FUZZY BASED SECURITY ALGORITHM FOR WIRELESS SENSOR NETWORKS IN THE INTERNET OF THINGS PARADIGM

BY

Ts'itso Maphats'oe

Submitted in fulfilment of the requirement for

MASTERS IN INFORMATION TECHNOLOGY

CENTRAL UNIVERSITY OF TECHNOLOGY FREE STATE

Supervised by

Dr Masinde Muthoni

Dedication

This work is dedicated to my parents; Malekhutla and Samuel, as a token of appreciation for their unlimited support with my every new journey. I am eternally gratefully for every opportunity they have afforded me.

To my colleagues at the Limkokwing University of Technology for giving me laughs in my times of depressions, I would like to show my appreciation in the form of this work.

To my supervisor Dr. Muthoni Masinde, thank you for introducing me to a whole new world and guiding me through it.

-Ts'itso

Declaration

This dissertation is a presentation of my original work. Wherever contributions of other people are used, due effort was expended to indicate and acknowledge them with references to literature. This work was done under the supervision of Dr. Muthoni Masinde, Department of Information Technology at the Central University of Technology, with considerable input from Professor Antoine Bagula, Department of Computer Science at the University of Western Cape

Ts'itso Emanuel Maphats'oe

Date: ………………………….

Signature: ……………………

In my capacity as supervisor of this dissertation, I certify that the above statements are true to the best of my knowledge

Dr. Muthoni Masinde

Date: ……………………….

Signature: ………………….

Acknowledgments

I would like to thank first and fore most my supervisor Dr. Masinde. In as much as you had so much to do, you did not relent in your pursuit to get me all the resources I needed for my experiments. You went over and above to cater to the need of our little research group, which is growing ever so quickly. From that fateful moment where I stumbled in your office, lost, trying to understand the procedures to be followed in order to be enrolled for a master's degree at the University, you have been with me every step of the way and guiding me to a whole new world in the field of information Technology, that of Internet of things.

Secondly, I would like to acknowledge the effort of Professor Antoine Bagula, who offered us ideas of alternative approaches whenever we had an obstacle in our way.

My special thanks go to my colleagues at the Limkokwing University of Technology in Maseru, who always demanded to see my progress and kept me on my toes.

Over and above all, I thank God, who gives me strength to move from one challenge to another.

Abstract

The world is embracing the idea of Internet of Things and Industrial Revolution 4.0. However, this acceptance of computerised evolution is met with a myriad of challenges, where consumers of this technology are also growing ever so anxious about the security of their personal data as well as reliability of data collected by the millions and even billions of sensors surrounding them.

Wireless sensor networks are the main baseline technology driving Internet of things; by their very inherent nature, these networks are too vulnerable to attacks and yet the network security tools designed for conventional computer networks are not effective in countering these attacks. Wireless sensors have low computational resources, may be highly mobile and in most cases, these networks do not have a central point which can be marked as an authentication point for the sensors, any node can join or leave whenever they want. This leaves the sensors and the internet of things applications depending on them highly susceptible to attacks, which may compromise consumer information and leave security breaches in situation that need absolute security such as homes or even the cars they drive. There are many possibilities of things that could go wrong when hackers gain control of sensors in a car or a house.

There have been many solutions offered to address security of Wireless Sensor Networks; however, most of those solutions are often not customised for African context. Given that most African countries have not kept pace with the development of these underlying technologies, blanket adoption of the solutions developed for consumption in the developed world has not yielded optimal results.

The focus of this research was the development of an Intrusion Detection System that works in a hierarchical network structured Wireless Sensor Network, where cluster heads oversee groups of nodes and relay their data packets all the way to the sink node. This is a reactive Intrusion Detection System (IDS) that makes use of a fuzzy logic based algorithm for verification of intrusion detections. This system borrows characteristics of traditional Wireless Sensor Networks in that it is hosted external to the nodes; that is, on a computer or server connected to the sink node. The rational for this is the premise that developing the system in this manner optimises the power and processing resource of nodes because no part of the IDS is found in the nodes and they are left to focus purely on sensing.

The Intrusion Detection System makes use of remote Over The Air programming to communicate with compromised nodes, to either shut down or reboot and is designed with the ZigBee protocol in mind. Additionally, this Intrusion Detection System is intended to being part of a larger Internet of Things integration framework being proposed at the Central University of Technology. This framework is aimed at developing an Internet of Things adoption strategy customised for African needs and regionally local consumers.

To evaluate the effectiveness of the solution, the rate of false detections being picked out by the security algorithm were reduced through the use of fuzzy logic systems; this resulted in an accuracies of above 90 %. The algorithm is also very light when asymptotic notation is applied, making it ideal for Wireless Sensors. Lastly, we also put forward the Xbee version of the Triple Modular Redundancy architecture, customised for Wireless sensor networks in order to beef-up on the security solution presented in this dissertation.

List of Acronyms

WSN	Wireless Sensor Networks
IDS	Intrusion Detection System
DOS	Denial of Service
SOA	Service Oriented Architecture
GPRS	General Packet Radio Service
TP	True Positive
FP	False Positive
TN	True Negative
FN	False Negative
UML	Unified Modelling Language
DSM	Domain Specific Modelling
OPM	Object Process Modelling
GUI	Graphical User Interface
OTA	Over the air programming
RAD	Rapid Application Development
TMR	Triple Modular Redundancy
MEMS	Micro Electrical Mechanical Sensors

Contents

List of Figures

List of Tables

Publications

MAPHATSOE, T. and MASINDE, M. , 2016, A Security Algorithm For Wireless Sensor Networks In The Internet Of Things Paradigm, Cunningham, P. and Miriam Cunningham,M., eds. In: *2016 IST-Africa Conference*, May 11-13 2016, IEEE xplore digital library, pp 1-10

1. Background and Motivation

1.0 Introduction

The Internet of Things archetype makes considerable use of Wireless Sensor Networks for basic data collection of the deployment environment. The advances in Micro Electrical Mechanical Sensor (MEMS) engineering and the widespread adoption of wireless communication protocols, for device to device communication, have aided immensely in the popularity of wireless sensor network development and usage. Unfortunately, wireless sensor networks also provide a new door way for malicious software as well as hackers to harm the integrity of data gathered for the numerous Internet of Things applications such as home security, farming as well as smart city applications such as pollution detection.

Wireless Sensor Networks (WSNs), have protection from malicious intrusion from the Internet through means of cryptography and firewalls in some instances. However, cryptography techniques cannot detect intrusion from within the network as well as physical tampering of a node from within the network. They can also be vulnerable to internet-based attacks such as denial of service attacks (DOS).

