployed across the smart building is proportional to the requirements of a typical smart home. For the network topology, a controlling unit (sink) is placed in the living room. Besides, about 85% of devices are logically regarded as sensor or sending nodes while 10% of the total number of nodes will act as malicious nodes within the network. It is further assumed that, malicious nodes will act from outside the building hence, in Figure 4.5, which depicts the topological representation of the deployment of the nodes the malicious nodes indicated in red are clearly outside the perimeter of the building whereas, the controller and sensor/sending nodes are all within the building. The nodes are typically fitted with the following: monitoring sensor light, temperature and humidity sensors. Figure 4.5 gives a detailed deployment of a 30-node layout plan for the simulation and testbed evaluation. The topology in Figure 4.5 was used for the simulation study. For the testbed experiment, 14 AS-XM1000 motes were acquired and used. Although the number of motes for the testbed does not match the number of nodes in the simulation study, however, the testbed was scaled accordingly to justify result findings. This research study submits that, these tasks are repeatable, consistent and without loss of generality during the simulation study and testbed experiments respectively.

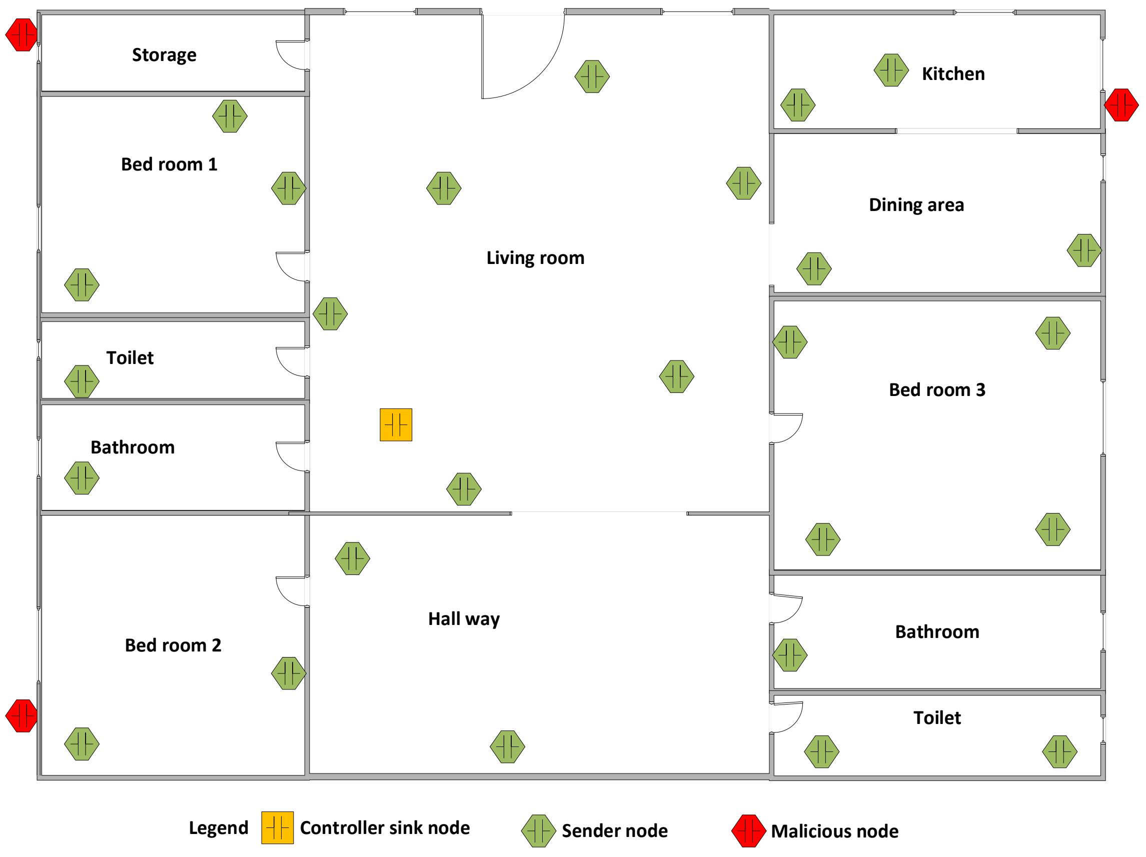


Figure 4.6: A 30-Node topology plan of a smart home with 1 sink node, 26 sensor nodes and 3 malicious nodes

4.7 Summary

A secure trust management system (*SecTrust*) for guaranteeing secure communications is proposed and specific research questions have been outlined from the outset in Chapter 1 and a suitable research method has been presented in this Chapter. The actualization of the proposed thesis study was conducted and validated thus:

i. A Trust-based test comparison against other global trust models was conducted using a suitable framework to benchmark *SecTrust* performance.
ii. *SecTrust* was embedded in the RPL IoT routing protocol (*SecTrust*-RPL) using the InstantContiki platform to simulate and gather the performance measurement of *SecTrust*-RPL protocol against the standard RPL routing protocol.
iii. A live testbed implementation was conducted to validate further the simulation performed in ii.

The process of this research thesis study has been constructive starting with problem formulation, trust solution modelling and evaluation, and validation of the research.

In Chapter 5, multiple attack models, as studied above are examined in the light of the performance of *SecTrust* against other global trust and recommendation systems.

5 *SecTrust* Vs Global Trust and Recommendation Systems: Design And Performance Comparison

Trust and Recommendation systems are used in distributed and peer-to-peer networks to assess mutual trust between parties based on expected belief in a party, experiential behaviour, and current response or feedback of a party. The derived trust values affect how, or with whom, an evaluating party must interact with. *SecTrust*, a secure Trust and Recommendation System was proposed in Chapter 4. This Chapter proposes an evaluation framework, which compares the *SecTrust* system with other global trust systems using the Quantitative Trust Management (QTM) framework [1]. The QTM framework delivers interpretation of policies, and the determination of access control based on the plan of action of the dynamically changing reputations of entities in a system. It is developed as a simulator, which incorporates incremental trust computation methods that permits the simulation of networks from small to substantial sizes. The QTM simulator emulates various configurations of network scenarios with specific considerations for malicious behaviour in diverse network settings. This framework [1] is provided as an open source, so that researchers could extend, compare and evaluate the effectiveness of their trust systems against the trust and recommendation systems available in the framework. QTM framework has the following trust and recommendation systems incorporated into it, and they include: EigenTrust, EigenTrust incremental, Trust Network Analysis with Subjective Logic (TNA-SL) and No Trust reputation systems.

The design and methodology of the *SecTrust* system was presented in Chapter 4. The goal of this Chapter however, is to embed the *SecTrust* system into the QTM simulator so that a detailed performance comparison could be performed between *SecTrust* and other global trust and recommendation systems specified in the QTM framework.

This Chapter achieves the DSRM's *rigor* and *relevance* testing of the *SecTrust* system through the design testing of *SecTrust* against other global trust systems. Figure 5.1 illustrates the design evaluation testing of *SecTrust* system.

The sections following explore these notable global trust systems; this is followed by an introduction of the QTM P2P-simulator and a description of the implementation of the global trust system in the simulator. A description is also given on how *SecTrust* has been implemented using Java. Finally, simulations are performed, which compare the performances of all the trusts systems while varying the network conditions like network size and the malicious node behaviour among others.

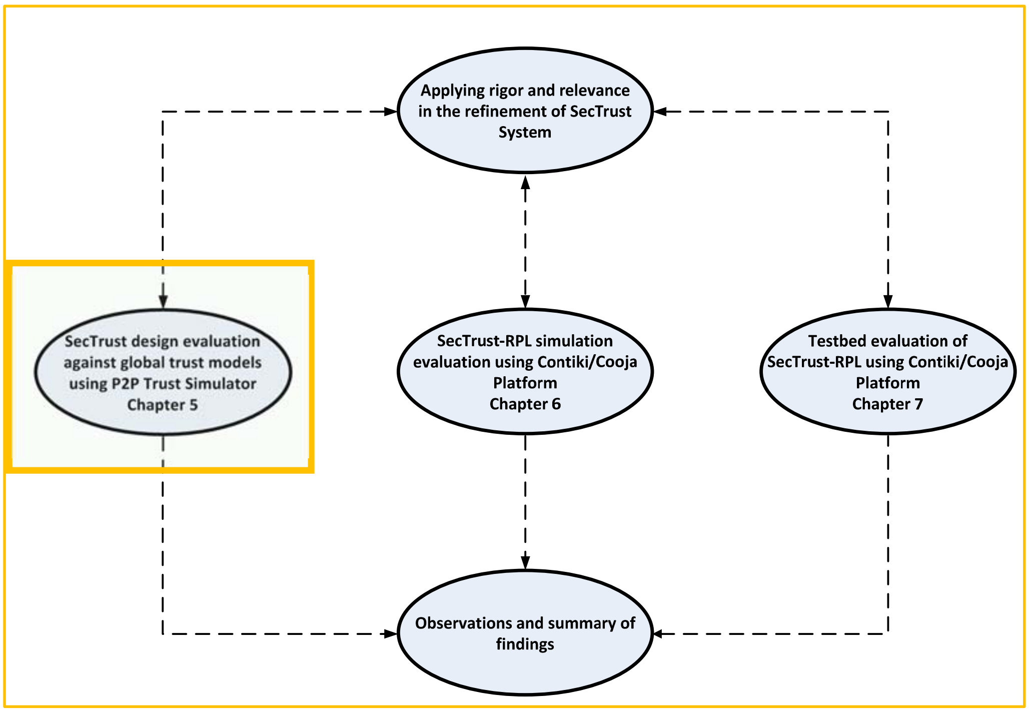

Applying rigor and relevance in the refinement of SecTrust System
SecTrust design evaluation against global trust models using P2P Trust Simulator Chapter 5
SecTrust-RPL simulation evaluation using Contiki/Cooja Platform Chapter 6
Testbed evaluation of SecTrust-RPL using Contiki/Cooja Platform Chapter 7
Observations and summary of findings

Figure 5.1: Design evaluation testing of *SecTrust* against global trust models

5.1 Global Trust and Recommendation Systems

In a peer to peer network, an evaluating peer (trustor) evaluates another peer's (trustee) trustworthy level to decide, if it can be relied upon to perform some expected tasks. Trust computation among nodes could be used to locate reliable partners that will cooperate to perform specific tasks. A discussion on trust models have been presented in Chapter 3 while this Chapter evaluates *SecTrust* against EigenTrust [197], a modified EigenTrust for speed up functionalities, TNA-SL [152, 153], and No Trust. This is done to assess the general behaviour and performance efficiency of *SecTrust* against these global trust and recommendation systems. The justification for the selection of these trust systems are: EigenTrust is based on P2P network design, which is efficient, lightweight, scalable and deployable as an embedded system. Furthermore, EigenTrust as a global reputation management system, has been implemented in file sharing network site like Gnutella. Although various improvements to EigenTrust have been put forward by researchers, this study is unaware of any that has been adopted as a replacement to EigenTrust. Therefore, verifying the efficacy of *SecTrust* against EigenTrust in this study is appropriate, and it provides a meaningful comparison. Nevertheless, to accommodate the various proposed versions of EigenTrust by other researchers, the EigenTrust Incremental is used as a comparative basis against *SecTrust*. EigenTrust Incremental has same features as EigenTrust, but has been improved with speed-up functionalities. The TNA-SL is a subjective trust system based on theoretic abstractions. The diversity of features therefore, displayed by these trust and recommendation systems presented above, provide a comprehensive benchmark for testing *SecTrust* to measure and assess its performance levels. A brief discussion of the global trust systems used for comparison against *SecTrust* are presented below.

5.1.1 EigenTrust

The EigenTrust [197] trust model is a system which utilizes local normalization together with global convergence by means of a vector multiplication matrix. It is based on past interactions, where past interactions could be viewed as a square matrix or an $n \times n$ matrix with n defined as the number of peers in the network. In EigenTrust, a party's dealings with another party is represented as a vector, and these vectors are locally significant in their scope of influence. For instance, in the EigenTrust matrix presentation of an n x n, a value in the i^{th} row and j^{th} column shows party j's transactions with the party i. Figure 5.2 describes how trust is computed in EigenTrust.

$$\begin{bmatrix} \begin{pmatrix} \boldsymbol{pos}: & 5 \\ \boldsymbol{neg}: & 6 \end{pmatrix} = \mathbf{0} & \begin{pmatrix} \boldsymbol{pos}: & 2 \\ \boldsymbol{neg}: & 1 \end{pmatrix} = \mathbf{1} & \begin{pmatrix} \boldsymbol{pos}: & 4 \\ \boldsymbol{neg}: & 2 \end{pmatrix} = \mathbf{2} \\ \begin{pmatrix} \boldsymbol{pos}: & 9 \\ \boldsymbol{neg}: & 2 \end{pmatrix} = \mathbf{6} & \begin{pmatrix} \boldsymbol{pos}: & 5 \\ \boldsymbol{neg}: & 4 \end{pmatrix} = \mathbf{1} & \begin{pmatrix} \boldsymbol{pos}: & 3 \\ \boldsymbol{neg}: & 7 \end{pmatrix} = \mathbf{0} \\ \begin{pmatrix} \boldsymbol{pos}: & 5 \\ \boldsymbol{neg}: & 3 \end{pmatrix} = \mathbf{2} & \begin{pmatrix} \boldsymbol{pos}: & 7 \\ \boldsymbol{neg}: & 5 \end{pmatrix} = \mathbf{2} & \begin{pmatrix} \boldsymbol{pos}: & 3 \\ \boldsymbol{neg}: & 2 \end{pmatrix} = \mathbf{1} \end{bmatrix}$$

Figure 5.2: Trust computation in EigenTrust model

$$t_0 = p \ and \ \ t_{k+1} = \left(0.5 \ x \ A'^{T} \ x \ t_k\right) + (0.5 \ x \ p)$$

Figure 5.3: Initialization and normalization of trust tuple computation

In computing the trust between two parties, the number of positive *(pos)* and negative *(neg)* communications are stored in what is referred to as *relations*. The feedback in every *relation* is computed as *max(0, (pos - neg))* which is illustrated in matrix A of Figure 5.2. A vector normalization of values generated is then carried out on the *relations*. In other words, the aggregate feedback values of the *relations* across a column is used to divide each element of the column. This is much like the complement of a matrix (A') and it is illustrated in Figure 5.3. A vector *p* of size *n* is normalized and initialized to record the trust forecast tuples. A trust tuple t_k is defined, which is computed from the formulae given in Figure 5.3, and for large *k*, vector t_k will converge to the left principal eigenvector of A' and the global trust vector (t_∞). To maintain consistency of trust values, and preserve the order of the computed trust values among the parties during testing, a large *k* value is maintained. An appropriate initialization of the vector *p,* is key to the successful implementation of the EigenTrust model. The EigenTrust model assigns more trust weight to some parties (*x*) in the network which are presumed *pre-trusted* nodes thus, the vector *p(i)* of the pre-trusted nodes = 1/*x* while all other non-pre-trusted nodes have vector value *p(i)* = 0.

5.1.2 EigenTrust Incremental

The EigenTrust Incremental system is a near replica of the EigenTrust implementation system described in Section 5.1.1. The difference between them however, is that, the EigenTrust Incremental implementation employs a snapshot comparison mechanism to bypass the memory and CPU intensive re-calculations performed during each cycle of trust evaluation and computation. This bypass is achieved through a mechanism, which computes and stores the number of cycle sequences between trust recalculation processes. This way the trust computation comparison and computation cycles will not have to be repeated but instead, cycles with identical snapshots are re-used. This process helps speed up the execution rate of the EigenTrust Incremental system. In contrast, EigenTrust does not make any snapshot comparison, but re-computes at every cycle to determine the trust values of the nodes. Even though, EigenTrust Incremental tends to provide a boost in the execution speed of the processes, it does overlook some negligible feedback changes that could over time reveal considerable malicious activities.

5.1.3 Trust Network Analysis with Subjective Logic (TNA-SL)

This TNA-SL trust system [152, 153] is an abstract and complex, but logical computation of trust. The TNA-SL trust system gives much weight to the past observable direct dealings between nodes. This model describes trust as a tuple of four functions, which include: *belief function* (*b*), *disbelief function* (*d*), *uncertainty function* (*u*), and a *base-rate* (α). The model assumes that the sum of (*b, d, u*) = 1, while α is defined as a real value which lies between 0 and 1, and stores the deductive trust notions. Table 5.1 illustrates TNA-SL's opinion calculation in computing the belief, disbelief and uncertainty levels.

Table 5.1: Opinion computation from past and direct interaction between nodes

Trust description	**Formula**
Belief	(pos / (pos + neg + 2.0))
Disbelief	(neg / (pos + neg + 2.0))
Uncertainty	(2.0 / (pos + neg + 2.0))
Base-rate	If node is pre-trusted the *Base-rate = 1.0* Else Base-rate = 0.5.

The TNA-SL system utilizes several logical operators in its process of deriving the trust value between nodes. One of such logics is the *Discount* computation. *Discount* is employed when the transitive trust recommendation of a node is desired. For example, given node *A* has a direct communication link with *B*, and node *B* has a direct communication link with *C*, the *Discount* trust of *C* from the perspective of *A,* using the transitional information provided by *B* can be mathematically represented in Equation 5.1.

$$\omega^{A:B}_{C} = \omega^{A}_{B} \otimes \omega^{B}_{C} \qquad (5.1)$$

Another logic computation is *Consensus. Consensus* is used to find the weighted average of two opinions. To illustrate this, if nodes *A* and *B* have opinions regarding node *C*, the *Consensus* trust of *C* from the perspective of *A* and *B* is expressed in the notation in Equation 5.2 below while Table 5.3 provides a summary of the trust derivation of *Discount* and *Consensus* trust notions.

$$\omega^{A\lozenge B}_{C} = \omega^{A}_{C} \Theta \, \omega^{B}_{C}$$

Table 5.2: Discount and Consensus trust operators in TNA-SL [152, 153]

Discount: $\otimes$	Consensus: Θ
$b^{A:B}_{C} = b^{A}_{B} b^{B}_{C}$	$b^{A\lozenge B}_{C} = b^{A}_{C} u^{B}_{C} + b^{B}_{C} u^{A}_{C} / denominator$
$d^{A:B}_{C} = b^{A}_{B} d^{B}_{C}$	$d^{A\lozenge B}_{C} = d^{A}_{C} u^{B}_{C} + d^{B}_{C} u^{A}_{C} / denominator$
$u^{A:B}_{C} = d^{A}_{B} + u^{A}_{B} + b^{A}_{B} u^{B}_{C}$	$u^{A\lozenge B}_{C} = u^{A}_{C} u^{B}_{C} / denominator$
$\alpha^{A:B}_{C} = \alpha^{B}_{C}$	$\alpha^{A\lozenge B}_{C} = \alpha^{A}_{C}$
Where $denominator = u^{A}_{C} + u^{B}_{C} - u^{A}_{C} u^{B}_{C}$	

The computation of trust in TNA-SL convolutes into a series of complex operations requiring the discovery of an acyclic *direct series-parallel graph* (DSPG) that exists between a requesting node and a probable source node to reduce the uncertainty of derived trust values. As a final point, given the DSPG, a lone distinguishing opinion is derived through the application of the Subject Logic operators (SL-operators). The expected value (*EV*) is computed as $EV = b + \alpha u$, which is transferred as a set of trust values into the matrix *A*.

In computing the trust for a network of *n* nodes, TNA-SL creates an *n x n* matrix *A,* which houses the opinions of every node towards its neighbour. The opinions are computed based on description of Table 5.1. Furthermore, the *Discount* and *Consensus* operators are used to overload the matrix *A* in a series of multiplications and additions. This series of operations makes the belief values tend to zero, which in turn weakens the trust value of TNA-SL. To address this, the matrix complement (A') is created to store all opinions with maximum confidence observed during the multiplication process. From the matrix A', the global trust is computed as $EV(A'_{(i,j)})$ *where* $EV(A')$ reflects the trust node j has towards node i.

5.1.4 No Trust and Recommendation System (None)

The implementation of a *No Trust* and recommendation system (None), simulates the absence of a trust and recommendation system that uses solely random file-providers, which act as protocol transactions packets. In its implementation, all trust values are assigned the same value during start up, after which the trust values can remain after every call. This system is adopted to generate a benchmark of performance against *SecTrust* and other trust systems.

5.2 Framework Description and Justification for a Baseline Evaluation

This section presents a framework upon which the comparison of *SecTrust* and other trust systems are based upon.

However, articulating a common framework for the comparison of these trust systems discussed above pose a challenge for the following reasons. Firstly, Trust and Recommendation systems find their application in various network architectural designs including service-oriented (SON), social and peer-to-peer (P2P) networks. Secondly, the realistic implementation of the simulation of various trust systems while taking into consideration all the various parameters that may influence trust, presents a potent influence on the accuracy of results that may be obtained while trying to keep the simulation close to an ideal real-world situation. Finally, while the simulation of a well-behaving node can be predicted, the accurate description (coding) of a malicious node as an independent malicious attacker and as part of a colluding attacking group presents a significant simulation challenge. This is because these malicious nodes (in real-life scenarios) could behave in multiple ways that simulation may not be able to predict regardless of the type of attack. To address this however, this study uses a mix of well-known principles and concepts covered in the literature review of this research work.

To proceed further, the definition of terms and the explanation of concepts used during the simulation and comparison of trust systems are discussed.

i. *Nodes, users and peers*: These are entities within a system or network with the capability of communicating or transmitting files or data amongst themselves. When these entities establish communication links among themselves, the *links* are termed *channels*. However, in graph theoretic terms, the network is the *graph* and the users or peers are termed *nodes* while the communication channels are referred to as *edges,* which could be bidirectional. The nodes randomly communicate with other nodes when required.

ii. *Client/Server*: A node (user or peer) could act as a *server* or *provider* when it provides files or data to another node. A node is referred to as a *client* or *requester* when it sends requests for files or data from another node in the network.

iii. *Files*: In this framework, items requested or transmitted within the network are *files,* which could typify data, services or a function call. It is assumed that a node could determine the validity of a file and thus, the node can classify a file as either *valid* or *invalid*. The validity of a file is considered perpetual while multiple and identical copies of the file can be found throughout the network. A node's trustworthiness is predicated on the quality of files it offers to other nodes. Every node in the network has some storage referred to as an *initial library* where *files* can be stored.

iv. *Query*: Nodes, from time to time broadcast to make specific *file* requests to other nodes in the network. This is termed *Query.*

v. *Transaction*: When a query request is granted and the request furnished, this is referred to as a *transaction*. A transaction could be *valid* or *invalid* depending on the trustworthiness of the node in providing valid files as determined by the evaluating node. For every transaction, a transaction validity can be computed, which can be used to measure the validity of files provided by a *provider* node.

vi. *Clean-up*: The process whereby a node eliminates unwanted *files* from its *library*. Malicious nodes will tend not to clean-up, but transmit these invalid files.

As a justification to the use of the P2P network infrastructure framework, which this study explores, P2P network is regarded as a robust architecture in the emulation of many system models with defined functionality boundaries. This position is further supported by the authors in [1]. They provided an example in which a test is required on a system with a set of some mutually and exclusive clients and service providers, that use trust and recommendation systems to extrapolate a node's (*provider*) reliability in the network system. They further submitted that with only minor adjustments, a P2P infrastructure framework could effectively simulate the given model. In the adjusted framework, the *files* transmitted in the network become the *services* while *Service Providers* act in the network only as *Servers* offering a library of services. *Clients* go into the network with a preliminary set of empty library and with the ability to request and accept *services*. Clients do not store (*clean-up*) the *services* they receive, and this is to ensure that they (*clients*) do not become *providers (servers)*.

The trust and recommendation systems are coded in the trace-simulator (QTM P2P) used to make the baseline comparison. Figure 5.4 shows the workflow architecture for the evaluation of the trust systems. In the framework architecture, static traces are generated into a file which is saved and then fed into the simulator that implements the trust and recommendation system with the defined environmental settings. From the architecture in Figure 5.4, numerous simulations implementing diverse trust and recommendation systems can be executed using the same trace.

Having static snapshots of each aspect of a network while testing the various trust and recommendation systems in the framework makes it ideal for the comparison of the trust and recommendation systems under investigation. In addition, the state of the trustworthiness of a trust system can be examined at an instance.

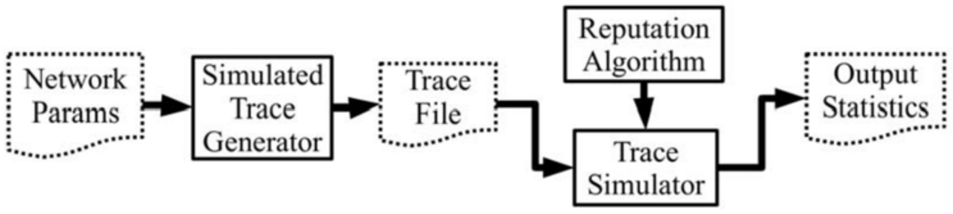


Figure 5.4: A workflow for the evaluation of trust systems as illustrated in [1]

5.3 QTM P2P Trust Simulator

The QTM P2P trust simulator is an evaluation tool for assessing the general efficiency and performance of trust and recommendation algorithms. The QTM P2P trust simulator comprises two simulation parts namely: i) *the trace generator*, which initializes and generates transaction seeds and ii) *the trace simulator*, which provides empirical results of the performance of the trust and recommendation system under investigation after receiving as input: i) the transaction seeds generated by the *trace generator* and ii) the specified trust and recommendation algorithm. The QTM P2P simulator has been ably developed using both the C and Java programming languages. It also incorporates with it the trust systems discussed in Section 5.1. Although, both the C and Java versions of the simulator are available online [1], the Java code is well documented, which makes it easy to comprehend and hence, extend the Java implementation of the P2P simulator. For this purpose, *SecTrust* was developed and embedded into the QTM P2P simulator using Java. The QTM simulator [1] was designed as a platform for the evaluation and comparison of Trust and Recommendation Systems.

5.3.1 Trace Generation

A trace is a succession of queries where a filename and the user (normally a client) seeking the specified file are outlined. The server choice for the download of the file is based on the specified trust and recommendation system, which is decided during simulation. The trace generator receives as input basic network parameters, processes these parameters and generates a script of the simulated network.

In accordance with the workflow of QTM, a network of users/nodes is generated having a library of user models. User (node) models are users/nodes with pre-programmed behaviour. Some examples include: misbehaving users/nodes, good behaving users/nodes amongst others. In this study, it is assumed that 10 - 20% of all users in the network behave maliciously. The framework further assumes that each user is endowed with valid and invalid files with multiple copies of the same files available throughout the network while the preliminary distribution of files.

The QTM framework also assumes that users exhibit behavioural preferences in two ways. The first preference shows that a user could decide to clean-up (delete) files downloaded. Good users gravitate towards cleaning up downloaded invalid files while their malicious counterparts will keep and propagate downloaded invalid files to destabilize the network. The framework further assumes that there are no clean-ups for the initial endowment of each user (an assumption of what each user comes into the network with) and users have a single opportunity at getting rid of a file right after it is downloaded.

In the second preference, clients provide globally observable feedbacks of the server regarding the quality of file served. Good users in the framework are inclined to providing honest feedback, providing positive scores for every valid file received while malicious users are inclined to providing dishonest feedbacks and providing negative scores against trusted servers while giving positive scores to malicious servers.

The framework provides malicious models which consists of a variety of strategies and adversarial motives from Sybil-like behaviour, bad-mouthing, good-mouthing to packet blackholing, clean-up and honest/dishonest feedback. These malicious models are adequately captured in the QTM P2P-simulation program. The following arguments are specified during the generation of the trace file and they include: total users in the network, modelled malicious behaviour of users, specified number of transaction files, file ownership probability, total simulated transactions (queries), maximum user connections (defaults to 2) and time period of bandwidth utilization.

1. User Models

Although, the motivations of users in a network are diverse and dynamic, a good user however, only needs to be furnished with a valid file and in turn, it provides an acknowledgment of its satisfaction (valid transaction). Malicious users on the other hand, desire to provide invalid files and perpetrate harm in a network based on the malicious intents of the user such as trading corrupted files or giving out false information or transmit files containing viruses. The adopted framework [1], describes in Table 5.3 the characteristics of the user models implemented.

Table 5.3: User initialization parameters

User Type	Clean-Up (%)	Honesty (%)
Good	90 – 100%	100%
Purely Malicious	0 – 10%	0%
Malicious Provider	0 – 10%	100%
Feedback Malicious	90 – 100%	0%
Disguised Malicious	50 – 100%	50 – 100%
Sybil Attacker	0 – 10%	Irrelevant

i. *Good*: Good users in this framework are primed for good behaviour. They are primed to give honest feedback, and they are alert to the activities in their file libraries. They *clean-up invalid files* that they may have downloaded. In the framework under study, good users have clean-up rate of 90-100%. The differential margin of 10% is meant to give some tolerance as no user can be expected to show perfect behaviour.

ii. *Purely Malicious*: This typifies a user that does not clean-up invalid files downloaded, but cleans up valid files. It does not give honest feedback of the files it has received. A typical clean-up rate for a malicious user is 0–10%.

iii. *Malicious providers and feedback malicious:* These two attack models exhibit similar misbehaving attributes. They exploit the recommendation system of trust systems, which evaluate trust on recommendation by giving much weight to recommendations from other malicious users. EigenTrust does not penalize malicious users, which provide dishonest feedback in the form of recommendations. Detecting a dishonest feedback (good-mouthing/bad-mouthing) predominantly proves to be difficult because of the unpredictable dynamics of misbehaving users in a network. They could decide to be truthful, partially truthful or completely dishonest. It becomes even more difficult when they collude to perpetrate these attacks. The malicious nodes could be honest among themselves, but remain dishonest to the global network or have a mix of in-betweens. In the case of the *feedback-malicious* model, it exploits the transitive trust (recommendation) of other users. It achieves this by maintaining its *valid files* (reputation), but recommends (refers) other unsuspecting, but good users to malicious peers where they could be compromised. Thus, the *feedback malicious* user serves as a route to trap unsuspecting users in the network. This is much like a wormhole attack. *Malicious providers* send invalid files to users, and this can be quickly detected since they will automatically have a bad feedback. Both cooperative strategies are implemented and tested.

iv. *Disguised Malicious:* The *disguised malicious* user acts alone but while it receives *valid* files, it refuses to make the files completely available to others. It slyly makes some available while retaining the others. This often means trust systems may find it difficult to classify these users as either good or malicious. This way, trust systems, which compute their trust on weighted averages, will think of them as "averagely good" users. *SecTrust* defeats this by increasing the penalty factor (β) for every misbehaviour of a user during trust computation. The authors of TNA-SL [1], similarly opined that their system avoids this pitfall by examining the total number of feedbacks by means of a beta-probability density function (PDF) approach.

v. *Sybil Attacker:* Sybil attackers exploit the low-entry requirements for networks. A Sybil attacker seeking to be a trusted member in the network waits for the moment when it becomes the provider in a transaction, the attacker erases its details and assumes a new identity. The Sybil attacker prefers to be perceived with a neutral personality rather than with the negative trust personality it had before. The QTM framework implements the Sybil attacker model although the framework does not permit the Sybil user to record feedbacks. This is to minimize network growth emanating from the profile details of expendable one-time Sybil users.

vi. *Malicious Collectives:* The threat posed by a group of colluding users presents serious danger to a distributed network even though, independent malicious users can randomly join in attacks such as giving positive scores (good-mouthing) to an unknown user and sending invalid files. Nevertheless, colluding and malicious cooperation among attackers pose greater risk to any distributed network. The Trust and Recommendation Systems are tested against these malicious collective attackers.

To measure the malicious infection perpetration model and invalid file proliferation in a distributed peer-2-peer network, the framework recommends that trace files generated have a balanced or realistic number of malicious peers, since it is impracticable for any trust system to recover from a network overwhelmed with malicious attackers. In consequence to this, this study adopts a 10 – 20% of total number of nodes as malicious. In addition, this study measures the performance of the trust systems while varying the size of the network to assess and evaluate the efficiency, trust effectiveness and scalability of the trust and recommendation systems.

2. User Library Definition

The QTM framework adopts the Zipf distribution [198] model, for which the authors established the support of the model by various research literatures as an accurate and accepted frequency modelling of file/service request system in many spheres of the Internet. During initial user library creation, clean-up is not undertaken, subsequently, a user must clean up its files. A user's initial file validity is predicated on the clean-up rate achieved by the file owner. Therefore, a user with clean-up rate of 10 will have a 10% probability that its file is valid while a user with a clean-rate of 85 will have an 85% probability that its file is valid.

3. Testing to the Limits: Intensification of Malicious Attacks

The comparative performance of some Trust and Recommendation Systems do not become apparent until they are stretched to their limits. With minimal malicious user empowerment, most of the trust systems seem to be at the top of their game. However, in the testing performed, malicious activities are varied to the point, where the performances of all the various trust systems under investigation begin to diverge significantly. Although, the parametric settings and attributes assigned to these malicious users may seem practically unrealistic, the idea here is to find the breaking point of the trust and recommendation systems to determine the trust system with superior secure communication performance.

i. Promoting Malicious Users

The assumptions made in this thesis provides malicious users with a lot more access to weaken the trust and recommendation system of a distributed network. This was done on purpose to experimentally assess the breaking point of the trust systems. As an example, a real-world network does not operate in a closed environment, users are mobile and they can freely join or leave a network at any time. In addition, actual networks are based on users with a common set of goals and which have trust among one another. Over the course of time other nodes could join, which may be good-intentioned or ill-intentioned. In a live network, the likelihood of having a swarm of misbehaving nodes operating in a network at a go is very remote. Their appearance and perpetuation of attacks in a network is rather sporadic, but consistent and sustained over a long period of time. Finally, as an empowering assumption for malicious users, any transaction could run to completion. Whether it is a valid or an invalid transaction. In a real-world scenario, this will normally be truncated; so, given an inadequate number of file copies or insufficient bandwidth conditions, the less trustworthy users are permitted to distribute files.

ii. Promoting Isolated Malicious Users

In the assumption of empowering isolated malicious users, the trust is computed among the users in a distributed manner. Thus, a user computes the trust value of its neighbour and submits the feedback (perhaps dishonest) to the global database, but the malicious user will locally keep a history of totally honest interactions. Good users calculate trust while selecting only valid nodes as specified by the trust systems. Malicious users however, do not compute or evaluate trust as specified by the trust systems. Hence, they (malicious users) seek to weaken the evaluation metric of the trust system while skewing good users' opinions.

iii. Promoting Malicious Collectives

Malicious collective nodes collaborate in creating invalid files that will be received by good users. Malicious collective nodes transmit honest feedback among themselves but send dishonest feedback (invalid) to good users. In this way, they cause the good users to have a distorted view of the network. Malicious users use their history of interactions for local trust evaluation, which they in turn send to their malicious collective group so they could have an exact assessment of the network. The Trust and Recommendation Systems are tested against this type of malicious behaviour.

4. Query Generation

With the trace file being generated, the next phase is the query generation, which defines who requests what file and in what number. Two modes are specified including: naïve and intelligent query generation. In the naïve mode, a user solicits a random file, which in the trace file is logged as a query. The intelligent mode provides procedures enabling the prohibition of large amounts of transactions that cannot be completed at runtime. Specifically, users are not permitted to request files they already have or have downloaded in the past as this ensures a user does not have several copies of the same file. In addition, files requested are assumed to be available in the network, as queries with no file result have no significance in the network, nor do they have any computational value. However, a file may be unavailable, not because it is not available, but because the bandwidth of the user(s) having the file may not be sufficient. Every user is assumed to possess the same likelihood to request files from the network. The Zipf distribution is used in specifying what file is requested for the initial population of a node's library.

5.3.2 QTM Trace Simulation

The trace simulator runs in an algorithmic manner as shown in Figure 5.5.

```
While more queries remain:
   Read query from trace file
   Broadcast query to determine potential providers
   Compute trust-values for relevant user pairs
   Select bandwidth-available source user
   Copy source file to requester's library
   Requester submits feedback concerning source
End
```

Figure 5.5: Basic loop procedure of QTM trace simulator

1. Load Balancing and Bandwidth Assignment

A possibility arises with trust and recommendation systems for trust dependencies to be predicated only on a select few users or nodes in the network. These users (nodes) face the danger of being excessively burdened with requests from other nodes. A restriction on bandwidth will inhibit requests from users from being fully processed or at best deliver poor service to the requesting users in the network [199]. Although, the framework [1] adopted for testing SecTrust's performance acknowledges load balancing as a functional requirement, and that network performance could be impacted as a result of a lack of it, the authors in [1] submitted that distributing the load across other less trusted nodes will by no means compromise security of the network on the long run.

EigenTrust and EigenTrust Incremental feature load-balancing functionality, which is implemented in the QTM P2P-simulator. The No Trust and TNA-SL trust systems do not address this functionality and hence, no load balancing feature is addressed in both trust systems. SecTrust, however, provides a trust-based ordered set of nodes that can be used by the bandwidth manager to allocate capacity to trusted and bandwidth-available nodes in the V4 and V5 categories as specified in Table 4.2 of Section 4.2.2.

2. Selection of Service Providers

The selection of a file provider is based on the characteristics of the *user* or *peer*. As illustrated in Table 5.4. A user in the network adopts a requester model based on its intentions. A malicious user who wishes to propagate spurious file libraries will select the *worst* source (provider). Also, a feedback-intentioned malicious user will select a *random* file provider in order to enhance the global rating of its colluding malicious member-users while denigrating *good* users. Expectedly, *good* users will select the *best* provider. It is noteworthy that in the selection of source/service providers, only users with available bandwidth can be selected while provider-users with the same trust values are chosen arbitrarily.

Table 5.4: A user-requester adoption model of source selection provider

Requester Model	Source
Good	BEST
Purely Malicious	WORST
Feedback Malicious	RANDOM
Malicious Provider	WORST
Disguised Malicious	RANDOM
Sybil Attacker	WORST

5.4 Embedding *SecTrust* in the QTM P2P Trace Simulator

As earlier discussed, the P2P-trace simulator can be extended to accommodate new trust systems. It has a set of developed Application Programming Interfaces (APIs) that makes the inclusion of new trust and recommendation systems possible. The trace simulator comprises four main modules namely: GENERATOR_LIB, CORE_LIB, TRUST_SYSTEM_LIB and SIMULATOR_LIB. A description of the four modules, their functionalities and interrelationship is further discussed in Appendix 1.

In Appendix 1, Table 11.1 provides the method summary of the Java implementation and integration of *SecTrust* in the QTM trace simulator. Figure 5.6 is an adapted and detailed QTM simulator trust management organisation diagram showing the interaction of the various modules and the incorporation of *SecTrust* into the trust system library.

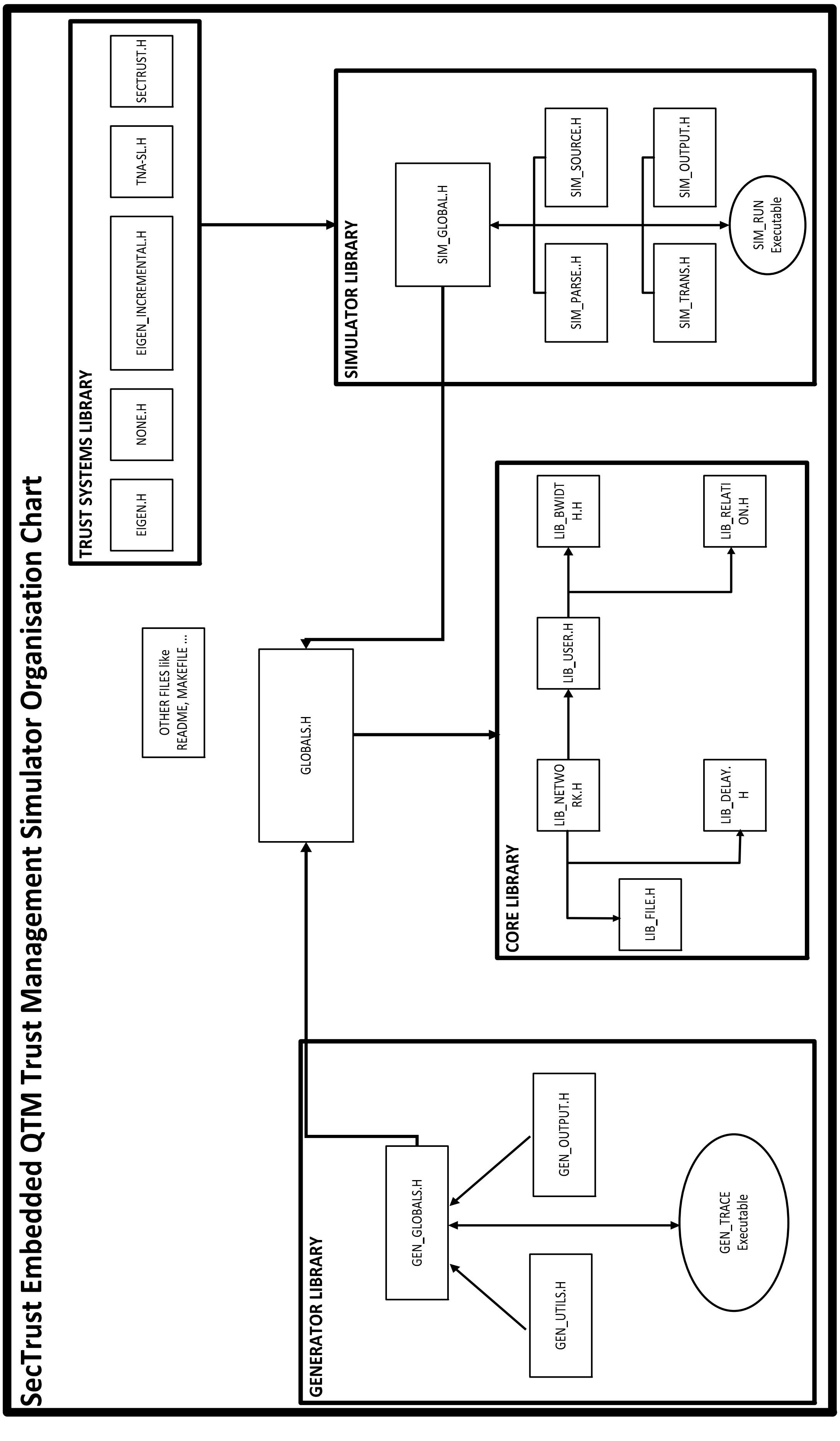


Figure 5.6: An adapted QTM trace simulator organisation chart [1].

SecTrust was developed as a Java program and included in the TRUST_SYSTEM_LIB of QTM P2P simulator. The implementation of SecTrust followed, and was based on the design provided in Chapter 4 (Design and Methodology of SecTrust). The details of the five-tuple trust level band have been described in Chapter 4.

Furthermore, the description of some key variables used in the Java implementation of SecTrust during integration with the QTM trace simulator is described next.

i. private final double penalty

This is the value which is applied to a node, which has not returned a positive/good feedback response. The penalty value (*β)* increases as the trust value of a node decreases as highlighted in Table 4.2.

ii. private double transitivePast

The transitivePast represents the percentage weighting, which the previous trust value has on a node. For example, if a node previously had a negative trust value rating, then this value will be multiplied with the transitivePast value to decrease its rating even further. This ensures that malicious nodes that occasionally flip between being good and bad are adequately penalized in the network.

iii. private double transitiveCurrent

This is the percentage weighting that the currently calculated trust value has on a node. If in the latest trust computation of a node, a bad feedback is returned then, the percentage weighting of the transitiveCurrent value will be higher than when it previously returned a bad feedback. In other words, there is a progressive percentage weight increase either positively or negatively depending on the current trust value of a node.

iv. private double formulaweight

This is the weighting ratio that determines the influence the trust formula has on a node. The total formulaweight is equal to 1.0. The direct trust value is set at 0.875 while the recommended and past behaviours amount only to 0.125 of the ratio. The current trust value of a node attracts a ratio weight of 0.875 because the study sought to determine the real-time behaviour of a node while placing less weight emphasis on its past behaviour because some malicious nodes can behave initially well, and later begin their malicious activity. However, the past behaviour still counts, except that it does not carry much weight. In addition, formularweight can be adjusted to suit specific scenario should it be required.

v. private double cutoff

This is the value which determines when a node is classified as a malicious or bad node. The current rating is set at 0.76 and above. This is as specified in the trust table (refer to Table 5.6). This implies that any node with a trust value below this threshold will be classified as malicious or selfish. Decreasing this value however, makes *SecTrust* more lenient towards the behaviour of nodes.

5.5 Metrics for the Evaluation of Trust Systems

This section discusses metrics used for the comparison of SecTrust against EigenTrust, EigenTrust Incremental, TNA-SL and No Trust systems. Metrics used are discussed below.

i. Effectiveness Metric

According to the [1], the effectiveness of a trust and recommendation system is given as the ratio of the number of valid transactional files received by users (nodes) to the number of transactions attempted by good users despite the malicious operations of bad users in the network. The effectiveness metric compares the relative effectiveness of SecTrust against the other trust system under study. The effectiveness metric is summarised in Equation 5.1 below:

$$Effectiveness\ Metric\ = \frac{Number\ of\ good\ user\ successes}{Number\ of\ good\ user\ transactions} \qquad (5.1)$$

ii. Transaction Validity Metric

A valid *Transaction* is the request and acquisition of a *file* after a broadcast has been made by a good user (node). Invalid transactions however, are transactional files being inserted into the network by malicious users. As previously discussed in section 5.2, malicious users do not clean up their *invalid files* so they could destabilize the network with spurious copies of transactional files. The transactional validity of a trust system is defined therefore, as the ratio of the *valid transactions* by good users to the number of *transactions completed* during the simulation period. This is specified in Equation 5.2 below:

$$Transaction\ Validity\ = \frac{Valid\ transactions}{Transactions\ completed} \qquad (5.2)$$

iii. Sybil Effect Metric

A Sybil attacker assumes multiple identities to operate in a network. After operating in a network maliciously for some time, it assumes a new identity and behaves well to gain the trust of the system after which, it begins to perpetrate its malicious behaviour again. This personality mutation continues until the network is fully compromised or destabilized. Therefore, how well a trust system is able to address this attack shows the adaptability of the trust system to the mutative behaviours of an attacker. The Sybil effect ratio is computed as the number of *Sybil user feedbacks* divided by the *feedbacks committed* in the network. The smaller the ratio the better the trust system is at mitigating against the Sybil attacks. The formula is shown in Equation 5.3.

$$Sybil\ effect\ ratio\ = \frac{Sybil\ user\ feedbacks}{Feedbacks\ committed} \tag{5.3}$$

5.6 Simulation Setup and Measurements

The simulation setup and simulation results of the trust and recommendation systems are presented. The simulation study demonstrates the behavioural reactions of the various trust and recommendation systems. The simulation study further presents different aspects of malicious users (nodes) under varying conditions such as purely malicious and purely collective malicious behaviour as discussed in Section 5.3.1.

To create a balanced view for the analytical comparison of all Trust and Recommendation systems against *SecTrust*, multiple malicious scenarios where simulated including: the purely naïve scenario, the purely collective scenario, the Sybil naïve scenario and the Sybil collective scenario. Furthermore, the size of nodes operating in the network was varied from 50 to 1000. This was necessary to test the performance, reliability and scalability of all trust system under malicious attacks. Other parameters were further considered and presented in each of the results' sections below. The simulation was performed on a desktop computer with the following configuration:

i. Platform: Microsoft Windows 10 Pro
ii. System Type: x64-based PC
iii. Processor: Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz, 3601 MHz, 4 Cores
iv. Memory Capacity: 16.0 GB RAM.

5.6.1 The Case of Trust Network Analysis with Subjective Logic(TNA-SL)

During the simulation test of the TNA-SL trust system, the trace files were generated just like the other trust systems. However, it was observed that when running the trace simulator for the TNA-SL algorithm, it simulation became intractable even when the least number of parameters was used including n=50 users, malicious users = 10 and number of transactions = 10,000. The simulation stops after processing 2,500 transactions, which was still a long way to the 10,000-transaction goal. The explanation for this is offered by the authors of TNA-SL [1, 152, 153], where they admitted to the complex computation of trust and transitive trust values, which as a result poses a large computational burden and further inhibits testing on its scalability to larger networks. Due to the intractability of TNA-SL, it will not be considered as part of the analyses of results presented.

5.6.2 Purely Naïve and Purely Collective Scenarios

The purely naïve scenario uses the global interaction data of user nodes as the basis for the computation of trust values. Hence, no other implicit data is considered or used for trust evaluation. The purely collective on the other hand, simulates a scenario where malicious peers share honest information amongst themselves. Table 5.7 provides a detailed simulation setup for 50 to 1000 user nodes while the results are presented next.

1. Effectiveness Metric for 50-Nodes

In the simulation of trust systems for 50 nodes, the number of transactions was set at 10,000 while the completion times for each of the trust algorithm is given in Table 5.8.

Table 5.5: Trust simulation parameter settings

Parameters	Value
Number of Users and Number of Transactions	50 users with 10,000 transactions
	100 users with 10,000 transactions
	500 users with 100,000 transactions
	1,000 users with 100,000 transactions
Number of Files	5,000
Max. User Connections	2
Bandwidth Period	1
Zipf Constant	0.4
Pre-Trusted Users	5
Good Behaving Users	40
Purely Malicious Users	10

Table 5.6: Completion Time for 50 Nodes in seconds

Trust System	**Purely Naïve (seconds)**	**Purely Collective (seconds)**
No Trust	0.071	0.119
EigenTrust	0.307	1.33
EigenTrust Incremental	0.098	1.08
SecTrust	0.321	1.29

Figures 5.7 and 5.8 show the effectiveness metric of the various trust systems in a 50-node network simulation under purely naïve and purely collective scenarios respectively. Under the purely naïve scenario, *SecTrust* had a better trust effectiveness measure of 0.90277 in 0.321 seconds compared to EigenTrust (0.90164) and EigenTrust Incremental (0.90089). However, EigenTrust and EigenTrust Incremental had faster simulation completion time of 0.307 and 0.098 seconds over SecTrust. No Trust (None) had the worst trust performance measure of 0.78773 although it had the fastest simulation completion time of 0.071 seconds.

In the purely collective scenario, EigenTrust Incremental had the best trust measure with 0.90114 in 1.08 seconds, while *SecTrust* (0.90076) followed closely with 1.29 seconds. Again, No Trust had the worst performance trust measure of 0.78451 in 0.119 seconds.

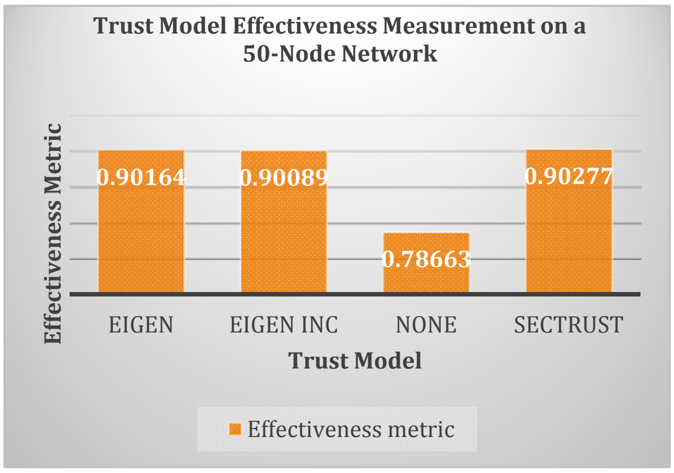


Figure 5.6: Trust system effectiveness of a 50-node network in a purely naive scenario

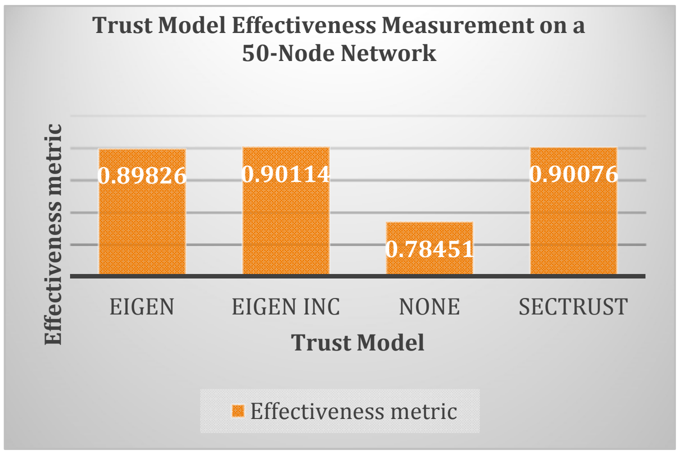


Figure 5.7: Trust system effectiveness of a 50-node network in a purely collective scenario

2. Effectiveness Metric for 100 Nodes

In this summary, the effectiveness of trust system for 100 nodes with transaction number set at 10,000 while the completion times in seconds for each of the trust algorithm is shown in Table 5.7. Figure 5.9 shows the result of the purely naïve effectiveness metric of the trust algorithms. *SecTrust* with a value of 0.94389 and a completion time of 1.041 seconds proves to be the better performing algorithm over EigenTrust and EigenTrust Incremental, which had 0.93588 and 0.93939 effectiveness values and completion times of 1.212 and 0.251 seconds respectively. Again, No Trust (None) had the worst effectiveness metric performance though with the faster completion time of 0.15 seconds.

The purely collective scenario as shown in Figure 5.10 further confirms the effectiveness of *SecTrust* over EigenTrust and EigenTrust Incremental. *SecTrust* recorded a 0.94264 effectiveness rating in 8.094 seconds over EigenTrust and EigenTrust Incremental, which had 0.93575 in 8.165 seconds and 0.94014 in 7.018 seconds respectively. No Trust had the least performance trust measure of 0.78673 in 0.263 seconds. EigenTrust Incremental displayed faster simulation completion time of 7.018 seconds due to the speed-up functionalities implemented in the algorithm. In critical terms however, EigenTrust Incremental may overlook some of the trust computations needed to provide thorough security among nodes, which if left over time, could sum up to prove a serious security flaw.

Table 5.7: Completion Time for 100 Nodes in seconds

Trust System	**Purely Naïve (seconds)**	**Purely Collective (seconds)**
No Trust	0.15	0.263
EigenTrust	1.212	8.165
EigenTrust Incremental	0.251	7.018
SecTrust	1.041	8.094

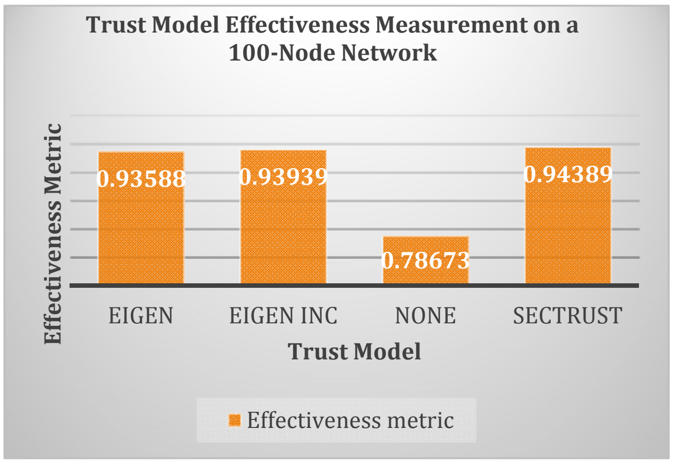


Figure 5.8: Trust system effectiveness of a 100-node network in a purely naive scenario

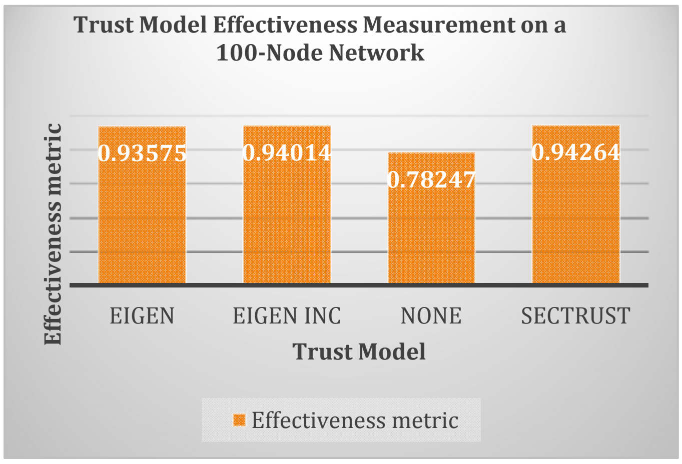


Figure 5.9: Trust system effectiveness of a 100-node network in a purely collective scenario

3. Effectiveness Metric for 500 Nodes

A simulation study was further conducted to test the scalability of trust systems in a large network setup using 500 nodes with a 100,000-transaction capacity while maintaining other parameters as displayed in Table 5.7.

Figure 5.11 shows the result of the purely naïve effectiveness metric of the trust algorithms with the completion times for both the purely naïve and collective is shown in Table 510. In the purely naïve scenario, *SecTrust* had the best effectiveness metric value of 0.95057 with a completion time of about 4.65 minutes, while EigenTrust Incremental was next with an effectiveness metric of 0.94744 and a completion time of 5.03 minutes. Expectedly, No Trust (None) showed the least effectiveness in performance with a completion time of 61.851 seconds.

Results for the purely collective scenario is as shown in Figure 5.12. EigenTrust Incremental (0.95239) displayed better effectiveness metric ratio over *SecTrust* (0.94274), EigenTrust (0.94166) and No Trust (0.80193) with a completion time of 2.41 hours. *SecTrust* had the next best performance over EigenTrust and No Trust. The speed-up functionalities in EigenTrust Incremental is clearly giving it an advantage as the network grows.

Table 5.8: Completion Time for 500 Nodes in seconds

Trust System	**Purely Naïve (seconds)**	**Purely Collective (seconds)**
No Trust	61.851	81.955
EigenTrust	301.801	8,848.784 (2.46 hrs)
EigenTrust Incremental	75.122	8,656.079 (2.41 hrs)
SecTrust	278.826	8,841.063 (2.46 hrs)

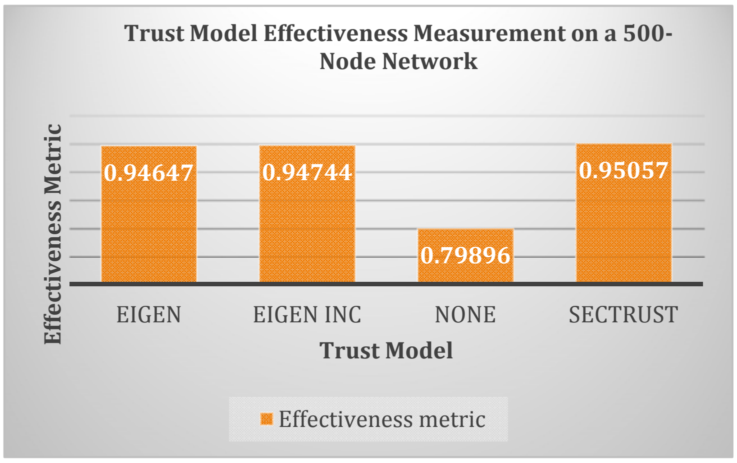


Figure 5.10: Trust system effectiveness of a 500-node network in a purely naive scenario

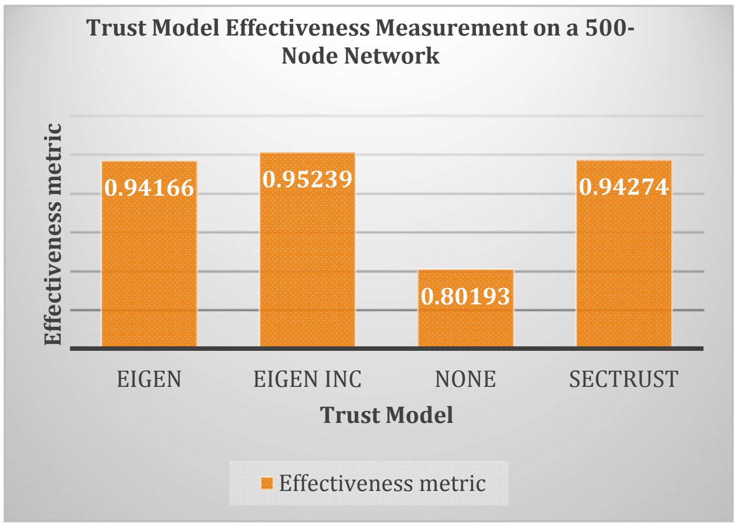


Figure 5.11: Trust system effectiveness of a 500-node network in a purely collective scenario

4. Effectiveness Metric for 1000 Nodes

As a final test of the effectiveness of all trust algorithms, a 1,000-node network size with a transaction capacity of 100,000 was further investigated while maintaining all other parametric settings as shown in Table 5.7. Under the purely naïve scenario of Figure 5.13, the *SecTrust* algorithm with a 0.95850 effectiveness value and a completion time of 23.59 minutes showed better performance over all remaining three trust algorithms. This was followed by EigenTrust Incremental with an effectiveness metric of 0.95230 and a completion time of 10.31 minutes. The difference in the effectiveness metric recorded between *SecTrust* and EigenTrust under the purely naïve scenario shows a 3.32% performance increase for *SecTrust* over EigenTrust. Therefore, as the network scales to over 10,000 nodes under the purely naïve scheme, the superior performance of *SecTrust* over other trust systems becomes very apparent.

The results of the purely collective scenario are shown in Figure 5.14 with the completion times tabled in Table 5.11. EigenTrust Incremental displayed better performance over *SecTrust*, EigenTrust and No Trust with an effectiveness metric of 0.95647 and an average completion time of 36.96 hours. Again, the speed-up function in EigenTrust Incremental obviously boosted the effectiveness metric and completion time of EigenTrust Incremental. *SecTrust* had the next better effectiveness metric of 0.95500 and an average completion time of 40.71 hours. EigenTrust and No Trust recorded 0.94664 and 0.78805 with average completion times of 48.74 hours and 9.90 minutes respectively.

Table 5.9: Completion Time for 1000 Nodes in seconds

Trust System	**Purely Naïve (seconds)**	**Purely Collective (seconds)**
No Trust	487.989 (8.13 mins)	593.899 (9.90 mins)
EigenTrust	1,524.778 (25.41 mins)	175,447.121 (48.74 hrs)
EigenTrust Incremental	618.444 (10.31 mins)	133,040.812 (36.96 hrs)
SecTrust	1,415.284 (23.59 mins)	146,554.542 (40.71 hrs)

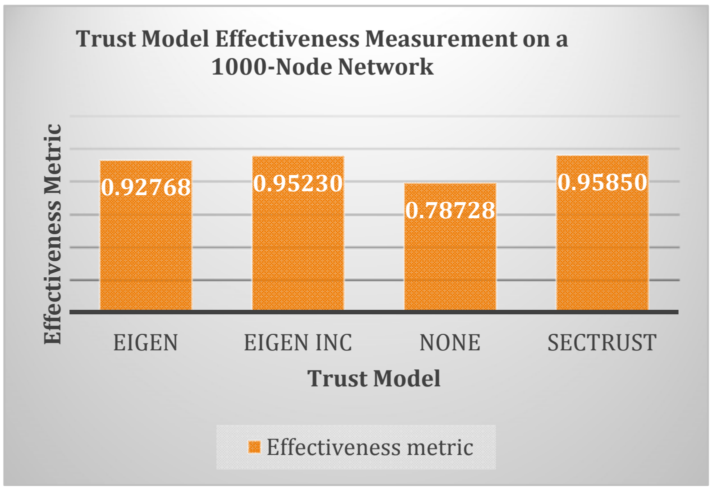


Figure 5.12: Trust system effectiveness of a 1000-node network in a purely naive scenario

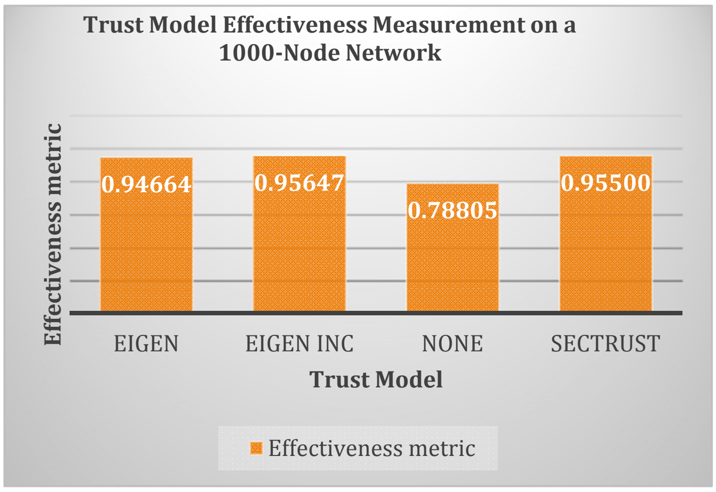


Figure 5.13: Trust system effectiveness of a 1000-node network in a purely collective scenario

5.6.3 Transaction Validity under Purely Naïve and Purely Collective Scenarios

The efficacy of a trust system could be further measured by the number of valid transactions carried out under a trust system or alternatively, the number of invalid transactions prevented by a trust system during a network simulation under malicious attacks. As defined in Section 5.5, the Transaction Validity metric of a trust system is given as the ratio of valid transactions by good users to the total number of transactions completed during the simulation period. Table 5.12 below gives the parametric settings for 50, 100, 500 and 1,000 user nodes with transaction settings of 10,000, 10,000, 100,000 and 100,000 respectively.

Table 5.10: Trust simulation parameter settings

Parameters	Value
Number of Users and Number of Transactions	50 users with 10,000 transactions
	100 users with 10,000 transactions
	500 users with 100,000 transactions
	1,000 users with 100,000 transactions
Number of Files	5,000
Max. User Connections	2
Bandwidth Period	1
Zipf Constant	0.4
Pre-Trusted Users	5
Good Behaving Users	40
Purely Malicious Users	10

1. Transaction Validity under the Purely Naïve Scenario

Figure 5.15 shows the transaction validity metric of all four trust systems under the purely naïve scenario. The Figure shows the performance of each trust system using 50, 100, 500 and 1000 user nodes. Table 5.13 shows the completion times for all trust systems under the purely naïve scenario. In general, *SecTrust* displayed better performance over all trust systems across all the simulated number of user nodes. During the simulation with 50 nodes, *SecTrust* had a transaction validity of 0.88340 while EigenTrust, EigenTrust Incremental and No trust recorded 0.83000, 0.83150 and 0.79100 respectively. An indication of *SecTrust*'s potency in defending against invalid file transactions in comparison to the other trust systems. Furthermore, simulation results reveal that with a simulation of 100 nodes, *SecTrust* recorded a transaction validity of 0.89680. EigenTrust, EigenTrust Incremental and No trust recorded 0.85840, 0.85880 and 0.78780 respectively. Results further confirm the effectiveness of *SecTrust* over the rest of the systems.

In addition to the effectiveness of a trust system in mitigating malicious attacks, the scalability of a trust system will further prove its effectiveness and relevance in providing adequate defence against attacks in large sized networks. Tests were conducted using 50, 100, 500 and 1000 nodes to determine the scalability of *SecTrust* in comparison with the trust systems under investigation. A look at Figure 5.15, reveals that *SecTrust* had a better transaction validity metric value of 0.94166. EigenTrust, EigenTrust Incremental and No Trust had 0.91315, 0.91688 and 0.79953 with completion times of 5.03 minutes, 75 seconds and 61.85 seconds respectively. Scaling finally to 1000 user nodes, *SecTrust* maintained a high consistent performance of 0.93025 and a completion time of 23.59 minutes as against EigenTrust, EigenTrust Incremental and No Trust, which had 0.91166, 0.91658 and 0.78861 with completion times of 25.41 minutes, 10.3 seconds and 8.13 minutes respectively.

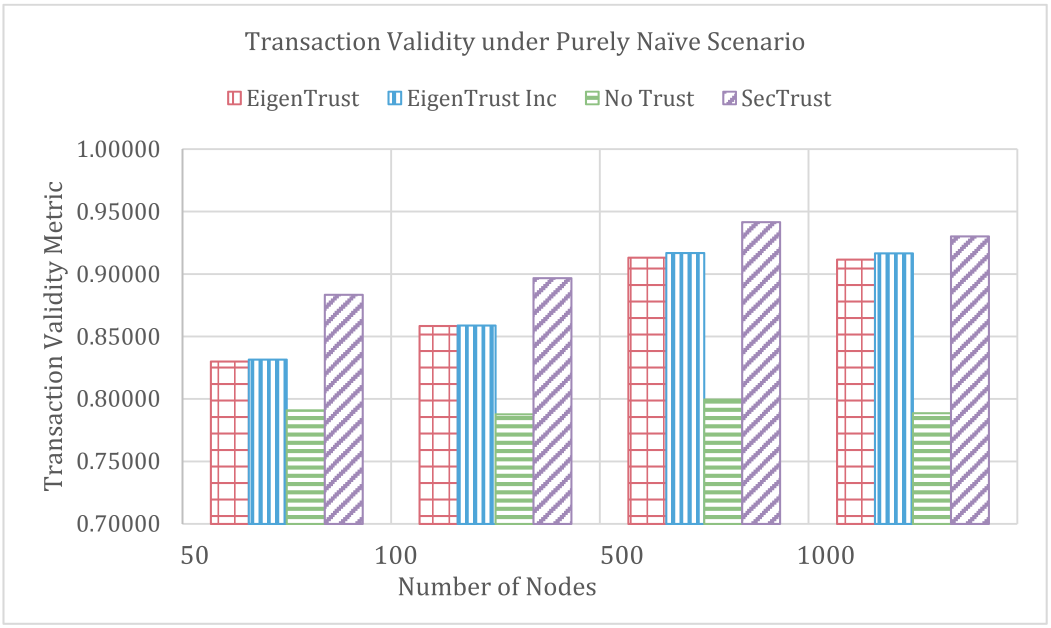


Figure 5.14: Transaction validity metric under a purely naïve scenario

Table 5.11: Trust systems completion times under purely naïve scenario (seconds)

Number of Nodes	EigenTrust	EigenTrust Incremental	No Trust	SecTrust
50 nodes	0.307	0.098	0.071	0.321
100 nodes	1.212	0.251	0.15	1.041
500 nodes	301.801 (5.03 mins)	75.122	61.851	278.826 (4.65 mins)
1000 nodes	1524.778 (25.41 mins)	618.444 (10.3 mins)	487.989 (8.13 mins)	1415.284 (23.59 mins)

2. Transaction Validity under the Purely Collective Scenario

Table 5.14 displays the completion times of the Transaction Validity across the 50, 100, 500 and 1000 nodes, and Figure 5.16 shows the simulation results under the purely collective simulation scenario.

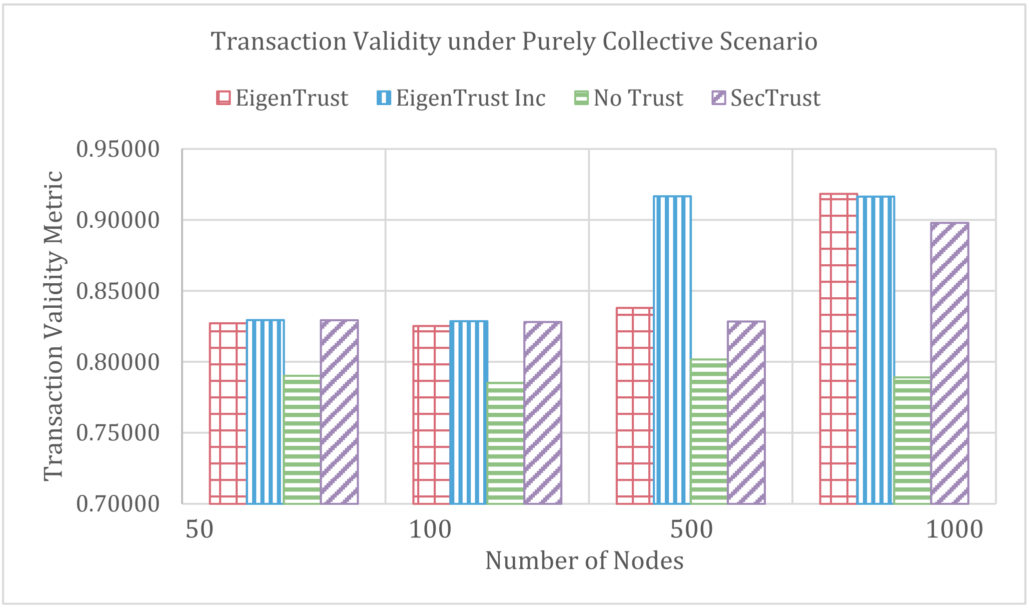


Figure 5.15: Transaction validity metric under a purely collective scenario

Table 5.12: Trust systems completion times under purely collective scenario (seconds)

Number of Nodes	**EigenTrust**	**EigenTrust Incremental**	**No Trust**	**SecTrust**
50 nodes	1.33	1.08	0.119	1.29
100 nodes	8.165	7.018	0.263	8.094
500 nodes	8848.784 (2.46 hrs)	8656.079 (2.41 hrs)	81.955	8841.063 (2.46 hrs)
1000 nodes	175,447.121 (48.74 hrs)	133,040.812 (36.96 hrs)	593.899 (9.90 mins)	146,554.542 (40.71 hrs)

In the simulation result of 50 nodes (Figure 5.16), EigenTrust, EigenTrust Incremental and *SecTrust* all appeared to have similar transaction validity metric values, although, EigenTrust Incremental had a slightly higher value of 0.82950 and this was followed by *SecTrust* with a value of 0.829240 and EigenTrust with 0.82720. No trust (None) had the least performance of 0.79020 transaction metric. However, No Trust had the fastest completion time of 0.12 second leaving EigenTrust, EigenTrust Incremental and *SecTrust* having completion times of a little over one second.

Furthermore, Figure 5.16 shows simulation result of all four trust systems using 100 nodes and a transaction count of 10,000. Again, EigenTrust Incremental had a marginally better value of 0.82870 over *SecTrust*, which had 0.82810. EigenTrust recorded a value of 0.82530 while No Trust recorded the least performance of 0.78530. All trust systems had completion time ranges of 7 to 8 seconds except for No Trust which had a completion time of 0.263 seconds.

In the simulation analysis with 500 nodes and a 100,000-transaction count, the performance difference becomes more apparent, as EigenTrust incremental with a transaction validity value of 0.91667 clearly showed better performance over *SecTrust* (0.82843), EigenTrust (0.83801) and No Trust (0.78905). From Figure 5.16, EigenTrust Incremental displays a wide margin difference compared with the rest of the trust systems. The reason for this could be attributed to the speed-up functionalities of the EigenTrust Incremental algorithm which bypasses the memory and CPU intensive re-calculations performed during each cycle of trust evaluation and computation thus, cycles with identical snapshots are re-used instead of being re-calculated. No Trust displays the fastest completion time of about 82 seconds, since it practically does not verify the validity of malicious transactions, while EigenTrust, EigenTrust Incremental and *SecTrust* observed completion times of 2.46, 2.41 and 2.46 hours respectively.

Finally, Figure 5.16 further shows the purely malicious collective result using 1000 nodes and 100,000-transaction count. EigenTrust and EigenTrust Incremental showed relatively similar transaction validity metric values of 0.91844 and 0.91648 and completion times of 48.74 and 36.96 hours respectively. *SecTrust* trails behind EigenTrust and EigenTrust Incremental with a performance value of 0.89795 and completion time of 40.71 hours and No Trust had a value of 0.78905 with completion time of 9.90 minutes.

The purely naïve scenario simulates malicious nodes acting independently while the collective scenario simulates nodes collaborating to destabilize the network. A key finding in this study is, *SecTrust* showed better performance under the purely naïve scenario while EigenTrust Incremental showed better performance under the purely malicious collective scenario.

5.6.4 Sybil Naïve

This section presents simulation results on how well a trust system can defend against Sybil activities. This section discusses results observed under the naïve scenario, in other words, Sybil attacking nodes acting independently. Two parameters are observed namely: i) the effectiveness metric of a trust system under Sybil attacks and ii) the Sybil effect metric (i.e., how well a trust system can defend against Sybil attack after personality change). The smaller the Sybil effect ratio, the better the trust system at mitigating Sybil attacks. These metrics have been presented in section 5.5. Table 5.15 shows the simulation setup for 50, 100, 500 and 1,000 user nodes with transaction capacity counts of 10,000, 10,000, 100,000 and 100,000 respectively. Table 5.16 shows the Sybil naïve scenario completion times.

Table 5.13: Trust simulation parameter settings

Parameters	Value
Number of Users and Number of Transactions	50 users with 10,000 transactions
	100 users with 10,000 transactions
	500 users with 100,000 transactions
	1,000 users with 100,000 transactions
Number of Files	5,000
Max. User Connections	2
Bandwidth Period	1
Zipf Constant	0.4
Pre-Trusted Users	5
Good Behaving Users	40
Sybil Attackers	10

Table 5.14: Trust systems completion times under the Sybil naïve scenario (seconds)

Number of Nodes	**EigenTrust**	**EigenTrust Incremental**	**No Trust**	**SecTrust**
50 nodes	0.342	0.098	0.08	0.3
100 nodes	1.19	0.237	0.166	1.021
500 nodes	291.698 (4.86 mins)	74.111	59.424	281.865 (4.70 mins)
1000 nodes	1565.791 (26.1 mins)	617.836 (10.30 mins)	493.162 (8.22 mins)	1426.923 (23.78 mins)

1. Effectiveness Metric

Under the Sybil naïve scenario in Figure 5.17, EigenTrust, EigenTrust Incremental and *SecTrust* displayed effectiveness values between 0.9 and 0.96 while the No Trust system displayed values that ranged around 0.78 and 0.80. To examine further the relative performances of EigenTrust, EigenTrust Incremental and *SecTrust*, Figure 5.18 provides a magnified view of the effectiveness metric of EigenTrust, EigenTrust Incremental and

SecTrust. *SecTrust* maintained consistent performance and scalability over EigenTrust and EigenTrust Incremental across the 50, 100, 500 and 1000 nodes during network simulation. EigenTrust shows a slight improvement over EigenTrust Incremental on the 1000 node simulation result. EigenTrust had the highest completion time across all number of nodes (50, 100, 500 and 1000) simulated and this was followed by *SecTrust*, EigenTrust Incremental and No Trust. The details are presented in Table 5.16 above.

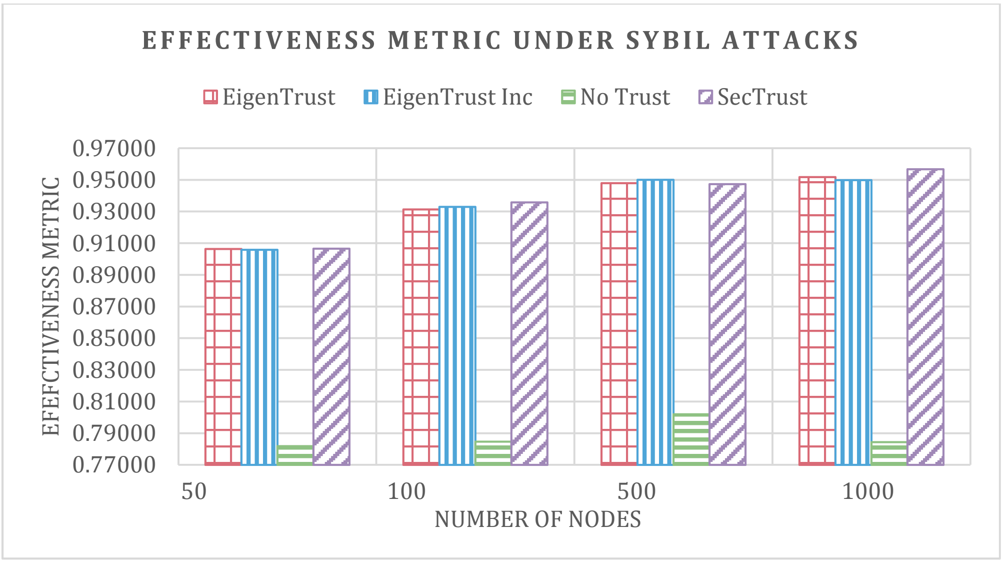


Figure 5.16: Effectiveness metric in a Sybil naïve scenario

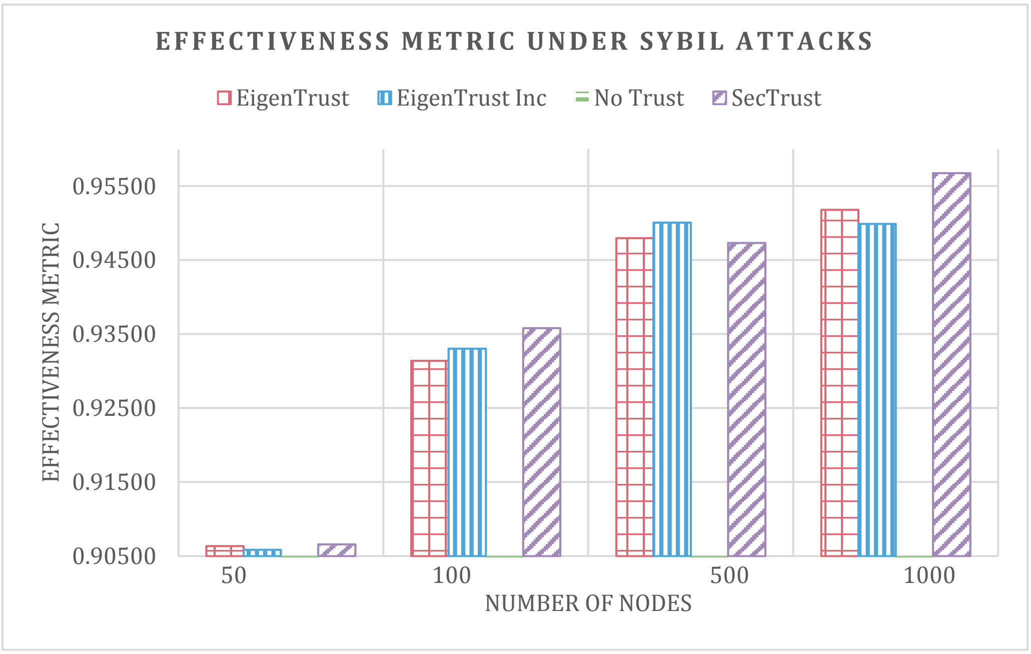


Figure 5.17: Enlarged effectiveness metric of EigenTrust, EigenTrust Incremental and *SecTrust* in a Sybil naïve scenario

2. Sybil Effect Ratio

Sybil attackers often take advantage of the minimal network entry requirements. And to be seen by other unsuspecting users as innocent, they assume different personalities. How well a trust system can defend against the attacks they perpetrate during and after their mutation in personality is reflected by the Sybil effect ratio. A higher Sybil effect ratio reflects the inability of a trust system to adequately mitigate the attacks posed by these Sybil attackers in the network while a lower Sybil ratio shows the effectiveness of a trust system to defend against Sybil attacks.

Figure 5.19 below shows the Sybil effect metric ratio of No Trust, EigenTrust, EigenTrust Incremental and *SecTrust* under a Sybil naïve scenario. No Trust is observed to have the highest Sybil effect ratio range of 0.34640 to 0.35005 on 50, 100, 500 and 1000-node during the simulation study; an indication of its vulnerability to more Sybil attacks compared to the other trust systems. The other three systems (EigenTrust, EigenTrust Incremental and *SecTrust*) displayed better results having a ratio range of 0.19876 to 0.23630. However, a magnification of the results of the Sybil effect ratio of EigenTrust, EigenTrust Incremental and *SecTrust* is presented in Figure 5.20. *SecTrust* displayed the least Sybil effect ratio across all nodes (50, 100, 500 and 1000) as compared with EigenTrust and EigenTrust Incremental. This shows the effectiveness of *SecTrust* in mitigating Sybil attacks in the distributed network while providing network scalability. Table 5.17 below shows the detailed Sybil effect metric ratio for all four trust systems.

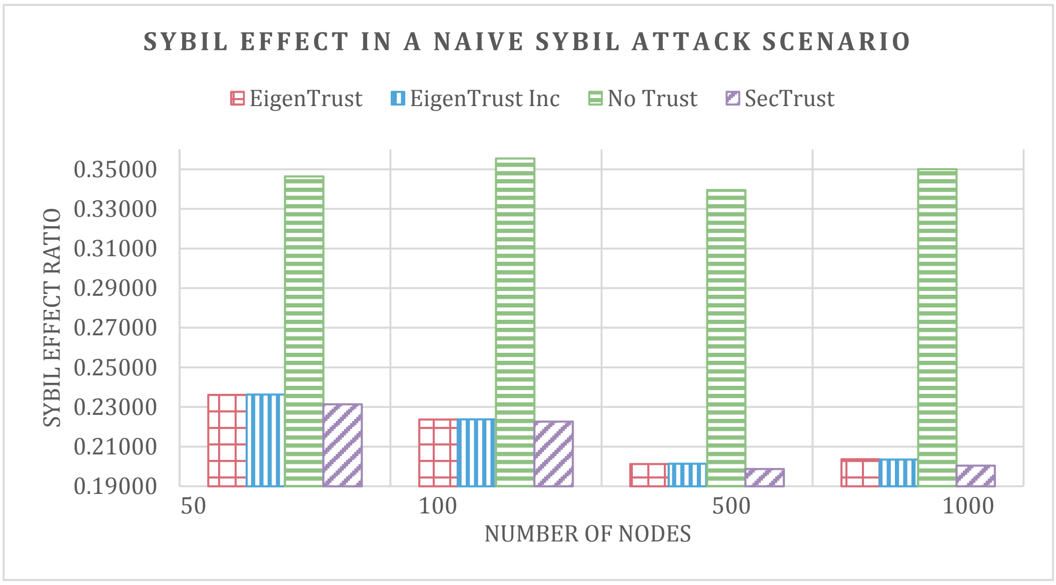


Figure 5.18: Sybil effect metric ratio in a Sybil naïve scenario

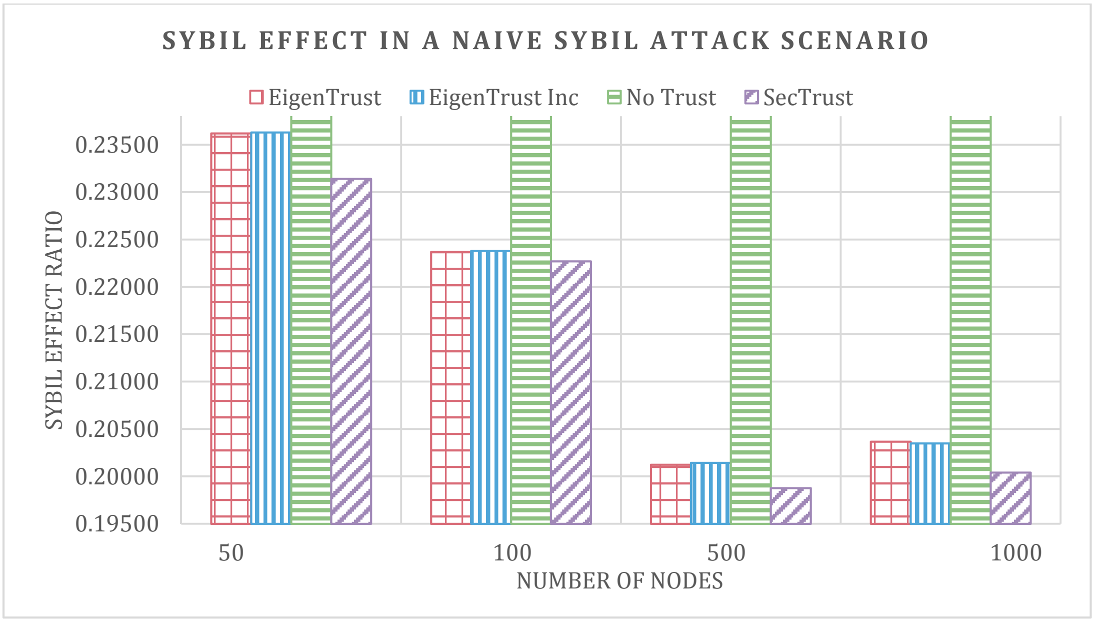


Figure 5.19: Enlarged Sybil effect metric ratio of EigenTrust, EigenTrust Incremental and *SecTrust* in a Sybil naïve scenario

Table 5.15: Sybil effect metric ratio in a Sybil naïve scenario

Trust System	50 Nodes	100 Nodes	500 Nodes	1000 Nodes
EigenTrust	0.23620	0.22370	0.20124	0.20367
EigenTrust Inc	0.23630	0.22380	0.20143	0.20349
None	0.34640	0.35550	0.33954	0.35005
SecTrust	0.23140	0.22270	0.19876	0.20040

5.6.5 Sybil Collective

The Sybil collective simulates performance of trust systems in a setting where Sybil attackers share honest information among themselves, but are dishonest to the global network community.

1. Effectiveness Metric

Under the Sybil collective scenario in Figure 5.21, EigenTrust, EigenTrust Incremental and *SecTrust* displayed effectiveness values above 0.9 while the No Trust system displayed values that ranged around 0.78414 and 0.79889. To examine further the relative performances of EigenTrust, EigenTrust Incremental and *SecTrust*, Figure 5.22 provides a magnified view of the effectiveness metric of EigenTrust, EigenTrust Incremental and *SecTrust*. *SecTrust* maintained consistent performance and scalability over EigenTrust and EigenTrust Incremental across the 50 to 1000 nodes during the network simulation.

EigenTrust Incremental showed better improvements over EigenTrust on 50, 100 and 500 nodes simulation, but displayed a lower performance on 1000-node simulation. *SecTrust* however, displayed a consistent and better performance over EigenTrust and EigenTrust Incremental in the 50, 100, 500 and 1000 nodes simulated, except for the 100 nodes simulation where EigenTrust Incremental had slightly better performance over *SecTrust*. Table 5.18 shows the completion times of all four trust systems.

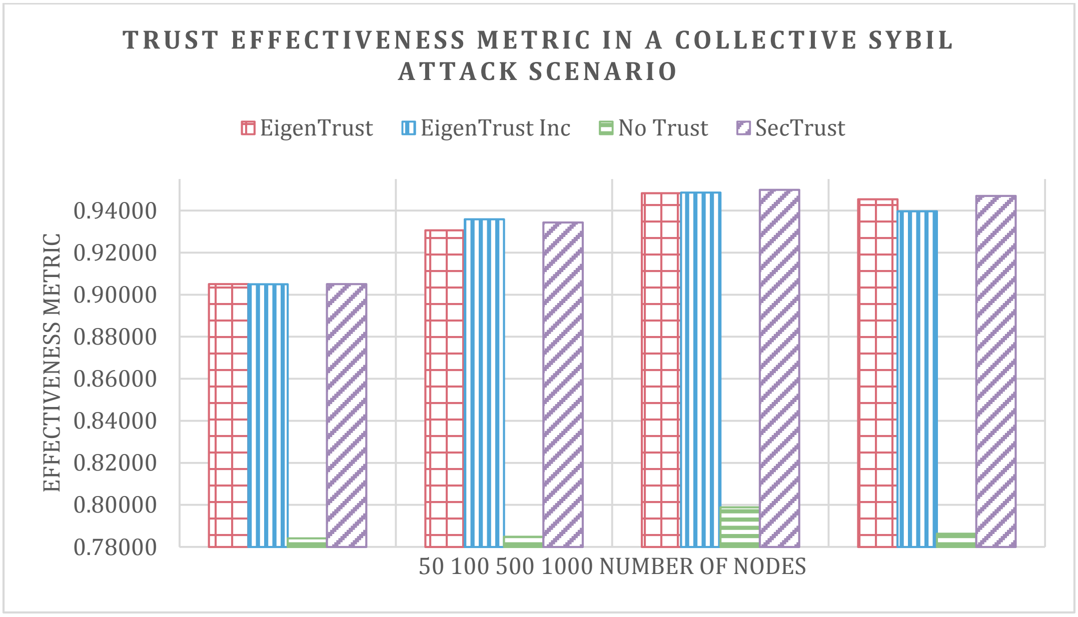


Figure 5.20: Effectiveness metric in a Sybil collective scenario

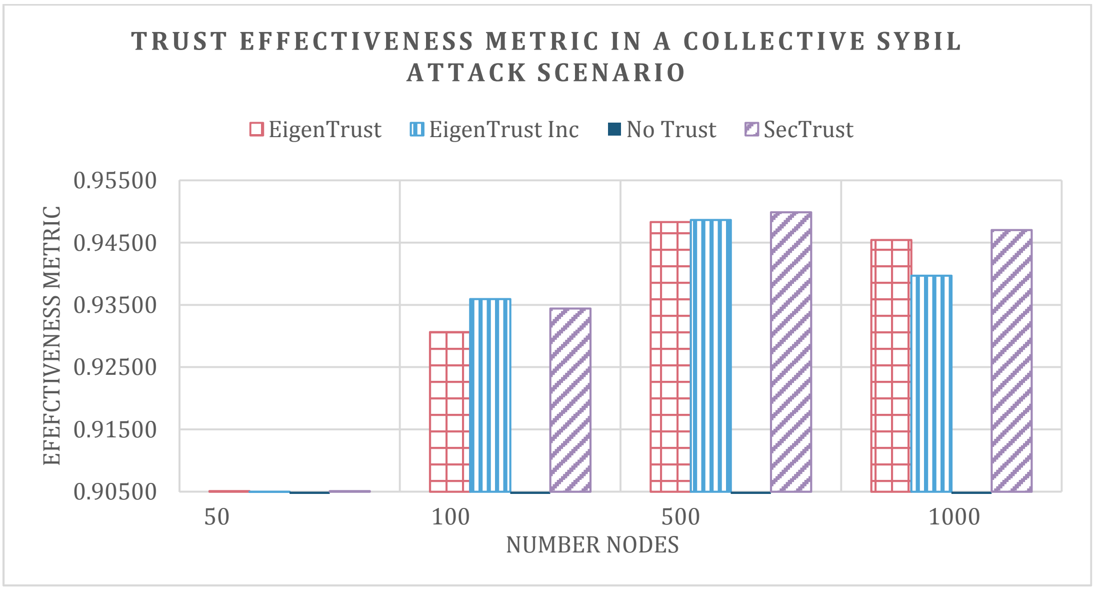


Figure 5.21: Effectiveness metric in a Sybil of EigenTrust, EigenTrust Incremental and *SecTrust* in a Sybil collective scenario

Table 5.16: Trust systems completion times under Sybil collective scenario (seconds)

Number of Nodes	EigenTrust	EigenTrust Incremental	No Trust	SecTrust
50 nodes	1.098	0.965	0.131	1.017
100 nodes	6.745	5.945	0.261	6.763
500 nodes	8435.188 (2.34 hrs)	9398.39 (2.61 hrs)	77.481	8429.156 (2.34 hrs)
1000 nodes	164181.286 (45.61 hrs)	132756 (36.88 hrs)	591.892 (9.87 mins)	141126.303 (39.20 hrs)

2. Sybil Effect Ratio

Figure 5.23 below shows the Sybil effect metric ratio of No Trust, EigenTrust, EigenTrust Incremental and *SecTrust* under the Sybil collective scenario. No Trust is again seen to have the highest Sybil effect ratio under the collective scenario with a range of 0.34175 to 0.35940 on 50, 100, 500 and 1000-node simulation studies; an indication of its weakness to defend against Sybil attacks as compared to the other trust systems. EigenTrust, EigenTrust Incremental and *SecTrust* however, showed lower Sybil effect metric ratio which ranged between 0.20136 and 0.24020.

Zooming into the Sybil effect ratios of EigenTrust, EigenTrust Incremental and *SecTrust* as presented in Figure 5.24 below, shows the comparative performance differences between the three systems. *SecTrust* displayed the least performance compared to EigenTrust and EigenTrust Incremental across all network sizes (50, 100, 500 and 1000 nodes). Under a collective scenario, *SecTrust* is generally observed to be inefficient in mitigating Sybil-like attacks or other malicious attacks. EigenTrust showed the best performance with Sybil effect ratios of 0.23640 on 50 nodes, 0.22250 on 100 nodes, 0.20136 on 100 nodes and 0.20332 on 1000 nodes respectively. Although EigenTrust Incremental overall had a faster time over EigenTrust, it can be observed however, that the penalty for the speed-up functionality of EigenTrust Incremental becomes apparent. A summary of the Sybil effect metric ratios of the trust systems is presented in Table 5.19.

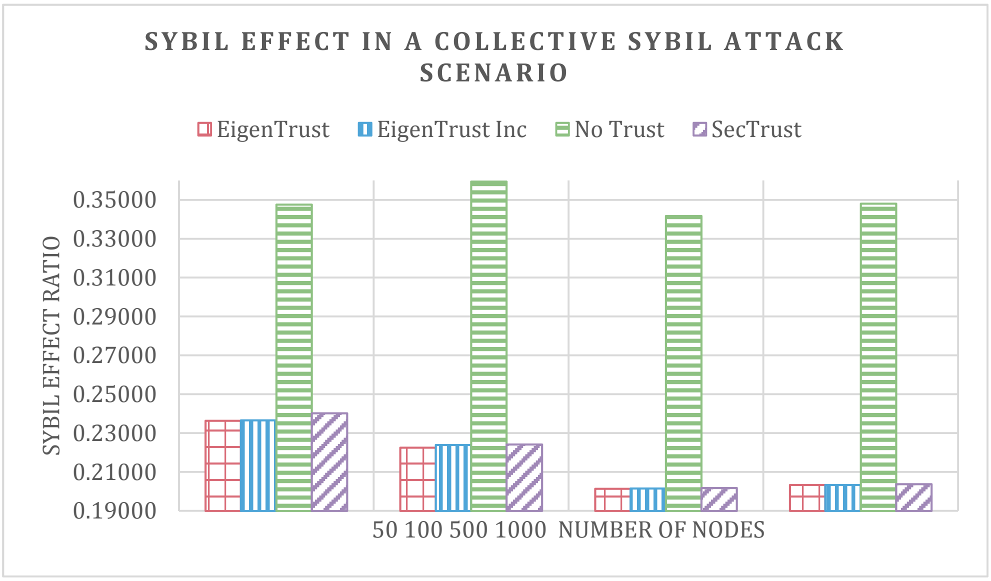


Figure 5.22: Sybil effect metric ratio in a Sybil naïve scenario

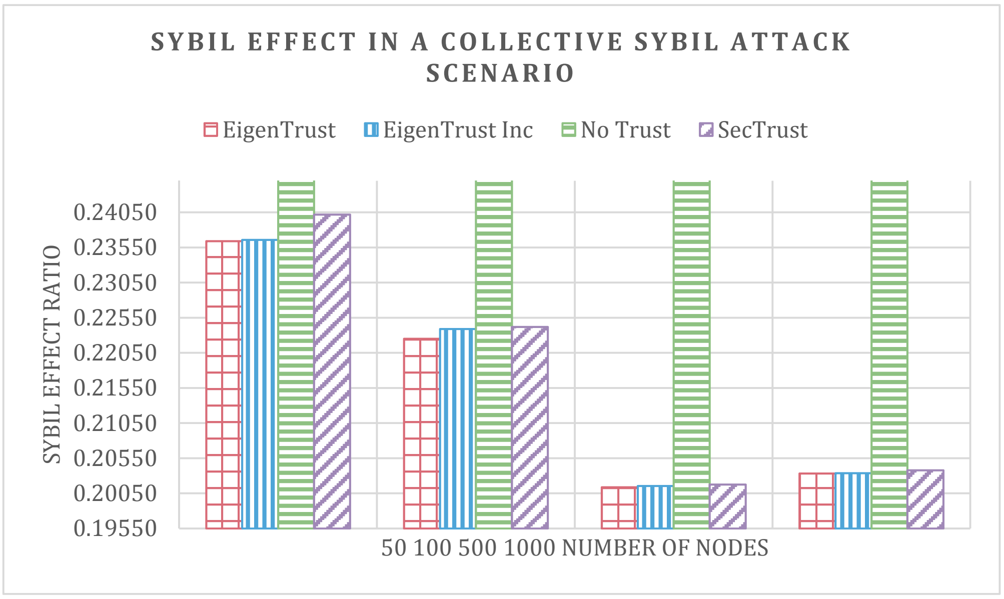


Figure 5.23: Enlarged Sybil effect metric ratio of EigenTrust, EigenTrust Incremental and *SecTrust* in a Sybil naïve scenario

Table 5.17: Sybil effect metric ratio in a collective scenario

Trust System	50 Nodes	100 Nodes	500 Nodes	1000 Nodes
EigenTrust	0.23640	0.22250	0.20136	0.20332
EigenTrust Inc	0.23660	0.22390	0.20156	0.20338
None	0.34760	0.35940	0.34175	0.34808
SecTrust	0.24020	0.22420	0.20176	0.20379

5.7 EigenTrust, EigenTrust Incremental, No Trust and SecTrust: A Comparative Framework Analysis

From the simulation study presented in this Chapter, it can be concluded that the SecTrust system shows better Trust effectiveness and Sybil isolation effectiveness among nodes in comparison to similar researched Trust systems. Table 5.18 shows a summary of the aggregated Trust and Sybil Effectiveness comparison of SecTrust against EigenTrust, EigenTrust Incremental and No Trust. Under all naïve scenarios SecTrust performed better than all the other Trust systems. However, under the collective scenarios, SecTrust closely trailed the EigenTrust Incremental.

Table 5.18: SecTrust Effectiveness Comparison

	Trust Effectiveness		**Sybil Effectiveness**	
Trust System	Naïve	Collective	Naïve	Collective
SecTrust	0.93893	0.93529	0.93663	0.93411
EigenTrust	0.92792	0.93058	0.93438	0.93237
Eigen Inc	0.93501	0.93754	0.93471	0.93231
No Trust	0.78990	0.78924	0.78823	0.78855

5.8 Summary

This Chapter provides an implementation of the proposed *SecTrust* system. QTM framework was used as guide in the development and modelling of the unique characteristics of the entities, behaviour, trust computation, trust formation, detection and isolation of malicious users. *SecTrust* system was subsequently developed (in Java) and integrated into the QTM framework.

The effectiveness of *SecTrust* was further benchmarked against other global trust and recommendation systems including EigenTrust, EigenTrust Incremental, TNA-SL and No Trust. Simulation evaluations were performed under various scenarios including purely naïve, purely collective, Sybil naïve and Sybil collective. In terms of performance, *SecTrust* under the purely Naïve (malicious attackers operating independently with no cooperation) and purely Sybil (Sybil attackers operating independently with no cooperation) was observed to be more effective over other trust systems. The performance of *SecTrust* under the collective scenarios, while being acceptable, did not outperform EigenTrust Incremental even though, it outperformed EigenTrust and No Trust.

Findings so far, enables this thesis to declare the comparative benefits of *SecTrust* over other trust systems so far investigated. The study therefore, asserts that within the limits of experimental error, *SecTrust* showed promising effectiveness and efficiency in trust management even in the presence of adversaries.

Finally, Social trust networks have benefitted from trust and recommendation systems, which have proven to be reliable and effective. A similar adoption in IoT, where thousands and millions of devices will be communicating with one another, seems justified as the adoption of a trust and recommendation system could prove to be well worth the effort.

Chapter 6 embeds the *SecTrust* system into the RPL routing protocol to defend against some IoT routing attacks discussed in Chapter 3. A fundamental reason for this is to provide a protocol that could scale in a large-sized IoT network while still being robust in the provision of adequate services and defence against routing attacks.

6 *SecTrust*: Simulation Study

In Chapter 5, a trust-based performance test on *SecTrust* was performed, which compared *SecTrust*'s ability against other global trust and recommendation systems in mitigating diverse malicious attacks. The trust-based performance tests in Chapter 5 show the following:

i. The quality of Trust effectiveness among nodes. Overall, *SecTrust*'s showed better Trust effectiveness over other trust systems investigated.

ii. The effectiveness and efficiency of *SecTrust*'s in comparison to other trust and recommendation systems in isolating malicious attacks.

iii. The quality of transactional validity of *SecTrust* in comparison to other trust systems. This is a pointer to the quality or reliability of transactions fulfilled by trusted nodes in a network.

iv. The scalability of *SecTrust*.

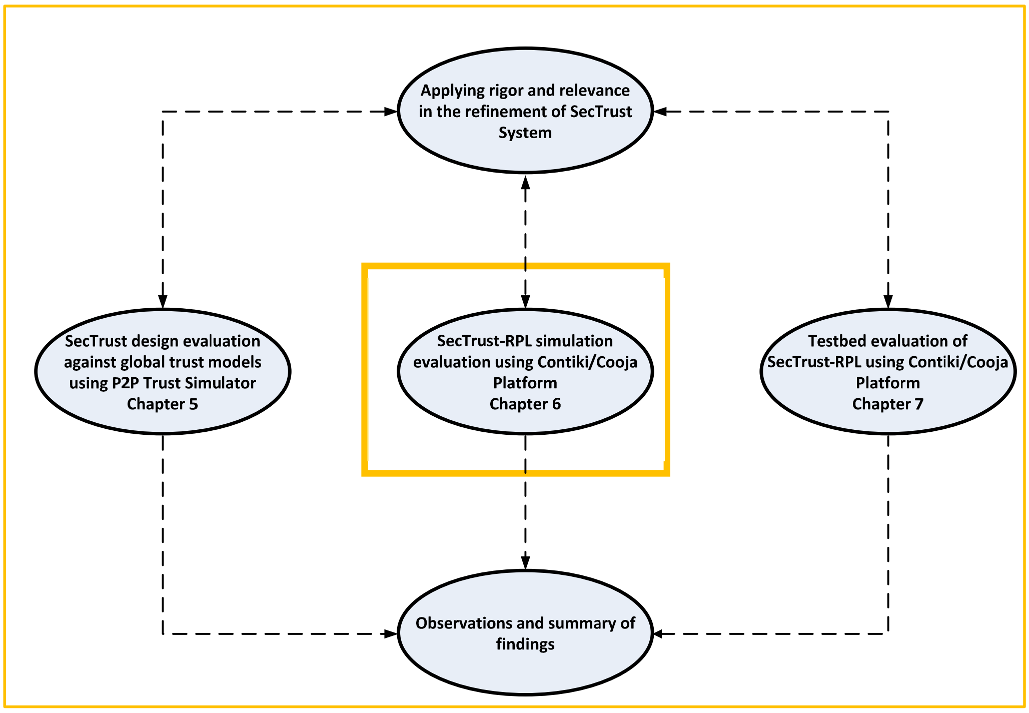

Applying rigor and relevance in the refinement of SecTrust System
SecTrust design evaluation against global trust models using P2P Trust Simulator Chapter 5
SecTrust-RPL simulation evaluation using Contiki/Cooja Platform Chapter 6
Testbed evaluation of SecTrust-RPL using Contiki/Cooja Platform Chapter 7
Observations and summary of findings

Figure 6.1: Simulation testing and validation stage of *SecTrust*

In this Chapter, an implementation of the design science testing of *SecTrust* is performed as discussed in the design and validation process in Chapter 4. The *rigor* and *relevance* testing of the *SecTrust* system is achieved through simulation. Figure 6.1 highlights the design science testing and validation stage of *SecTrust* through simulation in RPL. The *SecTrust* system was embedded in to the RPL routing protocol [62] where trust was computed for decision making during routing. This facilitated the choice of nodes made during routing to detect and isolate malicious nodes from being included in routing decisions. In Chapter 4, four attack types have been identified which are common IoT routing attacks that are addressed by this research work.

To proceed however, this Chapter begins by exploring the design principles and inner workings of RPL. This is necessary to ensure seamless integration of *SecTrust* design with RPL. The implementation of *SecTrust* in RPL protocol using InstantContiki 3.0 is further introduced to measure the performance of the *SecTrust* system in mitigating attacks proposed.

Furthermore, simulation results are presented and SecTrust in RPL (*SecTrust*-RPL) is compared with the implemented RPL standard in ContikiRPL.

SecTrust-RPL was compared with the standard RPL routing protocol for the following reasons:

i. This study has researched many secure RPL routing protocols, and there is none, for all intents and purposes, that addresses all four routing attacks including: Blackhole, Rank, Selective Forwarding and Sybil attacks.

ii. From the literature survey presented in this thesis, no literature study could be found that tested its Trust-based system against other globally renowned Trust-based systems (like this thesis has done) while also embedding such Trust-based system into the RPL routing protocol for secure routing operations and assessment.

iii. From the above mentioned survey, many secure RPL routing protocols were validated through simulation. This study is unaware of any Trust-based secure RPL routing protocol that presents both simulation studies, and testbed experiments as means of validation while addressing the four attacks proposed in this thesis.

iv. Although this thesis still considered some secure RPL routing protocols for possible comparison however, these Trust-based RPL protocols made some assumptions, which departed from the realistic boundaries of a smart building or a smart home that this thesis considered. Some studies considered factors like mobility, disaster/emergency and agricultural scenarios, which this thesis does not cover.

The simulation performances of both protocols were observed and analysed in the light of how well they could detect and isolate malicious attacks while coping with making optimal routing decisions. The concepts and some parts of the findings have been published in peer-reviewed conferences and journal [191, 192, 200]. This Chapter however, gives a rather revised and extensive version of papers published.

6.1 Simulation and Its Validity

Simulation – a programmable testing tool used for modelling and observing system behaviour in an environment with given parametric conditions. According to [201] simulation involves choosing the right model; discovering a method of implementation and enabling it to be run on a computer system to give output computation, observation and interpretation of results. Simulation provides a unique environment that permits a process to be performed and observing the results which could be used and applied for different purposes. A fundamental goal of simulation is to provide users with an economical means of testing experiments, analysing the results to have a foreknowledge of how it could be when eventually deployed in a real environment. Simulation further gives the opportunity to scale a testing scenario, which otherwise may be too expensive to perform or carry out in a real world system [202]. Since simulation assumes more or less a virtual reality of the real world, it therefore, can be modelled in diverse ways thus, revealing some otherwise unknown phenomena.

Although simulations are widely used today, however, the issue of the reliability and validity of simulation models and the results obtained come under question especially when they are regarded as epistemic equals with experiments, testbeds and analytic methods [203]. As opined by [203], the issue is always how accurate and trustworthy are the simulation results for the purpose intended? Of specific concern is, how accurate and trustworthy is Contiki/Cooja, the proposed network simulation tool for this study?

The development of Contiki/Cooja follows firm procedures of design, formulation and use. Contiki/Cooja does not only serve as a simulation tool, but as an emulator, which could be meshed and run with physical hardware devices [204]. Because of the foregoing, Contiki/Cooja was selected as the preferred tool for an accurate simulation study that will reflect a real-world scenario. Contiki/Cooja is discussed further in Sections 6.3 and 6.4.

6.2 RPL: IPV6 Routing Protocol for Low-Power and Lossy Networks (LLN)

Routing Protocol for Low power and lossy networks (RPL) is an IPv6 routing protocol standard developed by the Internet Engineering Task Force (IETF) [62]. RPL operates by initially discovering routes as soon as it becomes operational and this attribute classifies it as a proactive routing protocol. When the RPL protocol is initiated, it begins by creating a tree like topology called Directed Acyclic Graph (DAG). Every sensor node operating in a RPL network selects a parent node which acts like a packet gateway for that node. During network operations, if a node does not have a destination entry for a packet in its table, the node forwards the packet to its parent who in turn looks up its own routing table for a possible destination entry. If it exits, it simply forwards the packet to the recipient node however, if the node does not have the destination within its routing table, it then forwards the packet to its own parent for onward delivery. This process continues until the packet gets delivered to the right node recipient.

Route path selection is a key factor for RPL, unlike conventional routing protocols, RPL utilizes different factors to compute the best routing paths. Various routing metrics, route constraints and objective functions (OF) are among some notable factors used during routing. RPL has shown to be effective in various application areas of LLN because it segments packet processing, and makes routing decisions based on the specified OF like hop count, energy minimization or latency amongst others.

In the sections following, a description of RPL's topology, its routing mechanics and metrics, control messages, objective functions, constraints and trickle timer are discussed further.

6.2.1 Design Principles and Topology Formation in RPL

RPL forms a tree-like topology with a root node at the top (commonly referred to as the sink node) and leaves at the edges (known as sender nodes). RPL however, is distinct to a tree topology because it incorporates redundant links that is required in LLNs [62]. In RPL, the movement of traffic is described in terms of the "up" and "down" directions. Traffic moving from the sender nodes to the sink node is referred to as the "upward" traffic while traffic moving from the sink node to the sender nodes is regarded as the "downward" traffic. Figure 6.1 depicts a typical RPL network topology. During a RPL operation, links are required to be bidirectional to create both upward and downward routes to and from any node. For a node to be considered a parent, its reachability must be certain using an external mechanism that is being activated during the parent node selection process. This is to ensure the link properties and neighbour reachability is definite.

In RPL, a mechanism is used to access and convey control information (RPL route packet information) in packet segments, this helps in ensuring the mapping of data packets with a RPL instance, and it is a useful function for the validation of a RPL routing state such as when using the IPv6 Hop-by-Hop RPL option. Furthermore, RPL through some specialized mechanisms provides bidirectional communication between nodes. The bidirectional communication can be multipoint-to-point (sender nodes to sink node), point-to-multipoint (sink node to the sender nodes) and point-to-point (sender node to sender node). RPL is autonomous of any specific link layer features and this fulfils a key requirement of a layered IP architecture [22]. Moreover, RPL can operate on many link layer topologies, including constrained hosts, potentially lossy hosts and highly constrained hosts or router devices.

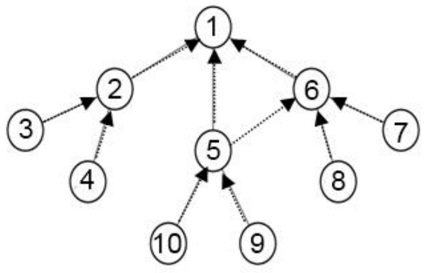


Figure 6.2: RPL Network Topology

6.2.2 Routing in RPL

RPL is considered a distance vector protocol, which performs routing either in a downward or upward format. Every information regarding the topology of RPL is maintained as a graph in what is called the Destination Oriented Directed Acyclic Graph (DODAG). The DODAG consists of paths from the sender nodes to the sink node. The sink node could act as an LLN border router (LBR) connected to a non-LLN network. When this occurs the DODAG is said to be grounded. Figure 6.3 shows a flowchart description of RPL routing operation.

1. Upward Routing

For an upward routing operation, the RPL protocol requires the information in the DODAG. The DODAG specifies the preferred parent of a node. Therefore, when a node needs to forward packets to the root node, it routes the packets through its preferred parent in the tree. The preferred parent in turn forwards the packet to its own preferred parent. This procedure continues until the packet is finally delivered to the root node.

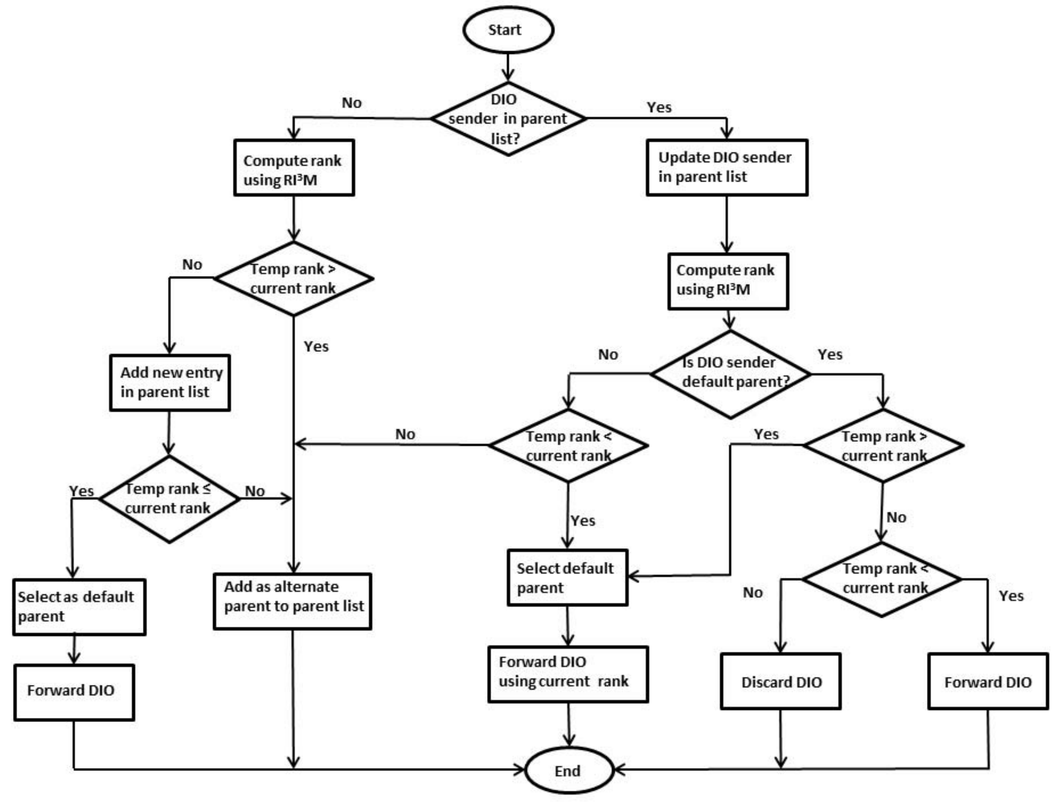


Figure 6.3: A RPL routing protocol operation flowchart [205]

During routing, every node maintains its Rank relative to its position in the DODAG tree. A node's Rank is a calculated 16-bit monotonic scalar value that is used for loop avoidance, and this is calculated according to the specified Objective Function (OF). Every DODAG is populated with parent information. The parent information are control and route information, which are used for routing and RPL network stability. The packets used by DODAG are: DODAG Information Object (DIO) and DODAG Information Solicitation (DIS) for transmitting the DODAG information. A DODAG formation is administered using the following guidelines namely;

i. Path metrics
ii. Objective Function (OF)
iii. Rank of a node for loop avoidance in the DODAG tree
iv. Any stipulated policies of a node [62].

In a RPL network, a RPL instance is known by its RPLinstanceID which may also comprise of other DODAGs identified by their DODAGID. Every DODAG comes with its own OF, routing metrics and the sink node. Each DODAG further uses its DODAGSequenceNumber to express the freshness of its topology information. A

DODAG is exclusively known by its combination of RPLinstanceID and DODAGID whereas, a DODAG version is known by the RPLinstanceID, DODAGID and DODAGSequenceNumber. In the build process of a DODAG tree, the following process ensues: A default root node is set via configuration and once set, the root node begins to multicast the link local DIO messages to its neighbours. After the DODAG formation an isolated node may solicit for DIO from the root or its potential parent node using DIS. Nodes receiving DIO from the root process these messages. As they process and discover that it is from a node with a lower Rank than theirs, they select it as their preferred parent, and they in turn broadcast their DIO to their neighbouring nodes who receive these messages, process them with a view to considering them as potential parents. Figure 6.4 shows the illustration of this operation.

From time to time, a node periodically re-evaluates the Rank of its parent, and sometimes it discovers that the Ranks of its parents is greater than its own a clear violation of the RPL protocol rules. The effect of this violation results in routing loops, and to address this, a local repair is initiated, so that the node re-aligns itself to another potential parent with a lower Rank than itself. However, over time, Rank inconsistencies build up in several parts of the network, which compels a global Rank repair leading to the creation of a new DODAG version as illustrated in Figure 6.5.

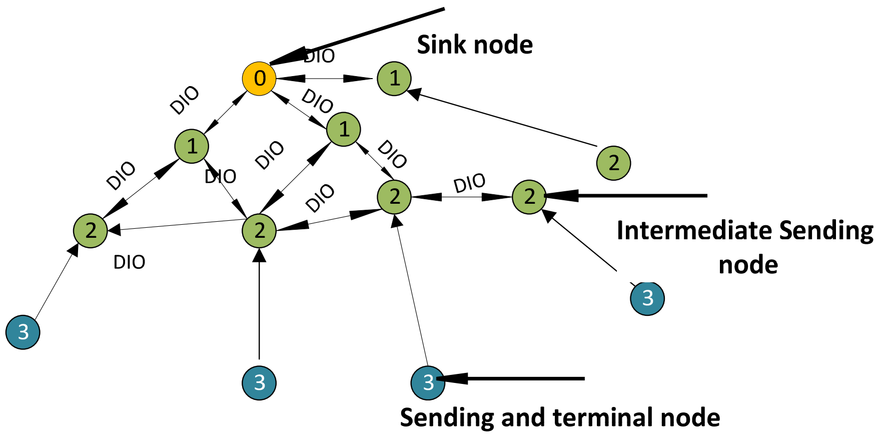


Figure 6.4: A RPL instance showing DIO broadcast

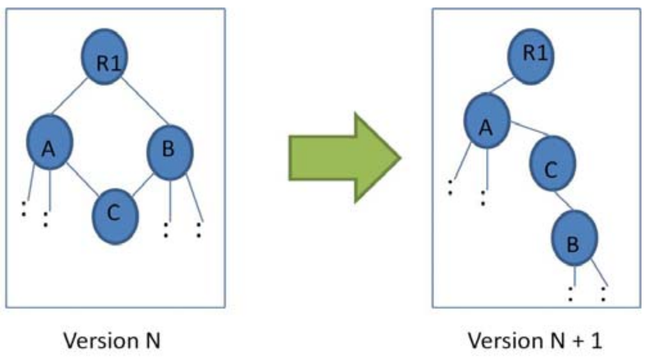


Figure 6.5: RPL DODAG Version - A RPL global repair due to Rank inconsistencies in the DODAG

2. Downward Routing

RPL uses the DAO messages to maintain the downward route topology; even though they are referred to as "downward" routes, DAOs are sent in an upward direction. This establishes RPL as a hierarchical network from the perspective of the flow of control messages. DAOs are sent only after the DODAG tree has been created through the exchange of DIO messages. In the specification of the IETF ROLL group [62], the forwarding tasks are separated from the routing task as the task of the forwarder is to accept datagrams and transmit them to the most suitable interface with regards to the information in the routing table. The router simply updates and maintains the routing information of the table.