The aim of this research was to investigate, discover and enhance or come up with an intrusion detection system (IDS) to detect security breaches coming from within the WSN as well as those that could by-pass the cryptography security measures implemented within the WSN, from the perspective of ZigBee based WSN.

Additionally, the scope of this research includes an implementation of the IDS application as software, followed by controlled experiments for intrusion detection tests to evaluate the effectiveness of the IDS.

1.1 Background

Wireless sensor networks have made it easy to visualize and implement a variety of novel ideas because of their flexible wireless nature and small portable size. The range of applications of WSNs are inclusive of simple every day uses such as automated doors; to life critical scenarios such as detection of natural disasters and possible overflow of a dam and even military applications, where they are often deployed to gather information in hostile environments (Stankovic & Wood, 2011). They can also be used to combat drought (Satish, et al., 2009) which is an advantage for the agricultural sector.

The term Internet of Things (IoT), was first used in a presentation in 1999 by Proctor and Gamble employee, Kevin Ashton, when linking the Radio Frequency Identifier (RFID) to the Internet (Dassault Systems, 2013). It has now evolved to describe a sector of information technology and research aimed at connecting all objects, electronic or living organisms, to the Internet (Lopez Research, 2014), with the aim of efficient communication of all interconnected objects. Each device or organism will have its own unique identification (in the form of an IP address) such that it can connect to the internet (David, et al., 2012). WSNs are a crucial aspect in the realization of the IoT implementation because sensor devices are small, portable and can be implemented in a variety of applications.

It is imperative therefore to conclude that WSNs have to remain as secure as possible from any form of intrusion or security breaches (Butun & Morgera, 2014). This will bring a sense of acceptance from the end users entering the era of the Internet of Things (IoT), who should also be confident that the information they receive is reliable and not tampered with.

For example, a farmer using a WSN to automate the irrigation system should have full confidence that the sensors will not let the soil dry beyond the desired point or let plants wither in heat before deploying the sprinkler system or alerting the farmer. Likewise, the military should be confident that information received from sensors has not been tampered with by enemies to give wrong information, as well as a dam relying on sensors to warn of a possible flooding before it happens (Seal, et al., 2012).

The type of security threats usually directed at WSNs can be classified by schemes such as Outsider-Insider and passive-active (Lee, et al., 2012). An example of an Outsider attack could be one coming from the Internet which could be active in harming the nature of the network such as Denial of Service (DOS) attack (Krishnan, 2014) or passive such as the listening of an authorized user authentication’s details. On the other hand, Insider attacks range from situations where someone physically captures a node and reprograms it such that it gives wrong information or for it to harm the integrity of the network from within (Honus, 2009), to someone, authorized with encryptions keys, using software to inject malicious software into the network (Lee, et al., 2012).

Intrusion Detection Systems (IDS) have long been researched on and employed in networked systems as secondary security measure to encryption mechanisms that usually focus on Outsider attacks (Soliman, 2012). They aim at detecting threats to the network and to report these threats, such that an action is performed to prevent threats before much damage is inflicted to the network (Priyanka, 2013). They can also keep a log of all threats to a network which could be vital for analysis purposes (Strikos, 2007).

1.2 Problem Statement

Africa, as a continent, has always lacked behind when matters of technological advancements are concerned (Masinde & Bagula, 2015). Generally, technological advancements come to Africa once they have been pioneered in other continents such as Asia, America and Europe. However, the technologies received from such continents are usually not augmented to be specific for the needs of Africa and the African landscape. For WSNs to be trusted in the African consumer markets, their security solutions should be customised to be compatible with African needs.

Given the above background information, it is evident that; there is need to develop or enhance existing IDS to improve on general knowledge in the realm of IDS technology, in the quest to come up with an exceptionally resource friendly IDS for WSNs. The IDS chosen should be portable and light enough, computational cost-wise, to work efficiently given the limited resources inherent in WSNs, given the possibility of remote deployment in Africa. Additionally, the IDS should also be effective enough to detect if one or more of the nodes has been physically or otherwise breached, as well as to detect if the traffic coming from the network side is not one that can constitute a DOS attack or some passive listening to pick up end-user authentication.

Furthermore, the resulting IDS should be compatible with the framework in (Masinde, 2013).

1.3 Motivation for the Research

The motivation of this study was partially the prospect of being a part of a larger research project inclined towards creating a generic integration framework for IoT, which is to be optimised for developing African countries, at the Central University of Technology, Bloemfontein, South Africa (Masinde, 2013).

The proposed integration framework will consist of four (4) layers, governed by the three components of the Service Oriented Architecture (SOA); service description, service advertisement and service composition respectively (Jiehan, et al., 2007). Layer 1 of the four layers consist of Information Generators such as WSNs, people and other objects that can produce the raw data being recorded for a specific domain, such as thermometer in weather applications. Layer 2 has the main task of integrating all the raw information from a variety of information generators from layer 1. Since the framework will be generic but used across different applications, layer 3 is the first application specific layer that receives the integrated data from layer 2. Layer 3 will be specific for the needs of each unique application. Layer 4 is the layer responsible for formatting the information output to a specific output device, ranging from computers, mobile phones or loud speakers and any other way that may benefit the end users.

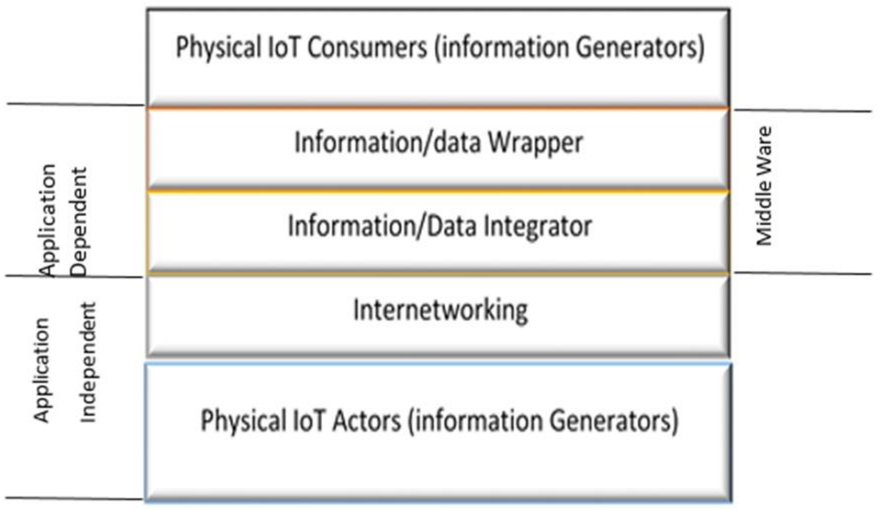

Physical IoT Consumers (information Generators)
Information/data Wrapper
Information/Data Integrator
Internetworking
Physical IoT Actors (information Generators)
Application Dependent
Application Independent
Middle Ware

Figure 1-1 Proposed IOT integration architecture at Central University of Technology (Masinde, 2013)

1.4 The Solution

An existing algorithm that had been used to combat flood attacks in ZigBee powered sensors was enhanced with fuzzy logic to create a desktop IDS for WSNs with a build in database. The ZigBee gateway device for a network connects to this application and even stores sensor data into the database for third party applications to use once the data has been considered legitimate.

The IDS developed is a reactive software and makes use of over the air programing (OTA) to either reset or shut down offending sensors motes. This dissertation documents all the steps taken to arrive at the solution.

1.5 Objectives of the Research

The overall objective of this research was to enhance and implement an intrusion detection algorithm for wireless sensor networks in the Internet of Things integration architecture. This was achieved through the following sub-objectives.

1. To investigate and analyse the concepts and designs of existing intrusion detection system in relation to WSNs.
2. To enhance an intrusion detection algorithm, as part of designing an intrusion detection mechanism, to improve security within WSNs. This algorithm should be:
    i. Able to deal with Jamming, Flooding and De-synchronization attacks, of which all fall under the Denial of Service, attack (DoS).
    ii. Light-weight and able to work well given the limited processing capabilities of WSNs while being as energy efficient as possible. That is because nodes within WSNs use batteries that could be

difficult to charge, if a WSN is deployed in a hostile area or if too many have been deployed over an area.

iii. Compatible with the framework (Masinde, 2013)

3. To implement and evaluate the intrusion detection algorithm

1.51 Research Questions

The research objectives were achieved by answering the following research questions:

1. What are the existing IDS that are relevant to WSNs?
2. To what extent does the physical placement of IDS in the network affect the efficiency of IDS?
3. Which artificial intelligence notation can be used in WSN development?

1.6 Methodology of Research and Evaluation Criteria

The work presented in this dissertation is in the form of an experimental research with the aim of enhancing an existing IDS algorithm.

With Algorithm enhancement, a dynamic algorithm, found to be suitable, was improved and analysed to determine both its correctness (as intrusions detector) and efficiency (computational time and computer resources consumption). The algorithm was considered correct with respect to intrusion detection because for every instance of the inputs (that is, when the specified properties of the inputs are satisfied), the algorithm halts, and the outputs produced satisfy the specified input/output relation. This algorithm was first augmented with properties of fuzzy logic and realised (coded), as part of an Intrusion Detection System software.

The implemented algorithm is used to run various tests to determine its performance. The scope of this project does not allow for the IDS solution to be tested on real physical outdoor applications but rather in a controlled environment where a threat can be simulated when and as needed.

Further experiments are carried out to determine the performance of the planned IDS, when dealing with varying intensities of Internal and External flood threats. Internal Threats are inclusive of physical tampering. In this instance, the IDS is evaluated on how quick it can detect a node being compromised and how quick it can offer a solution.

The results from the experiments are used in conducting an assessment, where relevance of threat, adequacy of information about threat provided to administrator, frequency of false and positive alarms and the number of steps used by the IDS to identify an intrusion are evaluated.

1.7 Limitation of the study

The evaluation of the algorithm is done indoors with limited number of nodes and simulated attacks as opposed to using real threats. The use of real world threats as a method for measurement could have only been possible if this was a study over a relatively long period of time. This limitation also implies that the evaluation could end up with results that are not a true perfect representation of threats in the real world.

1.8 Contributions of the Research

This research is instrumental in creating a dynamic custom-made security API and to make a contribution towards the development of a localised generic WSN applications integration architecture within the IoT paradigm. Furthermore, it will bring forth confidence, to end users of WSN based networks, that information harvested from networks is tamper free and those threats, which can compromise the integrity of the network, are drastically minimised.

Additionally, a customised approach in the development and deployment of the IDS will enhance the maintainability of a locally designed application of WSNs.

The result of this research is a dynamic, reactive Intrusion Detection System for ZigBee powered WSNs aimed specifically at ZigBee flood attacks. The IDS uses fuzzy logic to maximise accuracy. The IDS is resource friendly with regard to actual sensors as it is hosted externally to the network, but at the point in the network where all the data eventually ends. There is also a database that is ready to connect with third party applications that consume information coming from the network.

.

1.9 Dissertation Structure

This dissertation is structured to give every detail of the development of the IDS, to the testing of the completed solution.

Chapter 2-is the literature review: This is a review of the associated literature, pertaining to WSN, the IoT phenomenon, and threads directed at WSN, communication protocols used in WSN as well as proposed security solutions for WSN. On the issue of security we also explore literature behind the use of artificial intelligence notations to enhance security in WSN.

Chapter 3- Methodology: This chapter describes the research methods that were used in this study, also with justification for the selected approach.

Chapter 4- IDS Design: The chapter documents the design behind all the components that make up the final solution.

Chapter 5-Implementation: This chapter gives a description of how the algorithm was enhanced and coded into a working solution. It also describes how the various components work together to produce make one product.

Chapter 6- Evaluations, Discussions and Conclusions: This chapter describes the tests that were used to evaluate the IDS, the possible enhancements to the IDS as well as the conclusion of the dissertation.

2. Literature Review

2.1 Introduction

Wireless sensor networks (WSNs) consist of networks mainly composed of autonomous sensors for measuring a variety of data such as habitat and ecosystem monitoring, seismic monitoring, ground water contamination, water acidity and even civil structural health monitoring (Bhaskar, 2005). These sensors use diverse communications technologies such as Bluetooth, ZigBee, General Packet Radio Service (GPRS) and Wi-Fi (Libelium, 2016), to connect to the sink nodes, which are control nodes that are connected to computers or the Internet. Most of the sensors are small and easy to implement in a variety of surfaces, devices as well as existing infrastructure. This is the basis of the phenomenon referred to as Internet of Things (IoT). With IoT, everyday appliances and electronic devices are being fitted with sensors that relay the information they monitor to communication devices that end-users can interact with such as computers and cell phones (Weber & Richard, 2016).

This IoT sensation is giving birth to and driving forth innovative ideas such as smart cities, smart farming as well as automation of everyday devices in a home scenario. For example, in a smart city, traffic operation could work with embedded sensors that can alert motorists about busy routes, roads that have experienced an accident, the average speed of cars moving on a certain road as well as alternative roads that can be utilised in cases of emergency (Nikita, et al., 2016).