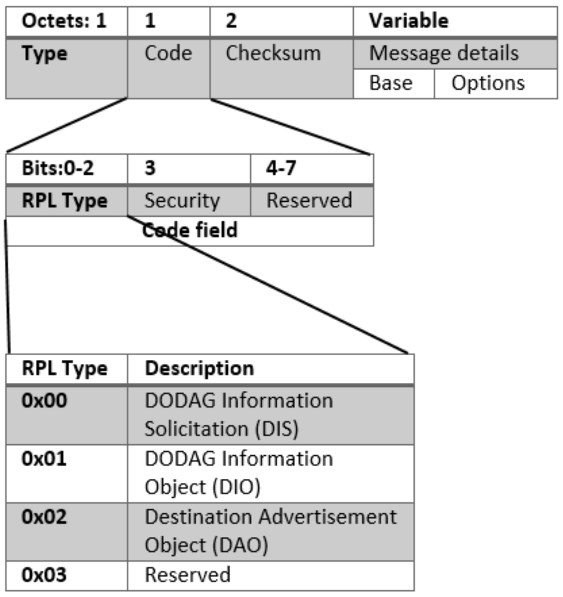
3. Control and Route Messages in RPL

For RPL to create and maintain its topology and routing table, it utilizes three control message types. These RPL messages are ICMPv6 packets with structure as shown in Figure 6.6. In the specification of RPL in [62, 206], and as shown in Figure 6.6, the RPL control message comprises of (a) an ICMPv6 header that is made up of the Type, Code and Checksum fields, (b) the message detail which contains a message base and an options field. The RPL *Code* field defines the type of RPL control messages and these control message types include: DODAG Information Object (DIO), DODAG Destination Advertisement Object (DAO) and DODAG Information Solicitation (DIS).

i. DODAG Information Object (DIO): DIO messages are transmitted by the root node in a RPL network to build the DAG, which is then then sent through a multicast in its DODAG structure. The DIO broadcast contains relevant network configuration information that allows nodes within the broadcast range to discover a RPL instance, to learn the configuration details, to choose a preferred DODAG parent, and to preserve the DODAG. The layout for the DIO base object is shown in Figure 6.7.

ii. DODAG Advertisement Object (DAO): RPL uses the DAO messages for creating a route from the leaf node back to the root sink node. To aid the downward route traffic, RPL propagates DAOs from the sender node to the sink node thus, creating a route path from any leaf node back to the root node. Figure 6.8 gives a layout of the DAO base object.

iii. DODAG Information Solicitations (DIS): Instances occur when a new node wishes to join the DODAG tree; or after a RPL network converges, some nodes may not receive the DIO messages and hence, they find themselves isolated from the DODAG tree. To address situations as described, the isolated nodes send out DIS messages to their neighbours soliciting for a potential parent that is already a member of the DODAG.

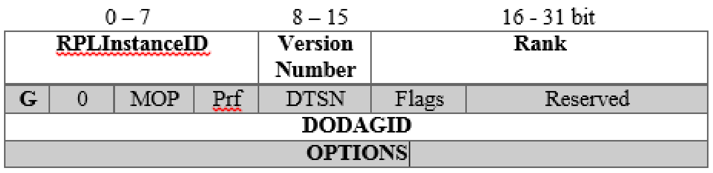


Octets: 1	1	2	Variable	
Type	Code	Checksum	Message details	
			Base	Options

Bits:0-2	3	4-7
RPL Type	Security	Reserved
Code field		

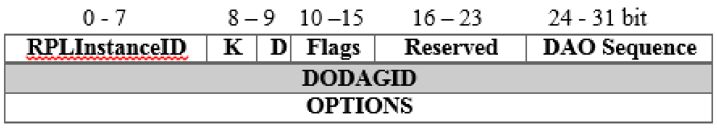
RPL Type	Description
0x00	DODAG Information Solicitation (DIS)
0x01	DODAG Information Object (DIO)
0x02	Destination Advertisement Object (DAO)
0x03	Reserved

Figure 6.6: Layout for IPv6 control and route message types



0 – 7				8 – 15	16 - 31 bit	
RPLInstanceID				**Version Number**	**Rank**	
G	0	MOP	Prf	DTSN	Flags	Reserved
DODAGID						
OPTIONS						

Figure 6.7: Layout for the DIO base object



0 - 7	8 – 9		10 –15	16 – 23	24 - 31 bit
RPLInstanceID	**K**	**D**	**Flags**	**Reserved**	**DAO Sequence**
DODAGID					
OPTIONS					

Figure 6.8: A layout of the DAO base object

6.2.3 Routing Metrics in RPL

Routing metrics are scalar values for determining the cost of a route path. The values are used for making optimal routing decisions especially when multiple routes are identified. In LLNs, this metric is used in making optimal routing decisions, and is defined in the Objective Function. The use of a scalar value for route determination makes it particularly attractive for embedding trust as a metric for route computation and the elimination of malicious nodes. Routing metrics are important to the successful creation and preservation of any network topology. Traditional networks employ the use of static metrics (hop count, bandwidth) for routing decisions. However, LLNs due to their unique properties and sphere of application, employ a broad range of routing metric dynamics. Different LLNs have different network requirements, which makes them unique in every respect such as optimal energy utilization, lossy link minimization and mobility. The IETF RFC 6550 specification of RPL [62] does not define specific forwarding metric policies. Furthermore, in the RPL draft from the IETF, constraints are also specified, which are used as filters for the specification of what should be included or excluded in the routing metric dynamics of RPL.

6.2.4 Defining the RPL Objective Function

A formal specification of how routes are defined, selected and optimized is regarded as the objective function (OF) of the RPL routing protocol. According to the IETF specification [62], different OFs could be specified for RPL.

Traffic in a RPL network is transported and delivered based on the defined OF which could be different for various traffic types. OFs are defined to optimize some particular metrics while also fulfilling specific constraint(s). Accordingly, the OF is used for effective routing path definition based on specific requirements. These requirements could be embedded in a series of programming logic in IoT motes, and utilized by RPL for routing purposes and this is explained further in the next section. A fundamental reason for the adoption of RPL for LLNs is - the separation of the OF from the central protocol specification [62] thus, making it easy for different OF specifications to be built into RPL and which in turn, makes it useful for a wide range of application scenarios.

6.2.5 The RPL Trickle Algorithm

The trickle algorithm is a timing mechanism used by RPL to conserve the energy resources of nodes in the network. RPL sends out DIO messages at every periodic interval called the trickle interval. Through this timing mechanism network freshness is maintained while the network nodes are not over burdened with excessive route updates. The minimum interval of any two DIOs corresponds to the *DIO Minimum Interval* that increases (doubling) until a maximum value is reached as determined by the *DIO Interval Doublings*. In embedding *SecTrust* into RPL, the trickle timer is used by the *Trust Monitoring and Update Process* to monitor, compute and update the trust system. In the RPL trickle algorithm, three parameters can be configured and they include: *Imin*, *Imax* and *k* -a redundancy constant of infinity [62], which are discussed next.

i. Imin

The *Imin* (minimum) provides the least time between two DIOs. DIOs are sent at intervals to limit redundant control traffic and hence, optimize the usage of the limited resources of the nodes. DIO broadcast is timed and controlled by the trickle algorithm of RPL which sets the minimum value as *Imin* and maximum value as *Imax*. The trickle timer value is set at the lowest value and this is doubled after each transmission until it reaches a certain threshold value of *Imax* and then resets while the process starts all over again. *Imin* is set by a RPL parameter called the *DIO Minimum Interval* and in InstantContiki, this variable constant is: *RPL_DIO_INTERVAL_MIN*.

The *Imin* is calculated as:

Imin = 2 ^ *RPL_DIO_INTERVAL_MIN*

Therefore, if *RPL_DIO_INTERVAL_MIN* = 10, the *Imin* = 2 ^ 10 =

1024 milliseconds (ms) =1 second.

This value becomes the smallest time interval between any two DIOs given that *RPL_DIO_INTERVAL_MIN* is set at 10.

ii. Imax

The *Imax* (maximum) is used as an upper limit for the frequency of times the *Imin* is doubled. In RPL, the *Imax* value is defined by the variable *DIO Interval Doublings.*

In InstantContiki, the variable constant is: *RPL_DIO_INTERVAL_DOUBLINGS* and it is calculated as:

Imax = Imin * 2 ^ *RPL_DIO_INTERVAL_DOUBLINGS*

Therefore, if *RPL_DIO_INTERVAL_DOUBLINGS* is set at the value of 10 and *Imin* is 1024 as illustrated above, then *Imax* = 1024 * 2 ^ 9 = 524,288 ms = 524.3 second = 8 minutes and 44 seconds.

iii. Redundancy Constant of Infinity (k)

This is a value set in the RPL protocol to limit DIO transmissions. This helps ensure that nodes within the network do not deplete their limited resources through the excessive transmissions of DIOs. In addition, it helps maintain network stability among nodes in the RPL network.

6.3 Contiki

A wireless sensor network operating system has several key components; the kernel, a program loader, libraries and a collection of processes [204]. It finds its use in embedded networked systems and smart objects.

The Contiki OS offers a number of mechanisms, which helps in embedding and programming smart object applications. It has memory allocation libraries, linked list with manipulation capabilities and communication abstractions; it is a pioneering operating system, which offers IP communication for smart objects. The Contiki OS and its applications are entirely developed using the C programming language thus making it possible for diverse architectures including the Texas Instruments' MSP430, and Crossbow's TelosB microcontrollers to be deployed in Contiki. It is an event-driven emulator that runs processes as event handlers. The Contiki system is divided in two parts: the *core* (kernel) and the *loaded* programs. The core is a collection of the Contiki kernel, program loader, language run-time libraries, and the communication stack for hardware communication using the device drivers [204]. The Contiki operating system offers different modules for various tasks hence, all tasks are logically divided into different directories based on the protocol layer. The routing module for example, is kept in a directory which is located at "*contiki/core/net/rpl*" and comprises of system files strictly for RPL routing. Other RPL routing protocol related files include: *rpl-dag.c* - this maintains the functionality for Directed Acyclic Graph (DAG) creation; and *rpl-icmp6.*c - offers the functionality for putting together all ICMP messages.

Contiki can be run on diverse hardware platforms such as the Z1 platform (Texas Instruments) SKY platform (Telos B), Mica-Z platform, all of which are based on the microcontroller MSP430 with radio chip CC2420 family. In reference to the used protocols, they all employ the 6LoWPAN mechanism for header compression and frame packet exchange over 802.15.4 radio links. However, in this thesis and for the purpose of implementing a secure routing system in RPL, the RPL protocol implemented in Contiki is of interest hence, coding is performed on the RPL protocol of the Contiki operating system. Figure 6.9 shows the Contiki protocol layer stack that is designed with the perspective of having a minimal set of operational TCP/IP features.

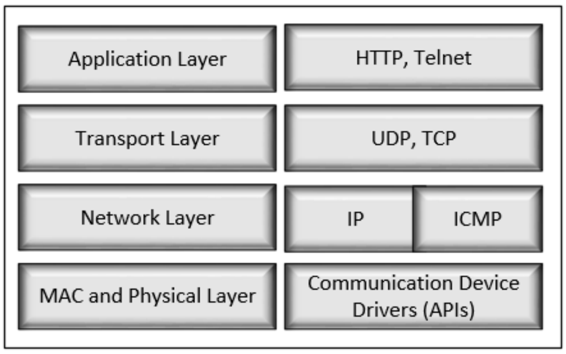


Figure 6.9: Protocol Stack in Contiki OS

6.4 Cooja

Cooja is a Java-based simulator developed primarily for the simulation of sensor networks running the Contiki OS [207]. Cooja is implemented using Java; however, it allows the programming and emulation of sensor nodes in the C programming language.

A key feature in Cooja which makes it unique from other emulators, is its capability of operating at three different levels namely: the Network Level, the Operating System Level and the Machine code instruction level [207]. Cooja runs Contiki programs, which could be compiled on a host CPU or on an MSP430 emulator (the micro-controller processor on which most sensors are built on). Simulation in Cooja allows for graphical interactions with simulated nodes, which is achieved through the use of plugins like network visualizer, radio logger, mote output and timeline. Cooja builds and stores the node simulation in *.xml* files having the extension "*.csc*" (Cooja Simulation Configuration). This *xml* build file stores information regarding the plugins, node type, node positions, random seed and radio medium and so on.

A fundamental benefit of Cooja in this research study, is its dual features of having a good simulation speed with testbed realism. Cooja is used in this research work, since it supports a common platform for simulation and testbed validation with its *checkpointing* feature [208]. The *checkpointing* feature is used to interchange between the execution of any simulation and testbed application. In this research work, an interchange was made between a simulation carried out in Contiki/Cooja with the physical testbed deployed using the XM1000 motes [209]. The state of a node is maintained both in the simulation and testbed scenarios. A network *checkpoint* is performed after which, the state of the network can then be transferred from the simulation scenario to the physical testbed deployed where the execution state is resumed. The state of a physical testbed can also be frozen and transferred to the simulation state and execution also resumed. In both cases, the log output is measured and the results evaluated to examine if the testbed results validate the simulation results and vice versa. In this study however, the testbed results are used as a validation tool for the simulation study conducted [210].

6.5 Radio Propagation Models in Cooja

Cooja offers simulation on five radio propagation models namely: No radio traffic, Directed Graph Radio Medium (DGRM), Unit Disk Graph Medium (UDGM) Constant Loss, Unit Disk Graph Medium (UDGM) Distance Loss and the Multi-Path Ray-tracer Medium (MRM) [22]. A description of each model is given below while a justification for using a specific model for the purpose of this research work is advanced.

i. No Radio Traffic Model: This disables a node's capability to communicate on a specific channel hence, cannot be used for simulation. This, however, may be needed to simulate a dead node that makes no transmission.

ii. Unit Disk Graph Medium (UDGM) Constant Loss Model: This medium models the packet transmission range as an ideal disk whereby all sensor nodes outside the disk range are not able to receive any packets whereas sensor nodes within the disk's range receive packets. In this model, the ratio of a node's present output power to the maximum output power is multiplied by a predefined maximum transmission range, which is then compared with the distance in the simulation. As an example, given a node with a radio transmission range of 100m and a current output power set at half the maximum of its capacity, then the disk will have a transmission radius of 50m.

iii. Unit Disk Graph Medium (UDGM) Distance Loss: This model which is also known as the link failure model is an extension of the UDGM constant loss model however, the distance loss model extends the constant loss model in two ways: i) the interferences are treated as attenuating factors during transmission while interfered packets are regarded as lost packets when the interference distance becomes larger than the transmission distance. ii) the success ratio of the receiver and transceiver can be defined before and during simulation. Therefore, the success of the transmission or reception of packets are predicated on the SUCCESS_RATIO_TX and SUCCESS_RATIO_RX probabilities in Cooja. Nevertheless, SUCCESS_RATIO_*TX* is deemed unsuccessful when all nodes do not receive a sent packet while SUCCESS_RATIO_*RX* is considered unsuccessful when a destination node does not receive a packet specifically sent to it.

iv. Directed Graph Radio Medium (DGRM): This model generates the network topology of nodes using the nodes' edges. DGRM is basically used to define the transmission success ratio of packets and to get the transmission delay on the asymmetric links.

v. Multi-path Ray-tracer Medium (MRM): This model uses the ray tracing system to generate the communication topology among nodes. A node's receiver power is computed using the *Friis* formula while interferences are treated as attenuators. This model also computes the reflections, refractions and diffractions.

In this thesis, the *UDGM with distance loss* has been adopted since it provides a real-world emulation of the lossy links and shared media collision among wireless sensor nodes. Figure 6.10 shows a representation of UDGM with distance loss in Cooja. The inner green circle represents the transmission range of node 1 while the grey circle shows its collision circle domain with other radios. The Figure also shows the percentage reception ratio between nodes 1 and 2. Node 3 is located within the collision range of node 1 hence, communication between nodes 1 and 3 cannot be established.

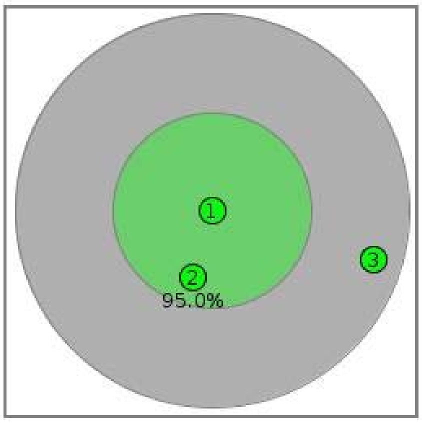


Figure 6.10: A UDGM distance loss illustration in Cooja

6.6 ContikiRPL

ContikiRPL is the RPL implementation in Contiki/Cooja, and this consists of several source files. The *rpl-icmp6.c* handles the ICMPv6 control messages (DIO, DAO and DIS) while *rpl-dag.c* handles the RPL DAG creation process.

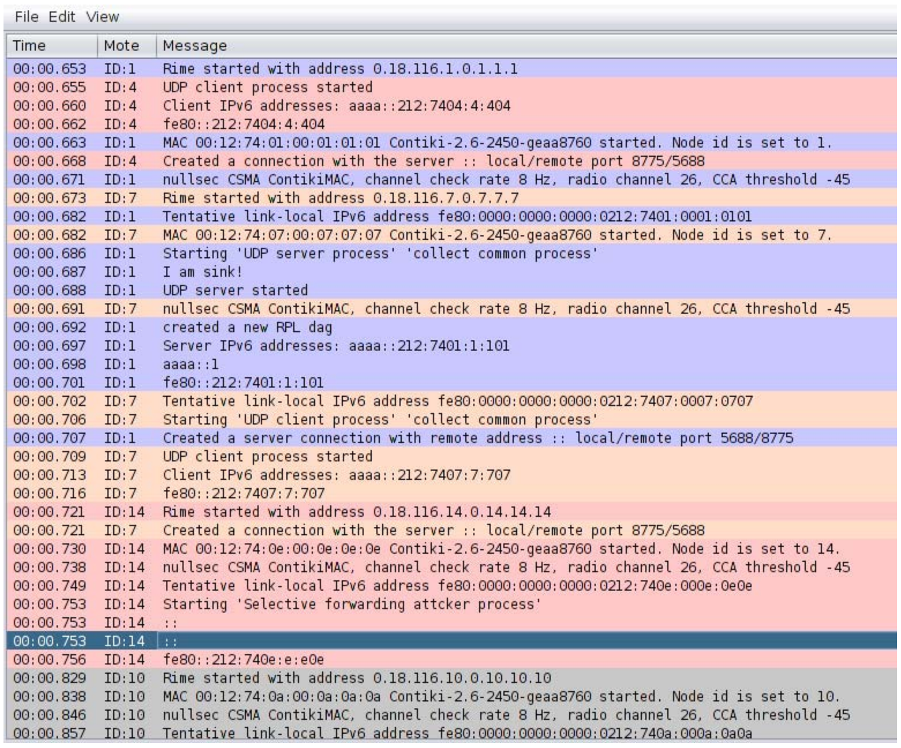
Figure 6.11 shows the log output of this implementation when the RPL process is started. When RPL is started, the sink node advertises itself as the sink and it begins sending out broadcast in the form of DIO messages so that surrounding nodes could know the DAG-ID and its configurations to join the network. When a child node (see Figure 6.4) obtains a DIO, it computes its Rank using the Rank of its parent and the cost of reaching the parent. Every node adheres to the specification in the OF for the selection of their preferred parent. As a standard in the IETF draft, and to avoid routing loops, a node considers another node a potential parent if, the potential parent's Rank is lower than its own. Thus, a node will typically add such a node with a lower Rank to its potential parent list during the determination and selection of a preferred parent. The DIO broadcast is shown in Figure 6.12. The formation of the upward traffic is achieved upon a node's selection of its preferred parent hence, the node begins the unicast of DAO messages to its parent to advertise its prefix. The parent node in turn updates its routing table thus, ensuring a downward traffic (refer to Figure 6.13). Figure 6.13 shows the unicast of DAO of node 3 to the sink node (node 1). When a node fails to receive a DIO after a period of 5 seconds, by default it sends a DIS broadcast. Nodes in receipt of the DIS immediately forward DIOs to the DIS-sending node. Figure 6.14 shows the DIS broadcast of node 2 to nodes 8 and 14. This is in a bid for node 2 to identify and select a preferred parent.

In the ContikiRPL implementation of RPL two OFs are defined and they are: Objective Function zero (OF0) and Expected Transmission Count (ETX).

i. Objective Function Zero (OF0): OF0 utilizes hop count in determining the best route to select. The hop count metric essentially counts the hops (nodes) between the source and the destination of the node. A hop count of 4 implies that there are 4 intermediate nodes between the source and the destination.

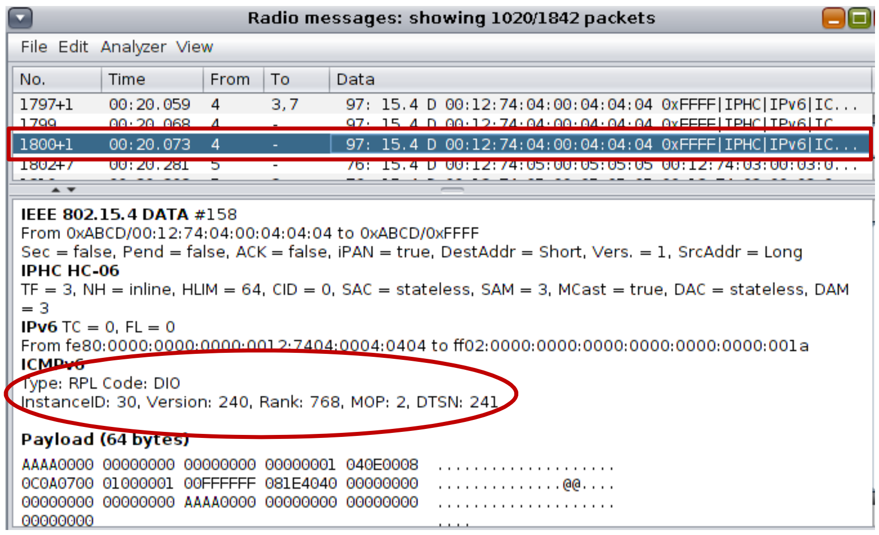
ii. Expected Transmission Count (ETX): The ETX metric is the expected number of transmissions needed to effectively transmit a packet from the source to the destination on the link. The ETX metric computes the shortest path from a node to the DODAG root which also represents the path with the least ETX sum from the defined source to the DODAG root node [211]. An ETX path with 4 links of 100% delivery ratio is 4, but an ETX path having 4 links of 50% delivery ratio is 8. The lower the ETX value the better the route path selection. The ContikiRPL uses the ETX metric as the OF in the implementation of the *Minimum Rank with Hysteresis Objective Function* (MRHOF).

In this proposed secure RPL protocol (*SecTrust-RPL)*, comparison is made with the *Minimum Rank with Hysteresis Objective Function* (MRHOF) implementation of RPL (MRHOF-RPL). MRHOF-RPL is specifically chosen for comparison with the proposed *SecTrust-RPL*, since MRHOF-RPL is shown to have better performance than the Objective Function zero (OF0) implementation of RPL (OF0-RPL) [212, 213].



```
Time        Mote   Message
00:00.653   ID:1   Rime started with address 0.18.116.1.0.1.1.1
00:00.655   ID:4   UDP client process started
00:00.660   ID:4   Client IPv6 addresses: aaaa::212:7404:4:404
00:00.662   ID:4   fe80::212:7404:4:404
00:00.663   ID:1   MAC 00:12:74:01:00:01:01:01 Contiki-2.6-2450-geaa8760 started. Node id is set to 1.
00:00.668   ID:4   Created a connection with the server :: local/remote port 8775/5688
00:00.671   ID:1   nullsec CSMA ContikiMAC, channel check rate 8 Hz, radio channel 26, CCA threshold -45
00:00.673   ID:7   Rime started with address 0.18.116.7.0.7.7.7
00:00.682   ID:1   Tentative link-local IPv6 address fe80:0000:0000:0000:0212:7401:0001:0101
00:00.682   ID:7   MAC 00:12:74:07:00:07:07:07 Contiki-2.6-2450-geaa8760 started. Node id is set to 7.
00:00.686   ID:1   Starting 'UDP server process' 'collect common process'
00:00.687   ID:1   I am sink!
00:00.688   ID:1   UDP server started
00:00.691   ID:7   nullsec CSMA ContikiMAC, channel check rate 8 Hz, radio channel 26, CCA threshold -45
00:00.692   ID:1   created a new RPL dag
00:00.697   ID:1   Server IPv6 addresses: aaaa::212:7401:1:101
00:00.698   ID:1   aaaa::1
00:00.701   ID:1   fe80::212:7401:1:101
00:00.702   ID:7   Tentative link-local IPv6 address fe80:0000:0000:0000:0212:7407:0007:0707
00:00.706   ID:7   Starting 'UDP client process' 'collect common process'
00:00.707   ID:1   Created a server connection with remote address :: local/remote port 5688/8775
00:00.709   ID:7   UDP client process started
00:00.713   ID:7   Client IPv6 addresses: aaaa::212:7407:7:707
00:00.716   ID:7   fe80::212:7407:7:707
00:00.721   ID:14  Rime started with address 0.18.116.14.0.14.14.14
00:00.721   ID:7   Created a connection with the server :: local/remote port 8775/5688
00:00.730   ID:14  MAC 00:12:74:0e:00:0e:0e:0e Contiki-2.6-2450-geaa8760 started. Node id is set to 14.
00:00.738   ID:14  nullsec CSMA ContikiMAC, channel check rate 8 Hz, radio channel 26, CCA threshold -45
00:00.749   ID:14  Tentative link-local IPv6 address fe80:0000:0000:0000:0212:740e:000e:0e0e
00:00.753   ID:14  Starting 'Selective forwarding attcker process'
00:00.753   ID:14  ::
00:00.753   ID:14  ::
00:00.756   ID:14  fe80::212:740e:e:e0e
00:00.829   ID:10  Rime started with address 0.18.116.10.0.10.10.10
00:00.838   ID:10  MAC 00:12:74:0a:00:0a:0a:0a Contiki-2.6-2450-geaa8760 started. Node id is set to 10.
00:00.846   ID:10  nullsec CSMA ContikiMAC, channel check rate 8 Hz, radio channel 26, CCA threshold -45
00:00.857   ID:10  Tentative link-local IPv6 address fe80:0000:0000:0000:0212:740a:000a:0a0a
```

Figure 6.11: A RPL process initialization

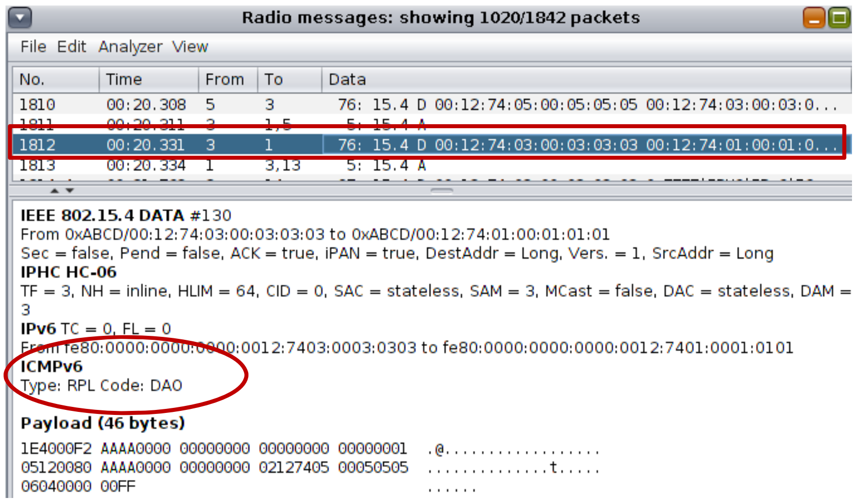


```
No.      Time        From  To    Data
1797+1   00:20.059   4     3,7   97: 15.4 D 00:12:74:04:00:04:04:04 0xFFFF|IPHC|IPv6|IC...
1799     00:20.068   4           97: 15.4 D 00:12:74:04:00:04:04:04 0xFFFF|IPHC|IPv6|IC
1800+1   00:20.073   4     -     97: 15.4 D 00:12:74:04:00:04:04:04 0xFFFF|IPHC|IPv6|IC...
1802+7   00:20.281   5     -     76: 15.4 D 00:12:74:05:00:05:05:05 00:12:74:03:00:03:0...

IEEE 802.15.4 DATA #158
From 0xABCD/00:12:74:04:00:04:04:04 to 0xABCD/0xFFFF
Sec = false, Pend = false, ACK = false, iPAN = true, DestAddr = Short, Vers. = 1, SrcAddr = Long
IPHC HC-06
TF = 3, NH = inline, HLIM = 64, CID = 0, SAC = stateless, SAM = 3, MCast = true, DAC = stateless, DAM
= 3
IPv6 TC = 0, FL = 0
From fe80:0000:0000:0000:0012:7404:0004:0404 to ff02:0000:0000:0000:0000:0000:0000:001a
ICMPv6
Type: RPL Code: DIO
InstanceID: 30, Version: 240, Rank: 768, MOP: 2, DTSN: 241

Payload (64 bytes)
AAAA0000 00000000 00000000 00000001 040E0008  ....................
0C0A0700 01000001 00FFFFFF 081E4040 00000000  ..............@@....
00000000 00000000 AAAA0000 00000000 00000000  ....................
00000000                                      ....
```

Figure 6.12: DIO broadcast by node 4 to surrounding nodes for who may potentially consider it as their preferred parent node

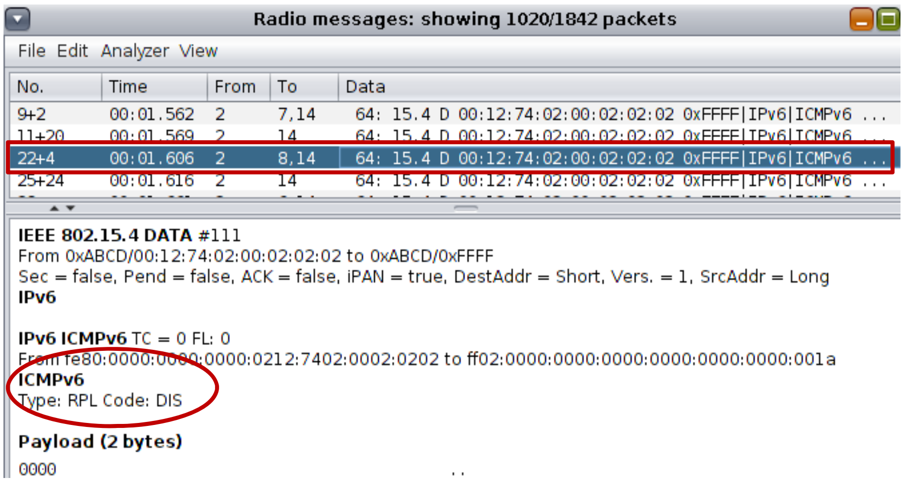


```
No.      Time        From  To    Data
1810     00:20.308   5     3     76: 15.4 D 00:12:74:05:00:05:05:05 00:12:74:03:00:03:0...
1811     00:20.311   3     1,5    5: 15.4 A
1812     00:20.331   3     1     76: 15.4 D 00:12:74:03:00:03:03:03 00:12:74:01:00:01:0...
1813     00:20.334   1     3,13   5: 15.4 A

IEEE 802.15.4 DATA #130
From 0xABCD/00:12:74:03:00:03:03:03 to 0xABCD/00:12:74:01:00:01:01:01
Sec = false, Pend = false, ACK = true, iPAN = true, DestAddr = Long, Vers. = 1, SrcAddr = Long
IPHC HC-06
TF = 3, NH = inline, HLIM = 64, CID = 0, SAC = stateless, SAM = 3, MCast = false, DAC = stateless, DAM =
3
IPv6 TC = 0, FL = 0
From fe80:0000:0000:0000:0012:7403:0003:0303 to fe80:0000:0000:0000:0012:7401:0001:0101
ICMPv6
Type: RPL Code: DAO

Payload (46 bytes)
1E4000F2 AAAA0000 00000000 00000000 00000001  .@..................
05120080 AAAA0000 00000000 02127405 00050505  ..............t.....
06040000 00FF                                 ......
```

Figure 6.13: DAO message sent by node 3 to node 1 for downward route formation



```
No.      Time        From  To    Data
9+2      00:01.562   2     7,14  64: 15.4 D 00:12:74:02:00:02:02:02 0xFFFF|IPv6|ICMPv6 ...
11+20    00:01.569   2     14    64: 15.4 D 00:12:74:02:00:02:02:02 0xFFFF|IPv6|ICMPv6 ...
22+4     00:01.606   2     8,14  64: 15.4 D 00:12:74:02:00:02:02:02 0xFFFF|IPv6|ICMPv6 ...
25+24    00:01.616   2     14    64: 15.4 D 00:12:74:02:00:02:02:02 0xFFFF|IPv6|ICMPv6 ...

IEEE 802.15.4 DATA #111
From 0xABCD/00:12:74:02:00:02:02:02 to 0xABCD/0xFFFF
Sec = false, Pend = false, ACK = false, iPAN = true, DestAddr = Short, Vers. = 1, SrcAddr = Long
IPv6

IPv6 ICMPv6 TC = 0 FL: 0
From fe80:0000:0000:0000:0212:7402:0002:0202 to ff02:0000:0000:0000:0000:0000:0000:001a
ICMPv6
Type: RPL Code: DIS

Payload (2 bytes)
0000                                          ..
```

Figure 6.14: DIS message sent by node 2 to nodes 8 and 14 for DIO solicitation

6.7 Embedding *SecTrust* in ContikiRPL

ContikiRPL is an implementation of the RPL protocol as specified in RFC 6550 [62]. It is implemented using two objective functions namely MRHOF and OF0. This research work specifies its own objective function and modifies the ContikiRPL using the *SecTrust* system. This creates a new RPL protocol based on the *SecTrust* framework with the implementation in ContikiRPL. This thesis considers this new protocol implementation as *SecTrust-RPL*.

SecTrust-RPL embeds the trust engine presented in Section 4.2 into the ContikiRPL so that routing decisions, malicious nodes detection and isolation are purely based on the trust among nodes. All nodes in the *SecTrust*-RPL independently process the trust relationship between them using *SecTrust* process engine so they could make their decisions about routing their data through their neighbour nodes in the network.

Although trust levels among nodes develop and evolve over time in *SecTrust*-RPL, a greater weight is placed on the current value (time-based) of the trust of a node. This ensures that a malicious node, although consistent in its good behaviour in the past, but decides to become malicious will quickly be detected and isolated from within the network.

6.7.1 Rank and Trust Computation in *SecTrust*-RPL

RPL maintains tree consistency through its Rank order. The Rank of a node is computed based on the Rank of its parent's Rank and the base Rank. When a node's base Rank is zero with no parent, the node is said to have an indeterminate Rank or infinite Rank (this node could possibly become the root node). However, if a node is linked to a parent with a base Rank of zero then the base Rank of the node becomes equal to the Rank of the parent's node.

If a node's base Rank is not zero and the node is linked to its parent, then the node's Rank is defined as:

node Rank = base Rank + min_hopRankinc (a minimum hop Rank increment defined in RPL specification of the parent in the DAG instance).

If a node's base Rank is not zero and the node has no defined parent, then the node's Rank is computed as:

node Rank = base Rank + RPL_INIT_LINK_METRIC (a RPL metric defined in the ContikiRPL).

The algorithm in Figure 6.15 (adapted from the IETF's RFC 6550 specification of RPL) embeds the Rank property of the RPL protocol as part of its metric to maintain consistency with the objectives of the RPL routing protocol. Figure 6.15 effectively summarises the description of the Rank computation described in this section.

```
Initialisations:
Initial node Rank assignment = BaseRank = BR
Rank of a node is NodeRank = NR
//Compute Rank of node
    if (node.parent == null && BR==0)
        NR => infinity
    else if (node.parent <> null && BR==0)
        NR=BR
    else if (node.parent <> null && BR>0)
        NR=BR+ min_hopRankinc
    else if (node.parent == null && BR>0)
        NR=BR+ RPL_INIT_LINK_METRIC
    else
        NR => infinity [inconsistency]
    end if
```

Figure 6.15: *SecTrust*-RPL adaptation of Rank computation as adapted from RFC6550

The trust mechanism computes the effective ICMPv6 and DIO packet exchange between nodes. In this thesis, the assumption is made that every attacking node starts out as well behaved, but over time begins its malicious activities. Furthermore, another assumption, which is also an assumption in ContikiRPL, is that every node operates in promiscuous mode hence, nodes overhear their neighbour packet transmissions. The number of packets a node can satisfactorily forward on behalf of the requesting node gives a true reflection of the reliable nature of any node. This concept of trust is embedded into ContikiRPL, so that the trust value of a node can be quantified and used in conjunction with the Rank specification for parent selection during routing.

When RPL is initially started, a comparison is made between nodes based on the expected transmission count and the Rank of the nodes. These are normal RPL operations to determine preferred parents and routing decisions. Trust values are computed using Equation given in 6.1. The computed trust values are sorted in descending order of trust magnitude. The corresponding trusted node(s) are selected for routing decisions while preserving the node Rank order in the network. From Equation 4.1 in Chapter 4, DT(N_i,N_j) represents the computed trust value of a node while $PF_{ji}(t)$, is the total number of ICMPv6 (DIO) packets forwarded by node N_j on behalf of node N_i at time *t*. $PS_{ij}(t)$ is the total number of ICMPv6 (DIO) packets sent to N_j by N_i at time *t*. As proposed in the design in Chapter 4, *t* is the time under which, a node (*j*) is being evaluated. β has an initial value of 1, which gives the penalty weight attached to any misbehaving node. The β value increases as a node's trust value traverses the trust threshold described in Table 4.2. This reduces the trust value of a misbehaving node hence, isolating it from being considered for routing decisions.

Figure 6.16a shows the algorithmic procedure implemented in *SecTrust*-RPL. It gives the computation of trust values, detection and isolation of malicious nodes during parent selection. The algorithm also ensures the Rank order of nodes are maintained as specified in [62]. Figure 6.17 below reiterates the flowchart of the *SecTrust* system implementation while Figure 6.18 gives the *SecTrust*-RPL implementation with the *SecTrust* engine coloured red in the flowchart. It can be observed from the flowchart (Figure 6.18) that all computations and decisions flow through the trust engine. The algorithms for the detection and isolation of Rank, Blackhole, Selective Forwarding and Sybil attacks are summarised in Figures 6.16b, 6.16c and 6.16d respectively.

Algorithm 1: Trust computation and trustworthy parent selection

```
      Let N1 ← one available item in the NeighbourList[ ]
      Let N2 ← another item next to N1 in the NeighbourList[ ]

Compute    $DT(N_i, N_j) = \frac{PF_{ji}(t)}{PF_{ji}(t)+\beta[PS_{ij}(t)-PF_{ji}(t)]}$

         If (N1.ETX<= ETX_Limit) & (N2.ETX<=ETX_Limit)
             If (N1.Rank <= Rank_Self) & (N2.Rank <+ Rank_Self)
                 Preferred_Parent = N1.DT($N_i$,$N_j$) > N2.DT($N_i$,$N_j$) ? N1 : N2;
             Else
                 If (N1.Rank <= Self_Rank) || (N2.Rank <= Self_Rank)
                     Preferred_Parent = N1.Rank < N2.Rank ? N1 : N2
                 Else
                       Preferred_Parent = NULL;
          Else
             If (N1.ETX <= ETX_Limit) || (N2.ETX <= ETX_Limit)
                Preferred_Parent = N1.ETX <= N2.ETX ? N1 : N2;
             Else
                 Preferred_Parent = NULL;
          Return Preferred_Parent
End program.
```

Figure 6.16a: *SecTrust*-RPL algorithm for trust computation, detection and isolation of malicious nodes during parent selection

Algorithm 2: Rank Attack Detection and Isolation

```
Begin
 Input: NeighbourNodeList[1..n]
 Input: Potential_Parents offers[1..j]
 Input: Preferred_Parent // derived from Algorithm 1
 From potential parents offer received [1..j]
 For i= 1,j
 begin
 //check for rank validity of potential parent offers against NeighbourNodeList[1..n]
   if (Potential_Parent(i).DIO_seq == NeighbourNodeList.DIO_seq
     if (Potential_Parent(i).rank == NeighbourNodeList.rank)
         return Preferred_Parent //Preferred parent is as in Algorithm 1
   else // Rank attack node, do not use it
     begin
       Insert node in MaliciousClassList do
       Preferred_Parent = NULL; //search for a new parent
     end; //of begin
   end if
 end;
 return Preferred_Parent
End; //of begin
```

Figure 6.16b: Algorithm for the detection and isolation of Rank attacks

Algorithm 3: Blackhole and Selective Forwarding Detection and Isolation

```
Begin //of program
 Input: Preferred_Parent // derived from Algorithm 1
 Input: Potential_Parents offers received [1..j]
 From potential parents offer received [1..j]
 For i= 1,j
 begin //check validity of potential parent against node's database
   if (potentialparent(i).DT(trustvalue) >= Threshold_Trust_value
      return Preferred_Parent //Preferred parent is as in Algorithm 1
   else //blackhole node or selective forwarding node, do not use it
     Preferred_Parent = NULL; //search for a new parent
   end if
 end; // of begin
 return Preferred_Parent;
End; //of program
```

Figure 6.16c: Algorithm for the detection and isolation of Blackhole and Selective Forwarding attacks

```
Algorithm 4: Sybil Detection and Isolation
Begin //of Sybil Detection
 Input: Preferred_Parent // derived from Algorithm 1

   Input: Potential_Parents offers received [1..j]
   For i= 1,j
   begin
   //check validity of potential parent against node's database
     if (potentialparent(i).DT(trustvalue) >= Threshold_Trust_value
       begin
         If (Potential_Parent(i).ipaddress == NodeList[1..j].ipaddress &&
            Potential_Parent(i).location(x,y) == NodeList[1.j].location(x,y))
              //Retain preferred parent as selected in trust parent algorithm
         else //node is sybil, isolate it
          Preferred_Parent = NULL;
         end if
       end; //of begin
     end if
   end; //of begin
  return Preferred_Parent
End;//of program
```

Figure 6.16d: Algorithm for the detection and isolation of Sybil attacks

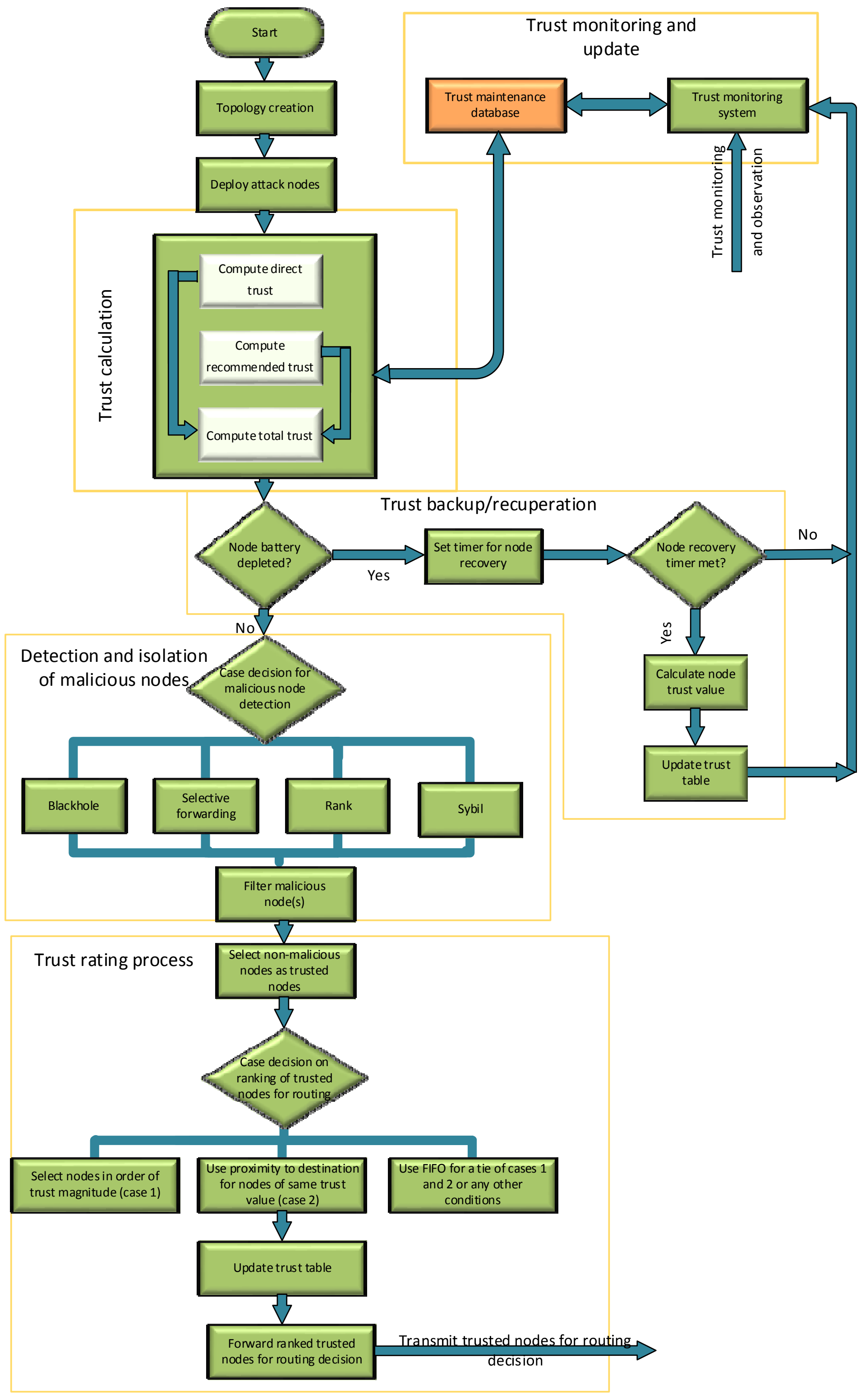

Start
Topology creation
Deploy attack nodes
Trust monitoring and update
Trust maintenance database
Trust monitoring system
Trust monitoring and observation
Trust calculation
Compute direct trust
Compute recommended trust
Compute total trust
Trust backup/recuperation
Node battery depleted?
Yes
Set timer for node recovery
Node recovery timer met?
No
Yes
Calculate node trust value
Update trust table
No
Detection and isolation of malicious nodes
Case decision for malicious node detection
Blackhole
Selective forwarding
Rank
Sybil
Filter malicious node(s)
Trust rating process
Select non-malicious nodes as trusted nodes
Case decision on ranking of trusted nodes for routing
Select nodes in order of trust magnitude (case 1)
Use proximity to destination for nodes of same trust value (case 2)
Use FIFO for a tie of cases 1 and 2 or any other conditions
Update trust table
Forward ranked trusted nodes for routing decision
Transmit trusted nodes for routing decision

Figure 6.17: ***SecTrust*** **engine showing all five processes for trust computation and processing**

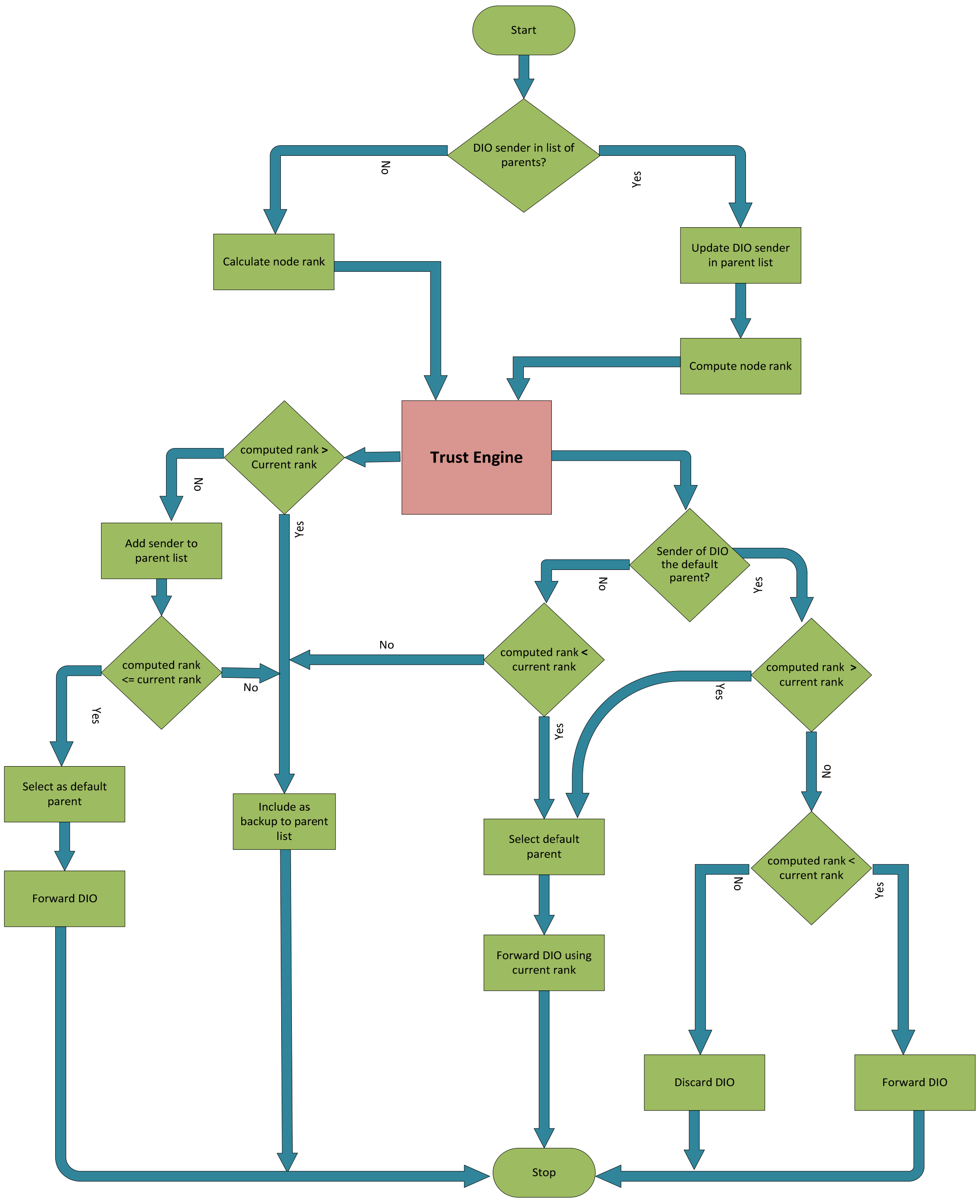

Start
DIO sender in list of parents?
No
Yes
Calculate node rank
Update DIO sender in parent list
Compute node rank
Trust Engine
computed rank > Current rank
No
Yes
Add sender to parent list
computed rank <= current rank
No
Yes
Select as default parent
Forward DIO
Include as backup to parent list
Sender of DIO the default parent?
No
Yes
computed rank < current rank
No
Yes
computed rank > current rank
Yes
No
Select default parent
Forward DIO using current rank
computed rank < current rank
No
Yes
Discard DIO
Forward DIO
Stop

Figure 6.18: *SecTrust*-RPL flowchart with the trust engine used for routing decisions and Rank consistency

6.8 Network Simulation setup

To replicate a real case scenario, this study emulates a smart IoT sensor deployment hence, the design and adoption of a smart building topology. The topology of a smart home was generated using 30 sky motes including 3 attack motes. It is assumed that at any point, a maximum of 10% of the total nodes could serve as attacking nodes hence, the use of 3 attacking sensor motes in the network design topology. This highlights a real case scenario for a typical network environment. In typical smart homes today, a network of 20 to 30 nodes are common although, according to predictions, this in the future could arguably increase to 250 nodes for any typical smart home [204]. Since the location size is predefined, the sensor devices are evenly distributed throughout the building while maintaining the root or sink node at the centre of the home. Figure 6.19 shows the topological view of the smart home with an even distribution of the sensor nodes with three attacker nodes outside the building. The sink node is shown in the orange colour and the sender nodes are designated in green while the attacker nodes are represented in red. The attacker nodes are shown to be planted outside the perimeter of the smart home to illustrate a real case scenario where a hacker could plant high-powered sensors to remotely gain access, and thereby infiltrate the internal network of the smart environment.

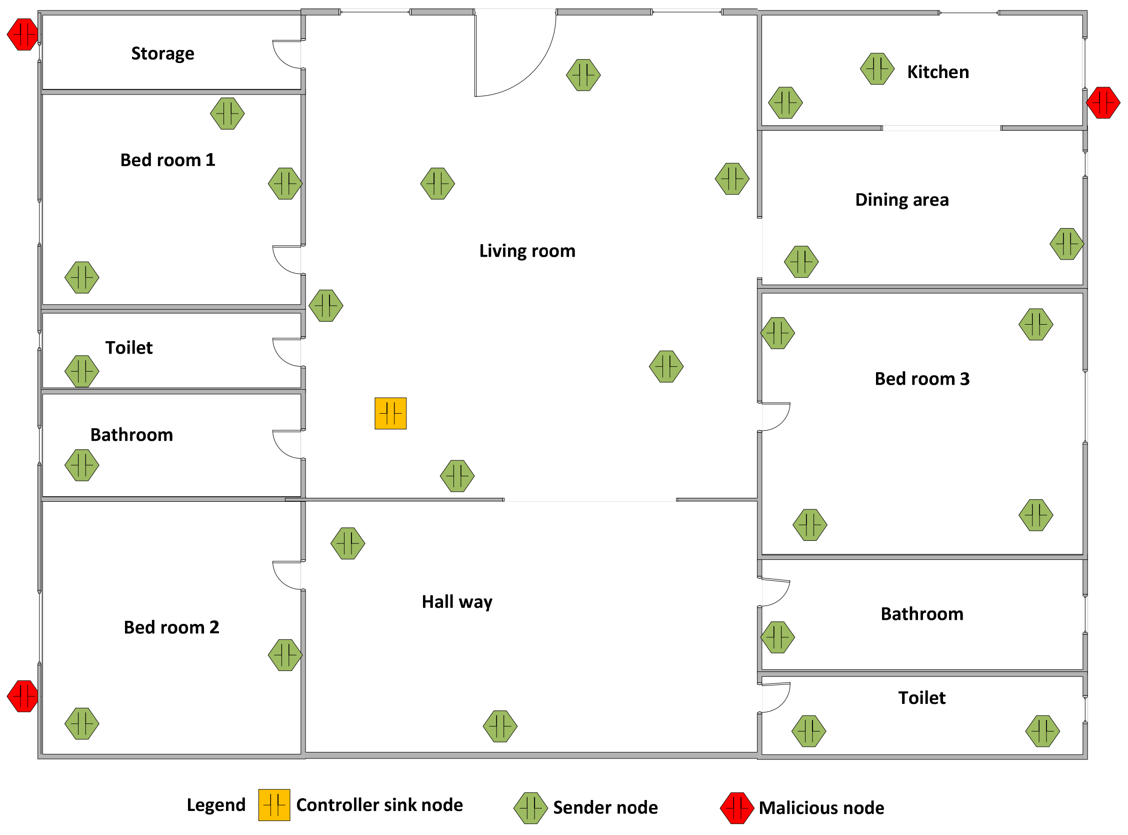


Figure 6.19: A topological view of a smart home with 27 nodes and 3 attacking nodes

A representation of the deployment of the 30 nodes in the Cooja simulator is shown in Figure 6.20. The sink node is shown in green while the sender nodes in orange, and the attacker nodes in red. The sink node is a combination and modified version of both the *udp-sink.c* and *udp-server.c* in Contiki/Cooja, which receives packets from all sender nodes. The sender node is also a combination and modified version of both the *udp-sender.c* and *udp-client.c*. These files were modified in order to measure effectively the details of control and route packets, received data packets, and dropped data packets during RPL operations. This process provides an effective measurement of network performances during network operations.

The mote output log file is created to measure the details of all routing communication during the simulation period. A C-script was developed to analyse and output the network performance. As presented in Table 6.1, the initial start delay of 5 seconds is given for the RPL network to converge. This is a sufficient time needed for the RPL network of 10 to 100 nodes to converge. This way, packets considered for evaluation are packets after network convergence. Table 6.1 provides other simulation set up details.

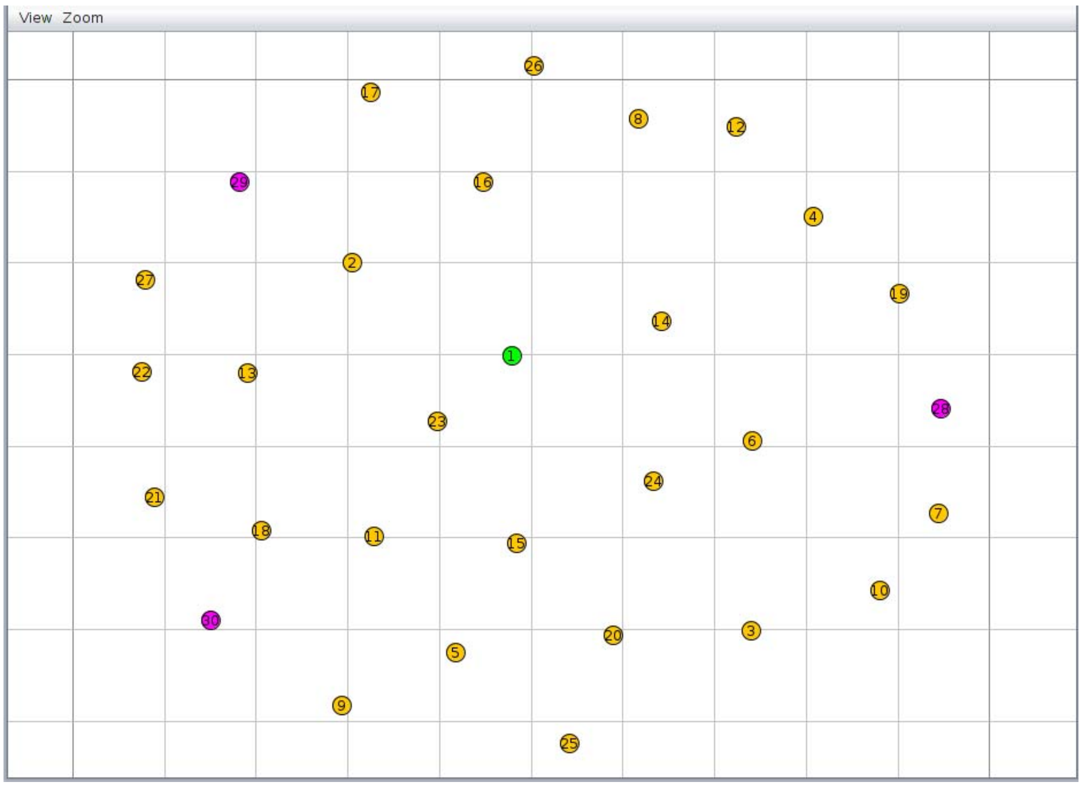


Figure 6.20: A RPL network of 30 nodes including 3 attacker nodes in Cooja Simulator

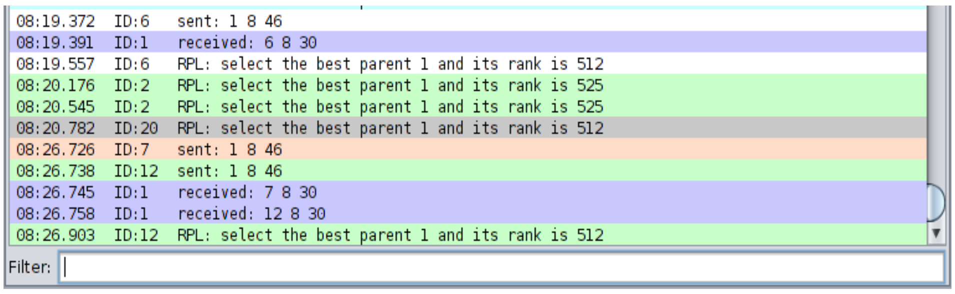
Table 6.1: Network simulation settings and parameters

Parameters	**Value**
Simulation tool	Contiki/Cooja 3.0
Simulation coverage area	70m x 70m
Total number of nodes	30
Malicious nodes	3 (Nodes 28, 29 and 30)
Malicious to legitimate node ratio	1 : 10
Node deployment environment	Smart building
RX Ratio	30-100%
TX Ratio	100%
TX Range	50m
Interference Range	50m
Routing Protocols	RPL and *SecTrust*-RPL
Network protocol	IP based
Start Delay	5 seconds
Simulation Time	60 minutes

The RPL operation mode was set to "No Downward" routes since in the smart home simulation the interest is on using multipoint to point traffic for the evaluation of the study (that is, sender nodes transmitting their packet to the sink node). DIO Min and DIO Doublings are maintained with the ContikiRPL default settings of 12 and 8 respectively. Since these nodes are lossy by nature, the reception ratio (RX) was set to a variable range of 30% to 100%. During the simulation study, the transmission ratio (TX) for all nodes was set to 100%, which shows a loss-free transmission as there was no need to introduce losses at the radio transmission end, but only at the reception end. However, for the attacker nodes their transmission and reception was set to 100%. The transmission range for all devices was set to 50m. This is the real coverage range of most sensor radios in the market today, while an interference range was set at 50m. Simulation was run for 60 minutes (simulation time) with 30 nodes including 3 attacker nodes. The results of the simulation findings are discussed next.

6.9 Simulation Results

The simulations performed in this study encompass deep analyses of the protocols' behaviour under study while the multiple simulation runs helped to confirm and verify results obtained under the given configuration settings. Running the Cooja simulations multiple times was indeed a mundane and time-consuming exercise. The data output log of simulation results gathered for just a single simulation run which spanned a 60-minute period was significantly large (spanning over 1200 pages of a word document) that a C-script was developed for data mining, analysis and presentation. Figure 6.21 shows a sample mote output log processed by the C-script for network performance measurement during RPL communication. The data log output where further tabled and presented in graphical format. See appendix 2 to access raw data generated during the simulation study.



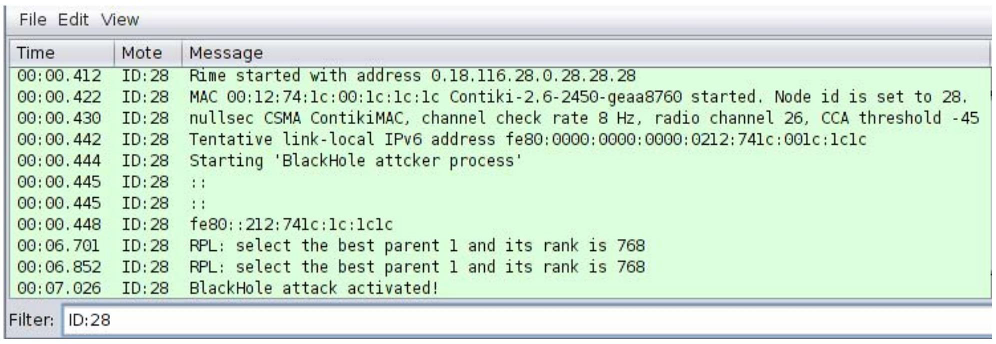
```
08:19.372  ID:6   sent: 1 8 46
08:19.391  ID:1   received: 6 8 30
08:19.557  ID:6   RPL: select the best parent 1 and its rank is 512
08:20.176  ID:2   RPL: select the best parent 1 and its rank is 525
08:20.545  ID:2   RPL: select the best parent 1 and its rank is 525
08:20.782  ID:20  RPL: select the best parent 1 and its rank is 512
08:26.726  ID:7   sent: 1 8 46
08:26.738  ID:12  sent: 1 8 46
08:26.745  ID:1   received: 7 8 30
08:26.758  ID:1   received: 12 8 30
08:26.903  ID:12  RPL: select the best parent 1 and its rank is 512
Filter:
```

Figure 6.21: A representation of smart building topology arrangement of 30 SKY mote sensors in Contiki/Cooja

The network shown in Figure 6.20 has 30 nodes and each node sends a 46-byte packet (30 bytes of data and 16 bytes of frame header detail) to the server (sink) periodically and after an initial start delay of 5 seconds (RPL convergence time). During RPL communication among nodes, sender nodes transmit packets to the sink node with the following stamp on each packet sent: time, source ID, packet type (sent or received), destination ID, sequence number and data size. This is shown in Figure 6.21. Packet sequence IDs are matched to ensure that packets sent are received by the sink node. Any sent packet sequence ID that is not matched by a corresponding sequence ID received by the sink node, has either been blackholed by the malicious node or affected by the lossy network link. However, lossy network links was eliminated because on nodes had strong reachability to each other, and furthermore, the packets dropped by the malicious nodes corresponded to the packets that did not reach the sink node. A complete log of sent and received packets were analysed and the results are discussed in the sections following.

6.9.1 *SecTrust*-RPL vs MRHOF- RPL under Blackhole Attacks

This section presents the simulation results of the RPL network using MRHOF objective function and the *SecTrust*-RPL under Blackhole attacks. During simulation start up, all nodes are considered well behaved, but after a certain time interval (6 - 10 seconds) they begin to display their malicious behaviour. This is when they are activated as shown in Figure 6.22. Figure 6.22 shows the activation of Blackhole attack on node 28 during RPL operation. Other attack nodes (29 and 30) were similarly activated.



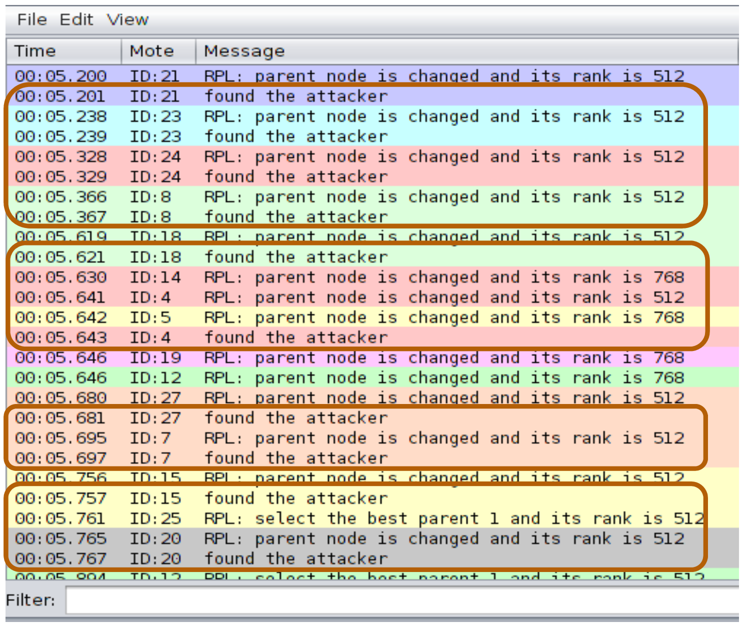
```
Time        Mote    Message
00:00.412   ID:28   Rime started with address 0.18.116.28.0.28.28.28
00:00.422   ID:28   MAC 00:12:74:1c:00:1c:1c:1c Contiki-2.6-2450-geaa8760 started. Node id is set to 28.
00:00.430   ID:28   nullsec CSMA ContikiMAC, channel check rate 8 Hz, radio channel 26, CCA threshold -45
00:00.442   ID:28   Tentative link-local IPv6 address fe80:0000:0000:0000:0212:741c:001c:1c1c
00:00.444   ID:28   Starting 'BlackHole attcker process'
00:00.445   ID:28   ::
00:00.445   ID:28   ::
00:00.448   ID:28   fe80::212:741c:1c:1c1c
00:06.701   ID:28   RPL: select the best parent 1 and its rank is 768
00:06.852   ID:28   RPL: select the best parent 1 and its rank is 768
00:07.026   ID:28   BlackHole attack activated!
Filter: ID:28
```

Figure 6.22: Blackhole attack activation on node 28 in a RPL simulation network

1. Detection and Isolation of Attack Nodes

In Figure 6.23, the *SecTrust*-RPL protocol could detect and isolate the Blackhole attacks during routing operations. A highlight of the attacks detected can be seen from the encircling brown pen-mark. In addition, Figure 6.24 presents a graph summary of attacks detected and isolated during RPL operation using *SecTrust*-RPL over a 60-minute simulation time. As many as 600 attacks were detected between the 40th and 45th minute of the RPL operation.

Conversely, in MRHOF's RPL implementation these attacks could not be detected, as there is no implemented mechanism to detect nor isolate Blackhole attacks. It is of note that in RPL routing, a node Rank change shows a re-alignment of a child-node to another preferred parent-node. Blackhole attack nodes advertise themselves to their neighbour nodes as having better routes in a guise to attract these unsuspecting nodes while eventually dropping their packets. In Figure 6.25, a comparison of the frequency of node Rank changes between the two routing systems is made. MRHOF-RPL showed high frequency in Rank changes reflecting its high level of vulnerability to Blackhole attacks while *SecTrust*-RPL protocol showed a very marginal level of vulnerability.



```
Time        Mote    Message
00:05.200   ID:21   RPL: parent node is changed and its rank is 512
00:05.201   ID:21   found the attacker
00:05.238   ID:23   RPL: parent node is changed and its rank is 512
00:05.239   ID:23   found the attacker
00:05.328   ID:24   RPL: parent node is changed and its rank is 512
00:05.329   ID:24   found the attacker
00:05.366   ID:8    RPL: parent node is changed and its rank is 512
00:05.367   ID:8    found the attacker
00:05.619   ID:18   RPL: parent node is changed and its rank is 512
00:05.621   ID:18   found the attacker
00:05.630   ID:14   RPL: parent node is changed and its rank is 768
00:05.641   ID:4    RPL: parent node is changed and its rank is 512
00:05.642   ID:5    RPL: parent node is changed and its rank is 768
00:05.643   ID:4    found the attacker
00:05.646   ID:19   RPL: parent node is changed and its rank is 768
00:05.646   ID:12   RPL: parent node is changed and its rank is 768
00:05.680   ID:27   RPL: parent node is changed and its rank is 512
00:05.681   ID:27   found the attacker
00:05.695   ID:7    RPL: parent node is changed and its rank is 512
00:05.697   ID:7    found the attacker
00:05.756   ID:15   RPL: parent node is changed and its rank is 512
00:05.757   ID:15   found the attacker
00:05.761   ID:25   RPL: select the best parent 1 and its rank is 512
00:05.765   ID:20   RPL: parent node is changed and its rank is 512
00:05.767   ID:20   found the attacker
00:05.804   ID:12   RPL: select the best parent 1 and its rank is 512
```

Figure 6.23: Detection of Blackhole attacking nodes during RPL operation

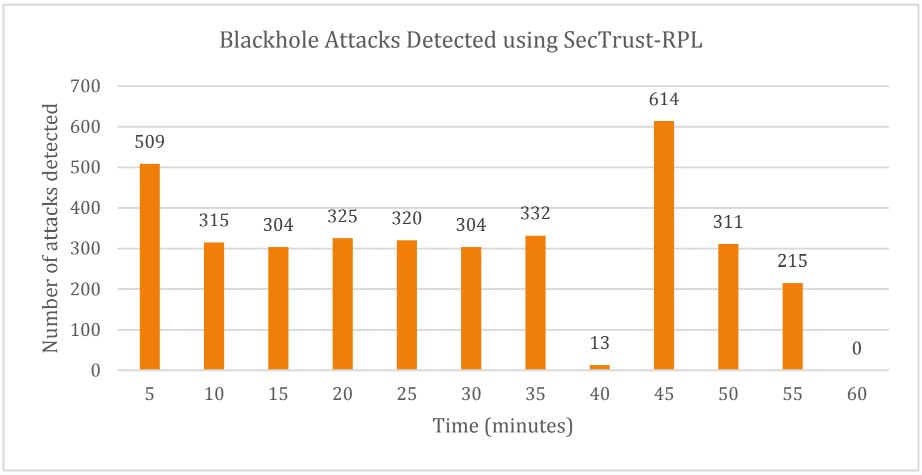


Figure 6.24: *SecTrust*-RPL detection and isolation of Blackhole attacks in RPL

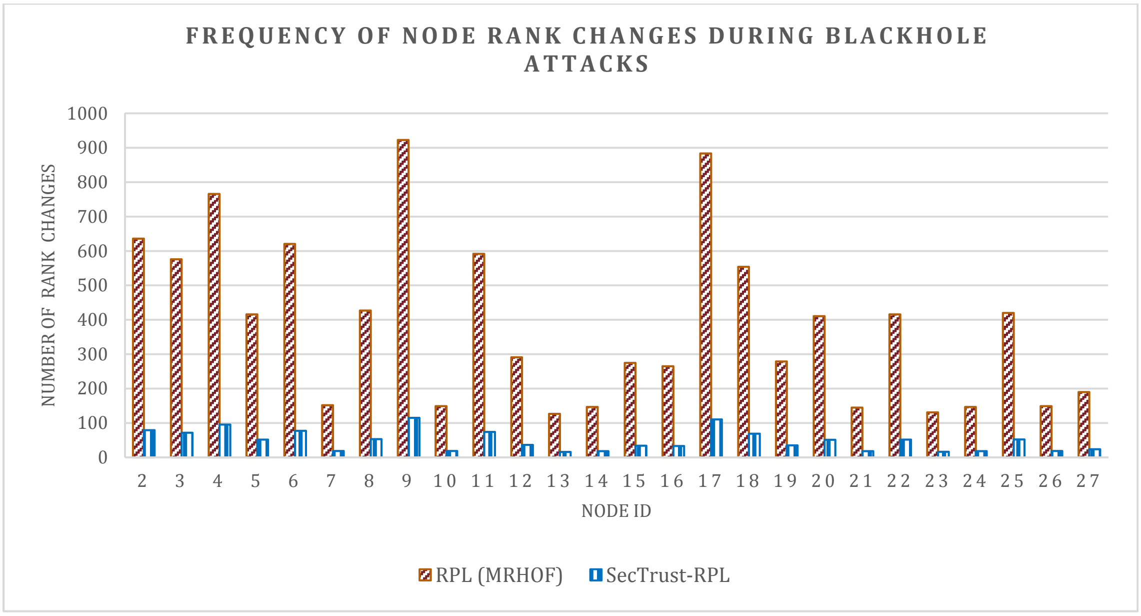


Figure 6.25: Node frequency Rank changes between MRHOF-RPL and *SecTrust*-RPL

2. Network Performance Measures

Although the new protocol could identify and isolate Blackhole attacks, it is imperative that the network performance measures including: network throughput, packet transmission delay and packet loss rate should be maintained or should not be impacted significantly. Therefore, a network performance measurement comparison between MRHOF-RPL and *SecTrust-RPL* under Blackhole attacks is presented.

In Figure 6.26, the *SecTrust*-RPL showed significant improvement in throughput over MRHOF-RPL. In fact, the throughput measurement of nodes 2 - 6, 8, 9, 15, 18, 19, 20, 22 and 25 was 0kbps under MRHOF-RPL as a result of the Blackhole attacks on the network. This is an indication that these nodes have selected a malicious node with a fake Rank as their preferred parent. This represents 50% of the total sending nodes in the network, which reflects a segmentation of the network in which 50% of the nodes are not able to send their packets to the sink node. The remaining nodes could send their packets to the sink. All nodes under *SecTrust*-RPL had throughput of 1.5kbps and higher and no node had a 0kbps throughput. Overall the throughput performance of *SecTrust*-RPL proved to be superior to MRHOF-RPL's.

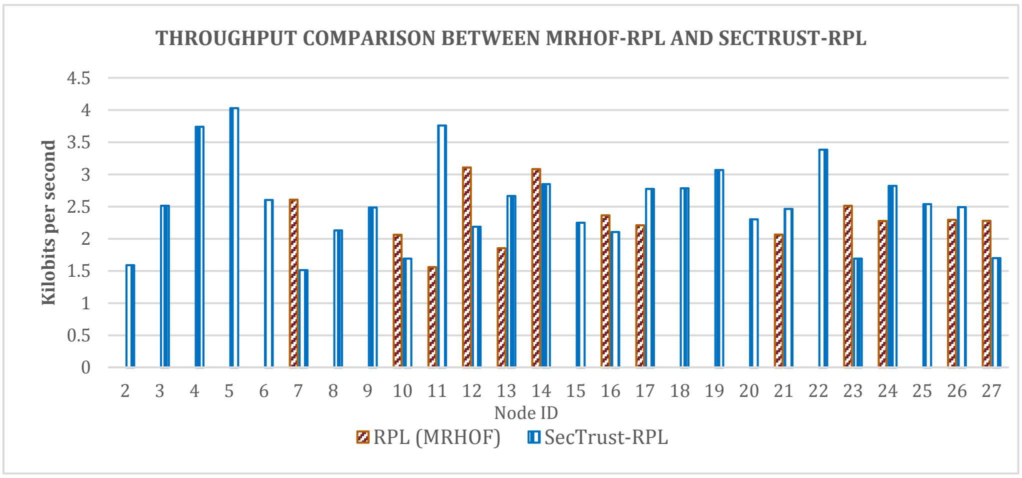


Figure 6.26: Throughput performance measurement between MRHOF-RPL and *SecTrust*-RPL

Figure 6.27 shows a graphical representation of the percentage of packet loss in RPL routing operation under Blackhole attacks. While the *SecTrust*-RPL protocol's packet loss stayed below 40%, MRHOF-RPL recorded a staggering 60 to 100% packet loss rate. Thus, the result presented in Figure 6.27 justifies *SecTrust*-RPL as a better performing protocol over MRHOF-RPL under Blackhole attacks.

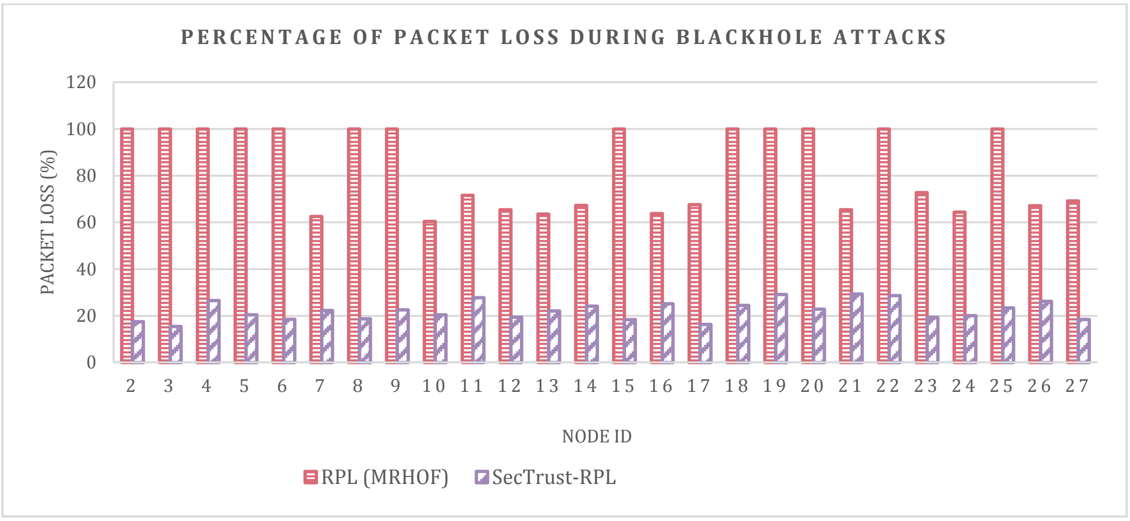


Figure 6.27: Packet loss rate comparison between RPL (MRHOF) and *SecTrust*-RPL

Figure 6.28 shows the packet transmission delay between MRHOF-RPL and *SecTrust*-RPL. In the transmission delay of MRHOF-RPL, the transmission delay of nodes 2-6, 8, 9, 15, 18 – 20, 22 and 25 was nil as there was no communication between these nodes and the sink node, which resulted in a transmission delay time of 0 milliseconds (ms).

The other nodes nevertheless, showed packet delay transmission time range of 70 to 150 milliseconds (ms). *SecTrust*-RPL however, displayed a packet transmission delay range of 30 to 155 ms. *SecTrust*-RPL had a better transmission delay range (35-155ms) when compared with MRHOF-RPL.

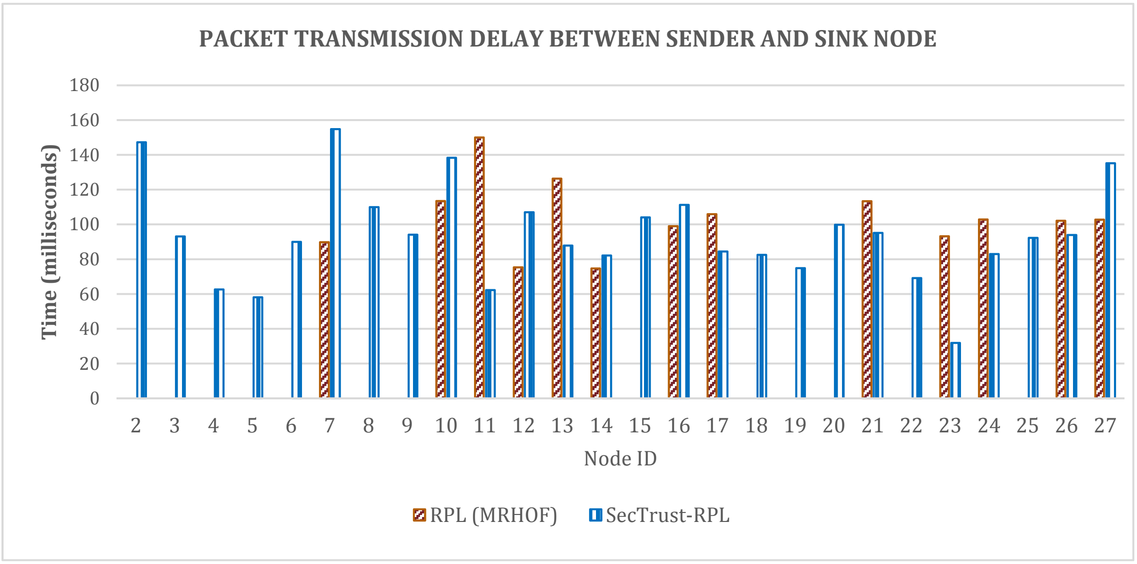
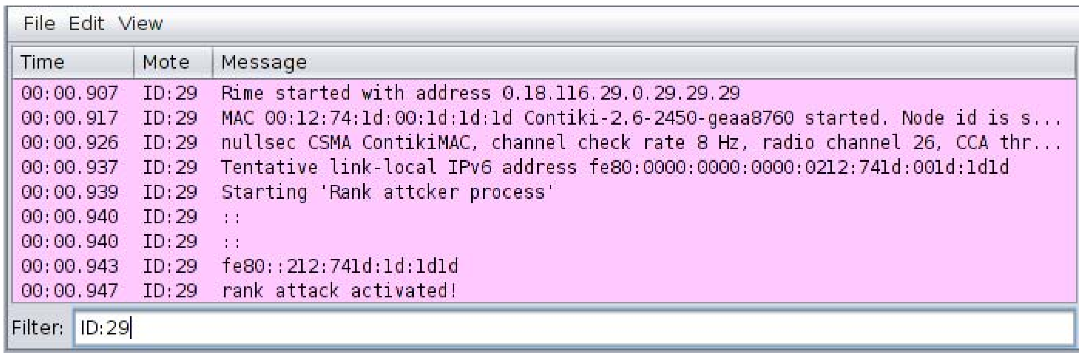


Figure 6.28: Comparison of packet transmission delay between RPL (MRHOF) and *SecTrust*

6.9.2 *SecTrust*-RPL vs MRHOF-RPL under Rank Attacks

This section presents the results of the simulation between MRHOF-RPL and *SecTrust*-RPL under Rank attacks. Figure 6.29 shows the activation of Rank attack on node 29 during RPL operation. Other attack nodes (28 and 30) were similarly activated.



```
Time       Mote   Message
00:00.907  ID:29  Rime started with address 0.18.116.29.0.29.29.29
00:00.917  ID:29  MAC 00:12:74:1d:00:1d:1d:1d Contiki-2.6-2450-geaa8760 started. Node id is s...
00:00.926  ID:29  nullsec CSMA ContikiMAC, channel check rate 8 Hz, radio channel 26, CCA thr...
00:00.937  ID:29  Tentative link-local IPv6 address fe80:0000:0000:0000:0212:741d:001d:1d1d
00:00.939  ID:29  Starting 'Rank attcker process'
00:00.940  ID:29  ::
00:00.940  ID:29  ::
00:00.943  ID:29  fe80::212:741d:1d:1d1d
00:00.947  ID:29  rank attack activated!
Filter: ID:29
```

Figure 6.29: Rank attack activation in a RPL simulation network

1. Detection and Isolation of Attack Nodes

From the result shown in Figure 6.30, *SecTrust*-RPL could detect and isolate the Rank attacks during routing operations. In the first 5 minutes 103 attacks were detected while the attacks decreased progressively. This is clearly understandable since RPL, being a proactive routing protocol, floods the network with control and route information before the network converges. This results in more control and route packets being transmitted, which malicious nodes could take advantage of to perpetrate their malicious behaviour. Conversely, MRHOF-RPL could not detect nor isolate the Rank attacks since it indeed has no mechanism to do so.

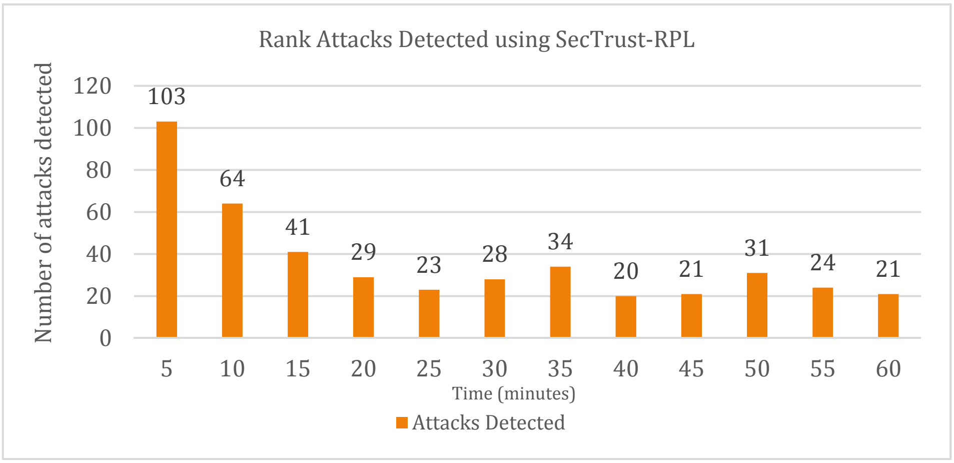


Figure 6.30: Rank attack activation in a RPL simulation network

During RPL routing, when a node's Rank changes, its child nodes re-aligns itself to another preferred parent node, which has a Rank value less than its own. This is necessary to maintain a loop-free topology. A Rank attack node advertise false optimal routes to their neighbour nodes. This causes the neighbouring nodes to detach themselves from its parents and select the malicious node(s) as their new preferred parent. This is in keeping with the RPL rule of having the Rank of nodes increase in a downward fashion to prevent routing loops. The result of this is a segmentation of the RPL network, which creates a disjointed network and isolate nodes from the sink. Figure 6.31 shows a comparison of the frequency of node Rank changes between the MRHOF-RPL and *SecTrust*-RPL. MRHOF-RPL showed significantly higher node Rank changes over *SecTrust*-RPL depicting a high level of vulnerability to Rank attacks. *SecTrust*-RPL consistently maintained low node Rank changes throughout the simulation period.

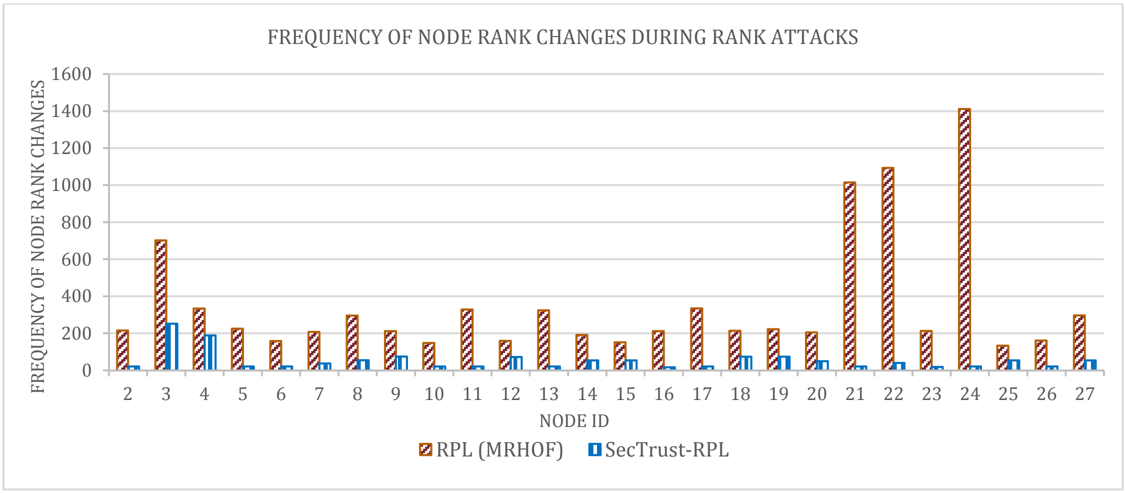


Figure 6.31: Comparison of frequency of node Rank changes between MRHOF-RPL and SecTrust-RPL during simulation

2. Network Performance Measures

The performance measurements of MRHOF-RPL and SecTrust-RPL are presented below. In Figure 6.32, the *SecTrust*-RPL showed significant improvement in throughput over the MRHOF-RPL maintaining a throughput range of 3.5 – 5.5 kbps with MRHOF-RPL having a throughput performance of 0.0 – 3.0 kbps. In fact, the throughput recorded of nodes 3,4,8,11,13, 21, 27 under MRHOF-RPL was 0kbps. This is an indication that packets sent by these nodes where not delivered to the sink node since the network had become segmented due to Rank attacks perpetrated by the malicious nodes in the network. As a result of the network segmentation, 30% of the nodes were not able to communicate with the sink node. In SecTrust nevertheless, throughput was 100% as all nodes communicated with the sink, and Rank attacks were isolate from untrusted nodes. In all, the throughput performance of *SecTrust*-RPL proved to be much better than MRHOF-RPL as depicted in Figure 6.32.

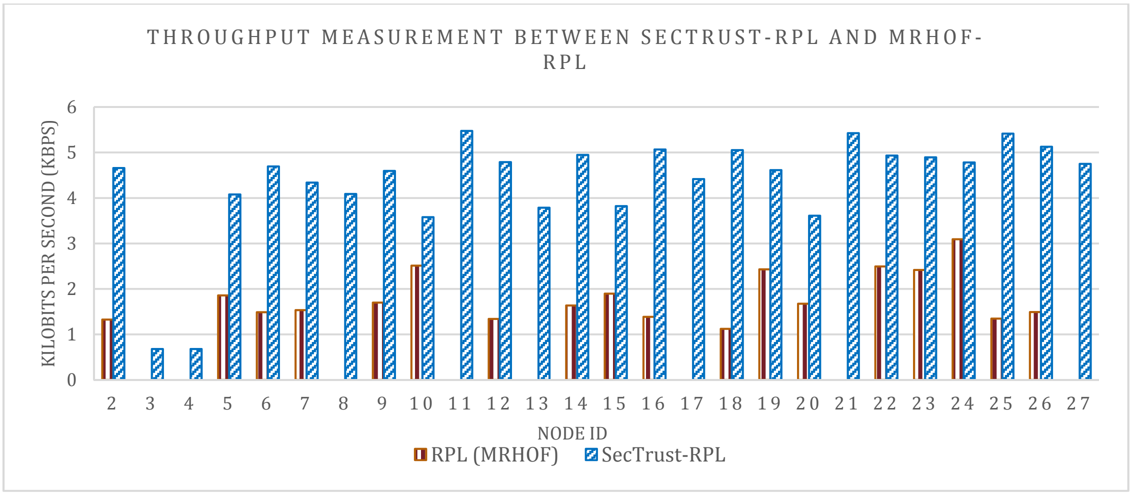


Figure 6.32: Network throughput performance comparison between *SecTrust*-RPL and MRHOF-RPL during Rank attacks

Figure 6.33 shows a graphical representation of the packet loss percentage during RPL routing operation under Rank attacks. While *SecTrust*-RPL protocol's packet loss rate averaged 22 – 23%, the standard MRHOF-RPL recorded a high packet loss rate range of 60 to 100%. As a result of the network segmentation caused by the Rank attacking nodes, MRHOF-RPL network experienced high packet loss rates. Conversely, the *SecTrust*-RPL protocol shows better promising defence against Rank attacks with less than 30% packet loss rates.

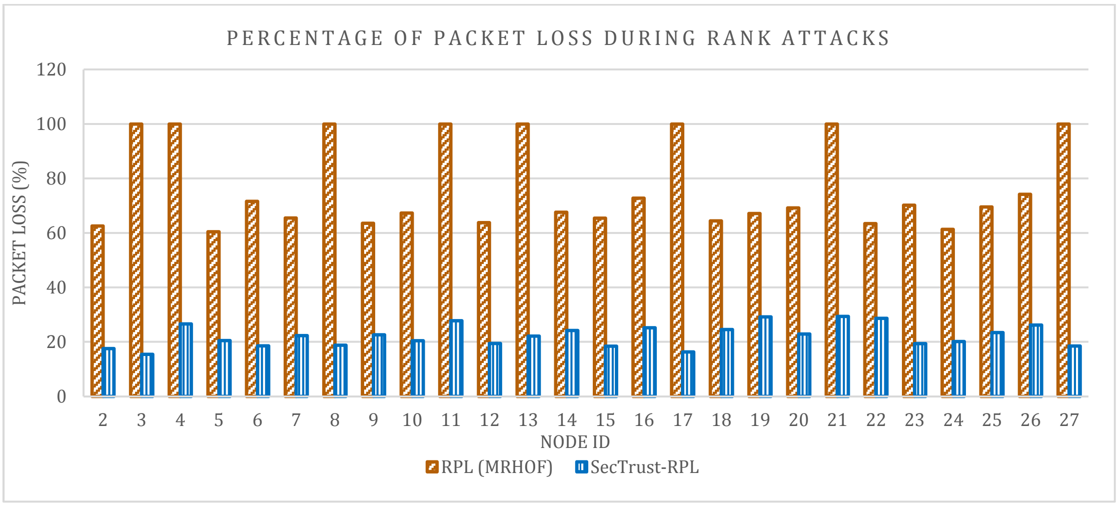


Figure 6.33: Packet loss percentage between *SecTrust*-RPL and MRHOF-RPL during Rank attacks

Figure 6.34 shows the packet transmission delay between MRHOF-RPL and *SecTrust*-RPL. In the transmission delay of MRHOF-RPL, the transmission delay observed in nodes 3, 4, 8, 15, 21, 11, 13, 21 and 27 was nil as there was no communication between these nodes and the sink node, which resulted in a transmission delay time of 0 milliseconds (ms). The other nodes nevertheless, showed packet delay transmission time range of 100 to 200 milliseconds (ms). *SecTrust*-RPL consistently maintained a low and stable packet transmission delay of 50 - 60 ms.

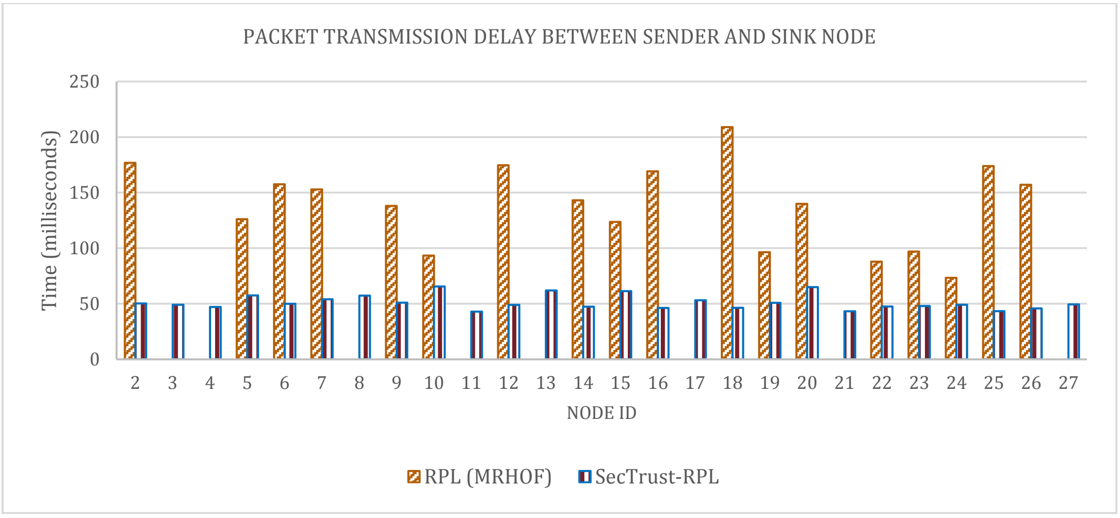
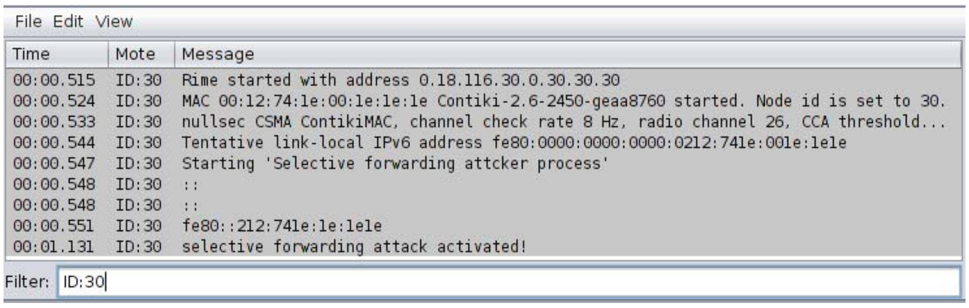


Figure 6.34: Packet transmission delay of *SecTrust*-RPL and MRHOF-RPL under Rank attacks

6.9.3 *SecTrust*-RPL vs RPL (MRHOF) Under Selective Forwarding Attacks

The results and discussion of the simulation study of MRHOF-RPL and *SecTrust*-RPL under Selective Forwarding attacks is presented in this section. Figure 6.35 shows node 30 being activated for Selective Forwarding attacks during RPL simulation. Other attack nodes (28 and 29) were similarly activated.



```
Time       Mote   Message
00:00.515  ID:30  Rime started with address 0.18.116.30.0.30.30.30
00:00.524  ID:30  MAC 00:12:74:1e:00:1e:1e:1e Contiki-2.6-2450-geaa8760 started. Node id is set to 30.
00:00.533  ID:30  nullsec CSMA ContikiMAC, channel check rate 8 Hz, radio channel 26, CCA threshold...
00:00.544  ID:30  Tentative link-local IPv6 address fe80:0000:0000:0000:0212:741e:001e:1e1e
00:00.547  ID:30  Starting 'Selective forwarding attcker process'
00:00.548  ID:30  ::
00:00.548  ID:30  ::
00:00.551  ID:30  fe80::212:741e:1e:1e1e
00:01.131  ID:30  selective forwarding attack activated!
Filter: ID:30
```

Figure 6.35: Selective Forwarding attacks activation in a RPL simulation network

1. Detection and Isolation of Attack Nodes

Selective Forwarding attack is a subtle variation of Blackhole attack where malicious nodes selectively drop packets during routing communication. The Selective Forwarding attack nodes were programmed to swarm the RPL network, and selectively drop packets during the initial period of the RPL operation. After the initial period of this attack, the nodes allow for network convergence, but continue to advertise spurious routes, and randomly drop route packets.

From the result shown in Figure 6.36, *SecTrust*-RPL could detect and isolate Selective Forwarding attacks during routing operations. The first 20 minutes of RPL operation reveals the detection of a swarm of Selective Forwarding attacks. In addition, after the first 20 minutes, attacks became minimal, but random which were also isolated by SecTrust-RPL.

On the other hand, MRHOF-RPL was not able to identify any of the Selective Forwarding attacks been perpetrated in the RPL network, and this is shown by the high frequency of node Rank changes as shown in Figure 6.37. A value much higher than that exhibited by *SecTrust*-RPL.

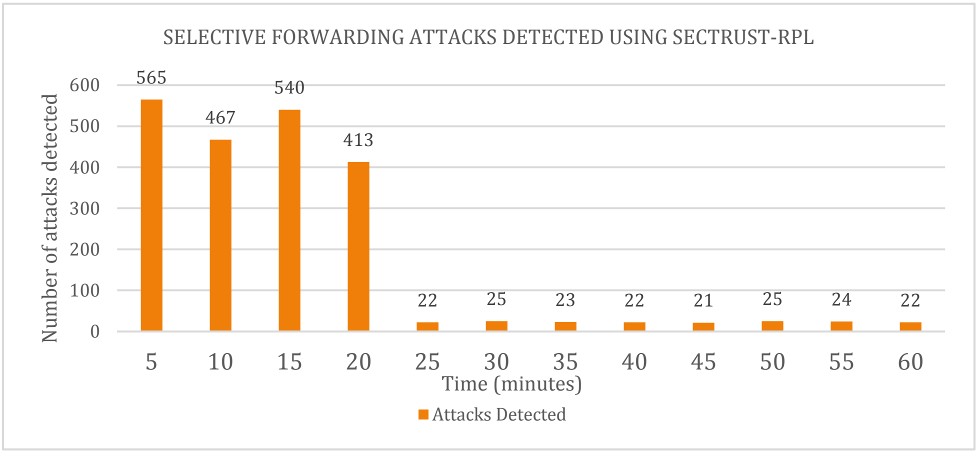


Figure 6.36: Selective Forwarding attacks detection and isolation a RPL simulation network

A RPL network with a stable topology will send route and control information based on the DIO trickle timer, and the timer value increases with a stable network. However, a RPL network environment with high network topology changes will cause frequent transmission of control and route information. The topology changes could be due to mobility of nodes or suspicious activities of some malicious nodes in the network. This thus, creates the necessity for node re-alignment with new parents hence, the reason for high frequency of node Rank changes. Since the nodes are stationary, it follows that the node Rank changes are purely due to the activities of the malicious nodes in the RPL network. Figure 6.37 below shows a comparison of the frequency of node Rank changes between the MRHOF-RPL and *SecTrust*-RPL. MRHOF-RPL showed significantly higher node Rank changes over *SecTrust*-RPL. This shows a high level of vulnerability to Selective Forwarding attacks. It can be observed from the Figure 6.37 that MRHOF-RPL had significant number of node rank changes in comparison to *SecTrust*-RPL, a reflection of a high network topology destabilization. As noted earlier that the high frequency of node Rank changes causes the RPL network to become destabilized and hence, affects both the efficiency and performance of the RPL network. *SecTrust*-RPL maintained a fairly consistent node Rank change value throughout the simulation period.

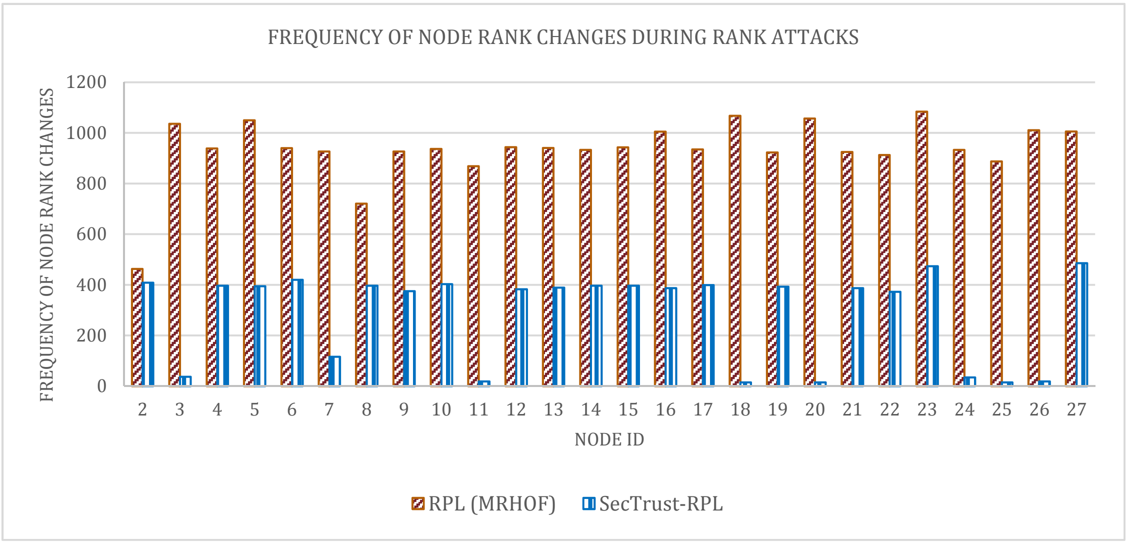


Figure 6.37: Comparison of frequency of node Rank changes during Selective Forwarding attacks in RPL network simulation

2. Network Performance Measures

From Figure 6.38 below, nodes 3, 16, 18, 20, 23, 26 and 27 under MRHOF-RPL recorded 0kbps throughput, an indication that these nodes aligned to malicious parents. In fact, the following are the number of packets sent by each of these nodes, which never got delivered to the sink node. Node 3 (packet sent, 52), Node 16 (packet sent, 52), Node 18 (packet sent, 52), Node 20 (packet sent, 52), Node 23 (packet sent, 52), Node 26 (packet sent, 52) and Node 27 (packet sent, 52). The remaining nodes had varying delivery packet rates that still showed they were also affected by the activities of the Selective Forwarding attacking nodes. *SecTrust*-RPL however, showed significant improvement in throughput over MRHOF-RPL, maintaining a much higher throughput range. To sum up, in the context of Figure 6.38, *SecTrust*-RPL displays superior performance over MRHOF-RPL.

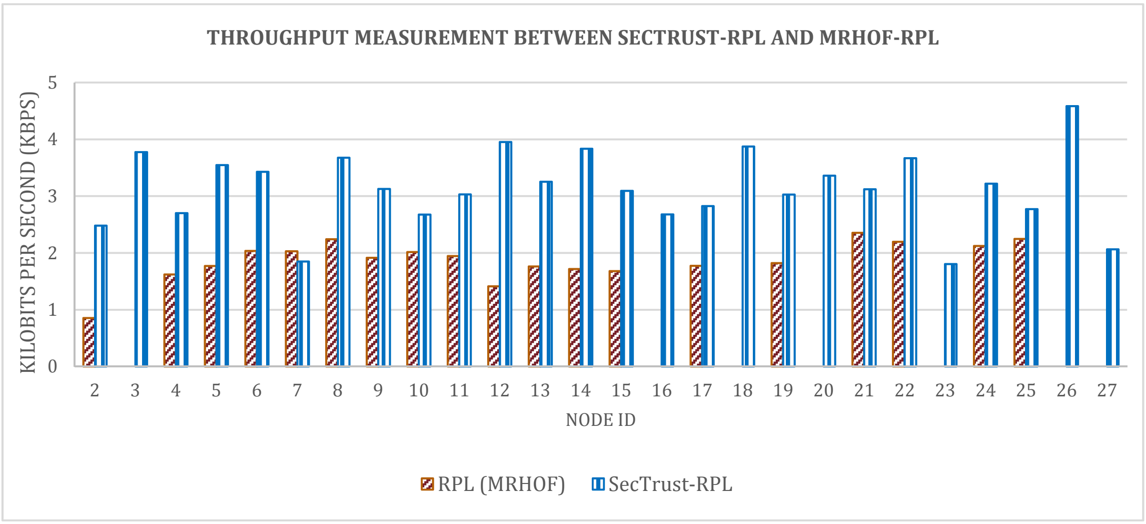


Figure 6.38: Network throughput performance comparison between *SecTrust*-RPL and MRHOF-RPL during Selective Forwarding attacks

Further to the result presented in Figure 6.38, Figure 6.39 below shows the packet percentage losses of each node under the two protocol schemes. It is quite clear that MRHOF-RPL had a higher percentage of lost packets during RPL operation as it had 60 – 100% packet loss rate while Selective Forwarding attack was active. *SecTrust*-RPL on the other hand, had a modest loss rate of less than 30% across all nodes. This proves the efficacy of *SecTrust* in delivering an acceptable network performance while isolating malicious nodes in the network.

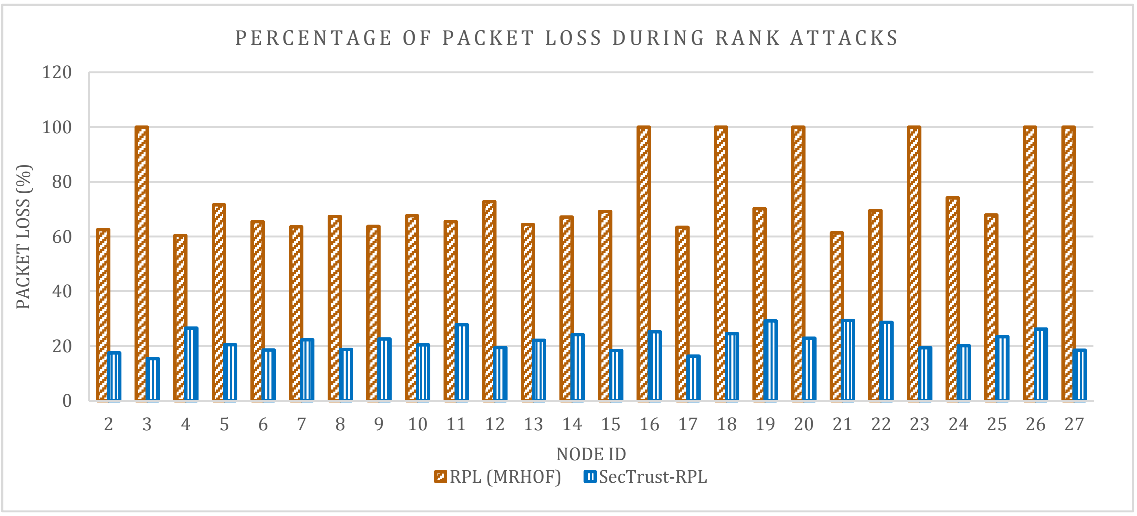


Figure 6.39: Percentage of packet loss comparison between *SecTrust*-RPL and MRHOF-RPL during Selective Forwarding attacks

Figure 6.40 shows the packet transmission delay between MRHOF-RPL and *SecTrust*-RPL. Under MRHOF-RPL, the transmission delay observed in nodes 3, 16, 18, 20, 23, 26 and 27 was nil as there was no communication between these nodes and the sink node which resulted in a transmission delay time of 0ms. The other nodes nevertheless, showed packet delay transmission time range of 34 to 260 ms. *SecTrust*-RPL maintained relative constancy in its node rank changes hence, the transmission delay was fairly constant and stable. The transmission delay range in *SecTrust*-RPL was between 51 to 129 ms.

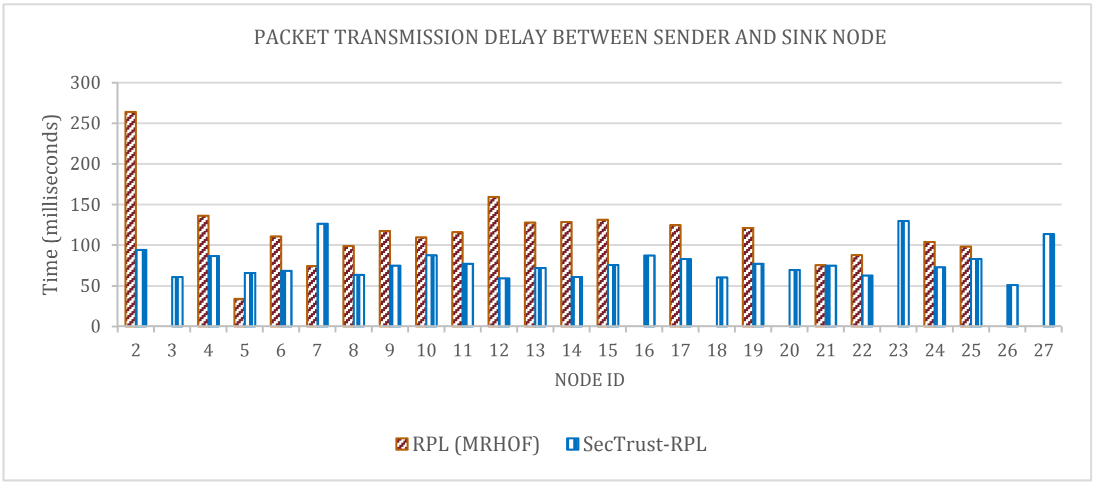
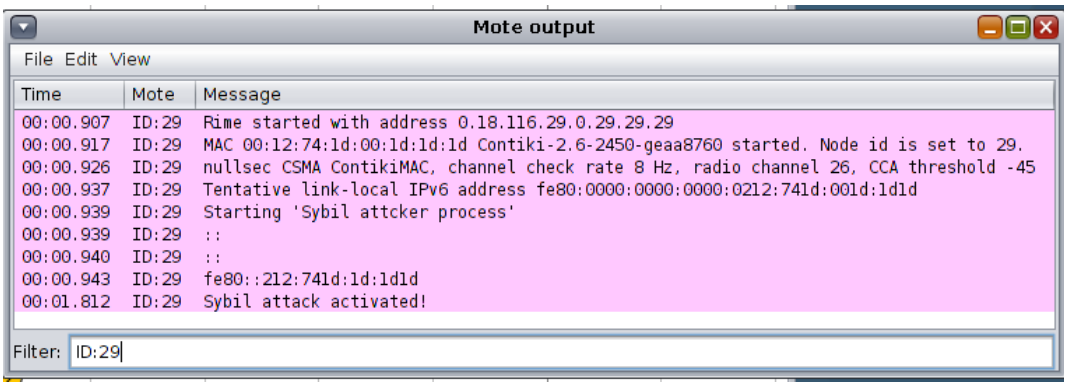


Figure 6.40: Packet transmission delay of *SecTrust*-RPL and MRHOF-RPL under Selective Forwarding attacks

6.9.4 *SecTrust*-RPL vs RPL (MRHOF) Under Sybil Attacks

This section presents the results of the simulation between MRHOF-RPL and *SecTrust*-RPL under Sybil attacks. Figure 6.41 shows the activation of attack node 29 during RPL operation. Other attack nodes (28 and 30) were similarly activated.



```
Time       Mote   Message
00:00.907  ID:29  Rime started with address 0.18.116.29.0.29.29.29
00:00.917  ID:29  MAC 00:12:74:1d:00:1d:1d:1d Contiki-2.6-2450-geaa8760 started. Node id is set to 29.
00:00.926  ID:29  nullsec CSMA ContikiMAC, channel check rate 8 Hz, radio channel 26, CCA threshold -45
00:00.937  ID:29  Tentative link-local IPv6 address fe80:0000:0000:0000:0212:741d:001d:1d1d
00:00.939  ID:29  Starting 'Sybil attcker process'
00:00.939  ID:29  ::
00:00.940  ID:29  ::
00:00.943  ID:29  fe80::212:741d:1d:1d1d
00:01.812  ID:29  Sybil attack activated!
```

Figure 6.41: Sybil attacks activation in a RPL simulation network

1. Detection and Isolation of Attack Nodes

ASybil attack behaves as though it is part of a larger network by assuming the identities of other nodes or by perpetrating multiple identities with the intention of creating a view of large number of nodes to its victims. It recommends other fake identity nodes linked to itself as trustworthy, or behaves as a new node that has just joined the network. In the RPL implementation of *SecTrust*, recommendation was not used as part of the trust computation mechanism. This is because in a RPL network, a child-node only knows its parent, but is unaware of its grandparent. Furthermore, the non-utilization of Recommendation attribute also helps in isolating sybil-like attackers, that may wish to exploit the Recommendation attribute of SecTrust's system. This way a Sybil operation under *SecTrust*-RPL becomes extremely difficult. In addition, node location was used since all nodes were stationary.

From the result shown in Figure 6.42, *SecTrust*-RPL could detect and isolate the Sybil attacks during routing operations. In the first 5 minutes 338 attacks were detected. Although attacks decreased, they were however, detected and isolated.

Conversely, MRHOF-RPL could not detect nor isolate any Sybil attacks since it indeed has no mechanism for performing this.

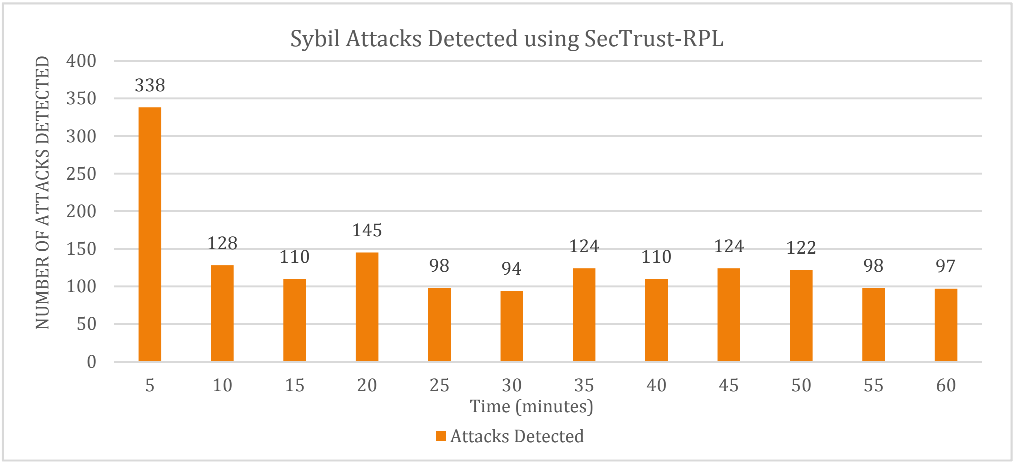


Figure 6.42: Sybil attacks detection and isolation in a RPL simulation network

Figure 6.43 shows a comparison of the frequency of node Rank changes between the MRHOF-RPL and *SecTrust*-RPL. From the observed Figure, MRHOF-RPL showed higher node Rank changes over *SecTrust*-RPL; an indication of a high vulnerability to Sybil attacks. *SecTrust*-RPL maintained better node rank consistency although, nodes 3, 19, 23 showed slightly higher node rank changes of 213, 178, 182 in comparison to their peers in *SecTrust*-RPL. However, the recorded values are still much lower than values observed under MRHOF-RPL.

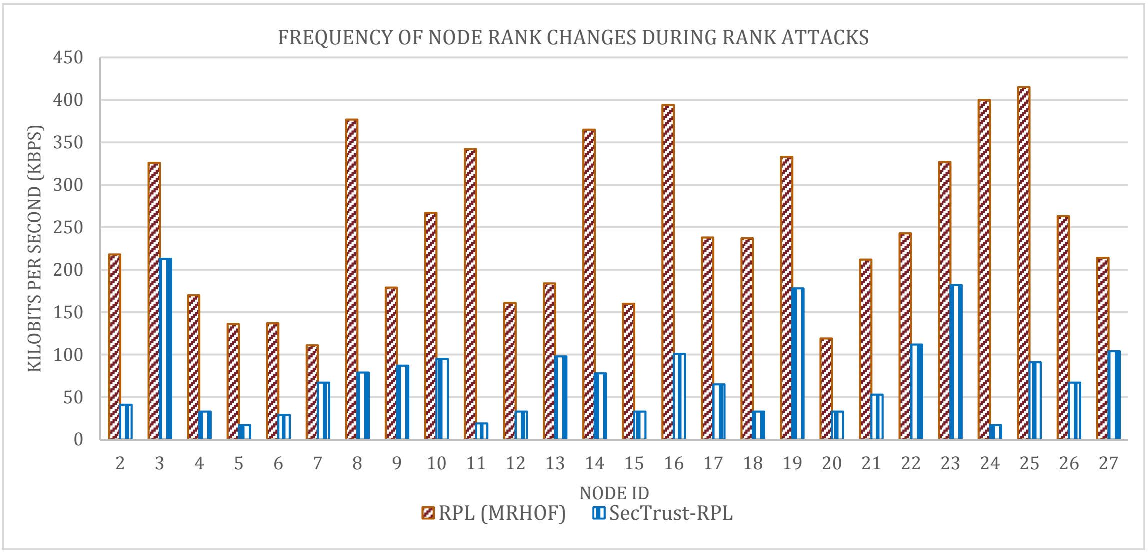


Figure 6.43: Comparison of frequency of node Rank changes during Sybil attacks in RPL network simulation

2. Network Performance Measures

From Figure 6.44, the throughput of nodes 3, 8, 10, 11, 16, 17, 19, 23 and 25 was 0 kbps under MRHOF-RPL hence, no throughput values are indicated in the graph. This reflects an attribute of a Sybil attack, which is the resource starvation of victim nodes [214] whereas *SecTrust*-RPL had no node with 0 kbps. The other nodes under MRHOF-RPL had a throughput range of 0.4 to 3kbps. Nodes under *SecTrust*-RPL displayed superior performance with throughput ranging from 1.4 to 3.9 kbps.

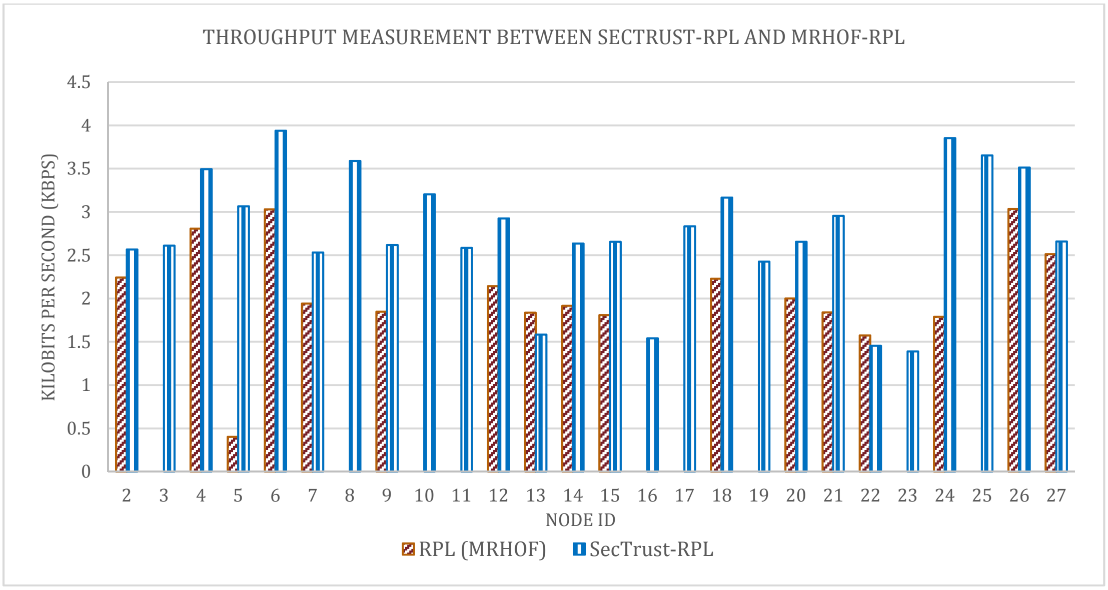


Figure 6.44: Network throughput performance comparison between *SecTrust*-RPL and MRHOF-RPL during Sybil attacks

The packet loss rate in MRHOF-RPL under Sybil attacks was significantly higher than in *SecTrust*-RPL. This is shown in Figure 6.45 where MRHOF-RPL maintained a steady rate of 60-100% packet loss rate while *SecTrust*-RPL ranged between 15 – 25%. Nodes 3, 8, 10, 11, 16, 17, 19, 23 and 25 under MRHOF-RPL had loss rates of 100%. Packet loss was higher in MRHOF-RPL due to resource starvation a consequence of successful Sybil attacks. *SecTrust*-RPL could provide good defence and an acceptable network performance while under the Sybil attack.