Additionally, city councils could be alerted automatically by dustbins when it is time to collect trash. In the home scenario, fridges are already being implemented with sensors to detect where there is shortage of certain items (Prapulla, et al., 2015). This idea can further be expanded to a point where the fridge can make an order, automatically, of the depleted item, on behalf of the owner. This is a brief indication of the possibility of numerous ideas that will be brought to fruition when IoT is fully embraced and applied.

The advent of IPv6 is also another driving force behind IoT as hundreds, thousands, millions and even billions of sensors will be able to connect to the Internet without the fear that the Internet Protocol will run out of address space (Antonio, et al., 2013).

When sensor nodes were first developed, they made use of simplex communication channels, that is, they could only sent signals to the sink nodes and not vice versa. Newer models are bidirectional meaning that communication between a sensor and a node or computer happens in a half-duplex or full duplex scheme (Silva Girão & Enache, 2007). However, to remain effective when deployed, these WSNs need protection from intrusions and pervasive attacks directed at defilement of their integrity.

This chapter therefore presents and discusses network security in the form of Intrusion Detection Systems (IDS) as well as WSN specific IDS, with particular reference to algorithms used for detection of intrusions.

Additionally, this chapter is divided into the following sections. Section 2.2 is dedicated to Wireless Sensor Networks, detailing their usage and their various deployment topologies. Section 2.3 look at the aspects considered in the design of algorithms used in intrusion detection as well their analysis. Section 2.4 and 2.5 discuss Intrusion detection systems and algorithms as well as a special focus on WSN IDS algorithms respectively in section 2.6. WSN simulators are discussed in section 2.7. In section 2.8 we discuss the protocol used in our approach to intrusion detection. Related work is presented in section 2.9.

2.2 Wireless Sensor Networks

WSNs can be deployed in a variety of topologies, such as star and mesh topology, depending on the preference and needs of the people using the sensors (Kosmerchock, 2012). For instance, in a star topology, all sensors connect to the sink node while in a mesh topology nodes can also communicate within themselves. Generally, there are two main network structures of wireless sensor nodes, being flat as well as hierarchical network structures (Abduvaliyev, et al., 2013). This research, however, gravitates towards study of the latter in an attempt to satisfy the aims set by the research. Under Hierarchical network structures there are tree, cluster as well as hierarchical tree networks.

2.2.1Tree network architecture

In this architecture, the base station of the network, being the main node connected to a computer or Internet for data analysis and aggregation, acts as the main parent of the network. WSNs work by relaying data packets from one node to a higher node in the network towards the base station, with each relay called a hop. The higher node acts as the parent of the lower node, as can be seen in Figure 2.1 below.

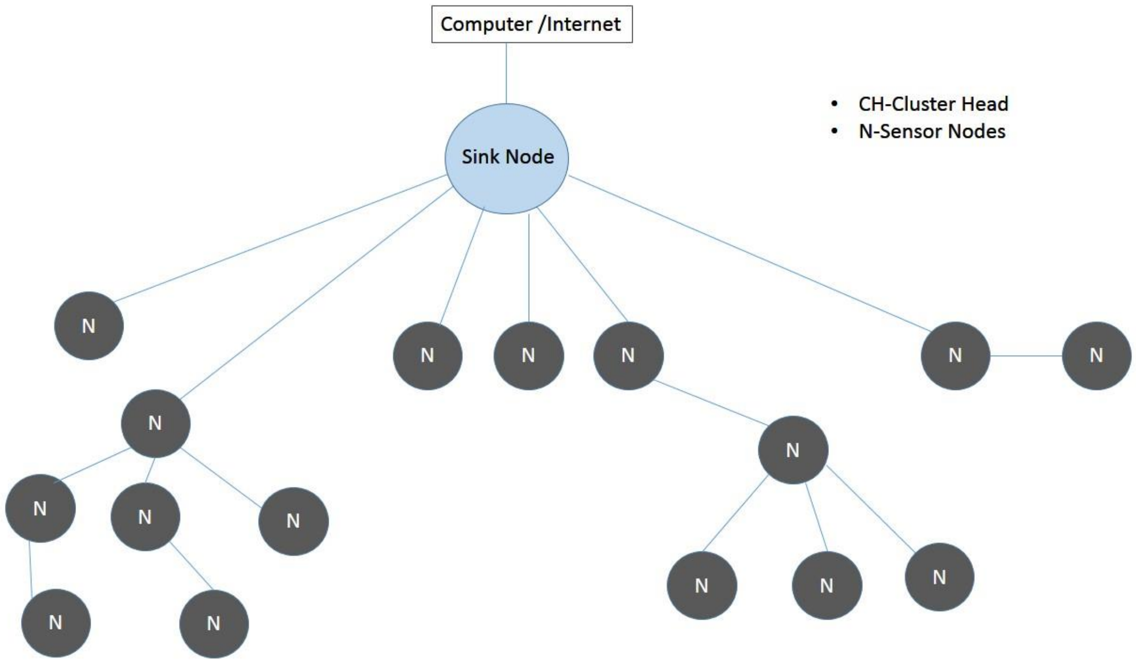


Figure 2-1The Tree Network Structure

2.2.2 Cluster network architecture

With the cluster network structure, there is division of nodes into smaller groups called clusters, where each cluster has a managing node called a cluster head that communicates directly with the base station, as can be seen in Figure 2.2.

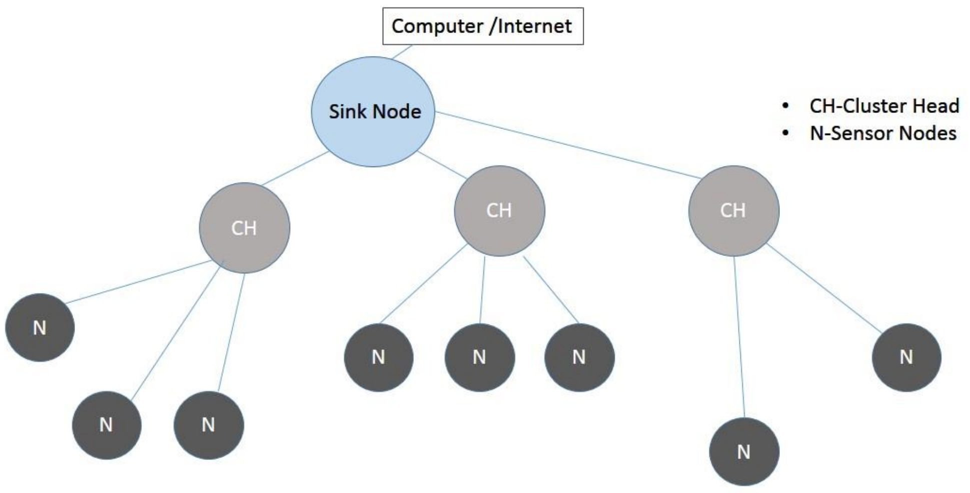


Figure 2-2Cluster Network structure

2.2.3 Hierarchical Tree Network Structure

In this topology network, there exist clusters of nodes where each cluster head communicates directly with the base station. Additionally, each cluster head acts as a parent node to group of nodes, the nodes in this scenario must be grouped in a fashion similar to that of a tree network structure. Figure 2.3 shows an example of a Cluster network.