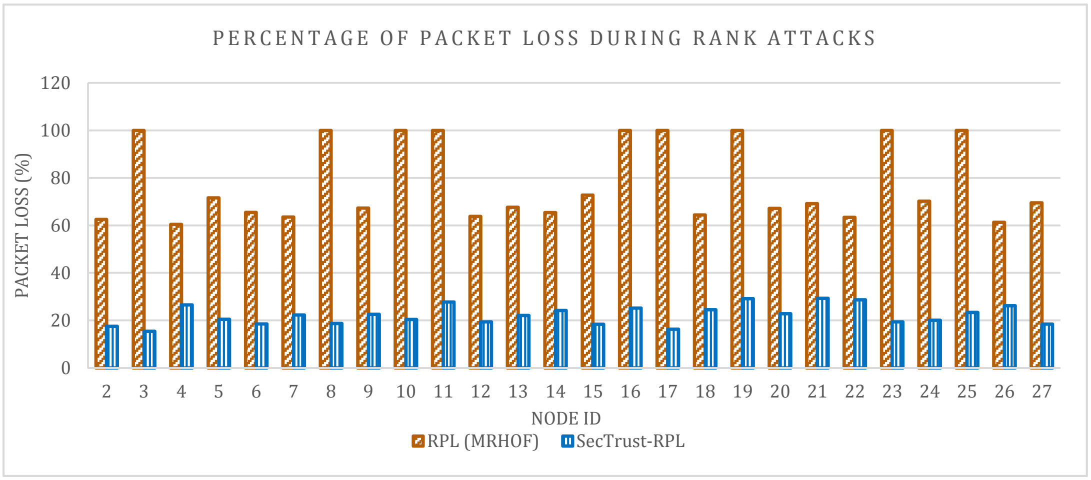


Figure 6.45: Percentage of packet loss comparison between *SecTrust*-RPL and MRHOF-RPL during Sybil attacks

Figure 6.46 shows the packet transmission delay between MRHOF-RPL and *SecTrust*-RPL. Under MRHOF-RPL, the transmission delay observed in 3, 8, 10, 11, 16, 17, 19, 23 and 25 was nil as there was no communication between these nodes and the sink node, which resulted in a transmission delay time of 0 milliseconds (ms). Node 5 had transmission delay of 580ms while the other nodes maintained a transmission delay time range of 17 – 129ms. *SecTrust*-RPL however, showed a wide range of transmission delay time (59 – 600ms). Particularly, nodes 3, 4, 7, 8, 13, 14, 16, 19, 22, 23 and 27 displayed high delay times. This reflects that Sybil attacks take a heavy toll on the network performance of *SecTrust*-RPL while identifying and isolating Sybil attacks.

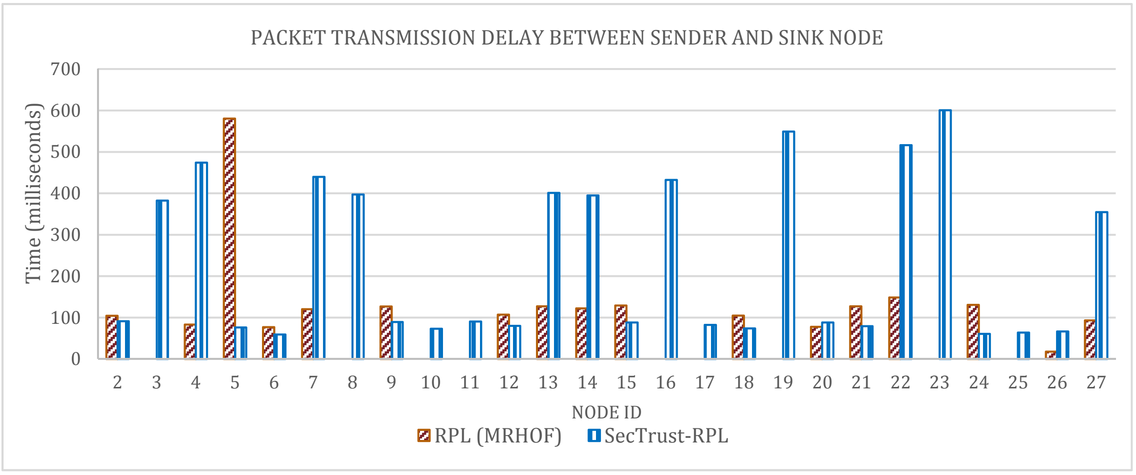


Figure 6.46: Packet transmission delay of *SecTrust*-RPL and MRHOF-RPL under Selective Forwarding attacks

6.10 Summary

SecTrust-RPL is a RPL protocol based on the *SecTrust* system (Chapter 4) that uses trust for the evaluation of nodes to make optimal routing decisions while isolating malicious nodes in the network. It computes its trust by examining the successful packet exchange between nodes to determine their reliability in forwarding packets to other nodes within the RPL network. It has a malicious node detection and isolation procedure that detects and isolates suspicious nodes. Other routines have also been embedded to ensure the proper functioning of *SecTrust*-RPL. The precision of the trust evaluation system is predicated on the characteristic features of the attacks discussed in this thesis. *SecTrust* has also been shown in Chapter 5 to scale well for large network sizes.

Furthermore, a simulation study was carried out to show the efficacy of the *SecTrust* system by embedding it into the RPL protocol (*SecTrust*-RPL) and presenting simulation results to show its efficacy in defending against various attacks as discussed in this

Chapter. Although this was only a simulation study, Chapter 7 provides a testbed validation of the *SecTrust* system. This involves the deployment of IoT sensor motes, and the collection of relevant control and route information under similar attacks. The results gathered during the testbed experiments are compared with the simulation study results for possible confirmation or variation. This helps to verify the validity and efficacy of the *SecTrust* system, and to reveal any discrepancies inherent in the trust system given a real-world environment.

7 *SecTrust*: Testbed Experiments

Chapter 6 described the simulation study of *SecTrust*-RPL, which is the embedding of *SecTrust* in ContikiRPL (*SecTrust*-RPL). The study provided a benchmark for the testing of the *SecTrust* system with the aid of a simulation platform (Contiki/Cooja). The simulation results gathered showed the efficacy of the *SecTrust* system in providing good defence against various IoT routing attacks.

Simulation in networks provide an effective way of developing scenarios, network topologies and modification of various attributes providing easy data collection and analysis. The use of simulation in modelling applications scenarios of a real world has often proved to be limited and challenging, although it has also shown to be steps in the right direction in the evaluation of systems for deployment and practical usage. In view of the above, simulation studies, have been perceived as having inadequacies in some real-world modelling cases and hence, are thought of as not being adequate to provide reliable results especially when compared with testbeds. A potential area of contention in simulation study in sensor networks is in the network layer where network routes in simulation studies are considered less reliable when compared with routes in testbeds. Therefore, testbeds become useful environments for the experimentation and validation of novel technologies under real world settings before proposed solutions can be exposed to the market place.

To validate the test results of the simulation study therefore, this Chapter explores the practical deployment of a small-scale testbed of *SecTrust*-RPL using XM1000 motes. Among other specific investigations this testbed undertakes, it investigates the effectiveness of *SecTrust*-RPL in a real-world scenario under acceptable network performance levels.

Further to the use of DSRM, the *rigor* and *relevance* testing of the *SecTrust* system is achieved through a testbed, and Figure 7.1 highlights the design science testing and validation stage of *SecTrust* with the aid of the proposed testbed.

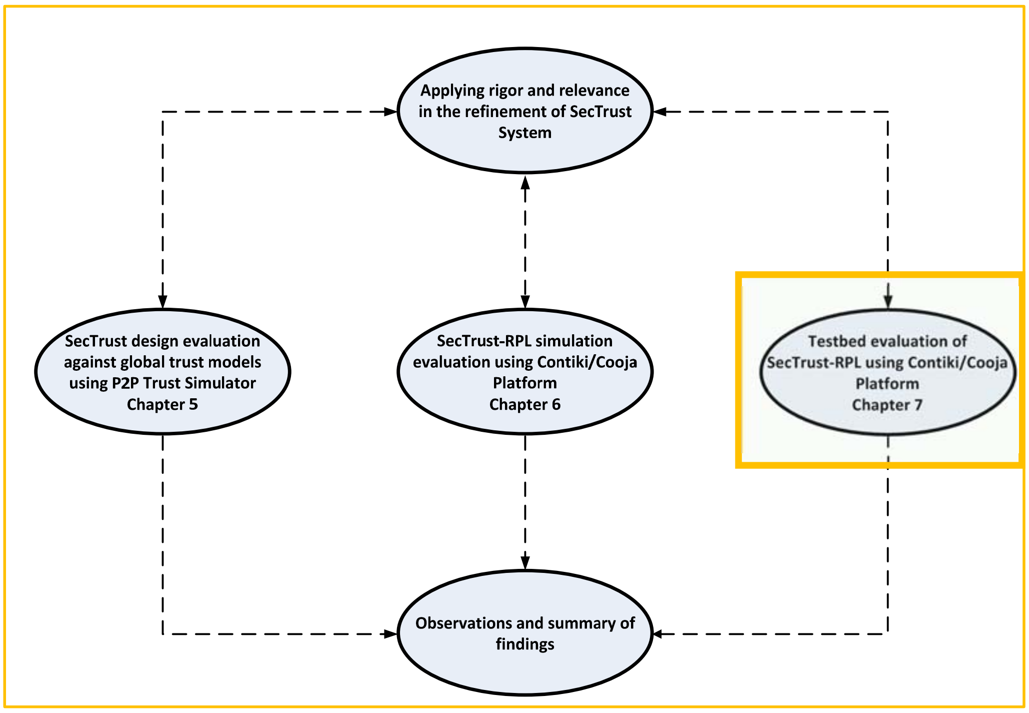

Applying rigor and relevance in the refinement of SecTrust System
SecTrust design evaluation against global trust models using P2P Trust Simulator Chapter 5
SecTrust-RPL simulation evaluation using Contiki/Cooja Platform Chapter 6
Testbed evaluation of SecTrust-RPL using Contiki/Cooja Platform Chapter 7
Observations and summary of findings

Figure 7.1: Testbed testing and validation stage of *SecTrust*

7.1 Advantages of Testbed Experimentation

Simulation has over the years proved to be a fundamental testing and diagnostic tool. This is especially true for network simulations, and this has become a global standard for wireless sensor network testing and evaluation. Many sophisticated simulators have been developed with advanced features like speedy deployment, systematic execution, variable and logic breakpoints, dynamic variable settings and the inspection of variable run-time. In addition, users can easily study deployed applications on networks with large scale of simulated sensor nodes having configurations which are much complex than what a real world scenario may provide [202]. However, as sophisticated as a simulation may be, it is often inadequate as an investigation platform for real-world deployments. Sensors, and indeed IoT sensors, have continued to improve in technological advancements; their form factors have become, and are continuing, to be smaller by the day while their processing capabilities have soared in recent times and some of the measuring devices on these sensors have not been adequately and accurately simulated using simulators. Specifically, present network simulators fall short on adequately simulating the intrinsic complexity of low power wireless links, environmental impacts (physical barriers, erratic environmental temperatures), and physical and MAC layer dynamics (radio signal interactions, irregular body object interruption and general sensor interrupt behaviours). Moreover, in simulation, unrealistic assumptions can be made, which can cause simulated models to depart from the boundary of realism. Although with a strict modeller policy of formulation and development this can be minimized [202].

A Testbed on the other hand, is a collection of deployed hardware infrastructure developed for physical network experimentation and integrated with software services for controlling and managing hardware and experiments executed on it. Therefore, a physical testbed experimentation becomes imperative as it is designed to support physical experimentation, which addresses the gaps that simulations are not able to fill. A fundamental feature of a testbed is its focus on a specific aspect of the total system. Testbeds provide practical real world hardware-software platforms for a complete functional test while not having a complete system. This helps in furthering a deeper understanding of the functional and operational requirements of the system and capturing specific behaviours under unique conditions, which otherwise would not have been captured during a simulation. Results gathered during testbed runs can be quantitatively measured and analysed from which, design decisions can be predicated on the theoretical and empirical findings.

In summary, in contrast with simulation studies, testbeds offer live experimental platforms where modelled real-world systems can be tested, studied and appraised from diverse experimental perceptions. Although some perceptive experimental views in a testbed can be limited due to cost. This becomes a limiting factor for testbeds as well.

7.2 Case Study of a Smart Building

Smart buildings today are not just containers of life but are progressively evolving to having lives of their own. Given the quantum of data accessible from a smart building such as the occupant information, humidity of the internal building, temperatures of the various compartments of the building, airflow rates and the amount of energy consumed in the building. Accessing, managing and adapting these data to suite the unique requirements of the occupants of the buildings gives the perception that these buildings are approaching sentience due to technological advancements. Today, the global market industry is investing in new and innovative ways in which, technological IoT solutions can be incorporated into buildings thereby, transforming these buildings into data-rich environments that exemplify the technology basis of IoT and cloud computing.

A key motivation for undertaking the smart building testbed scenario is the study, which shows that most IoT-based smart building/home systems give reliable data results under simulation testing scenarios, but provide unreliable performances when deployed in a real world scenario, where normal day-to-day activities are carried out [215]. Furthermore, this research testbed adopts the smart building case study because of the immense IoT work in both research and industry to make buildings smarter and have a life of their own.

Although in Chapter 6, a smart home case study (3-bed room smart home) was used however, for the testbed experiment, the Auckland University of Technology's (AUT) research lab was used since it is basically the same smart environment with normal activities within the building. Figure 7.2 gives the smart building layout of the Auckland University of Technology's PhD Research lab (WT building, room 404) where the XM1000 mote sensors where deployed for the testbed research study.

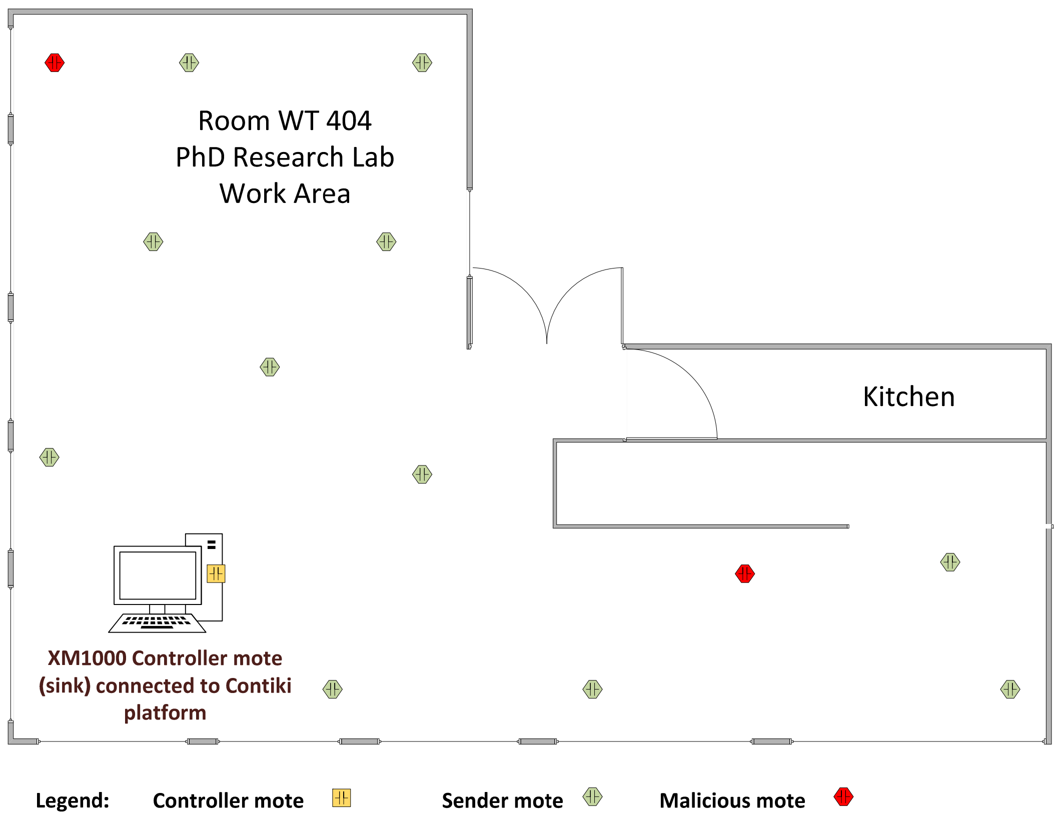


Figure 7.2: A topology view of 14 XM1000 motes with the sink mote attached to a desktop PC, and 2 deployed attacking motes

7.3 IoT Hardware and Software Platforms

The hardware of choice for this testbed study is the AS-XM1000 mote marketed by Advanticsys in Madrid, Spain. It is a commercially available mote, which was purchased for the purpose of this study. The XM1000 is a new generation of mote module based on the “TelosB” platform (same as the Tmote SKY used in Contiki) and developed by the University of California at Berkeley. The XM1000 mote (Figure 7.3) is based on the CC2420 single-chip radio with 2.4 GHz IEEE 802.15.4 compliant RF transceiver designed for low-power wireless applications using low voltages. It supports open-source operating systems and applications such as TinyOS and Contiki, and it has been deployed in several academic and scientific studies [216, 217]. The design of the XM1000 is built on the standard IEEE 802.15.4 using 2.4 GHz transceiver. It also comes equipped with an enhanced 116Kb-EEPROM and 8Kb-RAM while being embedded with humidity, temperature and light sensors. Table 7.1 below summarises the features of the AS-XM1000 mote [209]. This study deployed 14 XM1000 motes on a floor of the School of Engineering, Computer and Mathematical Sciences’ research laboratory.

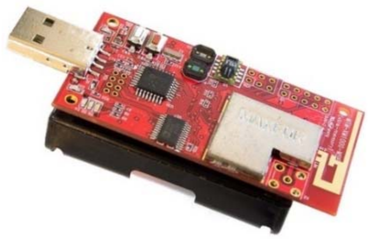


Figure 7.3: The AS-XM1000 mote [209]

Table 7.1: Summary of the features of the AS-XM1000 mote [209]

Processor		
Processor Model	TI MSP430F2618	Texas Instruments MSP430 family
Memory	116KB 8KB 1MB	Program flash Data RAM External Flash (ST® M25P80)
ADC	12bit resolution	8 channels
Interfaces	UART, SPI, I2C USB	Serial Interfaces External System Interface (FTI® FT232BM)
Radio		
RF Chip	TI CC2420	IEEE 802.15.4 2.4GHz Wireless Module
Frequency Band	2.4GHz ~ 2.485GHz	IEEE 802.15.4 compliant
Sensitivity	-95dBm typ	Receive Sensitivity
Transfer Rate	250Kbps	IEEE 802.15.4 compliant
RF Power	-25dBm ~ 0dBm	Software Configurable
Range	~120m(outdoor), 20~30m(indoor)	Longer ranges possible with optional SMA antenna attached
Current Draw	RX: 18.8mA TX: 17.4mA Sleep mode: 1uA	Lower RF Power Modes reduce consumption
RF Power Supply	2.1V ~ 3.6V	CC2420 Input Power
Antenna	Dipole Antenna / PCB Antenna	Additional SMA connector available for extra antenna
Sensors		
Light 1	Hamamatsu® S1087	Visible Range (560 nm peak sensitivity wavelength)
Light 2	Hamamatsu® S1087-01	Visible & Infrared Range (960 nm peak sensitivity wavelength)
Temperature & Humidity	Sensirion® SHT11	Temperature Range: -40 ~ 123.8 ºC Temperature Resolution: : ± 0.01(typical) Temperature Accuracy: ± 0.4 ºC (typical) Humidity Range: 0 ~ 100% RH Humidity Resolution: 0.05 (typical) Humidity Accuracy: ± 3 %RH (typical)
Electromechanical Characteristics		
Dimensions	81.90mm x 32.50mm x 6.55mm	Including USB connector
Weight	17.7g	Without batteries
Power	3V (2xAA Battery Holder Provided)	MICREL® MIC5207 Power Regulator

To maintain a common platform for the comparison and evaluation of both the simulation and testbed studies, the Contiki platform was used for the testbed experimentation, since ContikiRPL can be embedded into the XM1000 mote. Contiki uses the IPv6 protocol stack for communications among nodes in a smart network. Contiki is particularly suited for constrained devices like XM1000 because Contiki is an event-driven system and not a multithreaded emulator which requires a stack kept in memory for every thread. This feature makes Contiki run efficiently on emulated platforms like the Tmote SKY. However, a programmable multi-threaded abstraction known as *protothreads* is available in Contiki for coding using the multi-threading approach. A summary of the Contiki architecture is given in Figure 7.4.

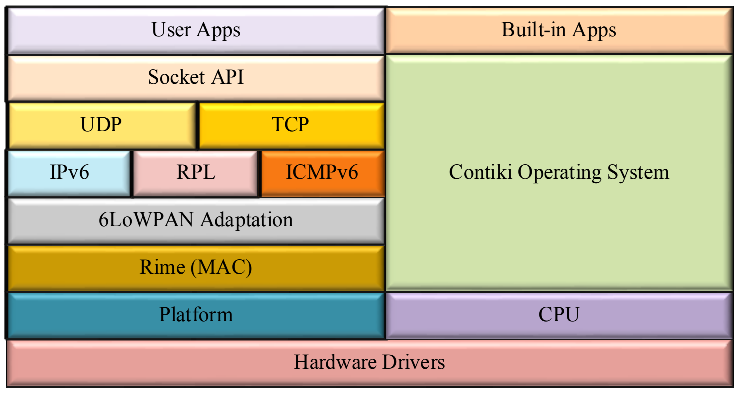


Figure 7.4: An architecture of the Contiki OS [204]

7.4 Contiki and Testbed Deployment

As discussed in Chapter 6, part of this research study focuses on a comparison of the *SecTrust*-RPL with the MRHOF Contiki implementation of RPL. Both protocols using Contiki were deployed and tested using the XM1000 motes. Routing communication among motes were logged using the different mote types presented. A brief of the mote types deployed is given below.

1. The UDP Sink

This is the main mote receiving control and routing communication details from other sending nodes. Every mote in the RPL network will seek to establish a path to the sink mote, which can be processed or forwarded to some other network or to the internet. In this thesis, the UDP sink code was modified to allow for the traffic measurement of route information and data packets received from all sending nodes. The modified code was embedded into one of the XM1000 motes, which was used as the sink mote.

2. The UDP Sender

This mote sends communication details like route path and information to the sink or server mote. The multi-hopping and multi-routing nature of the two protocols were observed and their effect on packet drop rate, delay time and throughput were studied. This experiment modified the UDP sender codes, so it could measure the packet drop rate, delay time and throughput rate because of the action of the malicious motes in the network. The modified code was embedded into 11 of the XM1000 motes which served as sender-motes.

3. Malicious Motes

A modified version of the UDP sink mote was used as malicious motes. This mote was coded with all the attributes of the malicious attacks proposed in this thesis. Two of the XM1000 motes were used for perpetration of attacks (motes 13 and 14).

7.5 Testbed Setup

To setup and deploy the testbed, three mote types were deployed namely: the UDP sink, the UDP sender and the attacking malicious motes. Motes deployed were stationary and could communicate with the UDP sink since all nodes were under the coverage area of the UDP sink mote. In all, one (1) sink mote (node), eleven (11) sender motes and two (2) attacker motes were deployed using the XM1000 motes. The sink mote was connected to the desktop PC running the Contiki/Cooja emulation program while the sender and malicious motes were evenly distributed across the testbed location.

7.5.1 Embedding ContikiRPL (MRHOF) and *SecTrust*-RPL in AS-XM1000

To complete the testbed setup, the respective UDP_sink, UDP_sender and malicious codes were embedded into the XM1000 motes using ContikiRPL (MRHOF) and *SecTrust*-RPL. Descriptions of the embedding process is given in the section following.

1. ContikiRPL (MRHOF) in AS-XM1000

To setup the testbed for ContikiRPL (MRHOF), one XM1000 mote was embedded with the UDP_sink code as explained in section 7.4, eleven XM1000 motes were embedded with the UDP_sender code while two XM1000 motes were embedded with the malicious codes of the Blackhole attack using the host desktop PC running the Contiki platform. In addition, the RPL protocol was set to MRHOF objective function.

The testbed was run under the Blackhole attacks and the results collected. After a successful run of the Blackhole attacks under the MRHOF-RPL protocol, the Blackhole attack code was erased from the two XM1000 motes, and it was then embedded with the Rank attack malicious codes. This testbed setup was also executed under the Rank attacks operation and the results were also observed. Furthermore, the process above was repeated for Selective Forwarding attacks and Sybil attacks respectively.

2. *SecTrust*-RPL in AS-XM1000

To setup the testbed for *SecTrust*-RPL, one XM1000 mote was embedded with the UDP_sink code as explained in section 7.4, eleven XM1000 motes were embedded with the UDP_sender code while two XM1000 motes were embedded with the malicious codes of the Blackhole attack using the host desktop PC running the Contiki platform. The protocol detail used was the *SecTrust*-RPL protocol developed, which is the trust-based system embedded into ContikiRPL.

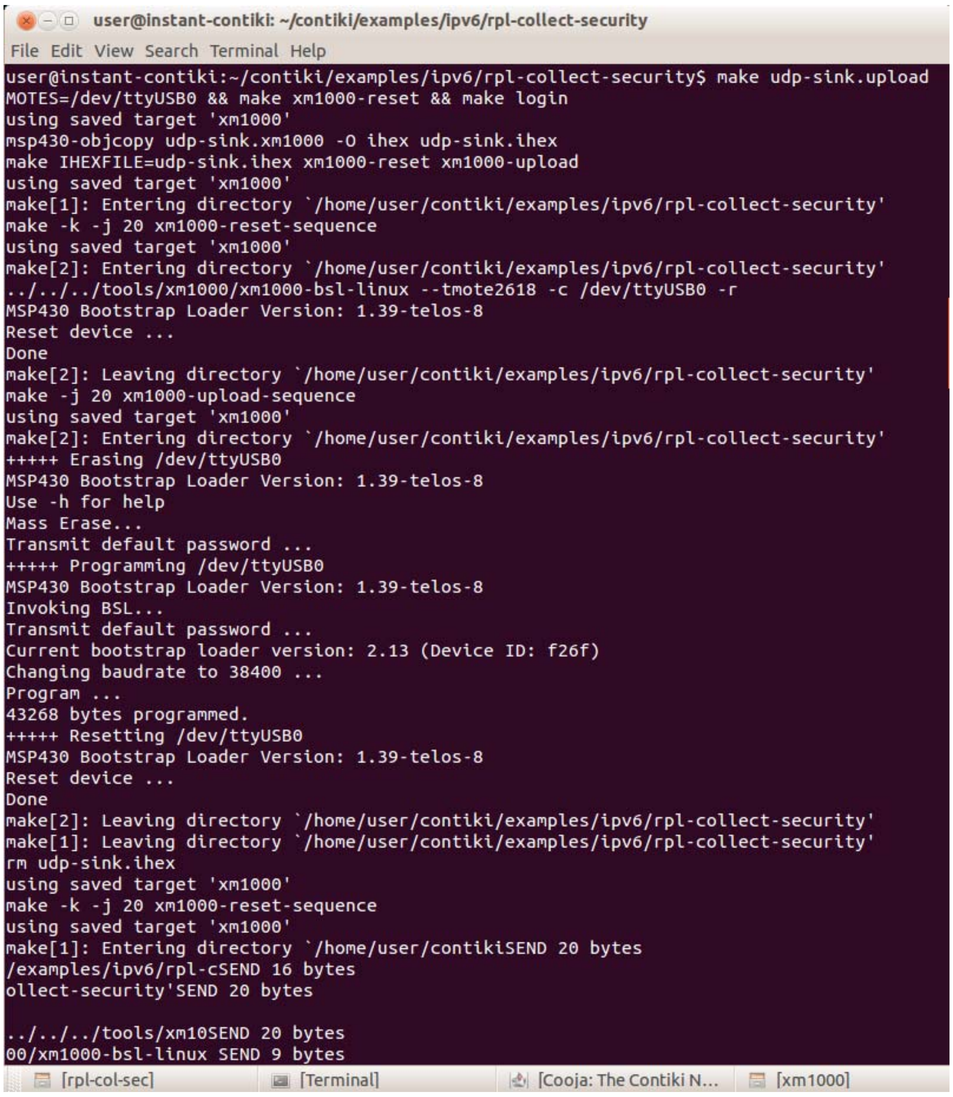
The testbed was run under the Blackhole attacks and the results collected. After a successful run of the Blackhole attacks under the *SecTrust*-RPL protocol, the Blackhole attack code was erased from the two XM1000 motes and it was then embedded with the Rank attack malicious codes. This testbed setup was also executed under the Rank attacks operation, and the results were similarly observed. Furthermore, the process above was repeated for Selective Forwarding and Sybil attacks respectively.

The motes were evenly deployed throughout the PhD research lab of the Auckland University of Technology on the fourth floor (WT building, room 404). The coverage area was approximately 30 metres by 30 metres. Further details on testbed setup is shown in Table 7.2. Figures 7.5, 7.6, 7.7 and 7.8 show the screen captures of the code embedding, compilation process of the UDP_sink, UDP_sender and Blackhole attack codes. Other attacks were similarly carried out. After embedding the codes into the motes, the sink mote, which serves as the controller mote remains connected to the USB interface of the Contiki desktop PC. This is required as it communicates with all the other motes deployed throughout the building. Also, all data sent by the deployed motes are channelled through the controller mote (sink) to the Contiki platform where it is logged for analysis.

Finally, the Contiki Collect View menu has the capability of showing the routing communication topology of all the XM1000 motes between the source mote (sink) and the sender motes under the interference of the malicious motes. This is presented in a later section.

Table 7.2: Network testbed settings and parameters

Parameters	Value
Emulation tool	Contiki/Cooja 3.0
Testbed coverage area	30m x 30m
Total number of XM1000 motes	14
Malicious nodes	2 (Nodes 13 and 14)
Malicious to legitimate node ratio	1 : 10 (approx.)
Mote deployment environment	Smart building
RX Ratio	30-100%
TX Ratio	100%
TX Range	50m
Interference Range	50m
Routing Protocols	MRHOF-RPL and *SecTrust*-RPL
Network protocol	IP based
Start Delay	5 seconds
Testbed testing time	7-day period



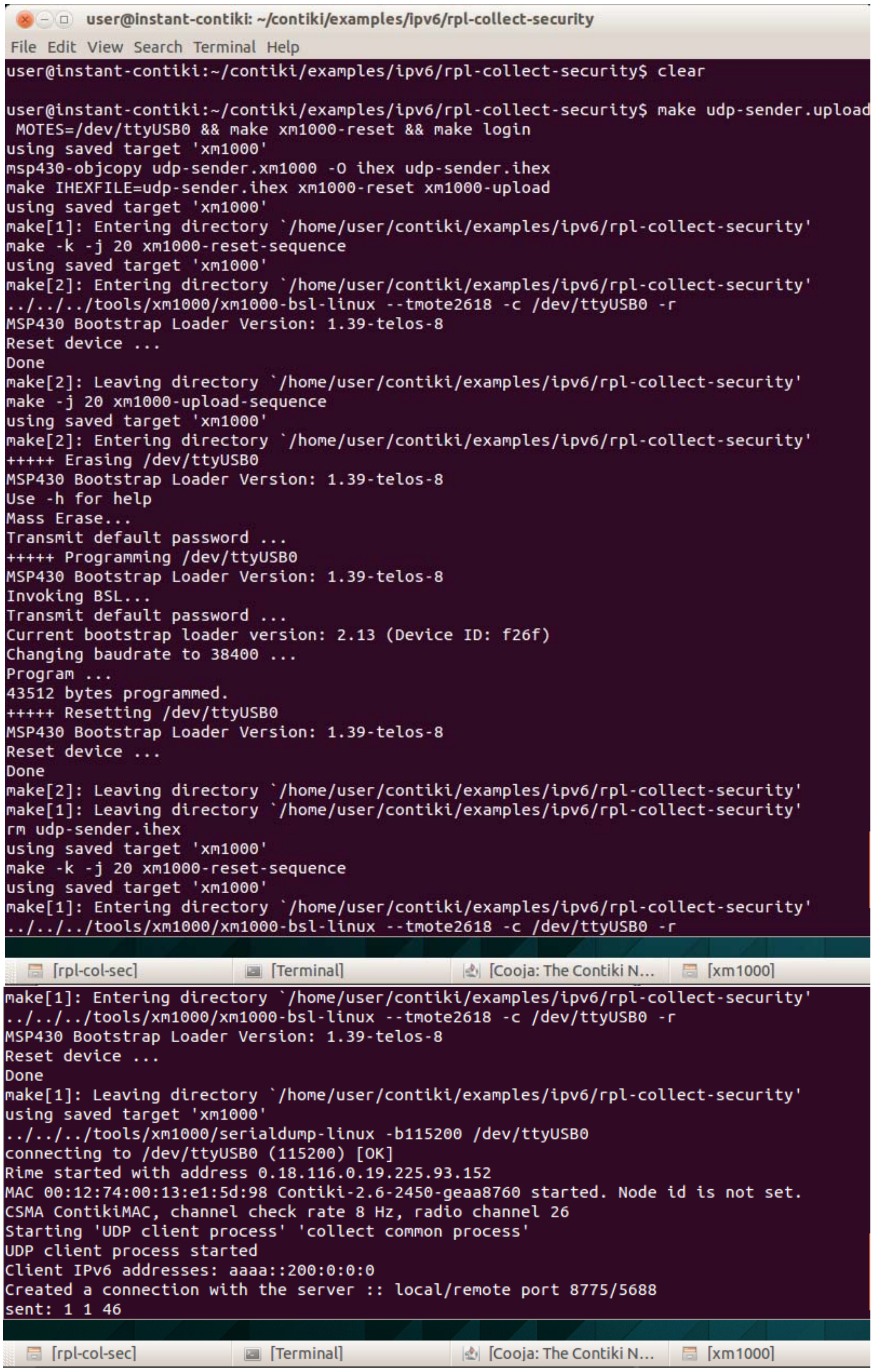
user@instant-contiki: ~/contiki/examples/ipv6/rpl-collect-security

File Edit View Search Terminal Help

```
user@instant-contiki:~/contiki/examples/ipv6/rpl-collect-security$ make udp-sink.upload
MOTES=/dev/ttyUSB0 && make xm1000-reset && make login
using saved target 'xm1000'
msp430-objcopy udp-sink.xm1000 -O ihex udp-sink.ihex
make IHEXFILE=udp-sink.ihex xm1000-reset xm1000-upload
using saved target 'xm1000'
make[1]: Entering directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
make -k -j 20 xm1000-reset-sequence
using saved target 'xm1000'
make[2]: Entering directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
../../../tools/xm1000/xm1000-bsl-linux --tmote2618 -c /dev/ttyUSB0 -r
MSP430 Bootstrap Loader Version: 1.39-telos-8
Reset device ...
Done
make[2]: Leaving directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
make -j 20 xm1000-upload-sequence
using saved target 'xm1000'
make[2]: Entering directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
+++++ Erasing /dev/ttyUSB0
MSP430 Bootstrap Loader Version: 1.39-telos-8
Use -h for help
Mass Erase...
Transmit default password ...
+++++ Programming /dev/ttyUSB0
MSP430 Bootstrap Loader Version: 1.39-telos-8
Invoking BSL...
Transmit default password ...
Current bootstrap loader version: 2.13 (Device ID: f26f)
Changing baudrate to 38400 ...
Program ...
43268 bytes programmed.
+++++ Resetting /dev/ttyUSB0
MSP430 Bootstrap Loader Version: 1.39-telos-8
Reset device ...
Done
make[2]: Leaving directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
make[1]: Leaving directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
rm udp-sink.ihex
using saved target 'xm1000'
make -k -j 20 xm1000-reset-sequence
using saved target 'xm1000'
make[1]: Entering directory `/home/user/contikiSEND 20 bytes
/examples/ipv6/rpl-cSEND 16 bytes
ollect-security'SEND 20 bytes

../../../tools/xm10SEND 20 bytes
00/xm1000-bsl-linux SEND 9 bytes
```

Figure 7.5: UDP sink code embedding and compilation on XM1000 mote

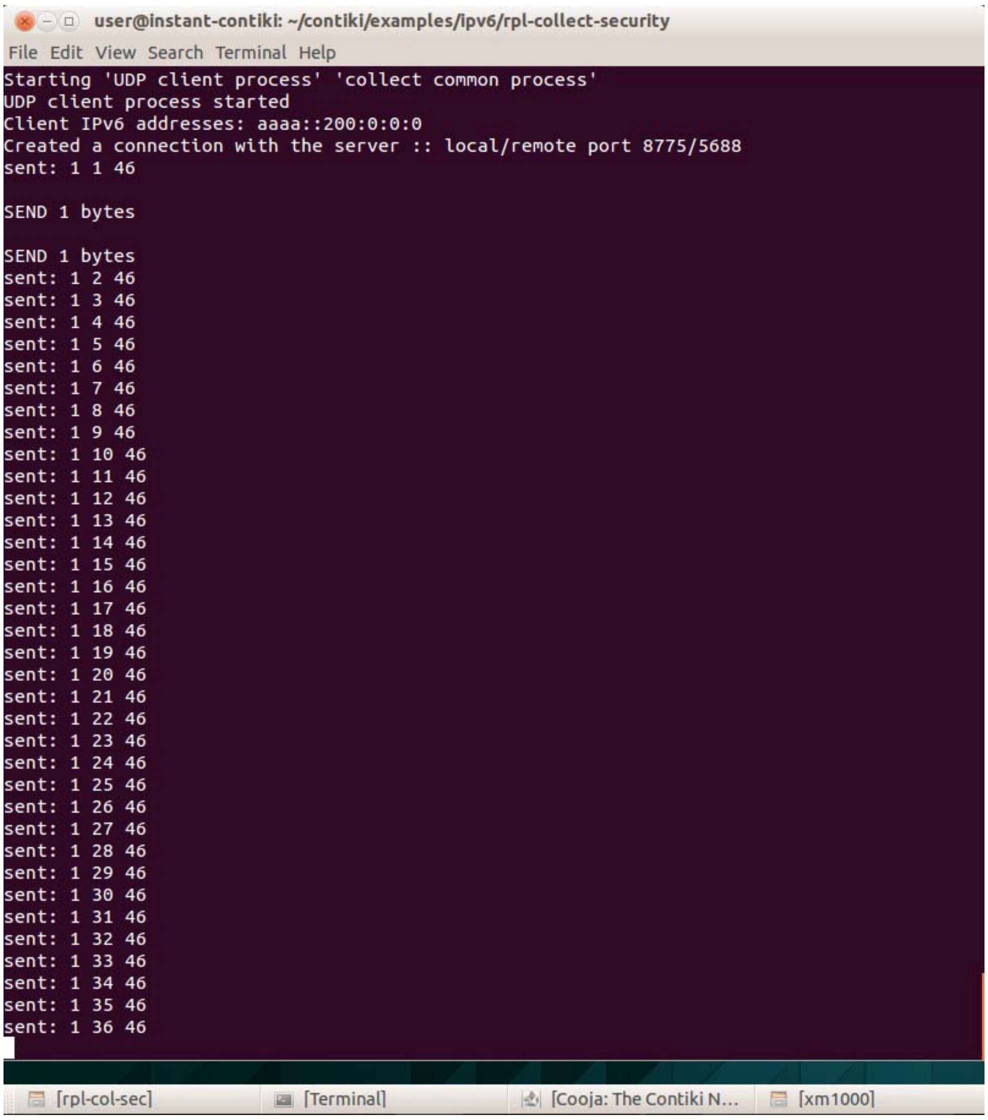

user@instant-contiki: ~/contiki/examples/ipv6/rpl-collect-security
File Edit View Search Terminal Help
[rpl-col-sec]
[Terminal]
[Cooja: The Contiki N...
[xm1000]

```
user@instant-contiki:~/contiki/examples/ipv6/rpl-collect-security$ clear

user@instant-contiki:~/contiki/examples/ipv6/rpl-collect-security$ make udp-sender.upload
 MOTES=/dev/ttyUSB0 && make xm1000-reset && make login
using saved target 'xm1000'
msp430-objcopy udp-sender.xm1000 -O ihex udp-sender.ihex
make IHEXFILE=udp-sender.ihex xm1000-reset xm1000-upload
using saved target 'xm1000'
make[1]: Entering directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
make -k -j 20 xm1000-reset-sequence
using saved target 'xm1000'
make[2]: Entering directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
../../../tools/xm1000/xm1000-bsl-linux --tmote2618 -c /dev/ttyUSB0 -r
MSP430 Bootstrap Loader Version: 1.39-telos-8
Reset device ...
Done
make[2]: Leaving directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
make -j 20 xm1000-upload-sequence
using saved target 'xm1000'
make[2]: Entering directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
+++++ Erasing /dev/ttyUSB0
MSP430 Bootstrap Loader Version: 1.39-telos-8
Use -h for help
Mass Erase...
Transmit default password ...
+++++ Programming /dev/ttyUSB0
MSP430 Bootstrap Loader Version: 1.39-telos-8
Invoking BSL...
Transmit default password ...
Current bootstrap loader version: 2.13 (Device ID: f26f)
Changing baudrate to 38400 ...
Program ...
43512 bytes programmed.
+++++ Resetting /dev/ttyUSB0
MSP430 Bootstrap Loader Version: 1.39-telos-8
Reset device ...
Done
make[2]: Leaving directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
make[1]: Leaving directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
rm udp-sender.ihex
using saved target 'xm1000'
make -k -j 20 xm1000-reset-sequence
using saved target 'xm1000'
make[1]: Entering directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
../../../tools/xm1000/xm1000-bsl-linux --tmote2618 -c /dev/ttyUSB0 -r
```

```
make[1]: Entering directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
../../../tools/xm1000/xm1000-bsl-linux --tmote2618 -c /dev/ttyUSB0 -r
MSP430 Bootstrap Loader Version: 1.39-telos-8
Reset device ...
Done
make[1]: Leaving directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
using saved target 'xm1000'
../../../tools/xm1000/serialdump-linux -b115200 /dev/ttyUSB0
connecting to /dev/ttyUSB0 (115200) [OK]
Rime started with address 0.18.116.0.19.225.93.152
MAC 00:12:74:00:13:e1:5d:98 Contiki-2.6-2450-geaa8760 started. Node id is not set.
CSMA ContikiMAC, channel check rate 8 Hz, radio channel 26
Starting 'UDP client process' 'collect common process'
UDP client process started
Client IPv6 addresses: aaaa::200:0:0:0
Created a connection with the server :: local/remote port 8775/5688
sent: 1 1 46
```

Figure 7.6: UDP sender code embedding and compilation on XM1000 mote

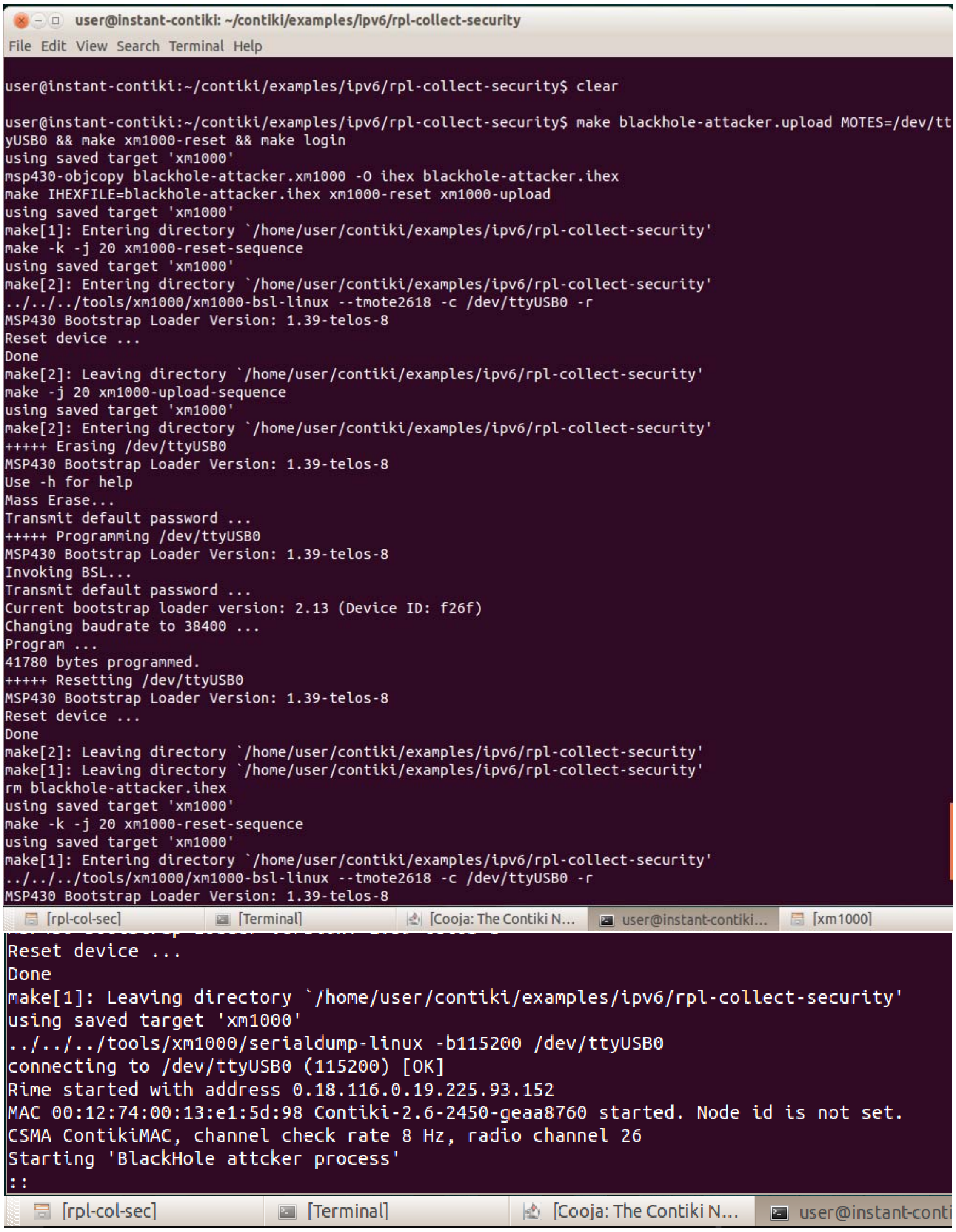

user@instant-contiki: ~/contiki/examples/ipv6/rpl-collect-security
File Edit View Search Terminal Help
[rpl-col-sec]
[Terminal]
[Cooja: The Contiki N...
[xm1000]

```
Starting 'UDP client process' 'collect common process'
UDP client process started
Client IPv6 addresses: aaaa::200:0:0:0
Created a connection with the server :: local/remote port 8775/5688
sent: 1 1 46

SEND 1 bytes

SEND 1 bytes
sent: 1 2 46
sent: 1 3 46
sent: 1 4 46
sent: 1 5 46
sent: 1 6 46
sent: 1 7 46
sent: 1 8 46
sent: 1 9 46
sent: 1 10 46
sent: 1 11 46
sent: 1 12 46
sent: 1 13 46
sent: 1 14 46
sent: 1 15 46
sent: 1 16 46
sent: 1 17 46
sent: 1 18 46
sent: 1 19 46
sent: 1 20 46
sent: 1 21 46
sent: 1 22 46
sent: 1 23 46
sent: 1 24 46
sent: 1 25 46
sent: 1 26 46
sent: 1 27 46
sent: 1 28 46
sent: 1 29 46
sent: 1 30 46
sent: 1 31 46
sent: 1 32 46
sent: 1 33 46
sent: 1 34 46
sent: 1 35 46
sent: 1 36 46
```

Figure 7.7: XM1000 UDP sender mote sending data packet to the sink mote


user@instant-contiki: ~/contiki/examples/ipv6/rpl-collect-security
File Edit View Search Terminal Help
[rpl-col-sec]
[Terminal]
[Cooja: The Contiki N...
user@instant-contiki...
[xm1000]

```
user@instant-contiki:~/contiki/examples/ipv6/rpl-collect-security$ clear

user@instant-contiki:~/contiki/examples/ipv6/rpl-collect-security$ make blackhole-attacker.upload MOTES=/dev/tt
yUSB0 && make xm1000-reset && make login
using saved target 'xm1000'
msp430-objcopy blackhole-attacker.xm1000 -O ihex blackhole-attacker.ihex
make IHEXFILE=blackhole-attacker.ihex xm1000-reset xm1000-upload
using saved target 'xm1000'
make[1]: Entering directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
make -k -j 20 xm1000-reset-sequence
using saved target 'xm1000'
make[2]: Entering directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
../../../tools/xm1000/xm1000-bsl-linux --tmote2618 -c /dev/ttyUSB0 -r
MSP430 Bootstrap Loader Version: 1.39-telos-8
Reset device ...
Done
make[2]: Leaving directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
make -j 20 xm1000-upload-sequence
using saved target 'xm1000'
make[2]: Entering directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
+++++ Erasing /dev/ttyUSB0
MSP430 Bootstrap Loader Version: 1.39-telos-8
Use -h for help
Mass Erase...
Transmit default password ...
+++++ Programming /dev/ttyUSB0
MSP430 Bootstrap Loader Version: 1.39-telos-8
Invoking BSL...
Transmit default password ...
Current bootstrap loader version: 2.13 (Device ID: f26f)
Changing baudrate to 38400 ...
Program ...
41780 bytes programmed.
+++++ Resetting /dev/ttyUSB0
MSP430 Bootstrap Loader Version: 1.39-telos-8
Reset device ...
Done
make[2]: Leaving directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
make[1]: Leaving directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
rm blackhole-attacker.ihex
using saved target 'xm1000'
make -k -j 20 xm1000-reset-sequence
using saved target 'xm1000'
make[1]: Entering directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
../../../tools/xm1000/xm1000-bsl-linux --tmote2618 -c /dev/ttyUSB0 -r
MSP430 Bootstrap Loader Version: 1.39-telos-8
```

```
Reset device ...
Done
make[1]: Leaving directory `/home/user/contiki/examples/ipv6/rpl-collect-security'
using saved target 'xm1000'
../../../tools/xm1000/serialdump-linux -b115200 /dev/ttyUSB0
connecting to /dev/ttyUSB0 (115200) [OK]
Rime started with address 0.18.116.0.19.225.93.152
MAC 00:12:74:00:13:e1:5d:98 Contiki-2.6-2450-geaa8760 started. Node id is not set.
CSMA ContikiMAC, channel check rate 8 Hz, radio channel 26
Starting 'BlackHole attcker process'
::
```

Figure 7.8: Blackhole malicious code embedding and compilation on XM1000 mote

7.5.2 Link failure Model

The testbed was performed using the UDGM with distance model. This is the same model adopted in Chapter 6 (Section 6.6) for the simulation of the network nodes. It provides a real-world emulation of the lossy links and shared media collision among wireless sensor nodes.

Figures 7.9 and 7.10 show the testbed setup using Contiki installed on a desktop PC as the operating platform while an XM1000 mote attached to the desktop PC communicates with other deployed motes. Figures 7.11 to 7.15 show the deployment of XM1000 motes in the PhD Research lab communicating with the sink mote attached to a host PC running Contiki.

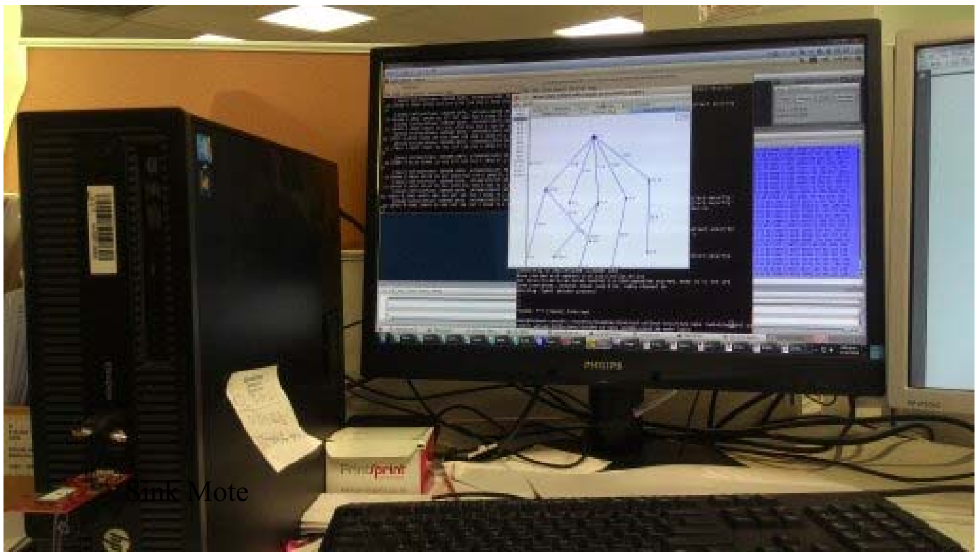


Figure 7.9: Testbed setup using Contiki and XM1000 mote as sink

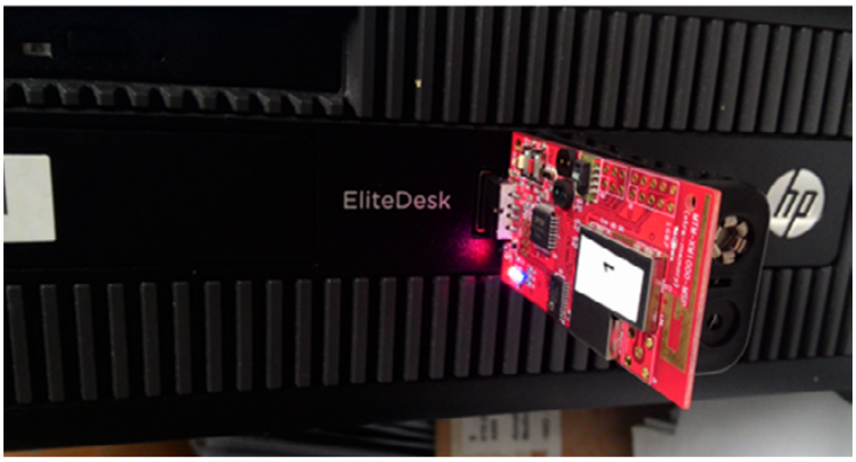


Figure 7.10: XM1000 mote as sink mote connected to Contiki via a desktop PC

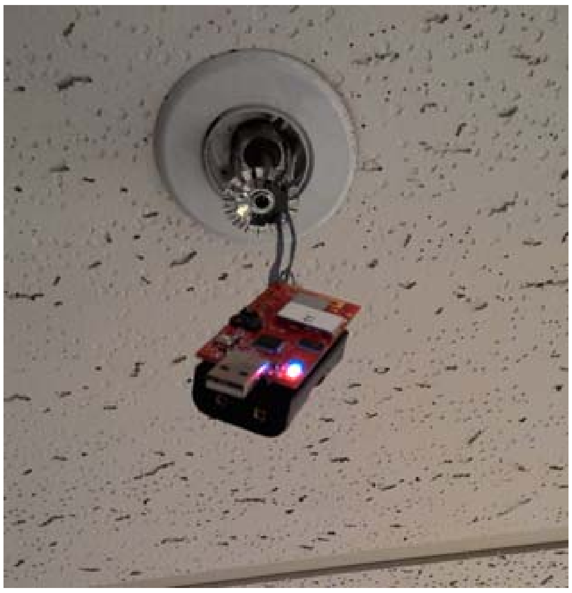


Figure 7.11: An XM1000 mote deployed in the PhD Research lab communicating with the sink mote attached to a desktop PC

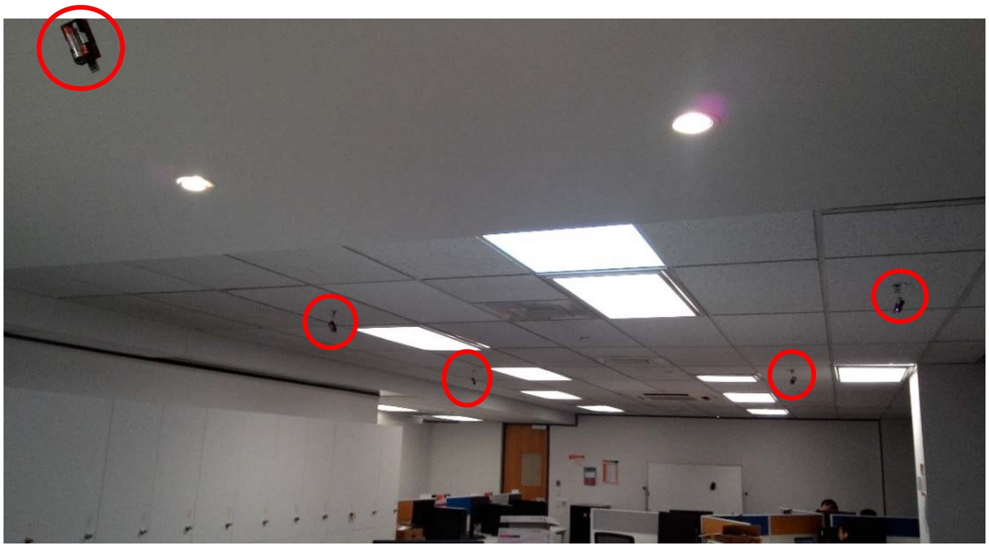


Figure 7.12: XM1000 motes deployed in the PhD Research lab of the AUT WT building

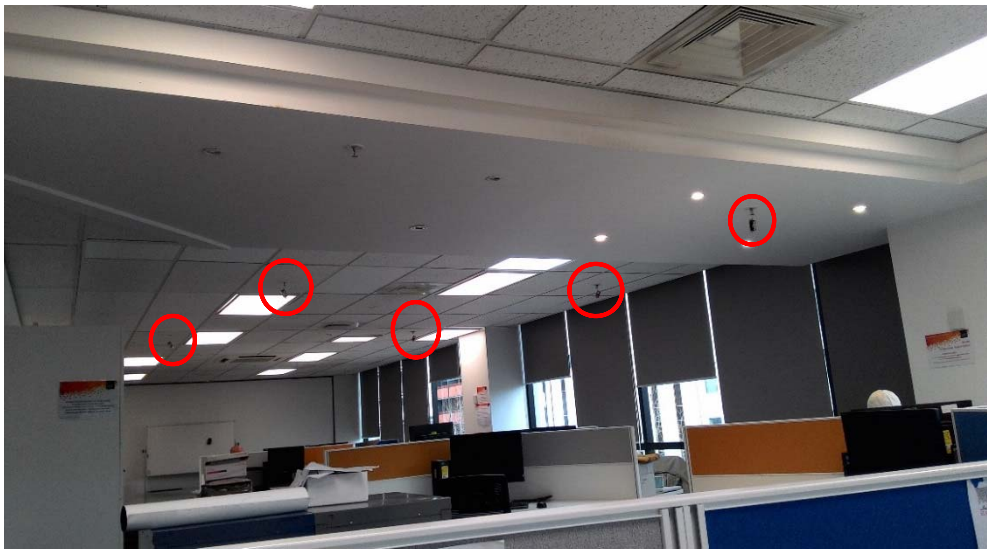


Figure 7.13: XM1000 motes deployed in the PhD Research lab of the AUT WT building

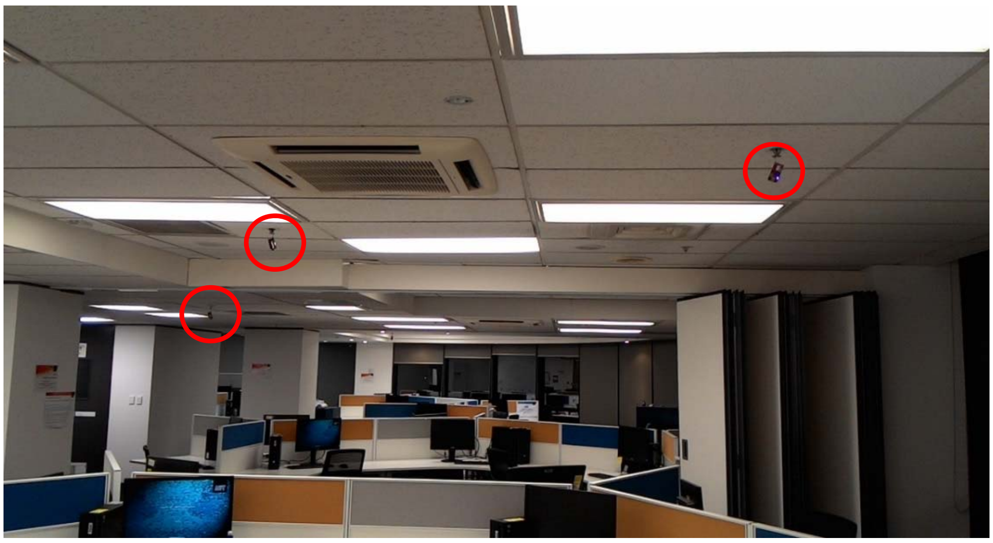


Figure 7.14: XM1000 motes deployed in the PhD Research lab of the AUT WT building

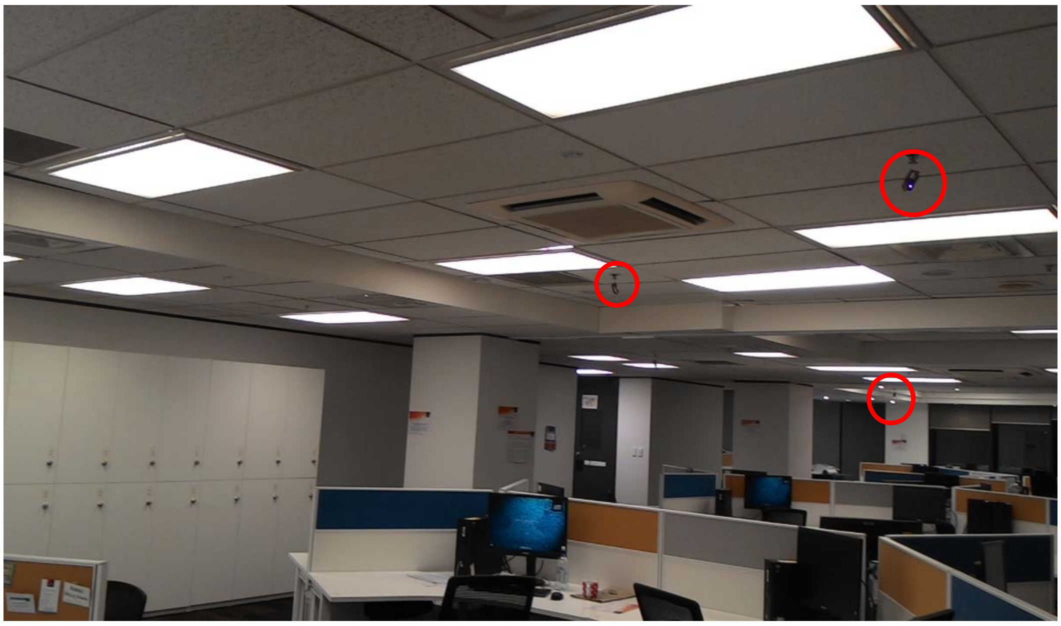
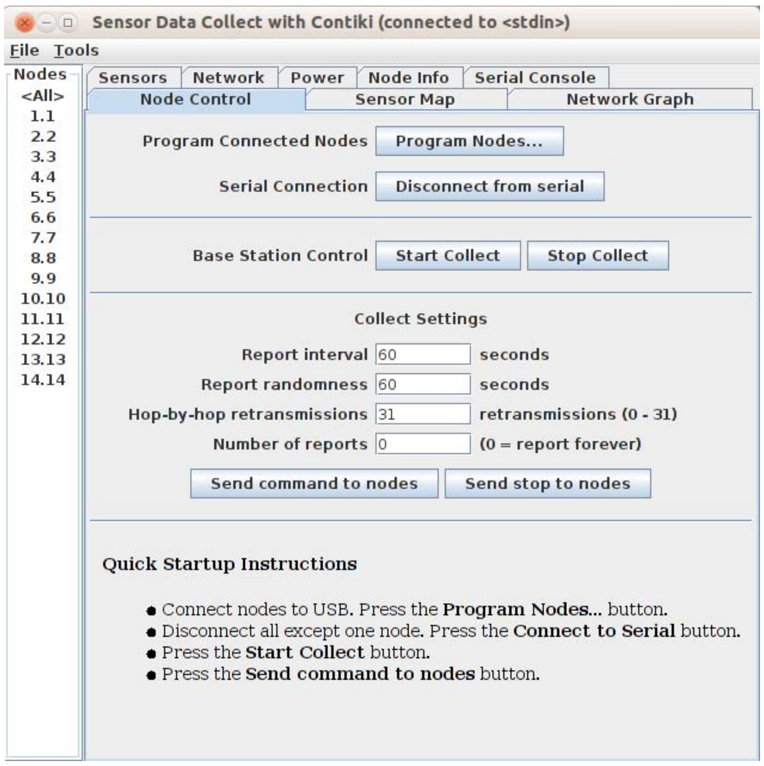


Figure 7.15: XM1000 motes deployed in the PhD Research lab of the AUT WT building

7.5.3 Data Logging and Capture

During the experimentation of the IoT testbed, the XM1000 sink and sender motes generate control and route traffic among themselves used for the creation of the network topology for route communication. These routes are used for the transmission of generated data packets. Contiki provides the ability to log generated route data, which was captured, analysed and this is discussed in the subsequent section. Figure 7.7 shows a sample of packet transmitted by one of the motes to the sink. The Figure shows *"sent: 1 2 46*" for example. This means a packet was sent to the mote 1 (sink mote) with serial packet number of 2 and a total of 46 bytes including headers. When it gets to the destination the header bytes are stripped which is an excess of 16 bytes thus, the actual payload packet sent is 30 bytes of information. This data packet sent is used for analysis to determine and compare the network performances between the two protocols. Also, the route and control information are logged using the collect view option in Contiki/Cooja and this is shown in Figure 7.16 below:



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** button.
- Press the **Start Collect** button.
- Press the **Send command to nodes** button.

Figure 7.16: A sensor data collector in Contiki/Cooja

The collection scheme mechanism for the deployed testbed is depicted in Figure 7.17. The dotted lines indicate the source and final destination of data packets while the solid lines show the topology route path created and used for packet delivery. See appendix 2 to access raw data generated during the testbed experiments.

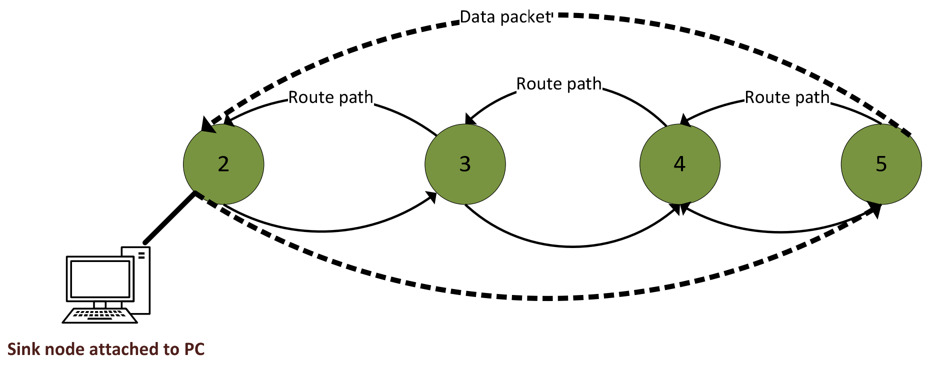


Figure 7.17: Testbed implementation of route and data collection scheme mechanism

7.6 Experimental Results

At the start of the experiment, the sink mote sends commands to all available motes so they could discover it, and a network topology can be built for route communication and data transmission. The experiments performed in this testbed involved a detailed analysis of the protocols’ behaviour under study as compared with the simulation study. Just like the simulation study conducted, the testbed deployment and experimentation process was rather tasking as the deployment and embedding of different malicious codes had to be done during the tests, to observe the performance of both protocols under different malicious attack types. The testbed was run for a 60-minute period and repeated multiple times over a period of seven consecutive days while changing malicious mote locations all throughout the experiment. However, all motes were static any time the testbed was run. The output log gathered after each experimental run (Figure 7.7) was rather voluminous that a C-script was written to mine data required for network performance analysis.

7.6.1 *SecTrust*-RPL vs MRHOF-RPL under Blackhole Attacks

This section examines the performance of MRHOF-RPL and *SecTrust*-RPL under Blackhole attacks while conducting the testbed experiment with XM1000 motes. Figure 7.18 shows an instance of the snap short capture of the network topology formation of XM1000 motes using *SecTrust*-RPL protocol in Contiki/Cooja (sensor data collection menu) under Blackhole attacks. *SecTrust*-RPL could isolate motes 13 and 14 from its route topology formation because of their malicious (Blackhole) behaviour within the network. On the other hand, Figure 7.19 shows the network topology formation of MRHOF-RPL protocol during the testbed testing. A quick look at the topology layout shows that MRHOF-RPL could not mitigate the effect of malicious activities of the Blackhole attacking motes (13 and 14) in the network. Consequently, three disjointed network topologies were formed which resulted in unsuspecting motes 2, 3, 12 and 6,7 selecting motes 13 and 14 respectively and routing their data through them.

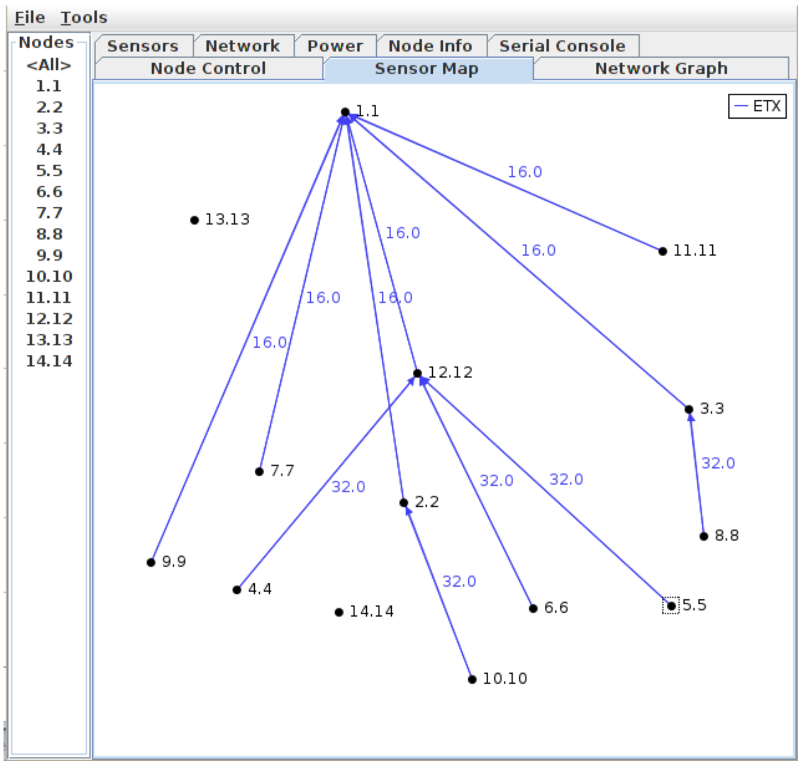


Figure 7.18: *SecTrust*-RPL route topology setup and isolation of Blackhole attacks during the testbed experiment with XM1000 motes

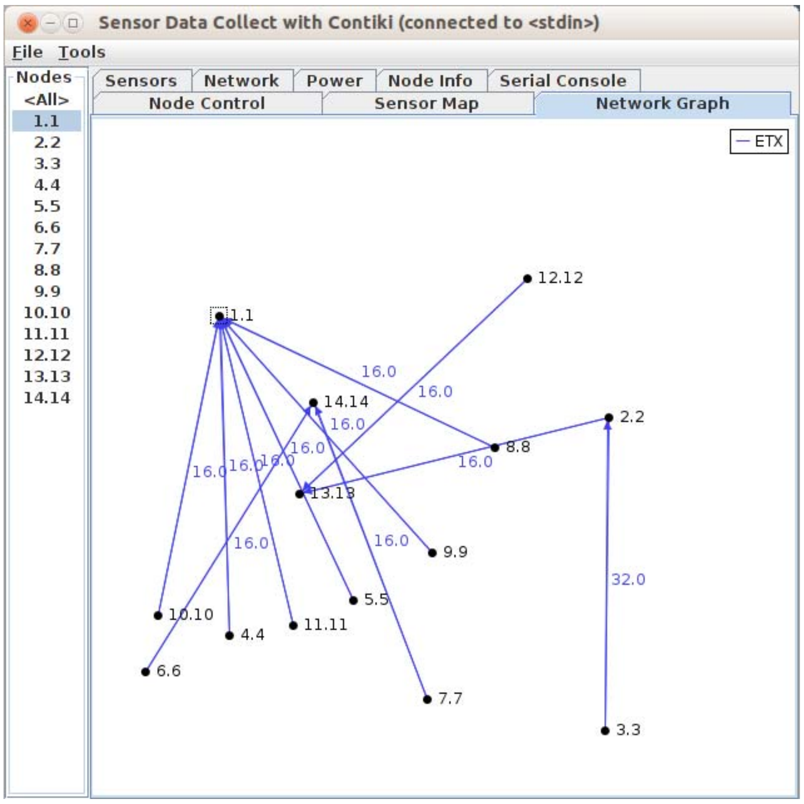


Figure 7.19: MRHOF-RPL route topology segmentation under Blackhole attacks during the testbed experiment with XM1000 motes

1. Detection and Isolation of Attack Nodes

From the result shown in Figure 7.20, *SecTrust*-RPL could detect and isolate Blackhole attacks during routing operations. The first five minutes of RPL operation witnessed the detection of a slightly higher level of initial Blackhole attacks among the malicious nodes. However, the level of Blackhole activity remained within the range of 14 – 20 detections per five minutes of RPL operation. Since *SecTrust*-RPL could identify and isolate the Blackhole attack motes, the malicious motes were not considered for routing decisions in the network as captured in Figure 7.18.

MRHOF-RPL protocol was unable to detect any of the Blackhole attacks being perpetrated in the RPL network, and this is reflected by the high frequency of mote Rank changes as shown in Figure 7.21. It is important to note that the coverage area of the testbed was approximately 30 metres by 30 metres, while the XM1000 motes had a transmission range of 50 metres. As a result, each XM1000 mote was within the coverage area of both the sink mote and the malicious attacking motes. RPL as part of its normal routing operations, periodically evaluates (trickle timer algorithm) the Rank value of its parent and other potential parents with a view of maintaining Rank consistency. RPL ensures that a mote selects a parent mote with the least Rank value to maintain a loop-free route topology. When it discovers another parent with a lower Rank value than its parent, it re-aligns itself with the new mote with lower Rank value. Thus, this new mote becomes its parent. However, an incessant alignment and re-alignment of motes from one parent to another causes instability of the network and hence, makes the network inefficient. Therefore, a high frequency of mote Rank changes among motes in the network shows that a network is experiencing a high level of child-parent re-alignment, which generally destabilizes a RPL network by making packets undeliverable. This causes network convergence to take longer time and introduces gross inefficiencies in the network. The Blackhole attack motes advertise themselves to their neighbours as better routes to their destination in a guise to attract these unsuspecting motes only to drop their packets in the end.

Now Figure 7.21 shows *SecTrust*-RPL maintained the number of its node Rank changes within 15 – 55 while MRHOF-RPL had a range between 16 – 246 motes Rank changes. From the Figure, it can be observed that MRHOF-RPL had significant number of mote Rank changes over and above that of *SecTrust*-RPL protocol; an indication that the activities of the Blackhole attack motes during RPL's operation had significant impact on MRHOF-RPL protocol.

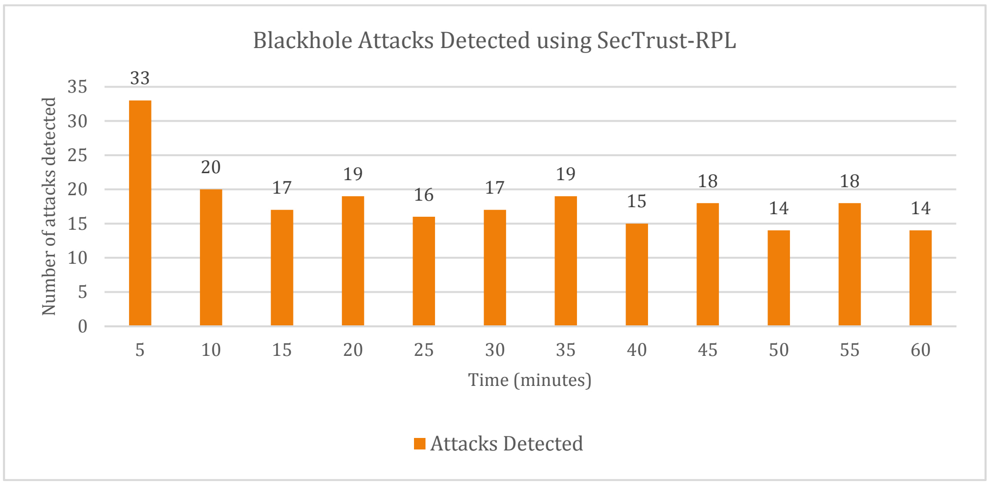


Figure 7.20: Blackhole attacks detection and isolation during the testbed experiment

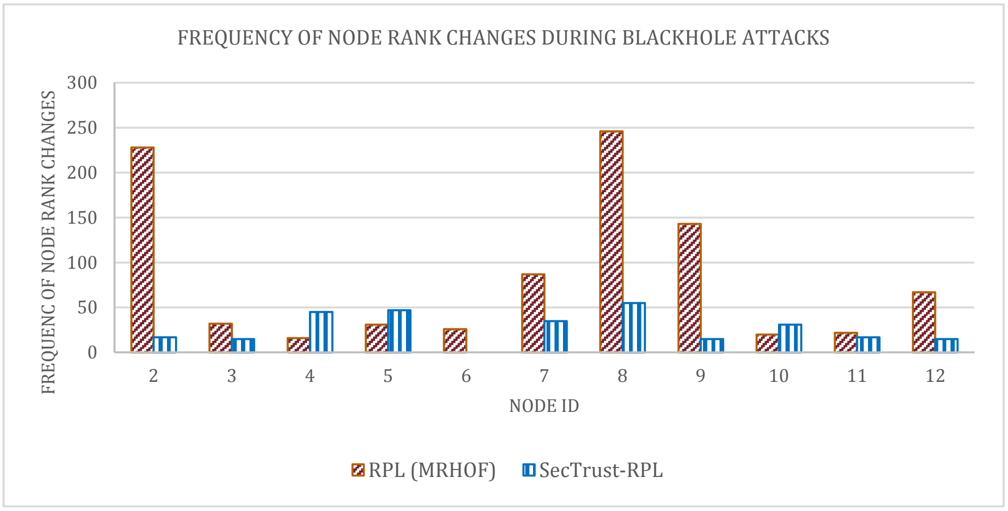


Figure 7.21: Comparison of frequency of node Rank changes under Blackhole attacks during the testbed experiment

2. Network Performance Measures

This section analyses the network performance measurements of MRHOF-RPL and *SecTrust*-RPL under Blackhole attacks during the testbed experiment using the XM1000 motes. In the throughput comparison between MRHOF-RPL and *SecTrust*-RPL in Figure 7.22, *SecTrust*-RPL protocol maintained a significant throughput measurement over and above MRHOF-RPL achieving almost 7 kbps in throughput. However, MRHOF-RPL had lower throughput measurement with mote 7 achieving the highest, which was 3 kbps and mote 8 delivering the lowest throughput of 0.34 kbps.

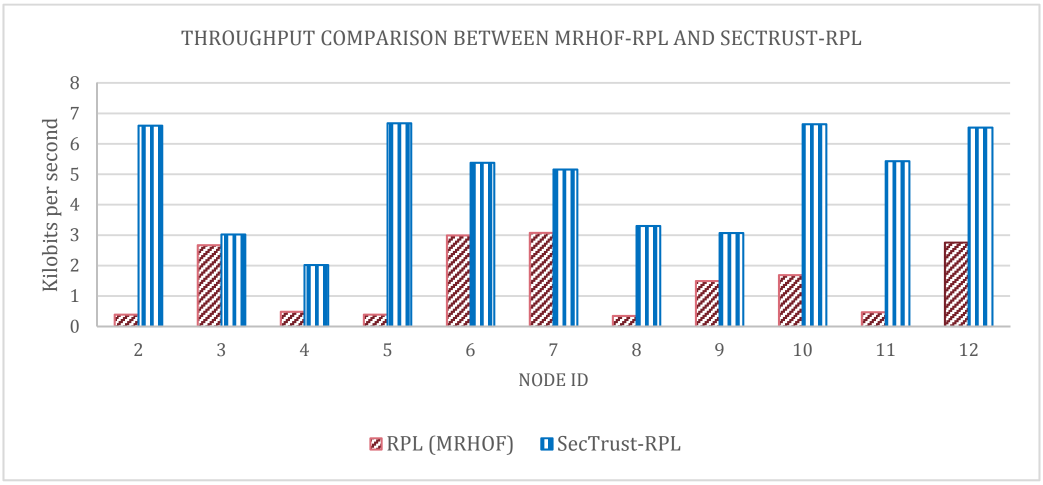


Figure 7.22: Network throughput performance comparison between *SecTrust*-RPL and MRHOF-RPL during the testbed experiment

Figure 7.23 below shows the packet loss rate between MRHOF-RPL and *SecTrust*-RPL. While *SecTrust*-RPL maintained a packet loss rate between 16% – 27%, MRHOF-RPL had a packet loss rate of 60% – 73%. It is observable from the graph that malicious Blackhole motes had significant impact on MRHOF-RPL protocol by inhibiting data packets from reaching the sink mote; more so than is the case with the *SecTrust*-RPL protocol. This clearly asserts *SecTrust*-RPL as having better throughput performance than MRHOF-RPL under Blackhole attacks.

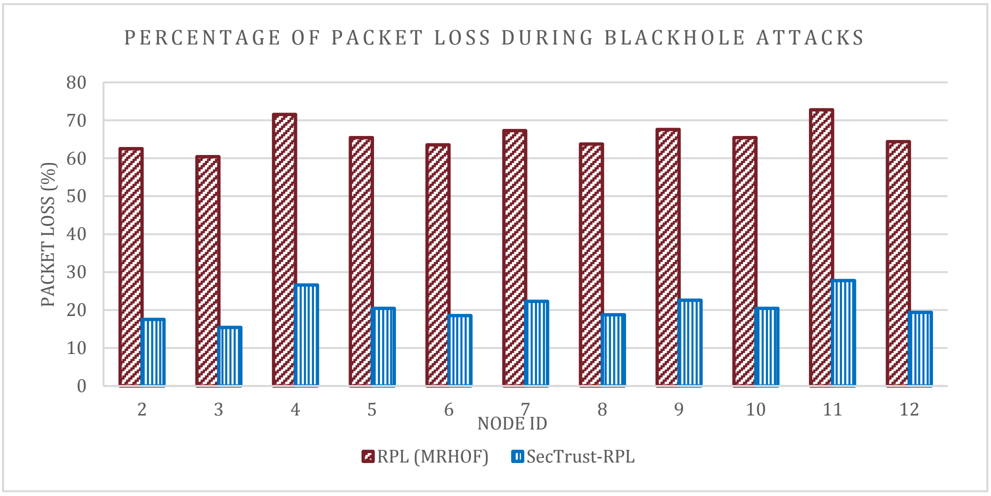


Figure 7.23: Percentage of packet loss comparison between *SecTrust*-RPL and MRHOF-RPL under Blackhole attacks during the testbed experiment

The packet transmission delay between MRHOF-RPL and *SecTrust*-RPL in Figure 7.24, shows the latency in the transmission of data from sender motes to the sink mote under Blackhole attacks. It is observed from Figure 7.24 that *SecTrust*-RPL had a transmission delay range of 35 – 230 ms while MRHOF-RPL had a transmission delay of 77 – 673 ms. Although, motes 3, 6, 7, 9 and 10 under MRHOF-RPL showed relatively low delay times, however, a further look at their corresponding packet loss rates show all five motes recording over 65% packet loss rate, an indication that only a few packets (some being the initial packets sent before the malicious attacks were triggered) got to the sink mote.

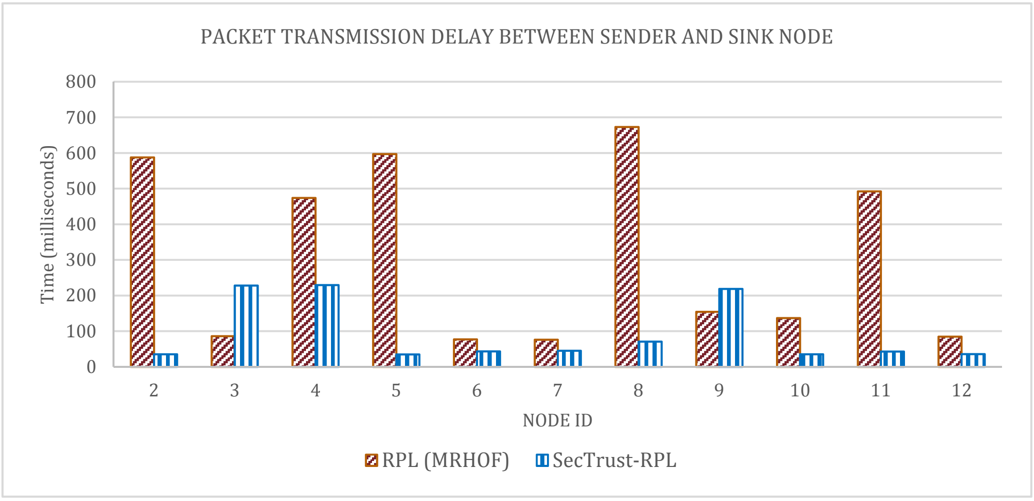


Figure 7.24: Packet transmission delay of *SecTrust*-RPL and MRHOF-RPL under Blackhole attacks during the testbed experiment

7.6.2 *SecTrust*-RPL vs MRHOF-RPL under Rank Attacks

This section examines the performance of MRHOF-RPL and *SecTrust*-RPL under Rank attacks during testbed experimentation. Figure 7.25 shows an instance of the snap short capture of the network topology formation of XM1000 motes using *SecTrust*-RPL protocol in Contiki/Cooja (sensor data collection menu) under Rank attacks. *SecTrust*-RPL could isolate mote 13 from its route topology formation because of its malicious (Rank) behaviour within the network. Malicious mote 14 could perpetrate its attacks against motes 8 and 9, apart from that, the network topology remained stable with the remaining motes transmitting to the sink mote (mote 1). Conversely, the topology of MRHOF-RPL in Figure 7.26 reveals the segmentation of the network topology into three parts with malicious motes 13 and 14 attracting motes 6, 7 and motes 2, 3 and 12 respectively. MRHOF-RPL could not defend against the malicious activities of the Rank attacking motes (13 and 14) in the network.

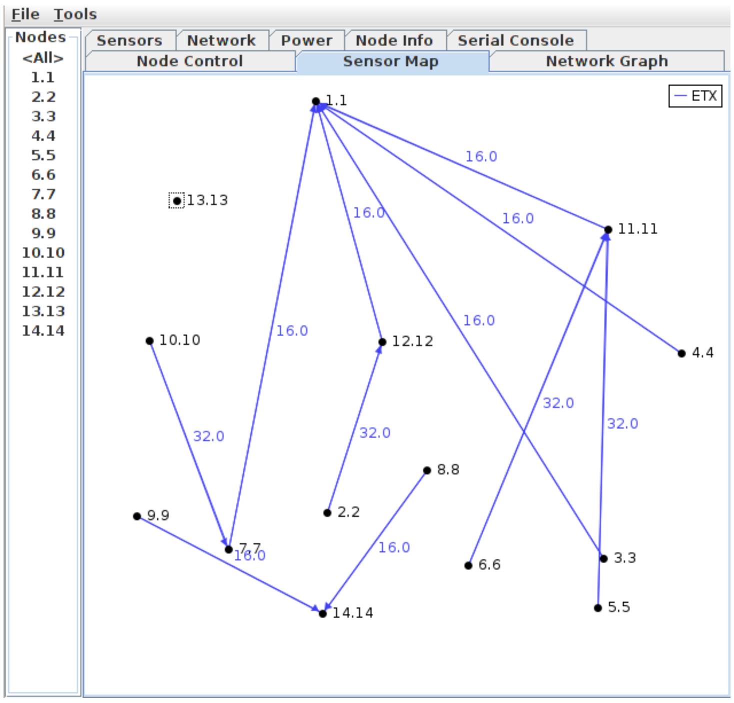


Figure 7.25: *SecTrust*-RPL route topology setup and isolation of Rank attacks during the testbed experiment with XM1000 motes

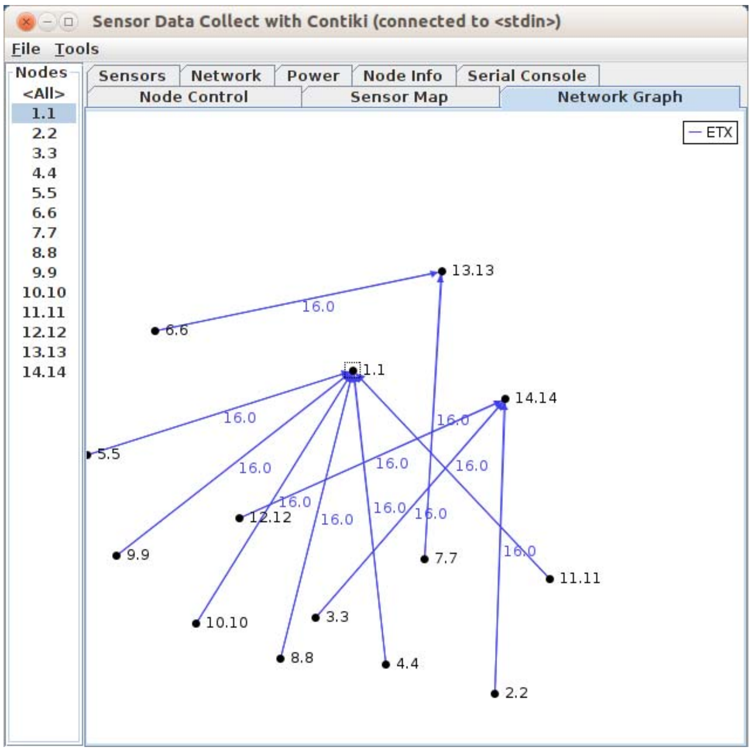


Figure 7.26: MRHOF-RPL route topology segmentation under Rank attacks during the testbed experiment with XM1000 motes.

1. Detection and Isolation of Attack Nodes

The testbed experiment in Figure 7.27 shows the detection of Rank attacks over a 60-minute simulation period of testing, with the first five minutes having the highest detection rate of 63 detections. The detection of attacks after the first 5 minutes maintained a rather stable pattern of 21 to 31 attack detections. MRHOF-RPL has no mechanism for detecting spurious Rank advertisements made by malicious motes 13 and 14 during the experiment. This is further supported with the mote Rank changes in Figure 7.28 discussed next.

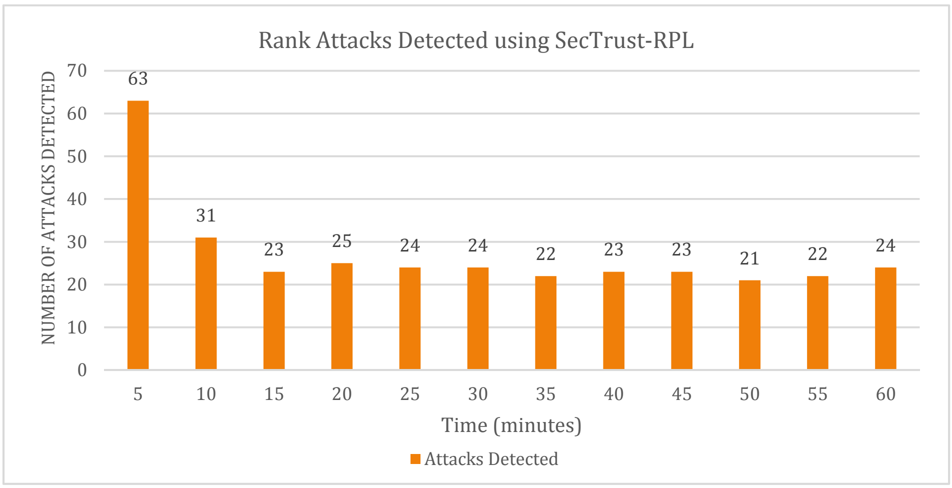


Figure 7.27: Rank attacks detection and isolation during the testbed experiment

Figure 7.28 below shows the frequency of mote Rank changes between MRHOF-RPL and *SecTrust*-RPL protocols during testing with motes 13 and 14 perpetrating Rank attacks. MRHOF-RPL had higher mote Rank changes, which ranged from 17 to 197. An indication that the network topology was under constant attack by malicious motes 13 and 14 hence, making the network rather inefficient as motes in the network constantly changed their preferred parents due to the spurious Rank advertisements made by these 2 malicious motes. However, *SecTrust*-RPL had much lower Rank changes with zero Rank changes on motes 5 and 8 and the highest being 41 on mote 12. As a result of the lower frequency of mote Rank changes, it is expected that the network performance and stability of *SecTrust*-RPL protocol will be superior to MRHOF-RPL protocol. This is discussed next.

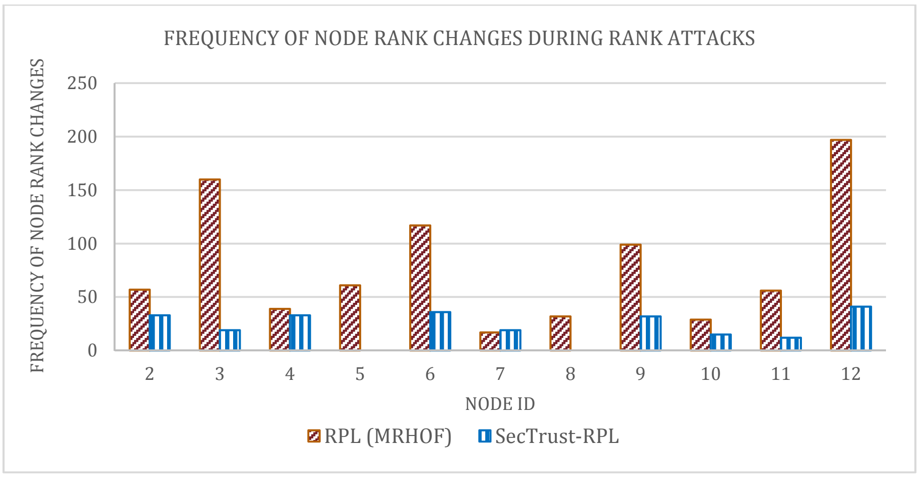


Figure 7.28: Comparison of frequency of node Rank changes under Rank attacks during the testbed experiment

2. Network Performance Measures

In the throughput comparison between MRHOF-RPL and *SecTrust*-RPL in Figure 7.29, *SecTrust*-RPL consistently maintained better throughput performance over MRHOF-RPL. *SecTrust*-RPL across all motes recorded higher throughput performance over MRHOF-RPL.

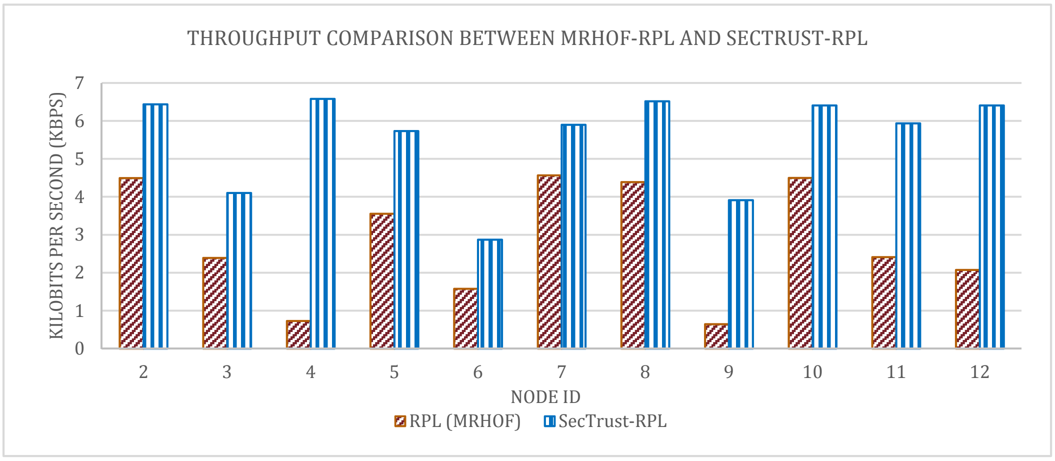


Figure 7.29: Network throughput performance comparison between *SecTrust*-RPL and MRHOF-RPL during the testbed experiment

Figure 7.30 shows the packet loss rate between MRHOF-RPL and *SecTrust*-RPL. While *SecTrust*-RPL maintained a packet loss rate between 15% – 28%, MRHOF-RPL had a packet loss rate of 60% – 73% during the RPL testbed experiment. It can be summarised from Figure 7.30 that the Rank attack activities of the malicious motes had significant scope of influence on MRHOF-RPL network operations than on *SecTrust*-RPL network operations. This shows *SecTrust*-RPL as having a much better network performance than MRHOF-RPL under Rank attacks.

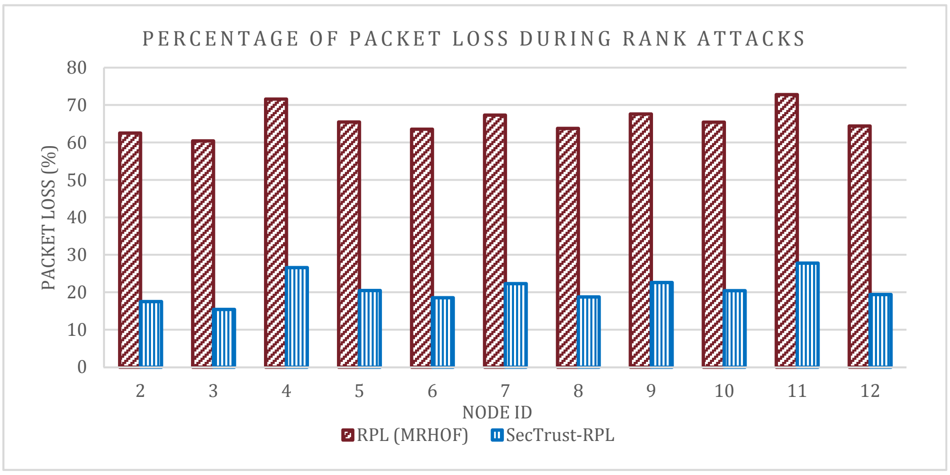


Figure 7.30: Percentage of packet loss comparison between *SecTrust*-RPL and MRHOF-RPL under Rank attacks during the testbed experiment

The packet transmission delay between MRHOF-RPL and *SecTrust*-RPL in Figure 7.31, shows the latency in the transmission of data packets from sender motes to the sink mote under Rank attacks during the testbed trials. The transmission delay range across all motes for MRHOF-RPL was 52 – 100 ms whereas *SecTrust*-RPL had a transmission delay range of 38 – 70 ms. In comparison, MRHOF-RPL experienced higher transmission delays during the trials. This could be attributed to the malicious interference of Rank attacking motes 13 and 14.

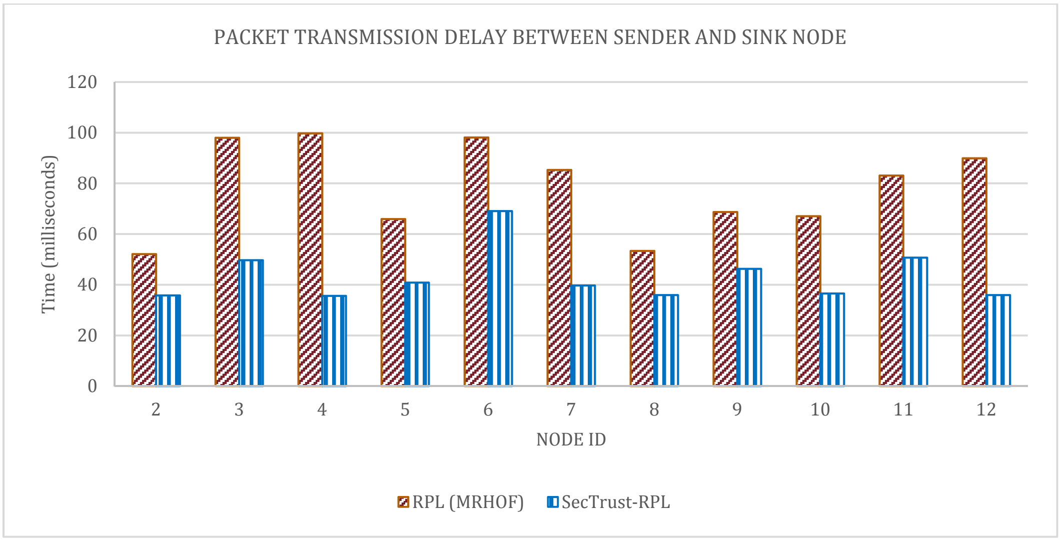


Figure 7.31: Packet transmission delay of *SecTrust*-RPL and MRHOF-RPL under Rank attacks during the testbed experiment

7.6.3 *SecTrust*-RPL vs MRHOF-RPL under Selective Forwarding Attacks

The testbed experiment was extended to test the performance of *SecTrust*-RPL and MRHOF-RPL under Selective Forwarding attacks using the deployed XM1000 motes. When the test was conducted an instance of the network topology as captured in the sensor data collection menu of Contiki/Cooja, and the topology instance is displayed in Figure 7.32. It reveals that *SecTrust*-RPL protocol could detect and isolate malicious motes 13 and 14 from its route topology formation. However, in Figure 7.33, which is the network topology formation of MRHOF-RPL, it could not mitigate the effect of malicious activities of the Selective Forwarding attacking motes (13 and 14) in the network. Motes 3, 6 and 8 were drawn to malicious mote 13 while motes 2, 7 and 12 were drawn to malicious mote 14. The rest of the motes however, were connected to the sink mote. The Figure in 7.33 creates a situation whereby packets sent to mote 1 by motes 2, 7, 12 cannot be delivered during the period they were connected to mote 13. Also, packets sent by motes 2, 7 and 12 cannot be received by mote 1 during the time they were connected to malicious mote 14, since the network is segmented and cannot connect to the central sink mote.

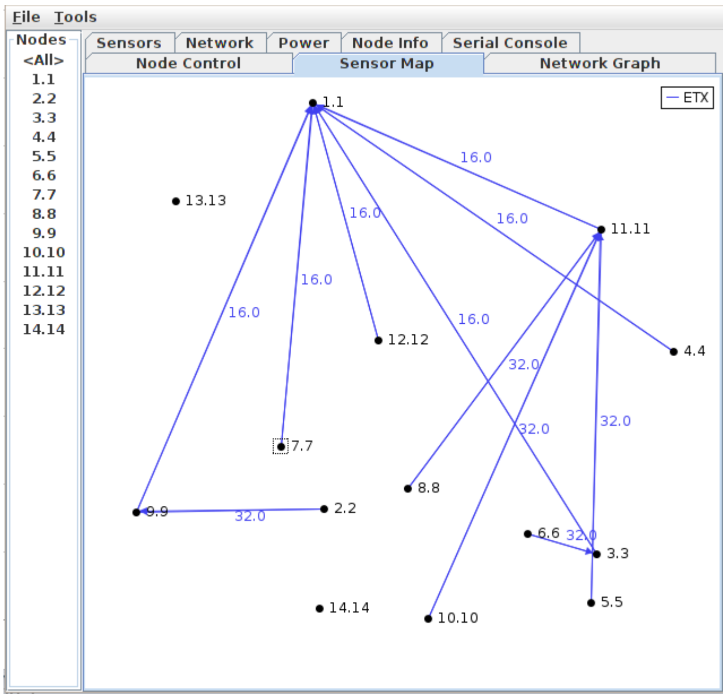


Figure 7.32: *SecTrust*-RPL route topology setup and isolation of Selective Forwarding attacks during the testbed experiment with XM1000 motes

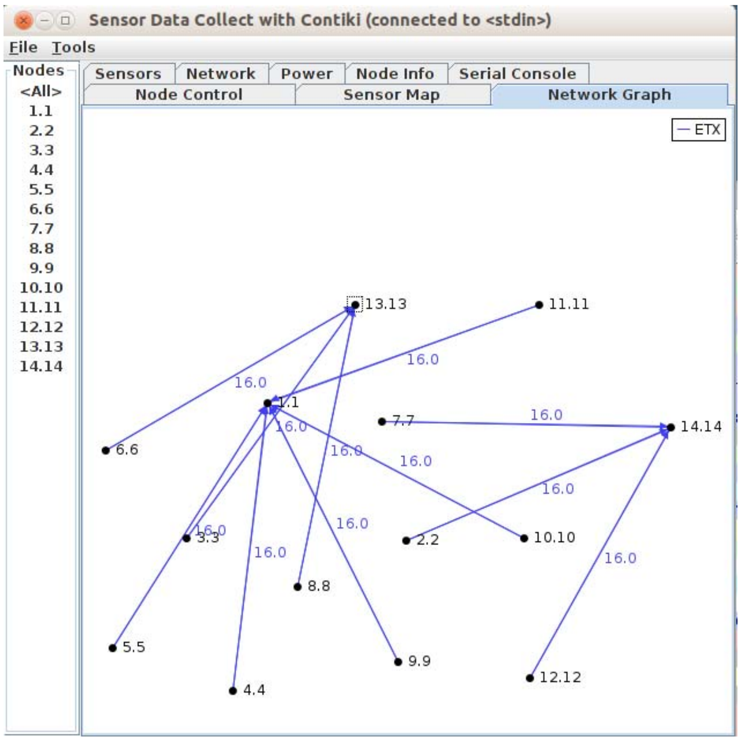


Figure 7.33: MRHOF-RPL route topology segmentation under Selective Forwarding attacks during the testbed experiment with XM1000 motes

1. Detection and Isolation of Attack Nodes

The testbed experiment in Figure 7.34 shows the detection of attacks perpetrated by malicious motes 13 and 14 perpetrating Selective Forwarding attacks. The first five minutes shows the detection of high amount of attacks while the remaining simulation period (10 – 60 minutes) shows a relatively stable number of attacks detection (50 – 75). The initial high flow of attacks and detection in the first five minutes is attributed to RPL's proactive routing nature, which floods the network with DIOs. Due to this, the Selective Forwarding attacking motes (13 and 14) rapidly, but selectively intercept and forward packets as per their malicious behaviour. Conversely, MRHOF-RPL protocol has no mechanism for detecting Selective Forwarding attacks perpetrated by malicious motes 13 and 14 during the experimentation period.

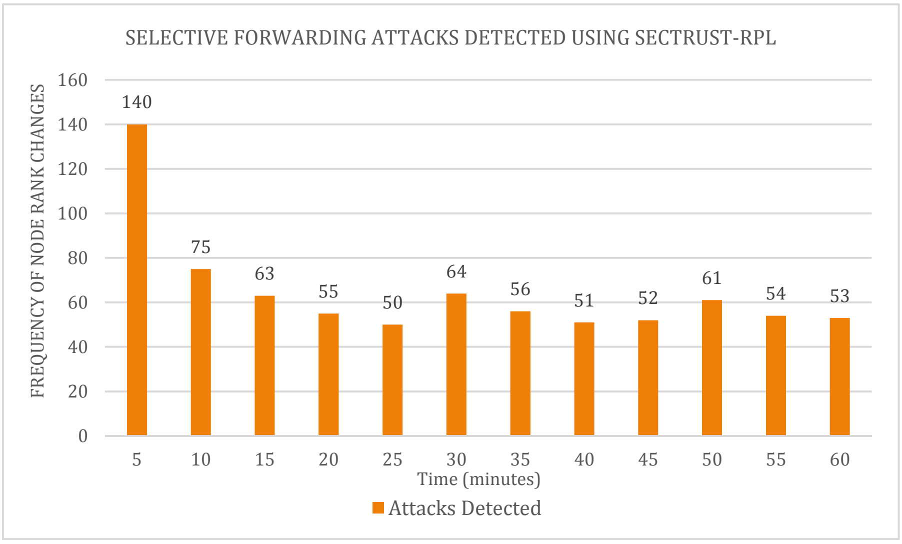


Figure 7.34: Selective Forwarding attacks detection and isolation during the testbed experiment

Figure 7.35 shows the frequency of mote Rank changes between MRHOF-RPL and *SecTrust*-RPL under Selective Forwarding attacks during the testbed operation. MRHOF-RPL had a significant number of mote Rank frequency changes. The frequency of Rank changes for MRHOF-RPL ranged from 122 – 661 while *SecTrust*-RPL had a range of 17 – 200. MRHOF-RPL protocol under Selective Forwarding attacks reveals high malicious mote activity. Conversely, *SecTrust*-RPL had much lower mote Rank changes, which could be considered moderate and consistent with RPL operations. This thus, implies *SecTrust*-RPL had better network performance and stability.

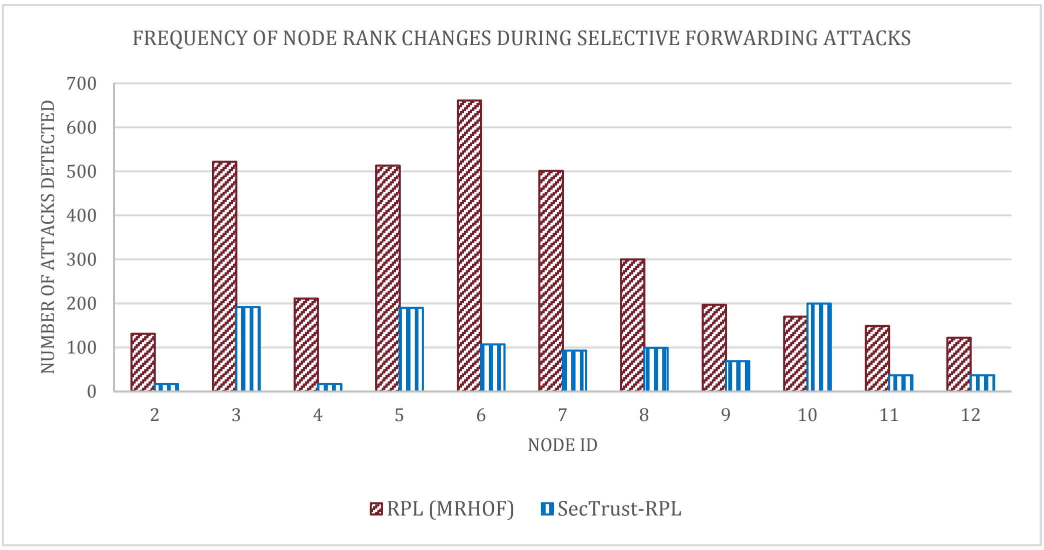


Figure 7.35: Comparison of frequency of node Rank changes under Selective Forwarding attacks during the testbed experiment

2. Network Performance Measures

In the throughput comparison between MRHOF-RPL and *SecTrust*-RPL shown in Figure 7.36, *SecTrust*-RPL displayed a better throughput performance over MRHOF-RPL. MRHOF-RPL consistently lagged behind *SecTrust*-RPL in throughput performance during the testbed trials. *SecTrust*-RPL maintained 4 – 6.5 kbps throughput range, while MRHOF-RPL had a range of 1 – 4 kbps.

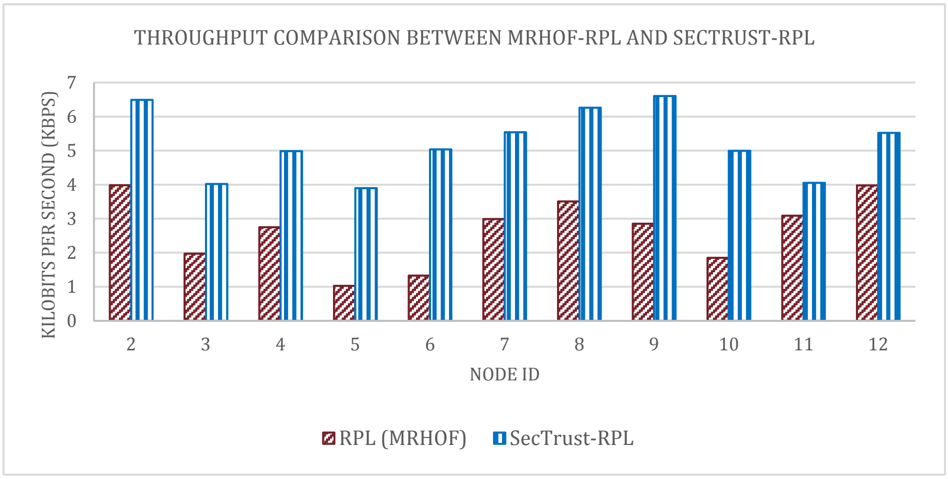


Figure 7.36: Network throughput performance comparison between *SecTrust*-RPL and MRHOF-RPL during the testbed experiment

Figure 7.37 shows the packet loss rates between MRHOF-RPL and *SecTrust*-RPL. *SecTrust*-RPL maintained packet loss rates between 15% – 27.7%, MRHOF-RPL on the other hand, had packet loss rates of 60% – 72.7%. It can be summarised therefore, from Figure 7.37, that the Selective Forwarding attacks of malicious motes 13 and 14 had a more significant impact against the MRHOF-RPL network than is the case for the network using the *SecTrust*-RPL protocol. This further shows *SecTrust*-RPL as having better network performance over MRHOF-RPL under the Selective Forwarding attacks.

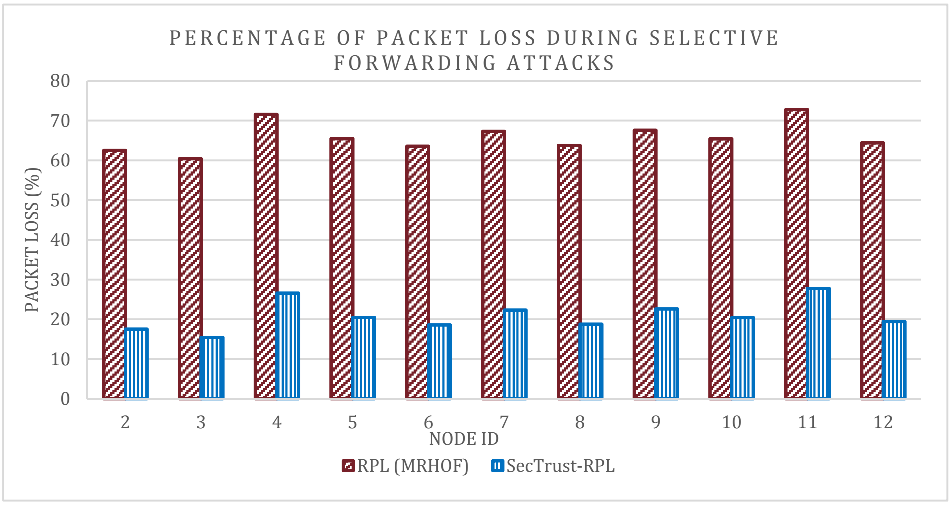


Figure 7.37: Percentage of packet loss comparison between *SecTrust*-RPL and MRHOF-RPL under Selective Forwarding attacks during the testbed experiment

The packet transmission delay between MRHOF-RPL and *SecTrust*-RPL in Figure 7.38, displays the latency in the transmission of data packets from sender motes to the sink mote under the Selective Forwarding attack. As observed from the Figure, *SecTrust*-RPL maintained superior performance in the transmission delay over MRHOF-RPL. *SecTrust*-RPL overall, recorded much less transmission delay times with the highest being mote 4 which recorded 47 ms. MRHOF-RPL however, recorded much higher transmission delay time with all motes recording transmission delay times above 50 ms.

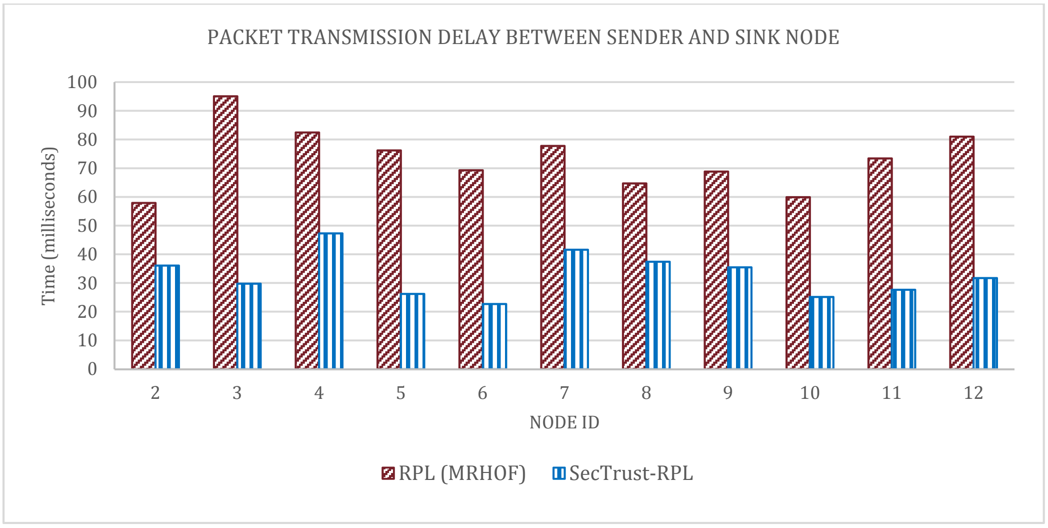


Figure 7.38: Packet transmission delay of *SecTrust*-RPL and MRHOF-RPL under Selective Forwarding attacks during the testbed experiment

7.6.4 *SecTrust*-RPL vs MRHOF-RPL under Sybil Attacks

This section presents the experimental results of the testbed trials between *SecTrust*-RPL and MRHOF-RPL protocols under Sybil attacks using the deployed XM1000 motes. During the testbed experiment, the network topology was captured and viewed using Contiki's sensor data collection. Figure 7.39 shows *SecTrust*-RPL protocol's detection and isolation of malicious mote 13 from its route topology formation while malicious mote 14 succeeded in attracting mote 10 to select it as its parent. However, in Figure 7.40, which is the network topology formation of MRHOF-RPL, the MRHOF-RPL protocol could not mitigate the effect of malicious activities of the Sybil attacking motes (13 and 14) in the network. Motes 3, 6 and 8 were attracted to malicious Sybil mote 13 while motes 2, 7 and 12 were drawn to malicious Sybil mote 14. The rest of the motes however, were connected to the sink mote. Due to the segmented network topology in Figure 7.40, motes 2, 3, 6, 7, 8 and 12 are unable to transmit their packets to the sink mote (mote 1) during the period of the experiment.

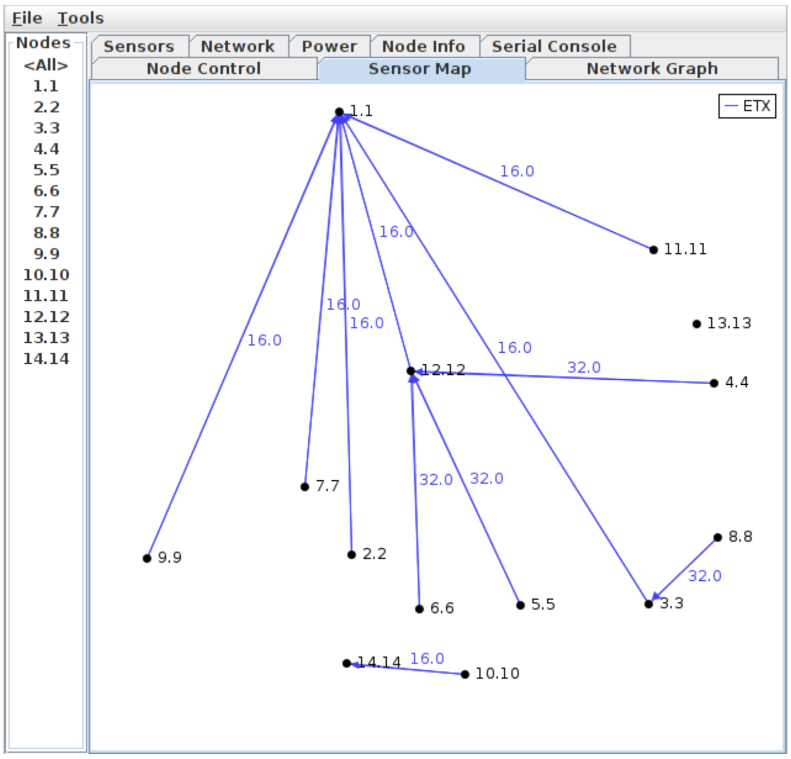


Figure 7.39: *SecTrust*-RPL route topology setup and isolation of Sybil attacks during the testbed experiment with XM1000 motes

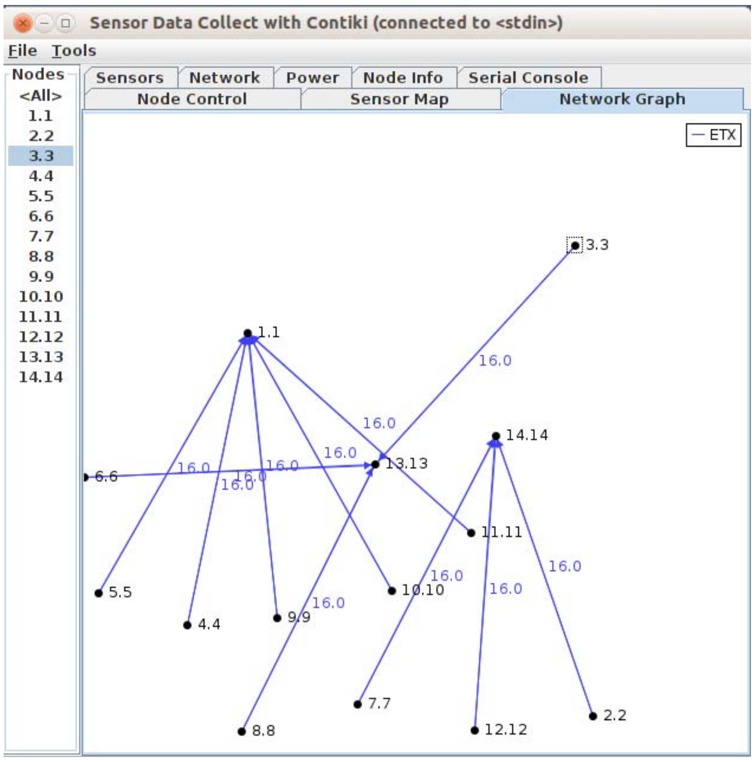


Figure 7.40: MRHOF-RPL route topology segmentation under Sybil attacks during the testbed experiment with XM1000 motes

1. Detection and Isolation of Attack Nodes

The testbed experiment in Figure 7.41 shows the detection of Sybil attacks over a 60-minute simulation period of testing with Sybil activities predominant within the first 25 minutes of the simulation testing time. The Sybil attack motes (13 and 14) masqueraded themselves thus, creating an illusion of multiple identities within a network. However, *SecTrust*-RPL could identify and isolate the Sybil motes from the RPL network in most cases. The 30th to 60th minutes witnessed moderate Sybil attacks with *SecTrust*-RPL detecting and mitigating these attacks. MRHOF-RPL however, could not detect nor defend against the masquerading malicious behaviour of the Sybil motes all through the testbed trials.