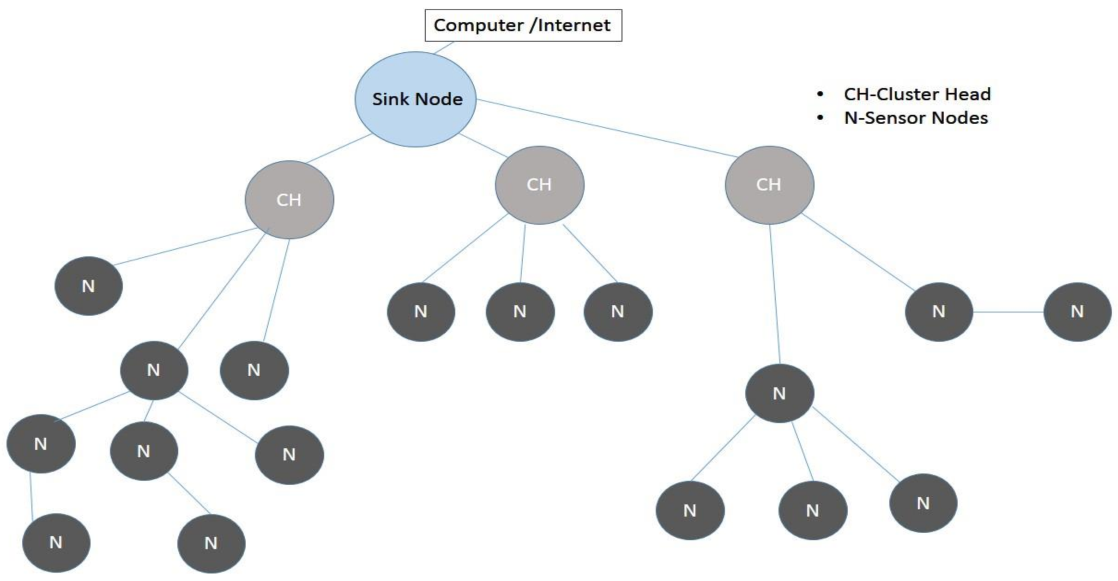


Figure 2-3 Hierarchical Tree Network Structure

2.3 Design and Analysis of Algorithm

Algorithm design can be described as a precise technique for creating a mathematical process for solving problems (Levitin, 2003). The success or the effectiveness of an algorithm is determined by its run time also commonly known as the BIG-OH (Mount, 2003). Big-oh (*O*) is used to check the efficiency of an algorithm in terms of processing time and memory requirements with varying input sizes (Kalle, et al., 2014).

Essentially, when dealing with algorithm analysis, the number of steps taken by an algorithm, to execute inputs of varying magnitudes are counted with the best, worst and average case complexities being determined (Tovey, 2002).

2.3.1 Best, Worst and Average case Complexities of algorithms

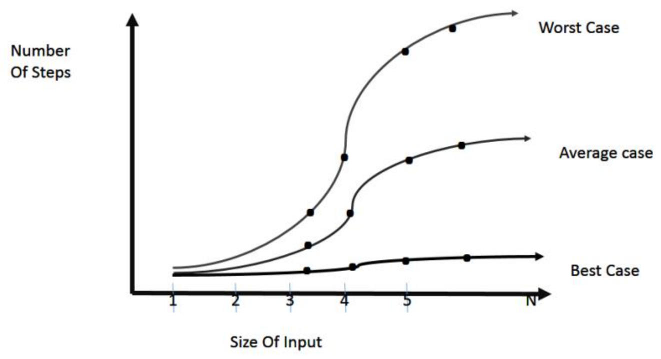


Figure 2-4 Best, Worst and Average Case Complexities. Image Source: (Anon., 1997)

For instance, one may consider an algorithm that searches for a certain key from the input set consisting of all possible sizes of the search interest, represented on the x-axis as shown in figure 2.4 above. The y-axis represents the number of steps the algorithm would need to take in order to find the key being searched for. In this scenario, the worst-case complexity of the algorithm is given by the function that takes the maximum number of steps with any given size of input, to find the search key, as can be seen in the graph. Conversely, the best-case complexity of the algorithm is produced by the function that takes the least number of steps to find a search key, given any size of the input. Finally, the average case complexity of the algorithm is specified by the function that takes the average number of steps to produce the desired output, given any size of input.

Nevertheless, in practice, the worst-case scenario is usually the measure that really counts, because it determines just how poor an algorithm can ever perform given any size of input (Engblom, 2002).

Thus said, it has been pointed out that when working with best-average-worst case complexities, it is often complex to figure out the exact number of a computer's random access memory executions because of the need to have much detailed information as well as specifications such as the approach used when coding the algorithm (Shaffer, 2009). This is where the Big-Oh notation comes in handy because it is simpler to describe the efficiency of an algorithm using basic upper and lower bounds of time complexity. Additionally, the big-oh places an abstraction on algorithm analysis that ignores details that have no impact in the comparison of algorithms (Jones & Pevzner, 2004).

For example, given an algorithm function with a multiplicative constant *g(n) =3n* and one without a constant multiplier, such as *f(n) =n*, the big-oh ignores the multiplicative constant and sees them as similar. This in fact makes sense when dealing with coding the algorithms in two different languages because the multiplicative constant would tell us nothing about the algorithm when one runs 3 times as fast in one language as compared to the other in a different language (Jones & Pevzner, 2004).

The formal definitions for big-oh can be given as follows (Koffman & Wolfgang, 2010);

- Upper bound (big-oh): $f(n) = O(g(n))$ means $c*g(n)$ is an upper bound on $f(n)$. This means that there exists a constant c such that $f(n)$ is always $<= c*g(n)$ where $n>=n_0$ for some constant n.
- Lower bound (big-omega): $f(n) =\Omega (g(n))$ means $c*g(n)$ is a lower bound on $f(n)$. This means that there exists a constant c such that $f(n)$ is always $>=c*g(n)$ for all $n>=n_0$.