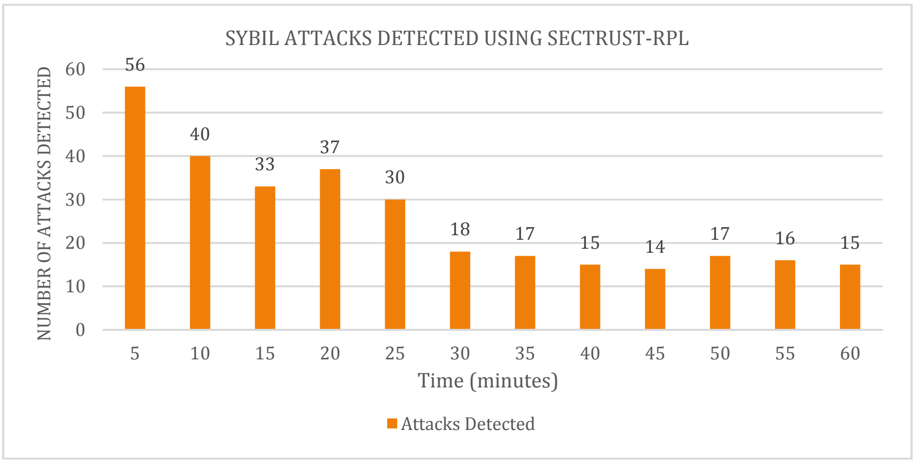


Figure 7.41: Sybil attacks detection and mitigation by *SecTrust*-RPL during testbed experiment

The graph in Figure 7.42 shows MRHOF-RPL had higher mote Rank changes which ranged between 34 and 99 however, *SecTrust*-RPL experienced mote Rank change range of 0 to 36. In fact, under *SecTrust*-RPL, motes 5, 10, 11 and 12 had 0 mote Rank changes; an indication that these motes adequately isolated Sybil attacks and maintained consistent connectivity with only trusted parent motes. MRHOF-RPL however, experienced several Sybil attacks, which influenced the topology of its network and hence, the frequent mote Rank changes experienced by the motes in the network (see Figure 7.42).

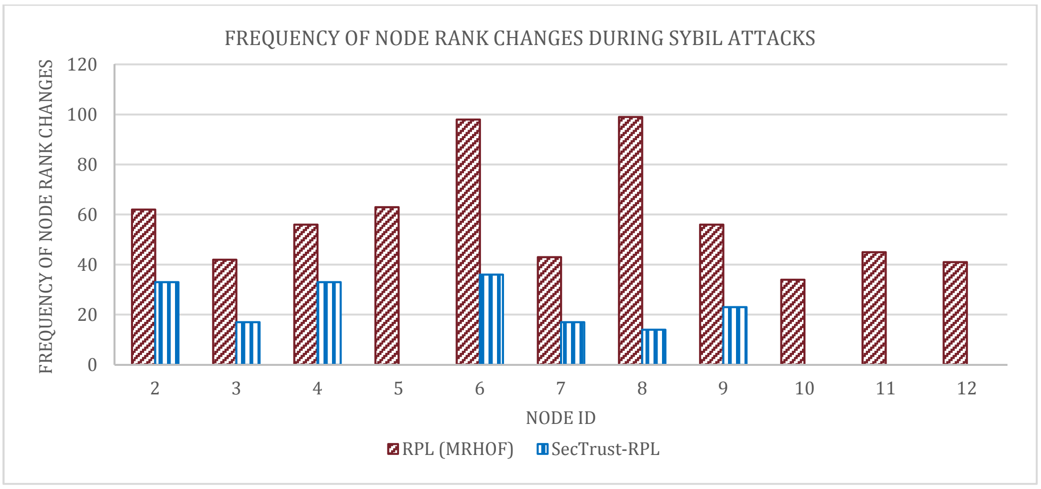


Figure 7.42: Comparison of frequency of node Rank changes under Sybil attacks during the testbed experiment

2. Network Performance Measures

Figure 7.43 below presents the throughput comparison between MRHOF-RPL and *SecTrust*-RPL; *SecTrust*-RPL displayed better throughput performance over MRHOF-RPL attaining throughput as high as 8.9 kbps and maintaining a throughput range of 5 – 8.9 kbps. The motes in MRHOF-RPL observed much lower throughput performance (0.6 – 4 kbps) with some motes having throughput of less than 1 kbps (motes 9 and 11).

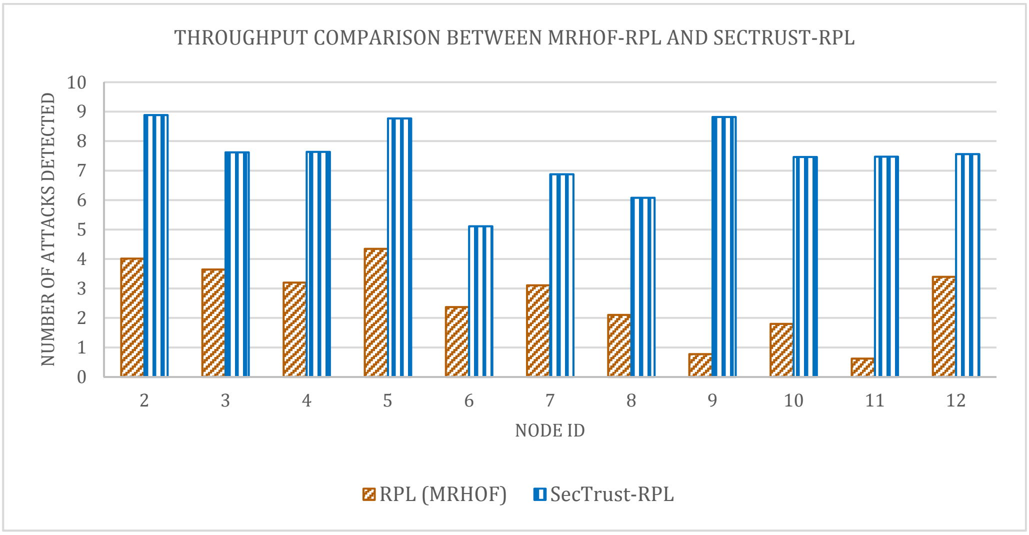


Figure 7.43: Network throughput performance comparison between *SecTrust*-RPL and MRHOF-RPL during the testbed experiment

Figure 7.44 shows the packet loss rates between MRHOF-RPL and *SecTrust*-RPL. The packet loss rate experienced by the MRHOF-RPL protocol exceeded that experienced by the *SecTrust*-RPL protocol. The MRHOF-RPL protocol experienced packet loss rates as high as 73% on mote 11 while the maximum experienced by *SecTrust*-RPL protocol was less than 28% also on mote 11. Given the packet loss rates experienced by the MRHOF-RPL protocol, it is clear that the Sybil attacks perpetrated by malicious motes 13 and 14 had significant scope of operational influence on MRHOF-RPL protocol's network, thereby affecting the security and network performance of the MRHOF-RPL network over and above the *SecTrust*-RPL protocol's network.

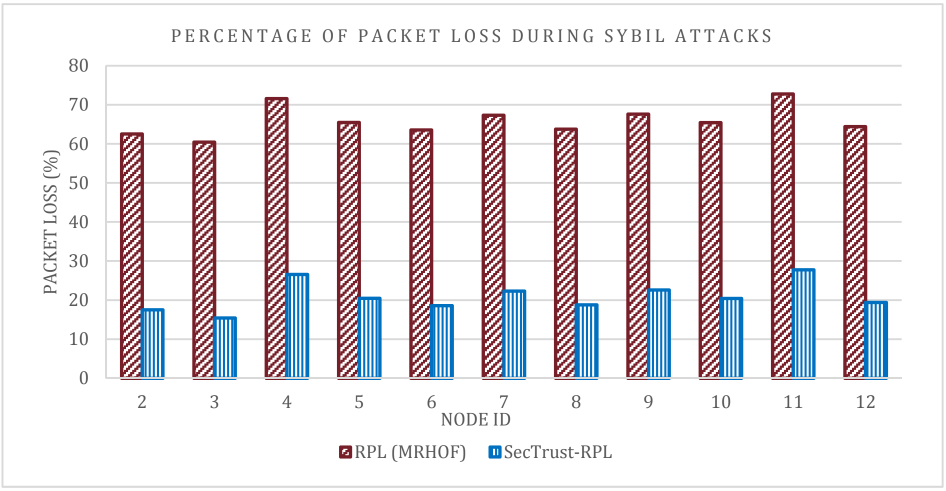


Figure 7.44: Percentage of packet loss comparison between *SecTrust*-RPL and MRHOF-RPL under Sybil attacks during the testbed experiment

The latency in the transmission of data packets from sender motes to the sink mote is shown in Figure 7.45 between MRHOF-RPL and *SecTrust*-RPL protocols. The transmission delay between the two protocols shows *SecTrust*-RPL having better transmission delay rate. A transmission delay range of 21 - 52 ms can be observed under *SecTrust*-RPL while MRHOF-RPL recorded a transmission delay range of 53 – 91 ms.

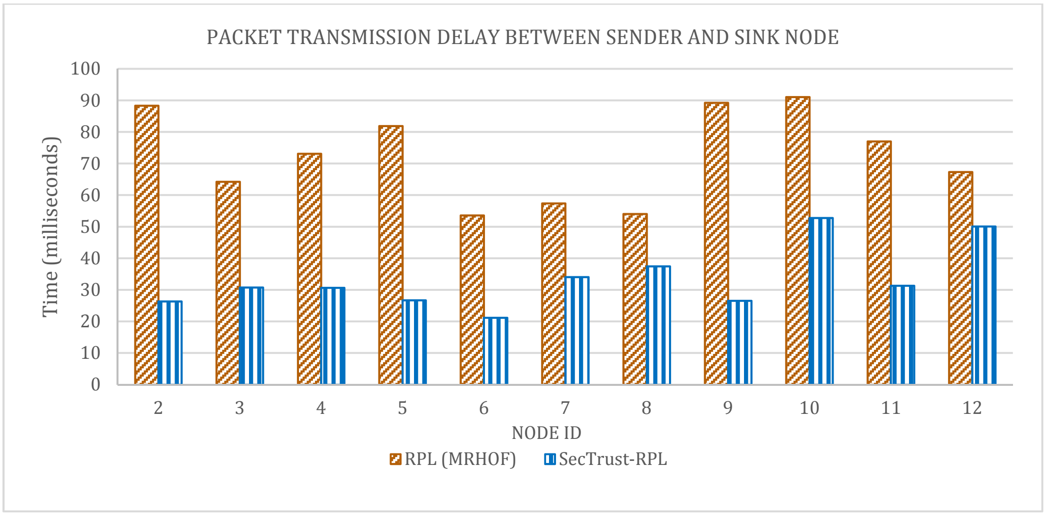


Figure 7.45: Packet transmission delay of *SecTrust*-RPL and MRHOF-RPL under Sybil attacks during the testbed experiment

7.7 Summary

This Chapter has presented a live testbed study of the *SecTrust* system, which was applied to the RPL routing protocol. A comparison of its performance with the IETF RPL standards (MRHOF) was similarly performed. In summary, this Chapter has presented testbed experiments of the performance of *SecTrust* system embedded with the RPL protocol while comparing its performance against the IETF version implemented in ContikiRPL. The experiments were performed under a battery of attacks with similar settings as those specified in Chapter 6. This testbed experiment serves as a validation of the simulation study carried out in Chapter 6. The experimental data gathered were further analysed to determine the efficacy of the *SecTrust* system in mitigating the attacks proposed in this study as compared with the IETF standard presented in ContikiRPL. The performance of *SecTrust* in RPL (*SecTrust*-RPL) was found to offer better routing security to IoT networks in comparison with the standard RPL IETF protocol as specified using the *Minimum Rank with Hysteresis Objective Function*.

In Chapter 8, statistical evaluation is carried out to measure the levels of confidence of simulated and experimented data. This is necessary not only to establish the confidence levels of data simulated and experimented data, but to establish the strong correlation among the data set generated.

8 Statistical Evaluation and the Implications of a Trust-based Design Framework for IoT

The deployment of IoT-enabled devices is growing at an unprecedented rate and statistics below confirm it. In a study conducted on behalf of Cisco by Forrester Research [218], 336 organisations with staff strength of over 1,000 were interviewed, and 29% to 38% of the organisations surveyed had deployed or were intending to implement IoT functionalities in their transport systems, healthcare systems, industrial applications, environmental monitoring systems, energy management systems, or the management of their infrastructure. From the total firms surveyed (336), 37% to 47% of them had plans to implement the IoT functionalities within the next 5 years. In addition, 43% of the firms believed that the deployment of IoT functionalities in their firms will help cut operational expenses while improving safety, managing risks and providing a better customer experience.

A fundamental reason for the growth of IoT is its ability to provide communications in a heterogeneous manner among various connected devices; from personal computers, routers to emerging devices including smart meters, sensors, smart phones and identification readers. It is obvious that several technologies are aiding the growth and communication of IoT. In smart objects, technologies like 6LoWPAN is used in Sensor Networks (IEEE 802.15.4), Bluetooth Low Energy (IEEE 802.15.1) deployed in Wireless Personal Area Networks, Wi-Fi Low Power (IEEE 802.11) utilized in Wireless Local Area Networks, and the Long-Term Evolution – Advanced (LTE-A) has been proposed for machine-to-machine communications. Of equal importance is the Near Field Communication (NFC) technologies used for the formation of peer-to-peer (P2P) networks and for identifying digital resources in personal devices like smart phones.

There is a global preponderance of Internet-connected smart devices, and IoT is transforming industries and creating new opportunities to use business models that are convergent on monetizing the upsurge of services ignited by the IoT transformation. This obviously has created a massive number of devices and operations in both scope and scale. If this growth is to be sustained, securing those devices becomes extremely critical. Vulnerability is no longer limited to the computing devices located in networks, but it can span to the mundane tasks that have been assigned to IoT systems; for example, misbehaviour of a street lighting system in a smart city, to faults affecting the energy or access control systems in residences and offices.

This study progresses by presenting the statistical evaluation to determine the levels of confidence of simulated and testbed data during simulation and testbed trials.

8.1 Statistical Evaluation of the Validity of the Simulation study and the Testbed Experiments

A statistical survey is a valuable measurement tool whereby a sample is chosen and the sample estimate can be used as a generalisation for the larger population. The validity of any statistical survey is measured by the randomness of the population sample. The simulation study and testbed experiment of any sort will be of limited value if the limits within which the simulation and experiments are performed cannot be guaranteed at a certain level of confidence. The possibility exists for results obtained during a study to be skewed due to several reasons without the researcher being aware of this. In consequence to this, the simulation study and testbed experiment conducted in this research computes the confidence intervals (CI) and the margins of error with the aim of determining the accuracy of estimates.

In statistical studies, the accuracy of a sample estimate is measured using the statistical survey's confidence interval and the margin of error. The CI provides a degree of the accuracy (or uncertainty) of study or experimented results in order to provide inferences on sampled population. CI is the given range of some values, which probabilistically holds an unknown population estimate. It follows that if data is randomly drawn from the set of data gathered during a simulation or testbed experiment, a given proportion of the confidence intervals will have the mean of the population. The confidence interval is expressed as the percentage of the mean population estimate. A fundamental benefit of using confidence intervals is in the display of variability of estimates in the population sample, that is, the width of the confidence interval. Furthermore, a confidence interval gives a true representation of estimates from a sample size. This implies that, small sample size estimates will give wider confidence limits hence, a limited confidence of the validity of study estimates. But selecting a larger sample size, will give narrow confidence limits implying a higher level of confidence of the validity of estimates.

Finally, random sampling was used in the selection of the data analysed discussed in this thesis.

8.1.1 Z-Confidence Intervals for Means and Margin of Error

Confidence intervals have proven to provide accurate estimates of the population sample since they provide intervals that are contained in the sample set. The accuracy of Confidence intervals is seen in composition of the precise point estimate and a margin of error around the point estimates. The margin of error is a measure of the level of uncertainty surrounding the population sample estimate [219]. From the foregoing, confidence intervals could be used to assess the precision of the sample estimate derived from the simulation and testbed studies carried out during this research study. For any specific sample estimate, a narrower confidence interval advocates a more accurate estimate of the population sample over a wider confidence interval [219].

The Z-Confidence Interval for statistical Means is applied to the Mean of data derived from the normal distribution of a sample estimate. Figure 8.1 shows a representation of confidence interval as a normal distribution graph. The representation which includes the shaded green area beneath the curve is bordered by the critical values on both sides. The Figure also illustrates a distribution based on the mean of the population given the sample's size, and the population's standard deviation.

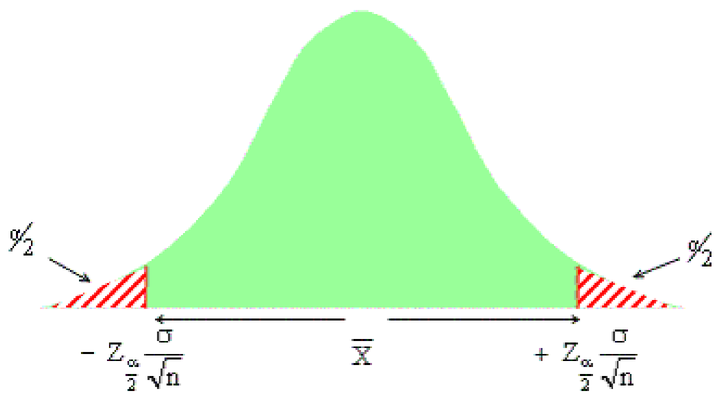


Figure 8.1: Confidence Interval as a normal distribution of a mean based population estimate

The simulation and testbed experiments undertaken in this study, assumes a normal distribution having large sample sizes, means and standard deviations. In essence, the assumption is made that the data set gathered during simulation runs and testbed experiments are all distributed like the bell curve depicted in Figure 8.1. In addition, this study assumes a 95% confidence interval. A 95 percent confidence level implies that 95% of the data set will contain the population mean while only 5 percent of the data set will probably not consist of the population mean. Also, from Figure 8.1, 2.5% of the time, the interval will be much less than the value of the population means, and 2.5% of the time, the interval will be much higher than the value of the population means.

The confidence coefficient (or degree of confidence) with the terminal points of the interval commonly referred to as Lower Confidence Range and Upper Confidence Range is shown in Equation 8.1.

$$\overset{\text{Lower Confidence Range}}{\overline{X} - Z_{\alpha/2}\frac{\sigma}{\sqrt{n}}} \le \mu \le \overset{\text{Upper Confidence Range}}{\overline{X} + Z_{\alpha/2}\frac{\sigma}{\sqrt{n}}} \qquad (8.1)$$

where: C.I. = $\overline{X} \pm Z_{\alpha/2}\frac{\sigma}{\sqrt{n}}$

σ = the standard deviation of the population.

$Z_{\alpha/2}$ = the Z value for the desired confidence level alpha (obtained from an area under Normal Curve table).

From equation 8.1, $Z_{\alpha/2}\frac{\sigma}{\sqrt{n}}$ is referred to as the margin of error, which is the quantum of random sampling error observable in the gathered dataset and $Z_{\alpha/2}$ is known as the critical value. Since this study assumes a 95% Confidence Interval, the $Z_{\alpha/2}$ of 95% (0.95) in the Z-table gives a value of 1.96 thus, the critical value is 1.96. This gives a margin of error of $1.96\frac{\sigma}{\sqrt{n}}$. The sections following present the computed confidence intervals and margin of error for the simulation study and testbed experiments undertaken. In each section, the summary of results is presented as table summaries.

8.1.2 Confidence Intervals and Margin of Error under Blackhole Attacks

The table presented in 8.1 gives a statistical summary of the frequency of node Rank changes of MRHOF-RPL and *SecTrust*-RPL during simulation and testbed experiments. A comparison of the lower and upper confidence ranges of MRHOF-RPL during simulation (387.89 - 388.18) and testbed experiments (77.18 - 77.55) show narrow ranges, a pointer to the validity of the simulation and testbed data estimates gathered. Similarly, the percentage margin of error for the MRHOF-RPL protocol during simulation (2.37%) and testbed investigation (2.90%) also reflect a negligible sampling error margin for the estimates generated during both tests.

The *SecTrust*-RPL confidence interval ranges during simulation (48.45 - 48.56) and testbed trials (38.42 - 38.66) similarly show narrow confidence values; an indication of a high confidence in sampled data estimates gathered during both tests. Likewise, the *SecTrust*-RPL's percentage margin of error during simulation (0.84) and testbed trial (1.87) both reveal an insignificant error sampling of data.

Furthermore, Table 8.2 shows the statistical summary of attacks detected by *SecTrust*-RPL during Blackhole attacks. The *SecTrust*-RPL's narrow confidence interval ranges of 296.72 - 296.95 and 18.31 - 18.35 gathered during simulation and testbed trial respectively, indicate a high confidence of the validity of simulation data estimates. Equally, the percentage margin of error values of 1.83% and 0.32% gathered during simulation and testbed investigations respectively, reflect a trivial sampling error margin.

Table 8.1: Statistical summary of the frequency of node Rank changes in MRHOF-RPL and *SecTrust*-RPL during Blackhole attacks

Description	**MRHOF-RPL (Simulation)**	***SecTrust*-RPL (Simulation)**	**MRHOF-RPL (Testbed)**	***SecTrust*-RPL (Testbed)**
Standard Deviation	237.60	29.70	84.47	38.43
Number of Nodes	26	26	11	11
Mean	388.04	48.51	77.36	38.55
Confidence Interval	0.15	0.05	0.18	0.12
Lower Confidence Range	387.89	48.45	77.18	38.43
Upper Confidence Range	388.19	48.56	77.55	38.66
Margin of Error	2.37	0.846	2.90	1.87

Table 8.2: Statistical summary of attacks detected by *SecTrust*-RPL during Blackhole attacks

Description	*SecTrust*-RPL (Simulation)	*SecTrust*-RPL (Testbed)
Standard Deviation	109.39	4.80
Time Frequency Range	3562	220
Mean	296.83	18.33
Confidence Interval	0.12	0.02
Lower Conf. Range	296.72	18.31
Upper Conf. Range	296.95	18.35
Margin of Error	1.83	0.32

Table 8.3 shows the statistical summary of the throughput rate of MRHOF-RPL during simulation and testbed investigations. The lower and upper confidence ranges of MRHOF-RPL during simulation (1.15 - 1.18) and testbed (1.51 - 1.55) reveal high confidence levels of data estimates generated during both studies. The percentage margins of error of MRHOF-RPL during both simulation (0.22%) and testbed trials (0.27%) also show insignificant levels of sampling error. Correspondingly, the *SecTrust*-RPL protocol's confidence interval ranges reported for the simulation study (2.54 - 2.55) and testbed trials (4.40 - 4.48) also point significantly to the high confidence level of the study's presented data. The margins of error of *SecTrust*-RPL protocols under both study

conditions namely; simulation study (0.08%) and testbed investigations (0.33%) also indicate a low sampling error level, which attests to the validity of the study data reported in this research.

Table 8.4 presents the statistical summary of the packet loss rate of MRHOF-RPL and *SecTrust*-RPL during simulation and testbed investigations under Blackhole attacks. The confidence interval ranges (lower and upper) of MRHOF-RPL protocol during simulation (82.92 - 83.34) and testbed trials (65.84 - 65.97) reflect a high confidence of data estimates of the simulation study. In addition, *SecTrust*-RPL protocol's interval ranges for simulation (22.20 - 22.30) and testbed trial (20.84 - 20.97) similarly show high degree of confidence of the study data. Finally, the margin of error of MRHOF-RPL protocols for both simulation and testbed tests, which are 0.37% and 0.13%, also show minimal data sampling errors of generated simulated and testbed data. *SecTrust*-RPL's margins of error under simulation (0.17%) and testbed trials (0.23%) equally confirm to the insignificant data sampling errors.

Table 8.3: Statistical summary of throughput rate of MRHOF-RPL and *SecTrust*-RPL during Blackhole attacks

Description	**MRHOF-RPL (Simulation)**	***SecTrust*-RPL (Simulation)**	**MRHOF-RPL (Testbed)**	***SecTrust*-RPL (Testbed)**
Standard Deviation	1.20	0.66	1.11	2.28
Number of Nodes	26	26	11	11
Mean	1.17	2.55	1.53	4.44
Confidence Interval	0.05	0.01	0.02	0.04
Lower Confidence Range	1.15	2.54	1.51	4.40
Upper Confidence Range	1.18	2.55	1.55	4.48
Margin of Error	0.22	0.08	0.27	0.33

Table 8.4: Statistical summary of packet loss rates of MRHOF-RPL and *SecTrust*-RPL during Blackhole attacks

Description	MRHOF-RPL (Simulation)	*SecTrust*-RPL (Simulation)	MRHOF-RPL (Testbed)	*SecTrust*-RPL (Testbed)
Standard Deviation	17.04	3.97	3.55	3.55
Number of Nodes	26	26	11	11
Mean	83.13	22.25	65.90	20.90
Confidence Interval	0.21	0.05	0.07	0.07
Lower Confidence Range	82.92	22.20	65.84	20.84
Upper Confidence Range	83.33	22.30	65.97	20.97
Margin of Error	0.37	0.17	0.13	0.24

8.1.3 Confidence Intervals and Margin of Error under Rank Attacks

The Table 8.5 presents a statistical summary of the frequency of node Rank changes of MRHOF-RPL and *SecTrust*-RPL during simulation and testbed experiments. The respective simulation and testbed trials confidence interval ranges of MRHOF-RPL (352.60 - 353.02 and 76.95 - 77.23) and *SecTrust*-RPL (53.03 - 53.21 and 45.88 - 46.12) indicate good levels of confidence. Similarly, the percentage margins of error for MRHOF-RPL protocol (3.35%) and testbed investigation (2.29) also reflect a good sampling error margin for both tests. Likewise, the *SecTrust*-RPL margin of error during simulation (1.43%) and testbed trial (1.89%) show trivial sampling error estimates of data analysed.

The statistical summary of Rank attacks detected is presented in Table 8.6. The *SecTrust*-RPL's narrow confidence interval ranges of 36.51 - 36.65 and 27.05 - 27.12 reported during simulation and testbed trials respectively, show a high confidence of the sampled data. Similarly, the margin of error values of 1.11% and 0.62% for simulation and testbed trials also show insignificant sampling error margin.

Table 8.5: Statistical summary of the frequency of node Rank changes in MRHOF-RPL and *SecTrust*-RPL during Rank attacks

Description	**MRHOF-RPL (Simulation)**	***SecTrust*-RPL (Simulation)**	**MRHOF-RPL (Testbed)**	***SecTrust*-RPL (Testbed)**
Standard Deviation	321.10	52.96	66.63	42.51
Number of Nodes	26	26	11	11
Mean	352.81	53.12	77.09	46
Confidence Interval	0.21	0.10	0.14	0.12
Lower Confidence Range	352.60	53.03	76.95	45.88
Upper Confidence Range	353.02	53.21	77.23	46.12
Margin of Error	3.35	1.43	2.29	1.89

Table 8.6: Statistical summary of attacks detected by *SecTrust*-RPL during Rank attacks

Description	*SecTrust*-RPL (Simulation)	*SecTrust*-RPL (Testbed)
Standard Deviation	23.19	11.09
Time Frequency Range	439	325
Average	36.58	27.08
Confidence Interval	0.07	0.04
Lower Conf. Range	36.51	27.04
Upper Conf. Range	36.65	27.12
Margin of Error	1.11	0.62

Table 8.7 gives a statistical summary of the throughput rate of MRHOF-RPL during simulation and testbed investigations. The narrow confidence interval ranges of MRHOF-RPL during simulation (1.25 - 1.27) and testbed (3.08 - 3.13) show high confidence levels of data estimates. Correspondingly, the percentage margins of error of MRHOF-RPL during both simulation (0.17%) and testbed trials (0.25%) also show low levels of data sampling errors of simulated and testbed data. Likewise, the *SecTrust*-RPL protocols' confidence intervals reported for the simulation study (4.31 - 4.34) and testbed trials (4.30 - 4.40) display a high significant confidence level of the study data presented. The margins of error reported for *SecTrust*-RPL protocols under simulation (0.11%) and testbed trials (0.37%) both show low level sampling errors attesting to the validity of study data reported in this research study.

Table 8.8 gives a statistical summary of the packet loss rate of MRHOF-RPL and *SecTrust*-RPL during simulation and testbed investigations while under Rank attacks. The confidence interval ranges of MRHOF-RPL protocol during simulation (76.73 - 77.12) and testbed trials (65.84 - 65.97) reflect a high confidence of data estimates of the simulation study while the *SecTrust*-RPL protocol under simulation (22.20 - 22.30) and testbed trials (20.84 - 20.97) similarly show narrow confidence intervals, which imply a high degree of confidence of study data presented. Finally, the margins of error of MRHOF-RPL protocols for both simulation and testbed tests, which are 0.35% and 0.13%, reportedly show minimal sampling errors while the testbed margins of error of *SecTrust*-RPL under simulation (0.17%) and testbed trials (0.23%) also indicate insignificant data sampling errors.

Table 8.7: Statistical summary of throughput rate of MRHOF-RPL and *SecTrust*-RPL during Rank attacks

Description	**MRHOF-RPL (Simulation)**	***SecTrust*-RPL (Simulation)**	**MRHOF-RPL (Testbed)**	***SecTrust*-RPL (Simulation)**
Standard Deviation	0.95	1.17	1.46	2.58
Number of Nodes	26	26	11	11
Mean	1.26	4.32	3.10	4.35
Confidence Interval	0.01	0.01	0.03	0.05
Lower Confidence Range	1.25	4.31	3.08	4.30
Upper Confidence Range	1.27	4.33	3.13	4.40
Margin of Error	0.17	0.11	0.25	0.37

Table 8.8: Statistical summary of packet loss rates of MRHOF-RPL and *SecTrust*-RPL during Rank attacks

Description	**MRHOF-RPL (Simulation)**	***SecTrust*-RPL (Simulation)**	**MRHOF-RPL (Testbed)**	***SecTrust*-RPL (Testbed)**
Standard Deviation	15.71657813	3.970552592	3.547734697	3.547734697
Number of Nodes	26	26	11	11
Mean	76.92	22.25	65.90	20.90
Confidence Interval	0.19	0.05	0.07	0.07
Lower Confidence Range	76.73	22.20	65.84	20.84
Upper Confidence Range	77.12	22.30	65.97	20.97
Margin of Error	0.35	0.17	0.13	0.23

8.1.4 Confidence Intervals and Margin of Error under Selective Forwarding Attacks

Table 8.9 presents the statistical summary of the frequency of node Rank changes of MRHOF-RPL and *SecTrust*-RPL during simulation and testbed experiments respectively while under Selective Forwarding attacks. The range of the confidence interval of MRHOF-RPL during simulation (949.79 - 950.05) and testbed experiments (315.89 - 316.29) indicated a good level of confidence. The lower and upper confidence ranges for *SecTrust*-RPL under both test cases (simulation and testbed) show values much less than 1.0 (315.03 - 315.35 and 96.05 - 96.31), an indication of a good confidence levels and high precision of studied data. In like manner, the percentage margins of error for MRHOF-RPL protocols during simulation (2.07%) and testbed investigation (3.16%) also reflect a good sampling error margin. In like manner, the *SecTrust*-RPL margins of error during simulation (2.54%) and testbed trial (2.06%) both reveal a trivial sampling error estimates from data analysed.

The Selective Forwarding attacks detection shown Table 8.10, shows establishes the authenticity of both simulation and testbed data. The *SecTrust*-RPL's confidence interval value ranges of 180.45 - 181.05 and 64.45 - 64.55 during simulation and testbed trials respectively, specify a high confidence of the validity of the simulation and experimental data estimates. Similarly, the margin of error values of 4.84% and 0.85% for both simulation and testbed trials reported also show good sampling error margins.

Table 8.9: Statistical summary of the frequency of node Rank changes in MRHOF-RPL and *SecTrust*-RPL during Selective Forwarding attacks

Description	**MRHOF-RPL (Simulation)**	***SecTrust*-RPL (Simulation)**	**MRHOF-RPL (Testbed)**	***SecTrust*-RPL (Testbed)**
Standard Deviation	325.48	229.99	186.11	66.87
Number of Samples	24698	8195	3477	1058
Mean	949.92	315.19	316.09	96.18
Confidence Interval	0.13	0.16	0.20	0.13
Lower Confidence Range	949.79	315.03	315.89	96.05
Upper Confidence Range	950.05	315.35	316.29	96.31
Margin of Error	2.07	2.54	3.17	2.06

Table 8.10: Statistical summary of attacks detected by *SecTrust*-RPL during Selective Forwarding attacks

Description	*SecTrust*-RPL (Simulation)	*SecTrust*-RPL (Testbed)
Standard Deviation	225.77	23.77
Time Frequency Range	2169	774
Average	180.75	64.5
Confidence Interval	0.30	0.05
Lower Conf. Range	180.45	64.45
Upper Conf. Range	181.05	64.55
Margin of Error	4.85	0.85

The statistical summary of the throughput rate of MRHOF-RPL and *SecTrust*-RPL protocols are shown in Table 8.11 during simulation and testbed investigations. The confidence interval ranges of MRHOF-RPL and *SecTrust*-RPL during simulation and testbed signify good confidence levels of data estimates investigated. Similarly, the protocols' (MRHOF-RPL and *SecTrust*-RPL) margins of error as shown in the table also testify to the validity of simulated and testbed data.

The statistical summary of the packet loss rates of MRHOF-RPL and *SecTrust*-RPL under Selective Forwarding attacks is presented in Table 8.12. MRHOF-RPL protocol showed a narrow confidence interval ranges of 75.50 - 75.87 and 65.84 - 65.97 during the simulation and testbed trials. Also, the reported margins of error for MRHOF-RPL under both testing conditions (simulation, 0.34% and testbed trials, 0.13%) are indicative of very low sampling error of data. In like manner, the confidence interval ranges of *SecTrust*-RPL protocol under both testing conditions (22.20 - 22.30 and 20.84 - 20.97) reveal a high confidence level of data estimates, while the margins of error for both simulation and testbed trials (0.165 and 0.234), show minimal sampling errors of simulated and testbed data.

Table 8.11: Statistical summary of throughput rate of MRHOF-RPL and *SecTrust*-RPL during Selective Forwarding attacks

Description	**MRHOF-RPL (Simulation)**	***SecTrust*-RPL (Simulation)**	**MRHOF-RPL (Testbed)**	***SecTrust*-RPL (Testbed)**
Standard Deviation	0.97	0.75	0.61	2.48
Number of Nodes	26	26	11	11
Mean	1.44	3.01	3.39	3.67
Confidence Interval	0.01	0.01	0.01	0.05
Lower Confidence Range	1.43	3.00	3.38	3.63
Upper Confidence Range	1.45	3.02	3.41	3.72
Margin of Error	0.16	0.09	0.10	0.39

Table 8.12: Statistical summary of packet loss rates of MRHOF-RPL and *SecTrust*-RPL during Selective Forwarding attacks

Description	MRHOF-RPL (Simulation)	*SecTrust*-RPL (Simulation)	MRHOF-RPL (Testbed)	*SecTrust*-RPL (Testbed)
Standard Deviation	15.10	3.97	3.55	3.55
Number of Nodes	26	26	11	11
Mean	75.67	22.25	65.90	20.90
Confidence Interval	0.19	0.05	0.07	0.07
Lower Confidence Range	75.50	22.20	65.84	20.84
Upper Confidence Range	75.87	22.30	65.97	20.97
Margin of Error	0.34	0.17	0.13	0.23

8.1.5 Confidence Intervals and Margin of Error under Sybil Attacks

Table 8.13 summarizes the confidence interval values of the frequency of node Rank changes of MRHOF-RPL and *SecTrust*-RPL during Sybil attacks for both the simulation and testbed trials. The confidence interval ranges for MRHOF-RPL under both testing conditions (295.69 - 296.15 and 52.99 - 53.19) show very narrow confidence result levels. In addition, the margins of error reported under both testing conditions for MRHOF-RPL (3.66 and 1.60) similarly reflect good sampling error margins. The *SecTrust*-RPL protocol displayed similar results of good confidence levels during simulation and testbed trials (111.17 - 111.37 and 32.96 - 33.23). The *SecTrust*-RPL margins of error during simulation (1.63%) and testbed trials (2.15%) both reveal a trivial sampling error estimates from data analysed.

Table 8.14 displays the statistical summary of attacks detected by *SecTrust*-RPL during Sybil attacks. *SecTrust*-RPL's further showed narrow confidence interval values of 132.24 - 132.43 and 25.63 - 25.700.10 under both testing conditions an indication of a high confidence of the validity of the studied data estimates. Similarly, the margins of error values of 1.60% and 0.54% further testify the reliability of both simulation and testbed data.

Table 8.13: Statistical summary of the frequency of node Rank changes in MRHOF-RPL and *SecTrust*-RPL during Sybil attacks

Description	**MRHOF-RPL (Simulation)**	***SecTrust*-RPL (Simulation)**	**MRHOF-RPL (Testbed)**	***SecTrust*-RPL (Testbed)**
Standard Deviation	321.14	87.67	38.60	40.99
Number of Samples	7694	2893	584	364
Mean	295.92	111.27	53.09	33.09
Confidence Interval	0.23	0.10	0.10	0.14
Lower Conf. Range	295.69	111.17	52.99	32.96
Upper Conf. Range	296.15	111.37	53.19	33.23
Margin of Error	3.66	1.63	1.60	2.15

Table 8.14: Statistical summary of attacks detected by *SecTrust*-RPL during Sybil attacks

Description	*SecTrust*-RPL (Simulation)	*SecTrust*-RPL (Testbed)
Standard Deviation	63.77	9.48
Time Frequency Range	1588	308
Mean	132.33	25.67
Confidence Interval	0.10	0.03
Lower Conf. Range	132.23	25.63
Upper Conf. Range	132.43	25.70
Margin of Error	1.60	0.54

The result presented in Table 8.15 is a statistical summary of the throughput rate of MRHOF-RPL and *SecTrust*-RPL protocols during simulation and testbed investigations. The confidence interval ranges of MRHOF-RPL (1.37 - 1.40 and 3.19 - 3.25) during simulation and testbed trials signify high confidence levels of data estimates investigated. Similarly, the margins of error reported stood at 0.188% for simulation, and 0.233% for testbed trials. The reported *SecTrust*-RPL confidence interval ranges (1.99 - 2.02 and 5.79 - 5.91) conducted under both testing conditions show high significant confidence level of the study data presented. Correspondingly, the margins of error reported for the *SecTrust*-RPL protocol under simulation (0.18%) and testbed trials (0.38%) both show low level sampling errors; a sign to the validity of simulation and experimental data reported during the research study.

The statistical summary of the packet loss rates of MRHOF-RPL and *SecTrust*-RPL during simulation and testbed trials under the Selective Forwarding attacks is shown in Table 8.16. MRHOF-RPL protocol showed a confidence interval ranges of 77.71 - 78.12 and 65.84 - 65.97 under both testing scenarios, a very good level of confidence of the study data. Also, the reported margins of error for MRHOF-RPL under both simulation and testbed trials of 0.36% and 0.13% respectively, are indicative of the very low sampling error.

A look at the confidence interval ranges of *SecTrust*-RPL protocol during simulation (22.20 - 22.30) and testbed trials (20.84 - 20.97) reflect a high confidence of data estimates. The margins of error of *SecTrust*-RPL protocols for both simulation and testbed reported were 0.165 and 0.234 respectively. A negligible sampling error of data analysed.

Table 8.15: Statistical summary of throughput rate of MRHOF-RPL and *SecTrust*-RPL during Sybil attacks

Description	**MRHOF-RPL (Simulation)**	***SecTrust*-RPL (Simulation)**	**MRHOF-RPL (Testbed)**	***SecTrust*-RPL (Testbed)**
Standard Deviation	1.13	1.30	1.39	3.01
Number of Nodes	26	26	11	11
Mean	1.38	2.00	3.22	5.85
Confidence Interval	0.01	0.02	0.03	0.06
Lower Confidence Range	1.37	1.99	3.19	5.79
Upper Confidence Range	1.40	2.02	3.25	5.91
Margin of Error	0.19	0.18	0.23	0.38

Table 8.16: Statistical summary of packet loss rates of MRHOF-RPL and *SecTrust*-RPL during Sybil attacks

Description	MRHOF-RPL (Simulation)	*SecTrust*-RPL (Simulation)	MRHOF-RPL (Testbed)	*SecTrust*-RPL (Testbed)
Standard Deviation	16.31	3.97	3.55	3.55
Number of Nodes	26	26	11	11
Mean	77.92	22.25	65.90	20.90
Confidence Interval	0.20	0.05	0.07	0.07
Lower Confidence Range	77.71	22.20	65.84	20.84
Upper Confidence Range	78.12	22.30	65.97	20.97
Margin of Error	0.36	0.17	0.13	0.23

8.2 Framing the Thesis' Contribution: Security in Embedded IoT Devices

IoT security and in particular, secure routing in IoT systems although a topical issue in the research community, is however, far from being comprehensively addressed as no security standardization has being adopted [220]. Consequently, most organisations resort to adopting their own developed standards, which they implement in their products. This obviously lacks the possibility of being able to integrate and interoperate with the solutions among other IoT manufacturers. Thus, the usage of different security frameworks by different IoT manufacturing organisations has caused a lot of naivety among IoT manufacturers. Below are some key identifiable security problems in secure routing for IoT and a justification of how the proposed framework in this study addresses some of the security concerns in IoT.

1. *Limited knowledge in IoT Security*

   Because of the nature of IoT as an emerging technology, most IoT device manufacturers are unaware of how to secure IoT devices properly; and this is made worse given the way these devices operate across heterogeneous networks. Currently, many IoT products are being released and many firms want to cash in on the potential financial benefits, but these vendors typically lack the specific IoT security skills to design secure products and ensure the networks they operate within are also secure. This study has investigated attacks, both current and potential that could compromise an IoT network system. As a result, this study offers a framework that IoT developers could understudy and implement as a security standard in their IoT systems. Chapter 4 of this thesis provides a detailed methodology for the implementation of a secure trust-based framework for IoT.

2. *Heterogeneity of network media*

   IoT devices operate within heterogeneous network media transmitting packets from wired networks to wireless networks like, Bluetooth enabled networks, Near Field Communication system (NFC) and Wi-Fi. Implementing a security framework that spans these network media, is clearly daunting, and it is an uncharted territory especially for resource-constrained devices [220-223]. The trust-based framework proposed is a P2P distributed system, thus it evaluates its neighbour to ascertain its trustworthiness to deliver packets as expected - the heterogeneity of the medium regardless. Thus, the issue of heterogeneity of network media is addressed in *SecTrust*.

3. *Safest approach to IoT security*

   Although there is a consensus on the need for security in IoT, there is however, a lack of standardization for the adoption of a common standard. Most firms, due to this limitation refuse to extend their boundaries beyond what they can manage and maintain in a secure way. Although this is obviously understandable, it has however limited the potential IoT adoption for many of these firms. Any framework proposed must address the threats that IoT networks may face and offer IoT developers and manufacturers a safe platform for its implementation and integration with their products. The design of *SecTrust*, as presented in Chapter 4 and applied in Chapters 5, 6 and 7 offers a platform for this. *SecTrust* can be integrated into IoT systems. This was demonstrated in Chapter 6 where *SecTrust* was integrated into RPL routing protocol, and subsequently, deployed into XM1000 motes as proof-of-concept of its workability on a live IoT network.

4. *Trustworthiness of Nodes*

   Due to the distributed nature of some IoT sensor nodes, they can communicate in a P2P fashion. There is however, no certainty or guarantee in the security of the communication among the nodes in the network. Although, IoT developers implement key encryption within a limited or defined scope of IoT systems, the limitation of the public key encryption becomes apparent when the communication between these nodes must traverse other heterogeneous networks. The *SecTrust* system addresses this limitation since it can be mapped to routing protocols that can be universally understood by every node in a system regardless of the network medium. In addition, every node evaluates its neighbour to determine its trustworthy behaviour, so a node can determine if a neighbour node can be sufficiently relied upon for a secure network communication.

5. *Trust management*

   Many trust systems rely on the use of a centralised trust processing system to accurately predict and isolate untrusted nodes in the network. In an IoT network system where thousands and even millions of IoT sensor devices are online, the use of central repositories for trust management and evaluation becomes purely impracticable due to the massive number of nodes. For an efficient trust management and evaluation, nodes should have their own trust database of their neighbours, which they could use to detect and isolate untrusted nodes. It is true that untrusted nodes could exploit this opportunity to perpetrate their malicious activities. To defeat this behaviour, the overhearing ability of peer nodes will enable trusted nodes to monitor the packet forwarding behaviour of nodes amongst others. This will in turn facilitate an accurate trust evaluation of peer nodes. *SecTrust* embodies this trust technique in the evaluation and trust management of nodes. The trust management technique can then be integrated into the protocol of choice. This is achieved in Chapter 4 of *SecTrust* design. The efficacy of *SecTrust* was presented in Chapter 5, and the implementation of this property is evidenced in Chapters 6 and 7 where *SecTrust* was embedded into RPL protocol.

6. *Scalability*

   Since IoT comprises a massive number of devices communicating together, it follows that any framework that will provide security must be scalable to comprehensively offer good protection for the large number of communicating IoT devices. The issue of a secure and scalable framework for IoT systems is a clear research challenge. This research study shows promising results with regards to scalability. The tests conducted in Chapter 5 wherein, *SecTrust* was tested against other global trust systems while progressively varying the number of nodes in the network shows that *SecTrust* scales well and provides better security and network performance in comparison to other trust systems studied.

7. *Detection and Isolation of Attacks*

A secure framework should have certain defence mechanisms to detect and isolate threats from the network and recover from any probable attacks. Diverse attacks have been investigated in this study (Chapter 3) and a good security framework should have a strategic approach to detecting and isolating attacks such as reporting an intrusion or isolating suspicious nodes in a network. A key feature of *SecTrust* is its ability to detect and isolate various routing attacks. This was clearly addressed in the design of *SecTrust* (Chapter 4) and implemented in Chapters 5, 6 and 7 respectively.

8. *Mobility*
   The IoT nodes in a network depending on the sphere of application could be stationary, semi-mobile or fully mobile. In semi-mobile and fully mobile IoT systems, IoT devices can join and leave the network. This obviously changes the network topology from to time. This study assumed the application of IoT in a static sphere of application, such as a smart home environment, where all IoT devices are stationary. The mobility feature was not tested nor assessed. However, this is a future work of research that could be undertaken.

9. *Energy Consumption*
   Various approaches have been presented to reduce and optimize the consumption of energy in IoT systems, such as the use of better cryptographic approaches however, more research study is needed to address the high-energy utilisation in security processes that guarantee secure communications in IoT systems. Energy efficiency was not an objective in the development of *SecTrust*. However, this is proposed as a future work of study for further improvements.

According to a survey conducted by McAfee [224], most firms are generally sceptical on the best and effective way to approach security to IoT. Most security approaches taken by firms have either been cautious, lack-lustre or completely none existent. In the study conducted by McAfee [224], most interviewees opined that the traditional model of security does not fit when applied to IoT systems. This situation is further compounded with the realization of the lack of a global standard for adoption. Clearly, a new and effective approach will be required that will accommodate the humongous number of devices expected with IoT evolution.

To sum up, the emergence of IoT will have its fundamental success rooted in the design and development of security embedded in it right from the outset. Achieving a secure and well-connected IoT will require a fundamental reorientation from industrial designers, technologists, and researchers to birth successfully a "securely connected" IoT network of devices. The IoT revolution may continue to prove challenging to the global IoT community because of the lack of the adoption of a comprehensive secure IoT framework while many organisations will continue to experience challenges in IoT deployment and growth, since they have failed to recognize the value of enshrining security in the design of connected IoT products.

8.3 Trust-Based Framework for IoT: An Evolutionary Path for Secure IoT Communication

There is no doubt that the IoT ecosystem brings with it a plethora of security vulnerabilities that could be exploited by malicious actors desiring to manipulate the flow of data and information in a network of connected IoT systems. The manipulations and tampering could lead to data loss, privacy loss; and for most businesses, an interruption of business, loss of clients, loss of revenue, and litigation, amongst others.

Many researchers and indeed the media have focused on securing the devices, but security in IoT goes beyond the devices. It also includes the protection of the data being processed and transmitted by these devices. With many IoT devices in operation, IoT-generated data becomes a significant part of the networks' traffic. In fact, the devices are only tools for transmitting the data that is so badly hunted by malicious predators. Any security framework that will ultimately be adopted for IoT, will largely depend on how the framework will specify the protection of both data and devices in a secure manner and on a massive scale. A promising concept that the IoT security framework could be hinged on, is the concept of Trust. Trust has proven to be an invaluable concept in every human relationship. Trust has been a fundamental fabric of human society and has been used in the creation and continual existence of all mankind [225]. Mankind has been able to filter out trusted allies from foes through the concept of trust. In a world of 7.4 billion people with many malicious individuals, only the concept of a trust evaluation in a relationship can help decipher the true identity of any entity.

Similarly, IoT today has peaked to over 3 billion devices online with the promise of over 20 billion by 2020 as presented in Chapter 3. The adoption of a trust-based security framework for a massive scale of IoT devices becomes imperative in filtering malicious actors from true allies in IoT networks. Trust can be deployed to manage and identify malicious actors in a network; this ultimately helps in maintaining an IoT network with high fidelity. Furthermore, a trust framework can offer to a high degree, a prediction of the behaviour of a node or device in a network while creating the adaptability of the network system to changing behavioural dynamics of the node in the future. This helps in network stability and adaptability to changes due to malicious actors, just as humans would do when betrayed in a relationship. Many IoT devices will communicate in different ways including via peer-to-peer over heterogeneous networks, and data generated could be processed in diverse ways that may not be limited to the immediate location of the devices that generated the data. Moreover, coupled with the ability of nodes to move freely while joining and leaving the network, most traditional security frameworks will obviously fall short of providing the needed robust security definitions in a network of this nature.

The justification for the adoption of a trust-based framework is summarised in Figure 8.2, which shows the proposed framework of this thesis. It shows the interaction of a trust system with IoT routing protocols. In the framework, the trust system components evaluate, observe and process the behaviour of IoT nodes. The trust system is further mapped into an IoT routing protocol (RPL) to provide the security triad (Confidentiality, Integrity and Availability). However, the framework in this thesis does not address Confidentiality and this can be seen from the Figure as a fade-out grey component. A possible future work. The outcome of the implementation of the trust system in an IoT routing system provides scalability of network size. This does not disrupt the internal workings of the routing protocol. Furthermore, since the trust system is embedded into the IoT protocol, trust is managed as part of the operational system of the protocol. This is particularly necessary since the security system is now integrated into the protocol rather than being an afterthought or an add-on. Node trustworthiness is a feature that results from embedding trust into the protocol. This gives nodes in the network the ability to evaluate the trustworthiness of another node before they can rely on it to transmit its data. This feature satisfies a security baseline requirement for an IoT network. The ability of a node to evaluate the trustworthiness of another node or device will address issues like the heterogeneity of media types, device types, device mobility and transparency and integration of devices across technologies.

A routing protocol will always seek the best or optimal route to transmit its data. Malicious nodes often take advantage of this by attracting packets from unsuspecting nodes to transmit their packets through them, which end up being compromised. In the framework proposed, the trust system embedded in the protocol provides transactions and route validity since the integrity of every node is verified before a packet is transmitted through the verified node.

Energy consumption and Mobility are equally important components in IoT routing, and are highlighted in the framework of Figure 8.2, but which this study does not address. These are indicated in fade-out grey. The framework presented addresses many attacks as studied in Chapters 6, and 7. It is of note that the aim of an attack is consistent, irrespective of the sphere of operation, which could include and not limited to, the industrial sector, commercial sector or the consumer market.

Finally, the trust concept however, may not be viewed as a total replacement for any security framework being considered, but it can be used in collaboration with other security systems for an effective and secure protection of IoT systems.

The Internet of Things is still in its infancy, and if its potential is to be fully realized, the propagation of IoT devices with little or no security consideration, is a cost the world cannot bear. Considering the potential scope of harm to the critical infrastructure of organisations, the personal privacy of individuals and the entire economies that depend on it, the *Coriolis Effect* is enormous. A critical, conscientious and consensus measure is therefore required.

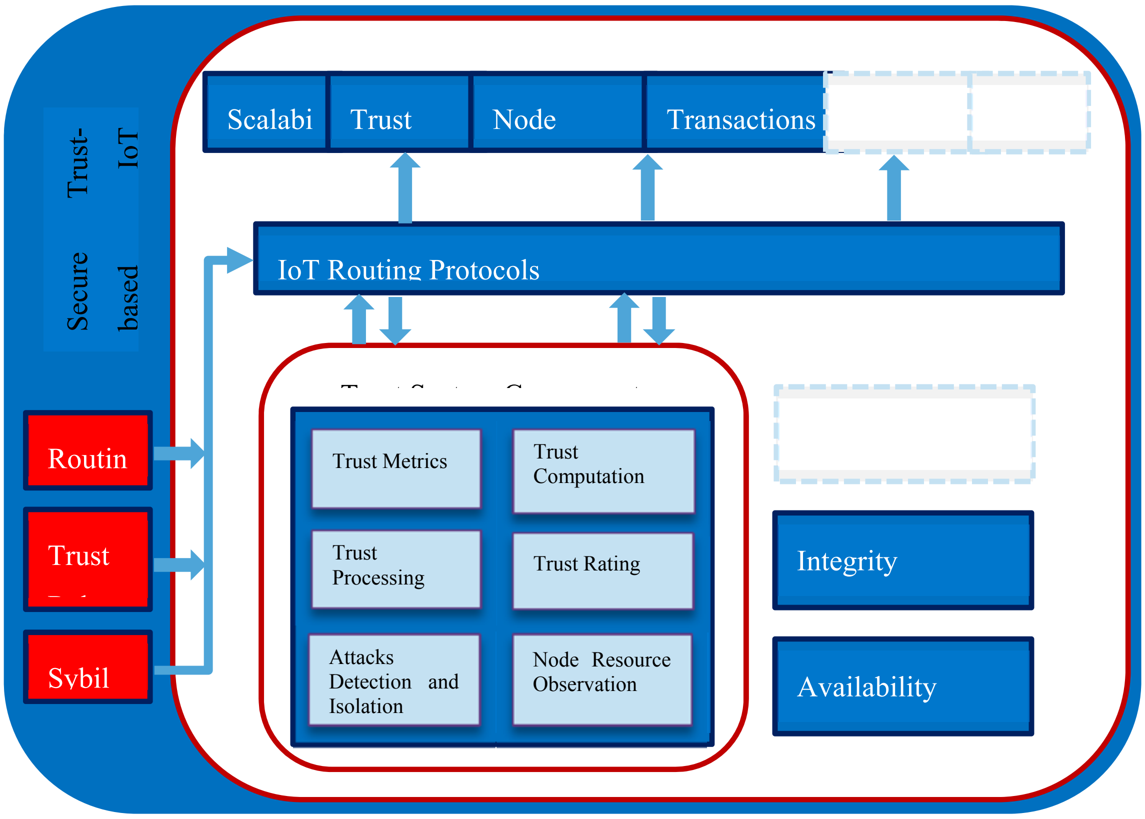


Figure 8.2: Proposed secure Trust-based IoT routing framework

8.4 Implications and Challenges for Trust-Based Frameworks in Secure IoT communications

Despite the unique features and benefits that trust management systems offer, they however have their own unique challenges. Most trust management systems are susceptible to several malicious attacks, which have been discussed in Section 4.2.2. This thesis however, articulates some challenges and implications that trust management models may face within an IoT context.

Most Trust Management Systems often find it difficult to address Sybil and Newcomer attacks. The Sybil attacker assumes multiple identities so it could deceitfully operate in a network, while the Newcomer attacker assumes a new identity as soon as its trustworthiness depreciates [226]. In addition, network collision is a factor that weakens trust evaluation among nodes, and while this is a normal activity in any network, malicious actors could exploit it to weaken the reputation of other trusted nodes in the network. A situation most Trust Management Systems have not adequately addressed [226].

Trust Management Systems utilize passive observation for monitoring node behaviour. A phenomenon referred to as promiscuous mode operation. In the evaluation of the trustworthiness of a neighbour node, Trust Management Systems overhear the communication channel and compute the direct and indirect (transitive) trust values. In fact, many Trust Management Systems including *SecTrust*, make this basic assumption of overhearing for their logic to hold true. The implication here is that, any security framework that will preclude the issue of devices operating in promiscuous mode could limit the adoption of a trust-based framework. In addition, for passive observation to be considered in the monitoring of a node, a node's communication channel must be free, since any node using asymmetric connection for antenna communication or power regulation, will render passive observation impossible [226].

The effectiveness of overhearing among nodes is impacted by the mobility of nodes, frequency of the collision of packets, and the ability of malicious actors to regulate transmission power for maximum network reach [227]. In an unstable network topology characterised by obscure collisions, unidirectional connections and partial network droppings, passive monitoring becomes infeasible [228]. Overhearing keeps a node awake and thus, requires CPU cycles, which creates computation overhead in the network. In the computation of transitive or indirect trust, recommendation from other nodes are collected and aggregated. Although aggregation improves trust estimation accuracy, complex trust aggregation generally tends to increase computational overhead. Any Trust Management Systems incorporating a recommendation system must maintain a healthy balance between an acceptable computational overhead and trust effectiveness [225]. As an additional overhead in trust computation, communication overhead is incurred in all trust propagation and trust recommendation updates. To propagate trust values widely throughout a network, most global Trust Management Systems utilize flooding mechanics to achieve this purpose. In Chapter 6, *SecTrust* leveraged on the DIO flooding of RPL in the integration of *SecTrust* into RPL. A critical analysis therefore, is required, to ascertain if the propagation of trust evidences to every node is essential.

Mobility of nodes from the perspective of trust-based systems will have both positive and negative impact to any trust-based framework. On one hand, the mobility of nodes within a network will help facilitate trust propagation throughout the network in a limited time; on the other hand, link connectivity between nodes could weaken due to increasing spatial distance between nodes, and thus, communication and trust propagation could be constrained. However, it is generally observed that higher node density achieves greater trust propagation [226].

8.5 Cloud Security Alliance's Software Defined Perimeter Vs SecTrust

In contrasting the Software Defined Perimeter architecture with SecTrust, key differential features have been identified, which elicits the advantages of SecTrust in comparison to SDP. As discussed in Chapter 3 (Section 3.11), that SDP is a recent approach to security in mitigating network-based attacks through the creation of dynamic perimeter networks in the Cloud, DMZ, and in Data Centres. Although, there are possibilities that SDP may address what this thesis has outlined to do, some arguable observations in SDP are however articulated further below:

1. Nature of the IoT devices. Many IoT devices are resource constrained (battery, CPU and memory) and this makes SDP unsuitable for use in this context. Now, SDP involves the use of cryptographic encryption for key exchange between initiating SDP host and SDP controller. This becomes a resource challenge for the IoT devices and a reason why Trust was pursued in this study.
2. Sparsely distributed network. For a sparsely distributed IoT network such as an agricultural farm, the resource constrained IoT devices may find themselves out of range from the SDP Controllers in order to authenticate and access resources, services from an Accepting SDP host.
3. SDP assumes the availability of key infrastructural devices such as the SDP Controllers. It begs the question then, that how will these IoT nodes authenticate and communicate in an infrastructure-less environment.

In summary, a Trust-based framework while not providing all the security protection as discussed in Chapter 3, it however, adds reliability and availability among IoT devices.

8.6 Summary

As the IoT revolution continues to grow and the issue of security gradually takes a centre stage, trust framework offers a promising platform for the secure communication of IoT devices. Even though, the trust concept has some limitations, such as the introduction of additional overheads, but which is generally considered minimal when compared to other security frameworks like Cryptography. Cryptography though, gives good security encryption, and is currently being used on various devices, but it does have its own limitations. As an example, Cryptography fails in providing data authentication protection against insider attacks or faulty nodes within a network, which trust accomplishes well.

A key design consideration of a trust framework, will be the design of a trust-based framework that could detect and isolate attacks across all layers of communications (physical to the application layers), which provides a comprehensive solution that can easily be adopted. Current Trust Management Systems compute trust values using factors such as packets forwarded, packet dropped etc. However, the incorporation of other parametric factors across other layers will provide the opportunity of enhancing the effectiveness of Trust Management Systems. Moreover, many Trust Management and Recommendation systems process node observations as either success or failure. This situation can be enhanced through the assignment of a weighted value system to every observation made whether they be successes or failures. This provides a more subjective way of rationalising node observations rather than perceiving them as pure Boolean values as currently adopted by most trust valuation systems.

Finally, Trust Management Systems should not be viewed as the sole replacement for other security systems. However, with proper alignment, these security systems could be melded into an integrated framework where these security systems complement each other. Cryptography and Trust Management Systems, for example, could be combined to produce a security framework, which ensures the security and stability of the Internet of Things.