- Big-Theta: *f (n) = Ɵ (g (n))* means there exist constants c_1 and c_2 such that c_1*g (n)* is an upper bound on *f (n)* and c_2*g (n)* is a lower bound on *f (n)* for all $n>=n_0$. In short $f (n) <=C_1*g (n)$ and $f (n)>= C_2*g (n)$. This roughly put means that g (n) is bound averagely around *f (n).*

Examples of the above are defined as follows:

Big-Oh

- $6n^2-100n+9=O (n^2)$ when c=3. $3n^2>6n^2-100n+9$
- $6n^2-100n+9=O (n^3)$ when c=1 and n> 3 then $n^3 >6n^2-100n+9$.
- $6n^2-100n+9\neq O (n)$ because for any C chosen where n>c, $c*n<6n^2$

Big-Omega

- $6n^2-100n+9=\Omega (n^2)$ when c=2 and n>100 then $2n^2 <6n^2-100n+9$.
- $6n^2-100n+9\neq\Omega (n^3)$ if c=1 and n>3 then $6n^2-100n+9< n^3$
- $6n^2-100n+9= \Omega (n)$ for any c chosen because $c*n<6n^2-100n+9$ when n>100c

Big-Theta

- $6n^2-100n+9=\Theta (n^2)$ because it is both in *O* and Ω.
- $6n^2-100n+9\neq\Theta (n^3)$ because it is only in Big-Oh
- $6n^2-100n+9\neq\Theta (n)$ because it is only in Big-Omega

Therefore, based on the above definitions, better conclusions can be drawn when functions are compared in terms of efficiency, given large enough or continuously increasing input. When used as part of an intrusion detection system, for example, algorithm analysis could be invaluable in the testing stage or simulation stage when checking how the algorithm will behave over time even with growing amounts of data packets being sent or an increase in intrusions.

2.4 Intrusion Detection Systems and Algorithms

Intrusion Detection Systems (IDS) could be in the form of hardware or software, and were introduced as a secondary security for networks (Ning & Jajodia, 2002). This is because IDS are more concerned about finding or sniffing out intrusions before or as they happen and alerting system administrators as opposed to just focusing on securing transmission of data, carried out using encryption schemes (SANS Institute, 2001). This is why encryption schemes are considered to be the first level of security for networks. Intrusion detection systems have been used for wired networks but with the introduction of WSNs, a need has risen to have intrusion detection systems in wireless networks as well (Khan, et al., 2014).

To make clear the need to have Intrusions Detection Systems in networks, it is important to outline the types of intrusions and the reasons as to why systems need to be protected from them.

The concept of intrusion was introduced in the 1980's and described as a deliberate unauthorised attempt to gain access to secured information, manipulate the information or render the system unusable or useless (Neethu, 2013).

2.4.1 Types of Intrusions

Various intrusions have been designed to cause harm to computer systems. These intrusions have been classified according to their attacking techniques (Cort, 2004).

- Exploits- these are hidden bugs within code of system or servers which will later allow the intruder to gain unauthorised access to a system
- Reconnaissance attacks-these attack types aim at finding weaknesses in system and to exploit such a weakness to gain access to systems.
- Denials of Service attacks- these kinds of attacks are build such that they flood the system with unnecessary requests until said system crashes.

However, the above-mentioned only form half of the class of intrusions, as they all fall under attacks directed to the system from outside the network. Some attacks however, do come from within the network and are classified as insider attacks (Probst, 2011). A user who already has authorization usually initiates insider attacks and this user could be anyone from a disgruntled employee to an employee who has retired or changed jobs (Rupert, 2009).

2.4.2 Intrusion detection systems (IDS)

An IDS is a system, which monitors data traffic coming in and out of a network to determine if there has been intrusions or attacks (Liao, et al., 2013). In the event of an apparent attack, some IDS are designed to alert system administrators who, in turn, take the appropriate response to protect data, while some IDS are programmed to act accordingly on their own in the case of an apparent attack (Cort, 2004).

These IDS can be categorised according to area of deployment in a system, form or reaction to an intrusion as well as detection technique. For example, when referring to area of deployment, which is the major classifier of IDS, the IDS could be Host based or Network based. These titles are self-explanatory in the sense that with **Host based IDS**, the IDS is installed on a particular node of a network, be it a work station or peripheral, to monitor traffic (Letou, et al., 2013).

On the other hand, Network based IDS are installed on strategic points of the network to monitor traffic passing through the network (Amiri, et al., 2011). However, both of these implementations have drawbacks. For instance, with Host Based IDS, one of the disadvantages include the fact that they can be difficult to maintain in a large network that has multiple computers and nodes, as the IDS is on every node of the network (Innella, 2001). This also means it is difficult to analyse attack attempts if many computers in a network are targeted. Additionally, Host Based IDS have the problem of interoperability and portability because they need to be installed on operating systems of different manufactures in a network (Sharma & Sinha, 2011). This implies that there will be instances of operation, when some operating systems that control nodes in a network are not compatible with the IDS especially in a large network setting. This fact of the IDS needing an operating system to operate on means that when attackers find vulnerability in the OS platform, they can use it as a back door to the IDS and destroy it from within. These types of attacks are very hard to trace and they leave vulnerability that is difficult to rectify. Furthermore, Host Based IDS are very poor at sniffing out outsider attacks (Anon., 2005).

Moreover, Host based IDS are not usually resource friendly. For example, most Host based IDS need to keep a record of system audit trails to find a source of intrusion and usually this ends up being a growing database that takes much system space as well as increasing Host Server load (Anon., 2005). However, this is a two-sided affair as Host based IDS make it easy to trace the source of intrusions, and this is a good feature when finding Insider attacks. Additionally, based on stored audit trails of a node in a network, if a node has been under attack previously, a similar reoccurrence can be sniffed out and stopped as it happens or before it happens. Moreover, all traffic to such a node can be stopped as a preventive measure. An example of a Host Based IDS is CyberSafe (Anon., 2014).

Network IDS on the other hand have the disadvantage that, since they have to monitor every packet in the network for known signs of intrusions, they sometimes have trouble doing this in networks at very high speeds (Li, et al., 2005). This does not seem like a problem that will go away soon as Internet and network speeds are constantly increasing with the evolution of technology. Furthermore, Network based IDS have to identify certain known signatures of attacks from packet traffic, implying that unknown or novel intrusions can go unnoticed and these are the ones that cause much harm (Li, et al., 2005). This is also amplified by the fact that there are just too many known attacks to be programmed into one IDS during creation (Anon., 2005). Network IDS tend to also be ineffective in networks where data packets are encrypted as they cannot evaluate packets for known signatures (Draˇsar & Vykopal, 2013).

However, just as in Host based IDS, Network based IDS also have their merits. For instance, as opposed to Host based IDS weakness of portability, Network based IDS are independent of platform on which they are running, making them desirable in large network settings that incorporate different operating system architectures (Albag, 2001). It has been stated that they are a lot easier to install and to have up and running in a relatively short time-frame (Bowden, 2007). This could be a useful feature in a setting where the deployment topologies are ever changing or there is re-scaling of a corporation. One example of a Host based IDS is Snort (Inella, 2001).

As an alternative, there are **Hybrid IDS** that try to combine the advantages of both approaches to provide the optimum solution for intrusion prevention. For example, a hybrid IDS would be able to check if a data packet is suspicious as in Network based IDS, ascertain as to whether it has been stored as a known attack and even provide tracing ability as in Host based IDS, to find the source of the attack (Aydin, et al., 2009).Table 2.1 displays the advantages and disadvantages of Host Based IDS as well as Network Based IDS.

Table 2-1 Advantages and Disadvantages of Host IDS and Network IDS

Classification of IDS	**Advantages**	**Disadvantages**
Network Based IDS	-Platform Independent -Relatively easy to configure and set up	-trouble coping with high speed networks -Novel attacks may go undetected
Host Based IDS	-can easily trace Inside attacks -stored audit trails makes it easy to combat re occurring attacks	-difficult to maintain in large area -Not resource friendly

With regard to type of reaction to intrusion, the IDS, be it Network or Host based can take one of two forms. It can have a passive reaction to an intrusion or it can be reactive in nature.

Passive reaction IDS respond to an intrusion by keeping a record of the intrusion and alerting the system administrator to take a preventive measure. **Reactive IDS**, however, usually respond by taking off the infected node from the network or re setting the connection from the infected node (Kumar & Chhabra, 2014).

Finally, there are also two types of detection techniques deployed in intrusion detection system. These are anomaly and signature detection techniques. With Anomaly Detection Technique, data packets are inspected in a network or from a node, for deviation from a set baseline that is deemed normal (Jyothsna & Rama, 2011). If the traffic is considered abnormal, an alarm is raised.

On the other hand, with Signature Based detection techniques, the IDS checks the data traffic in a network or from a node, for known sets of attack patterns, stored in its database. If a match is found an alarm is raised (Gaikwad, et al., 2012).

2.4.3 Algorithms used in Intrusion Detection

Various algorithms are used for the detection of impositions in intrusion detection systems. Most of these algorithms are borrowed from data mining algorithms and statistical analysis algorithms as they work similarly in data aggregation and analysis. For example, some popular algorithms from data mining such as fuzzy logic and genetic algorithms are used for anomaly detection (Hassan, 2013).

Fuzzy logic algorithms use rule-based logic to find information about the state of a system even when incomplete data is provided, because they try to incorporate the dynamics of human reason (Taylor & Meldrum, 2000). This characteristic makes them to not give answers as strict yes or not but also possibilities of the in between. Fuzzy logic systems therefore, can give computers the ability to produce calculated reasoning in a way similar to how a human would think (Pasupathi & Nishanth, 2014). There are not just black and white areas of logical deductions in fuzzy logic as grey areas, or the in between possibilities are accommodated for.

Fuzzy logic also gives computers a chance to make conclusion about a situation even when partial information, about that situation, is given (Johnson, 2002). For example, if a human were to somehow act as an IDS they would realise when a sensor is actually acting up and cannot be trusted, or if the sensor just needs to be rebooted, or isolated completely. This is very same concept that the fuzzy logic module brings into this application. The ability for a human to reason and to evaluate situations such as temperature as either hot, warm, mild or very cold, is afforded to computers through concepts of fuzzy logic.

Genetic algorithms, on the other hand, are inspired by the natural evolution of living organisms that include natural selection, inheritance and mutation (Scrucca, 2013). As such, they have been used to find solutions to search problems in computer sciences. Selection, inheritance and mutation mechanisms are the building blocks of genetic algorithms. With selection, a solution that is considered best has a higher chance of being selected in a search solution state. In inheritance, old solutions are combined to form new ones and in mutation, there are random changes done to the solutions space domain in a bit to keep evolving the best answer and making it the optimal answer (Ranjan Srivastava & Kim, 2009).

Statistical algorithms work by finding a deviation from a traffic pattern, when used in intrusion detection systems. This deviation is considered an anomaly (Obaid Amin, et al., 2009). An example of a statistical algorithm that can be used in intrusion detection system is the Cusum (Ravi, 2014). Cusum literally means cumulative sum and it is defined below.

Let n be a sample size greater or equal to 1 (n>=1)

Equation 2.1 Cusum evaluation

$$S_i = \sum_{j=1}^{i} (\bar{X} - \mu_0)$$

Where $\bar{X}$ is the average of the J^th sample and μ_0 is the in control system mean. If the Cusum value revolves around 0, the system is in control but if mean shifts value. For example if $\mu_1 > \mu_0$ or $\mu_1 < \mu_0$, it means the system is in abnormal state.

2.5 Intrusion Detection for Wireless Sensor Networks

Intrusion detection systems for WSNs work similarly to their wired counterparts and can even deploy similar algorithms and architectures, such as use of mobile agents (Vyavhare, et al., 2012). However, for IDS to be effective in WSNs, they have to be very efficient to cater for the limited resources in WSNs (Pleskonjic, 2003).

WSNs are more vulnerable to intrusions than wired networks. This is amplified by the fact that they can be physically captured and re-programmed or have their wireless signals interrupted and attacked (Mitrokotsa, 2008). It is important to understand the differences between wired networks and WSNs as well as how an intrusion detection system functions in wired networks as opposed to how it would work in a WSN.

For instance, some advanced intrusion detection systems in wired networks can detect an intrusion on a node in a network and alert all surrounding nodes to cut communication with such a node (Anon., 2016).

The problem arises as those IDS solutions cannot be directly applicable to solving security problems in WSNs. WSNs possess limitations such as power, memory and battery life (Abduvaliyev, et al., 2013). This implies that it is imperative for intrusions detection systems to be exceptionally lightweight, first, if they are to be of any use to a WSN.

To understand why there is need for intrusion detection systems in WSNs, it is also important to look at some of the types of intrusions normally targeted at WSNs as well as brief explanations about their nature. Below are types of intrusions in WSNs

2.5.1 Physical tampering

As previously mentioned, tampering of the hardware of the wireless sensor nodes remains one of the biggest problems targeted at the WSNs as there is little to be done to prevent it from ever happening (Padmavathi & Shanmugapriya, 2009).