In summary, statistical empirical results were presented showing levels of confidence of simulated and testbed data used during simulation and testbed trials. Of importance was the need to pay attention to the reliability of precision of the study data. Thus, this research undertook to analyse the level of reliability of data presented by measuring the confidence intervals, and the margins of error, as these offer statistical significance in determining how far from the central tendency of truth the data set is. Any good study data will most certainly want to see low confidence interval levels and margins of error values. This is important so that the entire range of data under evaluation typifies an effect that is expressive of the real-world scenario, which this study, precisely achieved.

9 Conclusions and Future Work

This thesis set out to propose the need for a secure routing framework based on trust and recommendation. A trust and recommendation framework is proposed, and simulation studies were performed, which proved the efficacy of the proposed system in the light of other global trust systems. Equally, the proposed framework was extended to include the security of routing data in an IoT context. A testbed experiment was further implemented to validate the simulation study. Performance measures were conducted and analysed while statistical inferences were computed and discussed.

The proposed trust and recommendation system (*SecTrust*) can also be extended to other systems that utilize recommendation systems, such as online trading systems, e-commerce systems, social media and online voting systems, amongst others.

The trust system was embedded in the RPL routing protocol (IETF standard routing protocol for the Internet of Things) with simulation been undertaken to compare its performance with the standard RPL routing protocol. Security vulnerabilities were measured while ensuring the delivery of acceptable network performance levels. This was conducted using Contiki/Cooja, an IoT based simulation and emulation platform.

This study was advanced further to provide a proof-of-concept especially to the industry by deploying a small-scale live testbed. The trust-based RPL (*SecTrust*-RPL) and the standard RPL were embedded in XM1000 motes using the Contiki/Cooja platform, and experiments were performed, which compared the security vulnerabilities and network performance of *SecTrust*-RPL and the standard RPL. The results of all simulations and experiments have been thoroughly presented and discussed in this thesis.

The implications and challenges of the adoption of a trust and recommendation system as a security framework for IoT was further explored, to examine critically, what the potential problems could be, and what issues are being faced in the emerging IoT industry. Due to the multi-faceted security vulnerabilities uncovered in the study, a mix of other proven secure frameworks in unison with trust and recommendation systems provide promising and an effective pathway for the adoption of a security framework that could guarantee secure and stable Internet of Things.

The outcome of this research work is underscored by peer-reviewed journals and conference publications. A survey study on secure routing for Internet of Things was carried out and the survey was published in the Journal of Network and Computer Applications (2016) [2]. The features of MANETs were studied with the aim of adapting some transferable features that could apply to IoT. The findings were reported in the 2015 International Conference on Information and Resources Management (Conf-IRM) [29]. The publications were a key component of Chapter 3 of this thesis.

An initial part of this research, which was later de-emphasized, was to investigate energy efficiency and sufficiency in curbing carbon footprints when IoT devices are multiplied on a large scale. The outcome of this study was presented at the 2015 IEEE Global Communications Conference (GLOBECOM) [56]. The study on energy was further investigated with the aim of examining the approaches and challenges of greening and optimizing the energy consumption of IoT sensor nodes through Energy Harvesting (EH). The study was published in the 2016 International Conference on Information and Resources Management [57]. These published papers were part of Chapters 2 and 3 of this thesis.

Chapter 4 of this thesis presented the design of the research study. The design was presented at the 14th International Conference on Pervasive Intelligence and Computing (2016) [191]. Simulation studies were performed to evaluate the performance of the proposed design while simulation studies were performed to compare it with other global trust systems. The simulation results were presented in Chapter 5 of this study. In addition, the proposed design was embedded in the RPL routing protocol for IoT and its efficacy and performance measurements were assessed against the current RPL standard. A simulation study was undertaken while testbed experiments were carried out to validate the simulation study. The findings have been presented in Chapters 6 and 7. The findings were published in the 26th IEEE International Telecommunication Networks and Applications Conference (2016) [200] and the Australian Journal of Telecommunications and the Digital Economy (2017) [192].

Challenges were encountered during this research study and alternate routes had to be pursued for the accomplishment of this study. Although this thesis report is being concluded however, some outstanding issues remain, which are discussed in Section 9.1.

9.1 Future Work

This study has investigated the security vulnerabilities in the routing layer of Internet of Things and has proposed a trust-based framework as a promising platform for secure routing of data in this disruptive technology. Even though the trust-based framework proposed addresses the research questions proposed in this study, there are still a few concerns that this study did not address. These concerns have been articulated as suggestive future research directions and they include:

- **Lightweight Encryption for Confidentiality**

To provide confidentiality as a possible future work of this thesis, a lightweight encryption system could be developed, and integrated into the *SecTrust* framework to complete the security triad of CIA (Confidentiality, Integrity and availability). Asymmetric and symmetric encryption systems could be investigated with a view to adapting it to the specifics of an IoT environment. Block ciphers have been studied as notable lightweight cryptographic primitives used in cryptographic constructions. They are used as fundamental secure blocks in ensuring the encryption and authenticity of data [229]. Block ciphers have proved effective in the design against various classical differential and linear attacks. The study of block ciphers in addressing confidentiality issues for resource constrained IoT sensor nodes will provide a platform for the effective defense, and mitigation of confidentiality and integrity related attacks. The study of Block ciphers is currently an active research area and some studies have been undertaken with a view of building ciphers dedicated to resource-constrained devices [230-232].

- **Optimized Trust-based Detection of Compromised Devices in WSN and IoT-based Networks**

This thesis demonstrated the effectiveness of dynamic trust management when applied to secure routing and detection and isolation of routing attacks in an IoT routing protocol (RPL). A possible research direction is to apply this study to consider other areas of application in other wireless constrained networks that can take advantage of the design scheme of the dynamic and robust trust management system proposed in this thesis. Some notable areas of application may include: (a) trust-based collaborative ad hoc applications in WSNs whereby trust computation and evaluation will be based on both the individual assessment and the communal assessment of a node. This will help defeat the complex malicious node behaviour and insider attacks; (b) an optimized trust-based node isolation for complex malicious nodes as reported in [233, 234].

- **Energy-based Trust Management System**

Another limitation in this research is that highly trusted nodes in the network could be overly relied upon to transmit packets to the sink node. As a result, they become depleted of energy, which eventually creates segmented networks from the sink node. This is called the *hotspot* problem. A future scope of study could be the introduction of a synergic selection of sensor node and energy-efficient aggregation scheme in distributed IoT networks. This could be incorporated into the trust-based framework to address the "hot spot" energy problem. This perhaps could produce a new trust-based energy framework, which offers secure communications among nodes while providing healthy energy balance in the network.

- **Trust and Recommendation Systems in Online Service Management**

Another possible future direction is to employ the trust and recommendation system proposed in this thesis to other related areas where trust and recommendation systems will prove beneficial in detecting and isolating attacks related to the ones covered in this study. Typical areas may include: Online adverting, Online purchasing websites and Online rating systems.

- **Integrity-based Trust Management System**

To ensure the integrity of the feedbacks entered, the authenticity of the feedback provider must be validated while guaranteeing the justification of the entry. One way could be through the provision of a service token to users at the end of a service delivery. The implementation of this system should be weighed against the benefits of a fast processing trust system to avoid delays. This adds a layer of security in addition to the confidentiality and availability already embedded. However, it ensures that packets are not tampered with in any way during transmission, and even if they are, they are easily detected and discarded.

- **Deceptive Recommendation Penalty**

Furthermore, it is of the general opinion of this study that the feedback system could be made more robust for better trust computation. Since a receiving node (client) gives feedback on the file service provided by a provider (server), the entry of variable types of feedbacks will provide for a more robust system of calculating trust among nodes. As an example, given a scenario whereby, a *User W* downloads a bad file from *User Z* as a result of the good recommendation received from *Users X* and *Y* about *User Z*. Although, the feedback given by *User W* about *User Z* will clearly be negative, there is however, the need to grant *User W* the opportunity to give useful feedbacks on *Users X* and *Y* since they colluded to deceive User *W*. This is particularly useful for providers who maintain a clean reputation, but refer unsuspecting users (clients) to malicious providers. Trust determination of this sort will adequately predict the referral trust of user nodes, which will mitigate malicious gateway scheme. This is a weakness in *SecTrust* hence, its weak performance against EigenTrust Incremental under the purely collective scenario as presented in Chapter 5. This attribute will effectively address attacks like Wormhole attacks.

9.2 Conclusions

The Internet of Things will continue to grow in both scale and scope and so will its adoption. It is projected that the effect of IoT will permeate every sector of the global society. The global task is to ensure that the emergence of this disruptive technology does not provide the opportunity for malicious actors to perpetrate their activities. The nature and heterogeneity of IoT makes it vulnerable to diverse security issues. To address the security vulnerabilities, a trust-based framework was proposed (*SecTrust*). *SecTrust*, a distributed trust and recommendation system detects common IoT routing attacks and isolates nodes issuing the attacks. Simulation studies and testbed experiments validated the feasibility of its viability and practicability.

The development of *SecTrust* as a security framework for IoT routing provides compelling performance benefits as evidenced by the simulation studies and testbed experiments. In the RPL routing protocol implementation of *SecTrust*, the computation and overhead costs is contingent on the number of behavioural parameters considered in the determination of the trustworthiness of a node. As a result, in the implementation of *SecTrust* across other application settings, the specific scenario and the extent of an attacker's intent needs to be clearly weighed.

The trust and recommendation system testing implemented in Chapter 5 although intensive, shows departures from the dynamic configurations of modern networks, and in subsequent work, more realistic scenarios could be adopted such as, the free entry and exit of nodes into the network due to node mobility, and the insertion of new *files* during network operations. This however, will strain the static notion of the framework adopted for this research. This therefore, necessitates the enhancement of the proposed framework to accommodate these changes.

Lastly, the future works suggested above, do not in any way undermine the validity of the research study undertaken in this thesis, but are suggestive areas where further research can be carried out to accommodate more complex scenarios. The suggestive future research works will even further prove the efficacy of the trust-based framework proposed in this thesis.

10 REFERENCES

[1] A. G. West, S. Kannan, I. Lee, and O. Sokolsky, "An evaluation framework for reputation management systems," 2010.

[2] D. Airehrour, J. Gutierrez, and S. K. Ray, "Secure routing for internet of things," *J. Netw. Comput. Appl.,* vol. 66, pp. 198-213, 2016.

[3] K. Zhao and L. Ge, "A Survey on the Internet of Things Security," in *Computational Intelligence and Security (CIS), 2013 9th International Conference on*, 2013, pp. 663-667.

[4] G. Xu, Y. Ding, J. Zhao, L. Hu, and X. Fu, "Research on the Internet of Things (IoT)," *Sensors & Transducers,* vol. 160, p. 463, 2013.

[5] S. Park, N. Crespi, H. Park, and S.-H. Kim, "IoT routing architecture with autonomous systems of things," pp. 442-445.

[6] International Telecommunication Union. (November, 2005, November 25, 2014). *ITU Internet Reports: The Internet of Things*. Available: http://www.itu.int/wsis/tunis/newsroom/stats/The-Internet-of-Things-2005.pdf

[7] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Generation Computer Systems,* vol. 29, pp. 1645-1660, 2013.

[8] D. Giusto, *The Internet of things: 20th Tyrrhenian workshop on digital communications*. New York: Springer, 2010.

[9] Spansion. (2014, November 25). *ZigBee mesh networks*. Available: http://core.spansion.com/article/contextual-computing-simplifies-iot/#.VG5sfskXJ8E

[10] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Computer Networks,* vol. 54, pp. 2787-2805, 2010.

[11] S. Sicari, S. Hailes, D. Turgut, S. Sharafeddine, and U. B. Desai, "Security, privacy and trust management in the internet of things era – SePriT," *Ad Hoc Networks,* vol. 11, pp. 2623-2624, 2013.

[12] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, "Internet of things: Vision, applications and research challenges," *Ad Hoc Networks,* vol. 10, pp. 1497-1516, 2012.

[13] D. Evans. (2011, 29 November, 2014). The Internet of Things: How the Next Evolution of the Internet Is Changing Everything. Available: http://www.cisco.com/web/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf

[14] Ericsson. (2011, December 2, 2014). More than 50 billion connected devices: Driving forces. Available: http://www.akos-rs.si/files/Telekomunikacije/Digitalna_agenda/Internetni_protokol_Ipv6/More-than-50-billion-connected-devices.pdf

[15] CTIA-The Wireless Association. (2014, December 2, 2014). Mobile Cybersecurity and the Internet of Things Empowering M2M Communication. [White paper]. Available: http://www.ctia.org/docs/default-source/default-document-library/ctia-iot-white-paper.pdf

[16] CISCO. (2013, December 2, 2014). *The Internet of Everything (IoE): Connections Counter*. Available: http://newsroom.cisco.com/feature-content?type=webcontent&articleId=1208342

[17] Symantec, "Internet Security Threat Report," Symantec Laboratory, Mountain View, CA, 2016.

[18] McAfee, "McAfee Labs Threats Report," Santa Clara, CA, 2016.

[19] Cisco, "2015 Annual Security Report," San Jose, CA, 2015.

[20] A. M. Law and A. M. Law, "How to build valid and credible simulation models," *IEEE Engineering Management Review,* vol. 33, pp. 57-57, 2009.

[21] N. Sarkar and J. Gutiérrez, "Revisiting the issue of the credibility of simulation studies in telecommunication networks: highlighting the results of a comprehensive survey of IEEE publications," *IEEE Communications Magazine,* vol. 52, pp. 218-224, 2014.

[22] T. Watteyne, A. Molinaro, M. G. Richichi, and M. Dohler, "From MANET To IETF ROLL Standardization: A Paradigm Shift in WSN Routing Protocols," *Communications Surveys & Tutorials, IEEE,* vol. 13, pp. 688-707, 2011.

[23] M. Ilyas, *The handbook of ad hoc wireless networks*. Boca Raton: CRC Press, 2003.

[24] A. Mishra, *Security and Quality of Service in Ad Hoc Wireless Networks*: Cambridge University Press, 2008.

[25] J. Manyika and M. Chui. (July, 2015, December 1, 2014). By 2025, Internet of things applications could have $11 trillion impact. *Insight Publications*. Available: http://www.mckinsey.com/mgi/overview/in-the-news/by-2025-internet-of-things-applications-could-have-11-trillion-impact

[26] Microsoft Inc. (2014, December 1, 2014). *Internet of Things*. Available: http://www.microsoft.com/windowsembedded/en-us/internet-of-things.aspx

[27] J. Holler and V. Tsiatsis, *From machine-to-machine to the internet of things: introduction to a new age of intelligence*. Amsterdam U6: Academic Press, 2014.

[28] D. Minoli, *Building the Internet of things with IPv6 and MIPv6: the evolving world of M2M communications*. Hoboken, New Jersey: John Wiley & Sons, Inc, 2013.

[29] D. Airehrour and J. Gutierrez, "An analysis of secure MANET routing features to maintain confidentiality and integrity in IoT routing," in *CONF-IRM 2015*, Ontario, Canada, 2015.

[30] D. Airehrour, J. Gutierrez, and S. K. Ray, "GradeTrust: A secure trust based routing protocol for MANETs," in *2015 International Telecommunication Networks and Applications Conference (ITNAC)*, 2015, pp. 65-70.

[31] IEEE Standards Association, "P2413 - Standard for an Architectural Framework for the Internet of Things (IoT)," ed: IEEE, 2014.

[32] D. Boswarthick, "Status of Machine to Machine Standards work in TC M2M and oneM2M," 2013.

[33] ETSI M2M, "ETSI M2M Architecture Overview," September, 2014.

[34] FI-WARE, "FI-WARE Architecture," December 30, 2014 2014.

[35] European Lighthouse Integrated Project. (2014). *Introduction to the Architectural Reference Model for the Internet of Things* Available: http://iotforum.org/wp-content/uploads/2014/09/120613-IoT-A-ARM-Book-Introduction-v7.pdf

[36] N. Rozanski and E. Woods, *Applying Viewpoints and Views to Software Architecture*, 2011.

[37] S. Haller, A. Serbanati, M. Bauer, and F. Carrez, "A Domain Model for the Internet of Things," in *Green Computing and Communications (GreenCom), 2013 IEEE and Internet of Things (iThings/CPSCom), IEEE International Conference on and IEEE Cyber, Physical and Social Computing*, 2013, pp. 411-417.

[38] IoT6. (2014, December 30, 2014). IoT6 architecture. Available: http://iot6.eu/iot6_architecture

[39] (2014, December 30, 2014). *DNS Service Discovery (DNS-SD),* . Available: http://www.dns-sd.org/

[40] (2014, December 30, 2014). *Multicast DNS,* . Available: http://www.multicastdns.org/

[41] S. Krco, B. Pokric, and F. Carrez, "Designing IoT architecture(s): A European perspective," pp. 79-84.

[42] S. Park, N. Crespi, H. Park, and S.-H. Kim, "IoT Routing Architecture with Autonomous Systems of Things," *IEEE World Forum on Internet of Things (WF-IoT),* pp. 442-445, 2014.

[43] Libelium. (2015). *Urban Resilience in the Smart City: River Flood and Forest Fire Early Detection*. Available: http://www.libelium.com/smart-city-urban-resilience-smart-environment/

[44] A. Asín and D. Gascón. (2011). *Smart Parking Sensor Platform enables city motorists save time and fuel*. Available: http://www.libelium.com/smart_parking

[45] W. H. Organisation, "Environment and Health: Data and Statistics (Europe)," ed, 2016.

[46] A. Bielsa. (2012). *Smart City project in Serbia for environmental monitoring by Public Transportation*. Available: http://www.libelium.com/smart_city_environmental_parameters_public_transportation_waspmote

[47] Libelium. (2015). *Water Quality Monitoring in Europe's Largest Fluvial Aquarium*. Available: http://www.libelium.com/water-quality-monitoring-europe-largest-fluvial-aquarium-zaragoza

[48] Government of new Zealand. (2013). *Smart Meters – Information for Households*. Available: https://www.meridianenergy.co.nz/assets/For-Home/Current-customers/Smart-meters-information-for-households.PDF

[49] D. Gascón and M. Yarza. (2011). *Wireless Sensor Networks to Control Radiation Levels*. Available: http://www.libelium.com/wireless_sensor_networks_to_control_radiation_levels_geiger_counters

[50] A. Bielsa, M. Boyd, and A. Asín. (2012). *Wireless Sensor Networks enhancing the efficiency and safety of logistics operations*. Available: http://www.libelium.com/wireless_sensor_networks_m2m_logistics_operations/

[51] Libelium. (2013). *e-Health Sensor Platform for Biometric and Medical applications*. Available: http://www.libelium.com/130220224710

[52] Redmill Marketing Associates. (2015). *Asserting a role in M2M – what's an operator's place?* Available: http://www.redmillcommunications.com/blog/asserting-a-role-in-m2m-whats-an-operators-place/

[53] Libelium. (2015). *Sustainable Farming and the IoT: Cocoa Research Station in Indonesia*. Available: http://www.libelium.com/sustainable-farming-and-the-iot-cocoa-research-station-in-indonesia

[54] I. Ishaq, D. Carels, G. Teklemariam, J. Hoebeke, F. Abeele, E. Poorter*, et al.*, "IETF Standardization in the Field of the Internet of Things (IoT): A Survey," *Journal of Sensor and Actuator Networks,* vol. 2, pp. 235-287, 2013.

[55] K. Machado, D. Rosário, E. Cerqueira, A. A. F. Loureiro, A. Neto, and J. N. d. Souza, "A routing protocol based on energy and link quality for Internet of Things applications," *Sensors (Basel, Switzerland),* vol. 13, pp. 1942-1964, 2013.

[56] D. Airehrour, J. Gutierrez, W. Liu, and J. Wu, "When Internet Raised to the Things Power: Are Energy Efficiency Standards Sufficient to Curb Carbon Footprints?," in *2015 IEEE Globecom Workshops (GC Wkshps)*, 2015, pp. 1-6.

[57] D. Airehrour, J. Gutiérrez, and S. K. Ray, "Greening and Optimizing Energy Consumption of Sensor Nodes in the Internet of Things through Energy Harvesting: Challenges and Approaches," in *CONF-IRM 2016 Proceedings. 42.*, Capetown, South Africa, 2016.

[58] D. E. Culler. (2011, October, 2014). The Internet of Every Thing - steps toward sustainability. Available: https://www.postscapes.com/internet-of-things-protocols/

[59] R. Hummen, J. Hiller, H. Wirtz, M. Henze, H. Shafagh, and K. Wehrle, "6LoWPAN Fragmentation Attacks and Mitigation Mechanisms," in *WiSec 2013*, Budapest, Hungary, April 17-19, 2013, pp. 55-66.

[60] J. Hui and P. Thubert, "Compression format for IPv6 datagrams over IEEE 802.15. 4-based networks," 2011.

[61] Z. Shelby, S. Chakrabarti, E. Nordmark, and C. Bormann. (2012. Available: https://tools.ietf.org/html/rfc6775

[62] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis*, et al.* (2012, RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. Available: https://tools.ietf.org/html/rfc6550

[63] A. Becher, Z. Benenson, and M. Dornseif, "Tampering with motes: real-world physical attacks on wireless sensor networks," presented at the Proceedings of the Third international conference on Security in Pervasive Computing, York, UK, 2006.

[64] R. Anderson and M. Kuhn, "Tamper resistance: a cautionary note," presented at the Proceedings of the 2nd conference on Proceedings of the Second USENIX Workshop on Electronic Commerce - Volume 2, Oakland, California, 1996.

[65] L. Anhtuan, J. Loo, A. Lasebae, A. Vinel, C. Yue, and M. Chai, "The Impact of Rank Attack on Network Topology of Routing Protocol for Low-Power and Lossy Networks," *Sensors Journal, IEEE,* vol. 13, pp. 3685-3692, 2013.

[66] T. Yashiro, S. Kobayashi, N. Koshizuka, and K. Sakamura, "An Internet of Things (IoT) architecture for embedded appliances," pp. 314-319.

[67] L. Wallgren, S. Raza, and T. Voigt, "Routing Attacks and Countermeasures in the RPL-Based Internet of Things," *International Journal of Distributed Sensor Networks,* vol. 2013, p. 11, 2013.

[68] Internet Engineering Task Force (IETF). (December, 2014, December 1, 2014). IPv6 over the TSCH mode of IEEE 802.15.4e (6tisch). Available: https://datatracker.ietf.org/wg/6tisch/charter/

[69] Z. Shelby, K. Hartke, C. Bormann, and B. Frank, "Constrained Application Protocol (CoAP). draft-ietf-core-coap-04 (Internet Draft)," ed: January, 2011.

[70] V. Karagiannis, P. Chatzimisios, F. Vazquez-Gallego, and J. Alonso-Zarate, "A survey on application layer protocols for the internet of things," *Transaction on IoT and Cloud Computing,* vol. 3, pp. 11-17, 2015.

[71] P. Desai, A. Sheth, and P. Anantharam, "Semantic Gateway as a Service architecture for IoT Interoperability," 2014.

[72] A. Mishra, *Security and Quality Of Service In Ad Hoc*

Wireless Networks. Cambridge, UK: Cambridge University Press, 2008.

[73] K. Chugh, L. Aboubaker, and J. Loo, "Case Study of a Black Hole Attack on LoWPAN-RPL," in *Proc. of the Sixth International Conference on Emerging Security Information, Systems and Technologies (SECURWARE), Rome, Italy (August 2012)*, 2012, pp. 157-162.

[74] S. K. Sarkar, T. G. Basavaraju, and C. Puttamadappa, *Ad Hoc Mobile Wireless Networks*

Principles, Protocols, and Applications, Second Edition ed. Boca Raton, FL: CRC Press 2013.

[75] M. O. Pervaiz, M. Cardei, and J. Wu, "Routing Security in Ad Hoc Wireless Networks ", S. Huang, D. MacCallum, and D. Z. Du, Eds., ed. Florida Atlantic University, Boca Raton, FL: Springer, 2005.

[76] P. García-Teodoro, L. Sánchez-Casado, and G. Maciá-Fernández, "Taxonomy and Holistic Detection of Security Attacks in MANETs," in *Security for multihop wireless networks*, S. Khan and J. Lloret Mauri, Eds., ed Boca Raton, FL: Auerbach Publications, 2014, pp. 3-11.

[77] Gartner, "Gartner Says 4.9 Billion Connected "Things" Will Be in Use in 2015," November, 2014.

[78] Hewlett Packard, "Internet of Things Research Study (2014 Report)," 2015.

[79] Smith. (2013, 30/10/2015). *Unpatched TRENDnet IP cameras still provide a real-time Peeping Tom paradise*. Available: http://www.networkworld.com/article/2223785/microsoft-subnet/unpatched-trendnet-ip-cameras-still-provide-a-real-time-peeping-tom-paradise.html

[80] European Union, "Opinion 8/2014 on the on Recent Developments on the Internet of Things," September, 2014.

[81] N. Islam and Z. A. Shaikh, "Security Issues in Mobile Ad Hoc Network " in *Wireless Networks and Security: Issues, Challenges and Research Trends*, S. Khan and A.-S. K. Pathan, Eds., ed Heidelberg, Germany: Springer-Verlag Berlin Heidelberg, 2013, pp. 49-56.

[82] B. Wu, J. Chen, J. Wu, and M. Cardei, "A Survey of Attacks and Countermeasures in Mobile Ad Hoc Networks," in *Wireless Network Security*, Y. Xiao, X. S. Shen, and D.-Z. Du, Eds., ed: Springer Science, 2007, pp. 110 -132.

[83] Gagandeep and Aashima, "Study on Sinkhole Attacks in Wireless Ad hoc Networks " *International Journal on Computer Science and Engineering (IJCSE),* vol. 4 pp. 1078-1084, June, 2012.

[84] V. P. Singh, S. Jain, and J. Singhai, "Hello Flood Attack and its Countermeasures in Wireless Sensor Networks," *International Journal of Computer Science Issues (IJCSI),* vol. 7, pp. 23-27, May, 2010.

[85] A. Hamid, R. Mamun Or, and H. Choong Seon, "Defense against lap-top class attacker in wireless sensor network," in *Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference*, 2006, pp. 5 pp.-318.

[86] K. Sanzgiri, B. Dahill, B. N. Levine, C. Shields, and E. M. Belding-Royer, "An Authenticated Routing Protocol for Secure Ad Hoc Networks," *IEEE Journal on Selected Areas in Communication, special issue on Wireless Ad hoc Networks,* vol. 23, pp. 598-610, March, 2005.

[87] B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens, "An On-demand Secure Routing Protocol Resilient to Byzantine Failures," in *Proceedings of the ACM Workshop on Wireless Security*, 2002, pp. 21-30.

[88] Y. Hu, A. Perrig, and D. Johnson, "Rushing Attacks and Defense in Wireless Ad Hoc Network Routing Protocols," in *ACM Workshop on Wireless Security (WiSe)*, 2003, pp. 30-40.

[89] H. Yih-Chun, P. Adrian, and B. J. David, "Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks," *Wireless Networks,* vol. 11, p. 21, 2005.

[90] J. Ibriq, I. Mahgoub, and M. Ilyas, "Secure Routing inWireless Sensor Networks," in *Handbook of Information and Communication Security*, P. Stavroulakis and M. Stamp, Eds., ed: Springer, 2010, pp. 553-559.

[91] P. L. R. Chze and K. S. Leong, "A Secure Multi-Hop Routing for IoT Communication," *IEEE World Forum on Internet of Things (WF-IoT),* pp. 428-432, 2014.

[92] R. Hummen, J. Hiller, H. Wirtz, M. Henze, H. Shafagh, and K. Wehrle, "6LoWPAN fragmentation attacks and mitigation mechanisms," presented at the Proceedings of the sixth ACM conference on Security and privacy in wireless and mobile networks, Budapest, Hungary, 2013.

[93] X. Anita, J. Martin Leo Manickam, and M. A. Bhagyaveni, "Two-Way Acknowledgment-Based Trust Framework for Wireless Sensor Networks," *International Journal of Distributed Sensor Networks,* vol. 2013, p. 14, 2013.

[94] K.-F. Krentz, H. Rafiee, and C. Meinel, "6LoWPAN security: adding compromise resilience to the 802.15.4 security sublayer," presented at the Proceedings of the International Workshop on Adaptive Security, Zurich, Switzerland, 2013.

[95] G. Mulligan, "The 6LoWPAN architecture," presented at the Proceedings of the 4th workshop on Embedded networked sensors, Cork, Ireland, 2007.

[96] S. Raza, H. Shafagh, K. Hewage, R. Hummen, T. Voigt, d. o. t. Akademin för innovation*, et al.*, "Lithe: Lightweight Secure CoAP for the Internet of Things," *IEEE Sensors Journal,* vol. 13, pp. 3711-3720, 2013.

[97] R. Giuliano, F. Mazzenga, A. Neri, and A. M. Vegni, "Security Access Protocols in IoT Networks with Heterogenous Non-IP Terminals," pp. 257-262.

[98] S. Raza, S. Duquennoy, J. Höglund, U. Roedig, and T. Voigt, "Secure communication for the Internet of Things—a comparison of link□layer security and IPsec for 6LoWPAN," *Security and Communication Networks,* vol. 7, pp. 2654-2668, 2014.

[99] P. Sang-Hyun, C. Seungryong, and L. Jung-Ryun, "Energy-Efficient Probabilistic Routing Algorithm for Internet of Things," *Journal of Applied Mathematics,* vol. 2014, 2014.

[100] J. Duan, D. Gao, D. Yang, C. H. Foh, and H.-H. Chen, "An Energy-Aware Trust Derivation Scheme With Game Theoretic Approach in Wireless Sensor Networks for IoT Applications," *IEEE Internet of Things Journal,* vol. 1, pp. 58-69, February, 2014.

[101] G. Piro, G. Boggia, and L. A. Grieco, "A standard compliant security framework for IEEE 802.15.4 networks," in *Internet of Things (WF-IoT), 2014 IEEE World Forum on*, 2014, pp. 27-30.

[102] A. Le, J. Loo, A. Lasebae, M. Aiash, and Y. Luo, "6LoWPAN: a study on QoS security threats and countermeasures using intrusion detection system approach," *International Journal of Communication Systems,* vol. 25, pp. 1189-1212, 2012.

[103] N. Accettura and G. Piro, "Optimal and secure protocols in the IETF 6TiSCH communication stack," in *Industrial Electronics (ISIE), 2014 IEEE 23rd International Symposium on*, 2014, pp. 1469-1474.

[104] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, "Transmission of IPv6 packets over IEEE 802.15. 4 networks," 2070-1721, 2007.

[105] K. Pister, P. Thubert, and T. Phinney. (2009, Industrial Routing Requirements in Low-Power and Lossy Networks. *IETF RFC 5673*. Available: https://tools.ietf.org/html/rfc5673

[106] K. Seo and S. Kent, "Security architecture for the internet protocol," 2005.

[107] S. Kent, "IP authentication header," 2005.

[108] S. Kent, "IP encapsulating security payload (ESP)," 2005.

[109] R. Riaz, K.-H. Kim, and H. F. Ahmed, "Security analysis survey and framework design for ip connected lowpans," in *Autonomous Decentralized Systems, 2009. ISADS'09. International Symposium on*, 2009, pp. 1-6.

[110] Z. Shelby and C. Bormann, *6LoWPAN: The wireless embedded Internet* vol. 43: John Wiley & Sons, 2011.

[111] J. Granjal, R. Silva, E. Monteiro, J. Sa Silva, and F. Boavida, "Why is IPSec a viable option for wireless sensor networks," in *Mobile Ad Hoc and Sensor Systems, 2008. MASS 2008. 5th IEEE International Conference on*, 2008, pp. 802-807.

[112] J. Granjal, E. Monteiro, and J. S. Silva, "Enabling network-layer security on IPv6 wireless sensor networks," in *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, 2010, pp. 1-6.

[113] J. Granjal, E. Monteiro, and J. S. Silva, "Network□layer security for the Internet of Things using TinyOS and BLIP," *International Journal of Communication Systems,* vol. 27, pp. 1938-1963, 2014.

[114] S. Raza, S. Duquennoy, T. Chung, D. Yazar, T. Voigt, and U. Roedig, "Securing communication in 6LoWPAN with compressed IPsec," in *Distributed Computing in Sensor Systems and Workshops (DCOSS), 2011 International Conference on*, 2011, pp. 1-8.

[115] S. Raza, S. Duquennoy, J. Höglund, U. Roedig, and T. Voigt, "Secure communication for the Internet of Things—a comparison of link□layer security and IPsec for 6LoWPAN," *Security and Communication Networks,* 2012.

[116] E. Kim and D. Kaspar, "Design and application spaces for IPv6 over low-power wireless personal area networks (6LoWPANs)," 2012.

[117] R. Hummen, H. Wirtz, J. H. Ziegeldorf, J. Hiller, and K. Wehrle, "Tailoring end-to-end IP security protocols to the Internet of Things," in *Network Protocols (ICNP), 2013 21st IEEE International Conference on*, 2013, pp. 1-10.

[118] R. Roman, C. Alcaraz, J. Lopez, and N. Sklavos, "Key management systems for sensor networks in the context of the Internet of Things," *Computers & Electrical Engineering,* vol. 37, pp. 147-159, 2011.

[119] T. Tsao, R. Alexander, M. Dohler, V. Daza, A. Lozano, and M. Richardson, "A Security Threat Analysis for Routing Protocol for Low-power and lossy networks (RPL)," 2014.

[120] A. Mayzaud, A. Sehgal, R. Badonnel, I. Chrisment, and J. Schönwälder, "A Study of RPL DODAG Version Attacks," in *Monitoring and Securing Virtualized Networks and Services*. vol. 8508, A. Sperotto, G. Doyen, S. Latré, M. Charalambides, and B. Stiller, Eds., ed: Springer Berlin Heidelberg, 2014, pp. 92-104.

[121] D. B. Parker, "Restating the foundation of information security," *Computer Audit Update,* vol. 1991, pp. 2-15, 1991.

[122] A. Dvir, T. Holczer, and L. Buttyan, "VeRA-version number and rank authentication in rpl," in *Mobile Adhoc and Sensor Systems (MASS), 2011 IEEE 8th International Conference on*, 2011, pp. 709-714.

[123] K. Weekly and K. Pister, "Evaluating sinkhole defense techniques in RPL networks," in *Network Protocols (ICNP), 2012 20th IEEE International Conference on*, 2012, pp. 1-6.

[124] S. Raza, L. Wallgren, and T. Voigt, "SVELTE: Real-time intrusion detection in the Internet of Things," *Ad Hoc Netw.,* vol. 11, pp. 2661-2674, 2013.

[125] S. O. Amin, M. S. Siddiqui, C. S. Hong, and J. Choe, "A novel coding scheme to implement signature based IDS in IP based Sensor Networks," in *Integrated Network Management-Workshops, 2009. IM'09. IFIP/IEEE International Symposium on*, 2009, pp. 269-274.

[126] H. Perrey, M. Landsmann, O. Ugus, T. C. Schmidt, and M. Wählisch, "TRAIL: Topology Authentication in RPL," *arXiv preprint arXiv:1312.0984,* 2013.

[127] K. Zhang, X. Liang, R. Lu, and X. Shen, "Sybil Attacks and Their Defenses in the Internet of Things," *Internet of Things Journal, IEEE,* vol. 1, pp. 372-383, 2014.

[128] P. Kasinathan, C. Pastrone, M. Spirito, and M. Vinkovits, "Denial-of-Service detection in 6LoWPAN based Internet of Things," in *Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on*, 2013, pp. 600-607.

[129] Cloud Security Alliance, "Software Defined Perimeter v1.0," ed, 2013.

[130] Z. Yan, P. Zhang, and A. V. Vasilakos, "A survey on trust management for Internet of Things," *Journal of Network and Computer Applications,* vol. 42, pp. 120-134, 2014.

[131] D. Gambetta, *Can we trust trust?* . Oxford, England, UK: Basil Blackwell, 1988.

[132] Stephen P. Marsh, "Formalising Trust as a Computational Concept," PhD, Computer Science, University of Stirling, Stirling, UK, 1994.

[133] R. T. Golembiewski and M. McConkie, *The centrality of interpersonal trust in group processes*, 1975.

[134] F. Bao and I.-R. Chen, "Dynamic trust management for internet of things applications," 2012, pp. 1-6.

[135] B. Fenye and C. Ing-Ray, "Trust management for the internet of things and its application to service composition," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2012 IEEE International Symposium on a*, 2012, pp. 1-6.

[136] J. Dubey and V. Tokekar, "Bayesian network based trust model with time window for Pure P2P computing systems," pp. 219-223.

[137] Y. Gao and W. Liu, "BeTrust: A Dynamic Trust Model Based on Bayesian Inference and Tsallis Entropy for Medical Sensor Networks," *Journal of Sensors,* vol. 2014, pp. 1-10, 2014.

[138] D. Melaye and Y. Demazeau, "Bayesian Dynamic Trust Model." vol. 3690, ed Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 480-489.

[139] J.-J. Qi, Z.-Z. Li, and L. Wei, "A Trust Model Based on Bayesian Approach." vol. 3528, ed Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 374-379.

[140] D. Wei and C. Junliang, "The Bayesian Network and Trust Model Based Movie Recommendation System." vol. 180, ed Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 797-803.

[141] K. Chatterjee and W. Samuelson, *Game theory and business applications* vol. 194. New York: Springer, 2014.

[142] R. Feng, S. Che, X. Wang, and J. Wan, "An incentive mechanism based on game theory for trust management," *Security and Communication Networks,* vol. 7, pp. 2318-2325, 2014.

[143] M. A. M. Isa, J.-l. A. Manan, R. Mahmod, H. Hashim, N. I. Udzir, A. D. Tanha*, et al.*, "Game theory: trust model for common criteria certifications & evaluations," *International Journal of Cyber-Security and Digital Forensics,* vol. 1, pp. 50-58, 2012.

[144] L. Yuanjie, X. Hongyun, C. Qiying, L. Zichuan, and S. Shigen, "Evolutionary Game-Based Trust Strategy Adjustment among Nodes in Wireless Sensor Networks," *International Journal of Distributed Sensor Networks,* vol. 2015, 2015.

[145] S. Che, R. Feng, X. Liang, and X. Wang, "A lightweight trust management based on Bayesian and Entropy for wireless sensor networks," *Security and Communication Networks,* vol. 8, pp. 168-175, 2015.

[146] Z. Kun, K. Márton, and B. Ginestra, "Entropy of Dynamical Social Networks: e28116," *PLOS One,* vol. 6, 2011.

[147] B. Sovan and G. Arnab, "Entropy Trust based Approach against IP Spoofing Attacks in Network," *International Journal of Computer Applications,* vol. 67, 2013.

[148] X. Anita, M. A. Bhagyaveni, and J. M. L. Manickam, "Fuzzy-Based Trust Prediction Model for Routing in WSNs," *The Scientific World Journal,* vol. 2014, pp. 1-11, 2014.

[149] R. A. Raje and A. V. Sakhare, "Routing in Wireless Sensor Network Using Fuzzy Based Trust Model," pp. 529-532.

[150] W. Xianghe, Z. Hong, M. Xuan, Q. Yong, L. Zhen, and L. Qianmu, "A Comprehensive Trust Model Based on Reputation and Fuzzy Theory," *Journal of Computers,* vol. 9, pp. 2666-2676, 2014.

[151] H. Zhang and D. Liu, *Fuzzy modeling and fuzzy control*. Boston, MA: Birkhäuser Boston, 2006.

[152] A. Jøsang, "A Logic for Uncertain Probabilities," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems,* vol. 09, pp. 279-311, 2001.

[153] A. Jøsang, R. Hayward, and S. Pope, "Trust network analysis with subjective logic," in *Proceedings of the 29th Australasian Computer Science Conference-Volume 48*, 2006, pp. 85-94.

[154] G. Shafer, *A mathematical theory of evidence* vol. 1: Princeton university press Princeton, 1976.

[155] F. Gómez Mármol and G. Martínez Pérez, "Trust and reputation models comparison," *Internet Research,* vol. 21, pp. 138-153, 2011.

[156] H. Xu, Y. Liu, S. Qi, and Y. Shi, "A novel trust model based on probability and statistics for Peer to Peer networks," pp. 2047-2050.

[157] Q. Yuan, "The Study on Trust Management Model Based on Probability in Distributed E-Commerce System." vol. 233, ed Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 312-320.

[158] C. Po-Jen and J. Yi-Jun, "Effective neural network-based node localisation scheme for wireless sensor networks," *Wireless Sensor Systems, IET,* vol. 4, pp. 97-103, 2014.

[159] S. Y. Monir Vaghefi and R. M. Vaghefi, "A novel multilayer neural network model for TOA-based localization in wireless sensor networks," in *Neural Networks (IJCNN), The 2011 International Joint Conference on*, 2011, pp. 3079-3084.

[160] P. Singh and S. Agrawal, "TDOA Based Node Localization in WSN Using Neural Networks," in *Communication Systems and Network Technologies (CSNT), 2013 International Conference on*, 2013, pp. 400-404.

[161] C. P. Subha, S. Malarkan, and K. Vaithinathan, "A survey on energy efficient neural network based clustering models in wireless sensor networks," in *Emerging Trends in VLSI, Embedded System, Nano Electronics and Telecommunication System (ICEVENT), 2013 International Conference on*, 2013, pp. 1-6.

[162] T. Dan, T. Shaojie, and L. Liang, "Constrained Artificial Fish-Swarm Based Area Coverage Optimization Algorithm for Directional Sensor Networks," in *Mobile Ad-Hoc and Sensor Systems (MASS), 2013 IEEE 10th International Conference on*, 2013, pp. 304-309.

[163] A. Gupta, O. J. Pandey, M. Shukla, A. Dadhich, A. Ingle, and P. Gawande, "Towards context-aware smart mechatronics networks: Integrating Swarm Intelligence and Ambient Intelligence," in *Issues and Challenges in Intelligent Computing Techniques (ICICT), 2014 International Conference on*, 2014, pp. 64-69.

[164] Z. Wei and L. Di, "Routing in Wireless Sensor Network Using Artificial Bee Colony Algorithm," in *Wireless Communication and Sensor Network (WCSN), 2014 International Conference on*, 2014, pp. 280-284.

[165] A. M. Zungeru, L.-M. Ang, and K. P. Seng, "Termite-Hill: Routing towards a Mobile Sink for Improving Network Lifetime in Wireless Sensor Networks," in *Intelligent Systems, Modelling and Simulation (ISMS), 2012 Third International Conference on*, 2012, pp. 622-627.

[166] S. Kannan and P. Viswanath, "Multi-Session Function Computation and Multicasting in Undirected Graphs," *Selected Areas in Communications, IEEE Journal on,* vol. 31, pp. 702-713, 2013.

[167] L. Yanhua and Z. Zhi-Li, "Random Walks and Green's Function on Digraphs: A Framework for Estimating Wireless Transmission Costs," *Networking, IEEE/ACM Transactions on,* vol. 21, pp. 135-148, 2013.

[168] H. Kowshik and P. R. Kumar, "Optimal Function Computation in Directed and Undirected Graphs," *IEEE Transactions on Information Theory,* vol. 58, pp. 3407-3418, 2012.

[169] P. Swaruba and V. R. Ganesh, "Weighted voting based trust management for intrusion tolerance in heterogeneous wireless sensor networks," in *Information Communication and Embedded Systems (ICICES), 2014 International Conference on*, 2014, pp. 1-7.

[170] J. Sai, S.-f. Yuan, M. Ting-huai, and T. Chang, "Distributed Fault Detection for Wireless Sensor Based on Weighted Average," in *Networks Security Wireless Communications and Trusted Computing (NSWCTC), 2010 Second International Conference on*, 2010, pp. 57-60.

[171] M. E. Rusli, R. Harris, and A. Punchihewa, "Markov Chain-based analytical model of Opportunistic Routing protocol for wireless sensor networks," in *TENCON 2010 - 2010 IEEE Region 10 Conference*, 2010, pp. 257-262.

[172] M. Vasim babu and A. V. Ramprasad, "Discrete antithetic Markov Monte Carlo based power mapping localization algorithm for WSN," in *Advanced Communication Control and Computing Technologies (ICACCCT), 2012 IEEE International Conference on*, 2012, pp. 56-62.

[173] L. Xiaolong and F. Donglei, "Markov Chain Based Trust Management Scheme for Wireless Sensor Networks," *Journal of Networks,* vol. 9, p. 3263, 2014.

[174] M. Agrawal and P. Mishra, "A comparative survey on symmetric key encryption techniques," *International Journal on Computer Science and Engineering (IJCSE),* vol. 4, pp. 877-882, 2012.

[175] M. G. Rashed, S. E. Ullah, and R. Yasmin, "Secured message data transactions with a Digital Envelope (DE)-A higher level cryptographic technique," in *International Conference on Engineering Research, Innovation and Education 2013*, 2013.

[176] J. P. Wang, S. Bin, Y. Yu, and X. X. Niu, "Distributed Trust Management Mechanism for the Internet of Things," *Applied Mechanics and Materials,* vol. 347-350, pp. 2463-2467, Aug., 2013.

[177] Y. Ben Saied, A. Olivereau, D. Zeghlache, and M. Laurent, "Trust Management System Design for the Internet of Things: A Context-Aware and multi-Service Approach," *Computers & Security,* vol. 39, Part B, pp. 351-365, 2013.

[178] P. N. Mahalle, P. A. Thakre, N. R. Prasad, and R. Prasad, "A fuzzy approach to trust based access control in internet of things," in *Wireless Communications, Vehicular Technology, Information Theory and Aerospace & Electronic Systems (VITAE), 2013 3rd International Conference on*, 2013, pp. 1-5.

[179] G. Dólera Tormo, F. Gómez Mármol, and G. Martínez Pérez, "Dynamic and flexible selection of a reputation mechanism for heterogeneous environments," *Future Generation Computer Systems*.

[180] L. Gu, J. Wang, and B. Sun, "Trust management mechanism for Internet of Things," *Communications, China,* vol. 11, pp. 148-156, 2014.

[181] Y. B. Liu, X. H. Gong, and Y. F. Feng, "Trust system based on node behavior detection in Internet of Things," *Tongxin Xuebao/Journal on Communications,* vol. 35, pp. 8-15, 2014.

[182] D. Chen, G. Chang, D. Sun, J. Li, J. Jia, and X. Wang, "TRM-IoT: A trust management model based on fuzzy reputation for internet of things," *Computer Science and Information Systems,* vol. 8, pp. 1207-1228, 2011.

[183] J. Luo, X. Liu, and M. Fan, "A trust model based on fuzzy recommendation for mobile ad-hoc networks," *Computer Networks,* vol. 53, pp. 2396-2407, 2009.

[184] G. Wu, Z. Du, Y. Hu, T. Jung, U. Fiore, and K. Yim, "A Dynamic Trust Model Exploiting the Time Slice in WSNs," *Soft Computing,* vol. 18, pp. 1829-1840, 2014.

[185] X. Li and L. Ling, "PeerTrust: supporting reputation-based trust for peer-to-peer electronic communities," *IEEE Transactions on Knowledge and Data Engineering,* vol. 16, pp. 843-857, 2004.

[186] L. Xiaoyong, Z. Feng, and D. Junping, "LDTS: A Lightweight and Dependable Trust System for Clustered Wireless Sensor Networks," *Information Forensics and Security, IEEE Transactions on,* vol. 8, pp. 924-935, 2013.

[187] F. Bao, R. Chen, M. Chang, and J.-H. Cho, "Hierarchical trust management for wireless sensor networks and its applications to trust-based routing and intrusion detection," *Network and Service Management, IEEE Transactions on,* vol. 9, pp. 169-183, 2012.

[188] M. J. Probst and S. K. Kasera, "Statistical trust establishment in wireless sensor networks," in *Parallel and Distributed Systems, 2007 International Conference on*, 2007, pp. 1-8.

[189] P. B. Velloso, R. P. Laufer, D. de O Cunha, O. C. M. B. Duarte, and G. Pujolle, "Trust management in mobile ad hoc networks using a scalable maturity-based model," *Network and Service Management, IEEE Transactions on,* vol. 7, pp. 172-185, 2010.

[190] K. Namuduri, Y. Wan, and M. Gomathisankaran, "Mobile Ad Hoc Networks in the Sky: State of the Art,Opportunities, and Challenges," in *Association of Computing Machinery (ACM), ANC 2013*, Bangalore, India, July 29, 2013.

[191] D. Airehrour, J. Gutierrez, and S. K. Ray, "A Lightweight Trust Design for IoT Routing," in *2016 IEEE 14th Intl Conf on Pervasive Intelligence and Computing*, 2016, pp. 552-557.

[192] D. Airehrour, J. A. Gutierrez, and S. K. Ray, "A Trust-Aware RPL Routing Protocol to Detect Blackhole and Selective Forwarding Attacks," *Australian Journal of Telecommunications and the Digital Economy,* vol. v. 5, n. 1, pp. 50-69, 2017.

[193] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A Design Science Research Methodology for Information Systems Research," *Journal of Management Information Systems,* vol. 24, pp. 45-77, 2007.

[194] S. T. March and G. F. Smith, "Design and natural science research on information technology," *Decision Support Systems,* vol. 15, pp. 251–266, 1995.

[195] C. Dai and W. Gong, "Model of Services Trust Threshold Assess Based on Fuzzy Theory," in *2010 2nd International Conference on E-business and Information System Security*, 2010, pp. 1-4.

[196] Gartner Inc., "The Future Smart Home: 500 Smart Objects Will Enable New Business Opportunities," GartnerSeptember 8, 2014

[197] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The EigenTrust Algorithm for Reputation Management in P2P Networks," in *Proceedings of the 12th international conference on World Wide Web*, 2003, pp. 640-651.

[198] K. Zipf George, "Human Behavior and the Principle of Least Effort: An Introduction to Human Ecology," ed: Reading Mass: Addison-W esley, 1949.

[199] R. Cavagna, C. Bouville, and J. Royan, "P2P Network for very large virtual environment," in *Proceedings of the ACM symposium on Virtual reality software and technology*, 2006, pp. 269-276.

[200] D. Airehrour, J. Gutierrez, and S. K. Ray, "Securing RPL Routing Protocol from Blackhole Attacks Using Trust-based Mechanism," in *26th IEEE International Telecommunication Networks and Applications Conference*, University of Otago, Dunedin, New Zealand, 2016, pp. 115-120.

[201] E. Winsberg, "Simulated experiments: Methodology for a virtual world," *Philosophy of science,* vol. 70, pp. 105-125, 2003.

[202] P. J. Fortier and H. Michel, *Computer Systems Performance Evaluation and Prediction*: Butterworth-Heinemann, 2002.

[203] E. Winsberg, "Computer Simulations in Science," in *The Stanford Encyclopedia of Philosophy*, E. N. Zalta, Ed., Summer 2015 ed: Stanford University, 2015.

[204] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki - A Lightweight and Flexible Operating System for Tiny Networked Sensors," presented at the Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks, 2004.

[205] A. I. Sabbah, A. El-Mougy, and M. Ibnkahla, "A Survey of Networking Challenges and Routing Protocols in Smart Grids," *IEEE Transactions on Industrial Informatics,* vol. 10, pp. 210-221, 2014.

[206] O. Gaddour and A. Koubâa, "RPL in a nutshell: A survey," *Computer Networks,* vol. 56, pp. 3163-3178, 2012.

[207] K. Roussel, Y.-Q. Song, and O. Zendra, "Using Cooja for WSN Simulations: Some New Uses and Limits," in *EWSN 2016 — NextMote workshop*, Graz, Austria, 2016, p. 319–324.

[208] F. sterlind, A. Dunkels, T. Voigt, N. Tsiftes, J. Eriksson, and N. Finne, "Sensornet Checkpointing: Enabling Repeatability in Testbeds and Realism in Simulations," presented at the Proceedings of the 6th European Conference on Wireless Sensor Networks, Cork, Ireland, 2009.

[209] Advanticsys, "XM1000," in *XM1000 Mote*, ed, 2016.

[210] K. W. Sally, O. H. Jason, and S. Nigamanth, "Testing and Debugging Sensor Network Applications," in *Distributed Sensor Networks, Second Edition*, ed: Chapman and Hall/CRC, 2012, pp. 711-728.

[211] R. Sharma and T. Jayavignesh, "Quantitative Analysis and Evaluation of RPL with Various Objective Functions for 6LoWPAN," *Indian Journal of Science and Technology,* vol. 8, p. 1, 2015.

[212] H. Altwassi, M. Qasem, M. B. Yassein, and A. Al-Dubai, "Performance evaluation of RPL objective functions," 2015.

[213] Q. Q. Abuein, M. B. Yassein, M. Q. Shatnawi, L. Bani-Yaseen, O. Al-Omari, M. Mehdawi*, et al.*, "Performance Evaluation of Routing Protocol (RPL) for Internet of Things," *Performance Evaluation,* vol. 7, 2016.

[214] J. Newsome, E. Shi, D. Song, and A. Perrig, "The Sybil attack in sensor networks: analysis & defenses," in *Third International Symposium on Information Processing in Sensor Networks, 2004. IPSN 2004*, 2004, pp. 259-268.

[215] H. Ghayvat, S. Mukhopadhyay, J. Liu, A. Babu, M. E. E. Alahi, and X. Gui, "Internet of Things for smart homes and buildings," *Australian Journal of Telecommunications and the Digital Economy,* vol. 3, 2015.

[216] A. Reinhardt and D. Burgstahler, "Exploiting platform heterogeneity in wireless sensor networks by shifting resource-intensive tasks to dedicated processing nodes," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2013 IEEE 14th International Symposium and Workshops on a*, 2013, pp. 1-9.

[217] S. Jarupadung, "Event processing in wireless sensor networks. ," Masters, University of Surrey, 2015.

[218] Forrester Research Inc and Cisco, "Security: The Vital Element Of The Internet Of Things," 2015.

[219] S. Nakagawa and I. C. Cuthill, "Effect size, confidence interval and statistical significance: a practical guide for biologists," *Biological Reviews,* vol. 82, pp. 591-605, 2007.

[220] C. Kharkongor, T. Chithralekha, and R. Varghese, "A SDN Controller with Energy Efficient Routing in the Internet of Things (IoT)," *Procedia Computer Science,* vol. 89, pp. 218-227, 2016.

[221] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Gener. Comput. Syst.,* vol. 29, pp. 1645-1660, 2013.

[222] M. M. Hossain, M. Fotouhi, and R. Hasan, "Towards an analysis of security issues, challenges, and open problems in the internet of things," in *Services (SERVICES), 2015 IEEE World Congress on*, 2015, pp. 21-28.

[223] 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, pp. 2481-2501, 2014.

[224] MC Afee. (2014). *Securing the Internet of Things: OEM capabilities assure trust, integrity, accountability, and privacy*. Available: https://mcafee-uat.mcafee.com/br/resources/solution-briefs/sb-securing-the-iot.pdf

[225] K. Govindan and P. Mohapatra, "Trust Computations and Trust Dynamics in Mobile Adhoc Networks: A Survey," *IEEE Communications Surveys & Tutorials,* vol. 14, pp. 279-298, 2012.

[226] J. Guo, I.-R. Chen, and J. J. P. Tsai, "A survey of trust computation models for service management in internet of things systems," *Computer Communications,* vol. 97, pp. 1-14, 2017.

[227] M. E. Mahmoud and X. Shen, "An Integrated Stimulation and Punishment Mechanism for Thwarting Packet Dropping Attack in Multihop Wireless Networks," *IEEE Transactions on Vehicular Technology,* vol. 60, pp. 3947-3962, 2011.

[228] H. Xia, Z. Jia, X. Li, L. Ju, and E. H. M. Sha, "Trust prediction and trust-based source routing in mobile ad hoc networks," *Ad Hoc Networks,* vol. 11, pp. 2096-2114, 2013.

[229] F. Abed, C. Forler, and S. Lucks, "General classification of the authenticated encryption schemes for the CAESAR competition," *Computer Science Review,* vol. 22, pp. 13-26, 11// 2016.

[230] F. Zhang, S. Guo, X. Zhao, T. Wang, J. Yang, F. X. Standaert*, et al.*, "A Framework for the Analysis and Evaluation of Algebraic Fault Attacks on Lightweight Block Ciphers," *IEEE Transactions on Information Forensics and Security,* vol. 11, pp. 1039-1054, 2016.

[231] B. Wang, L. Liu, C. Deng, M. Zhu, S. Yin, Z. Zhou*, et al.*, "Exploration of Benes Network in Cryptographic Processors: A Random Infection Countermeasure for Block Ciphers Against Fault Attacks," *IEEE Transactions on Information Forensics and Security,* vol. 12, pp. 309-322, 2017.

[232] S. Koteshwara and A. Das, "Comparative study of Authenticated Encryption targeting lightweight IoT applications," *IEEE Design & Test,* vol. PP, pp. 1-1, 2017.

[233] A. Cui and S. J. Stolfo, "Methods, systems, and media for inhibiting attacks on embedded devices," ed: Google Patents, 2016.

[234] T. Dimitriou, E. A. Alrashed, M. H. Karaata, and A. Hamdan, "Imposter detection for replication attacks in mobile sensor networks," *Computer Networks,* vol. 108, pp. 210-222, 10/24/ 2016.

11 APPENDICES

APPENDIX 1: EMBEDDING SECTRUST IN THE QTM P2P TRACE SIMULATOR.......................... 280

APPENDIX 1: EMBEDDING SECTRUST IN THE QTM P2P TRACE SIMULATOR

As earlier discussed, the P2P-trace simulator can be extended to accommodate new trust systems. It has a set of developed Application Programming Interfaces (APIs) that makes the inclusion of new trust and recommendation systems possible. The trace simulator comprises four main modules namely: GENERATOR_LIB, CORE_LIB, TRUST_SYSTEM_LIB and SIMULATOR_LIB. A description of the four modules, their functionalities and interrelationship is further discussed below. In addition, the integration of *SecTrust* into the QTM trace simulator is also presented.

i. GENERATOR_LIB

This module consists of two classes namely: The GeneratorOutput class and the GeneratorUtils class. The GeneratorOutput class supports the driver program in outputting the trace data file while the GeneratorUtils class supports the TraceGenerator program with the creation of network users, file libraries and user transactions.

ii. CORE_LIB

The core library is where the details and computation of the various activities of the peer to peer network is articulated. The module consists of seven classes with their functions detailed below.

- BWidthUnit: This class unit manages the users' bandwidth levels.
- FileCopy: The FileCopy unit stores and manages the unique instance of a file.
- Globals: The Globals class gives an operational platform setting for common network parameters among users.
- Network: This unit defines the class that manages the data and behaviour regarding users, files, and their bandwidth.
- Relation: This class maintains the past interactions between two users in the network.
- Statistics: This class maintains the statistics during trace simulation.
- Transaction: This class stores the variables that uniquely identify the transactions occurring between two users.

- User: This is the definition of a user and its characteristics in a network. This is also referred to as a node, or a peer.

iii. TRUST_SYSTEM_LIB

The module houses the trust systems available in the simulator while the TrustAlg class provides the trust algorithm interface which defines the methods required by a class for the implementation of a specific trust and recommendation system in use at run time. The available trust systems include:

- EigenTM: This provides the class implementation of the EigenTrust system as described in [197].
- EtIncTM: This is the EigenTrust Incremental trust implementation, which offers a speed-up version of the EigenTrust system.
- NoneTM: The NoneTM is a class implementation, which conforms to the Trust algorithm of absence of a trust and recommendation system.
- TnaSlTM: This is the implementation of TNA-SL trust system which conforms with the Trust Network Analysis with Subjective Logic (TNA-SL) concept as proposed by [152, 153].

iv. SIMULATOR_LIB

This library consists of 5 classes listed below:

- SimulatorInput: This class unit provides supports to the TraceSimulator program in reading and analysing data from the provided trace file.
- SimulatorMalicious: This class unit provides the coordination of malicious user behaviour, particularly in the user feedback among the coordinated malicious activities.
- SimulatorOutput: This is the unit responsible for outputting the data gathered from the TraceSimulator.
- SimulatorSource: A complex class object managing the choice of source selection of users and file availability based on trust values for the TraceSimulator program.

- SimulatorUtils: This unit class provides utility support for the TraceSimulator program.

Table 11.1: Java implementation method summary for *SecTrust*

Method Summary for *SecTrust*	
java.lang.String	**algName**() Interfaced: Text name of this trust algorithm (spaces are okay).
private int	**calcnumberpackets**(Relation rel) Calculate a 'feedback integer' using global feedback data.
void	**computeTrust**(int user, int cycle) Interfaced: Compute trust, exporting trust values to Network.
private double[]	**constantVectorMult**(double constant, double[] vector) Linear Algebra: Constant-vector multiplication.
java.lang.String	**fileExtension**() Interfaced: File extension placed on output files using this algorithm.
protected boolean	**hasConverged**(double[] vec1, double[] vec2) Test if the difference between the two vectors is below the set threshold limit.
private void	**normalizeVector**(int new_vec) Normalize a single vector of the persistent matrix.
private double[]	**sectrust1**(double[] prev_vector) Calculate values ready for further calculation within the trust class
protected double[]	**trustMultiply**(int user, int max_iters) Perform matrix multiply as a means of aggregating global trust data.

private boolean	`formula(Relation rel)` Adds the weight of the sectrust formula to the node if it has a negitive impact on the network (bad node)
void	`update(Transaction trans)` Interfaced: Given coordinates of a feedback commitment, update as needed.
private double[]	`vectorAdd(double[] vector1, double[] vector2)` Linear Algebra: Vector-vector addition.
private double[]	`vectorMatrixMult(double[] vector, double[][] matrix)` Linear Algebra: Vector-matrix multiplication.

APPENDIX 2: DATA FOR THE SIMULATION AND TESTBED

This is stored by the AUT Research and Technology. To access data contact: ART@aut.ac.nz