2.5.2 Black Hole attack

Nodes in a wireless network set-up work by routing messages through shortest paths all the way to the sink. In this attack, a malicious node works to pull all traffic towards itself. It first listens to all routing requests and replies to a victim node, with the messages that it has the shortest path to the sink. From there, it can choose whether it wishes to drop all those packets (Gondwal & Diwaker, 2013). This is harmful to the integrity of the network if for instance, the base station never gets the data packets. The data from the subject monitored would all be lost.

2.5.3 Node replication attacks

In the case that a node is physically captured and its cryptography schemes deciphered, an attacker would have power to replicate one of the nodes and use that node to send unwanted information through the network (Khan, 2013).

2.5.4 Hello Flood attacks

WSNs usually work by sending a packet from one sensor to another until the final destination, when deployed in large numbers over a wide area. For this to happen, the node wishing to send data uses its routing protocol to send a Hello “message” to all nearby nodes to find the nearest neighbour. The neighbouring node then responds and if it is indeed the most nearest, the data packet is passed onto it.

With this kind of attack, this very same concept is exploited. Here an attacker that has enough power may keep sending Hello “messages” over a large field of sensor nodes with the purpose of flooding them (Pal Singh, et al., 2010). All nodes that receive the Hello will try to establish contact with the attacking node endlessly without knowing they are being flooded and this creates traffic within the network, as most legitimate nodes will fail to access the data transfer channel.

2.5.5 Wormhole attack

This particular attack is dangerous, as it does not need any node to be compromised, for it to work. In a wireless network setting, all packets being transmitted have a potential to be overheard and in this instance, the attacker merely makes an offer to sensor nodes to provide a path to base station with the most minimal number of hops (Buch & Jinwala, 2011). It can then transfer these packets to any other location through wireless media, meaning that there is capturing of important information. Instead of the information being directed to the base station, it is then sent toward the location the attacker wants.

2.5.6 Sybil attack

With this kind of attack, a malicious node pretends to be two or more nodes within the network using their legitimate identities (Abirami.K & Santhi.B, 2013). This effectively kills the collaborative nature of the setup of WSNs, as nodes in a sensor network work by relying on each other.

2.5.7 Other Attack Types

All the above intrusions are considered well known or well researched, however there is a whole new class of intrusions directed at WSNs of which are less known. Given below is a summary of the lesser-known attacks (Abduvaliyev, et al., 2013).

Fabrication during reprograming- Many advanced sensors allow re-programming of the sensors and at times while still in field deployment. This re-programming is often carried out remotely, often. This feature becomes vital for fabrication attacks, which are possible when an intruder manages to get into the network and re-programs the nodes without physically altering them.

External stimuli-In some WSNs, some stimuli are considered of higher importance and when they occur, they are given precedence over all packets in the channels to the base station. For example, if sensors are doing seismic monitoring and a particular high reading occurs. When used as an attack, the attacker keeps sending frequent fake stimuli of high importance to the network, possibly causing a jam.

Homing- With homing, the attacker targets the cluster heads to use them for listening on the sensor nodes. Cluster heads are responsible for managing a certain batch of nodes in a region.

Neglect and greet- with this attack, a compromised node tends to give priority and precedence to its data packets as opposed to helping relaying packets from other sensors. It will also develop a tendency to drop all packets from other sensors after acknowledging and accepting them.

Forced delay- with this attack, a compromised node will hold on to data packets before sending them. This attack is made with the intention to make the network slow and to lower its integrity.

Unfairness- Finally, with this attack, a node is made to miss its transmission deadline when it has requested the use of the data packet transfer channel. Table 2.2 is a summary of the above mentioned attacks in WSNs.

Table 2-2 Attacks aimed at WSNs

Common WSN attacks	**Less Common WSN attacks**
Physical Tampering	Fabrication during programming
Black Hole Attack	External stimuli
Node Replication	Homing
Hello-Flood	Neglect and greet
Wormhole	Forced delay
Sybil	Unfairness

2.6. Intrusion Detection systems for Wireless Sensor networks

This sub section looks at some of the Intrusion Detection Systems that have been proposed for WSNs as well as their drawbacks and advantages.

There are three major classes of intrusion detection techniques that have been used for protection of WSNs. These are misuse detection, specification based detection as well as anomaly detection schemes.

2.6.1 Anomaly Detection schemes;

There are several Intrusion detection systems for WSNs classified under anomaly detection schemes. With anomaly detection schemes, just as in wired network intrusion detection systems, there is a set of rules of what is considered normal and if packets transmitted do not conform to such a threshold, an alarm is raised (Islam & AshiqurRahman, 2011). Below is a collection of examples of such intrusion detection schemes.

Statistical Model-Based approach- A sensor builds a model idea about its neighbouring nodes as well as packets they are to expect from such a neighbour. If packets do not match with what is expected from the neighbour, they are discarded (Onat & Miri, 2005).

Centralised approach - An intrusion detection system is implemented at the base station with a collection of information about all nodes and their functional status and all this information is used to check for anomalies (Abduvaliyev, et al., 2013).

Clustering algorithm approach-A fixed-width clustering algorithm is used to model normal behaviour of sensor clusters. If sampled packet size of a cluster is less than the threshold value, it is considered an anomaly (Loo, et al., 2006).

Isolation Table - This is a hierarchical system that includes the base station, cluster heads as well as nodes. Isolation table records anomalies and the cluster heads are responsible for isolating nodes from the network that have been compromised. These tables are generated by cluster heads and sent to the base station for recording (Chen, et al., 2009).

2.6.2 Specification Detection Schemes;

These schemes rely on the system administration to specify behaviour that can be deemed correct and then monitor the system for behaviours that are not normal.

Pre-defined watchdog approach-The system makes use of watchdogs which are nodes that watch on the behaviours of other nodes. When more than half of the watchdogs raise an alarm about a certain node, that node is removed from reliable nodes and the base station is made aware of such a move (Krontiris, et al., 2007).

Decentralised approach-In decentralised IDS, the system goes into a stage of data acquisition, then adding rules to the acquired data followed by intrusion detection, where raised failures are compared with what is referred to as number of expected failure to check for a possible occurrence of an intrusion (Da Silva, et al., 2005).

2.6.3 Misuse Detection Schemes

Misuse detection schemes rely on a defined set of attack signatures of which the network should be protected against. An example is the Watchdog (Abduvaliyev, et al., 2013). With this watchdog intrusion detection, each node has, within it, an IDS and every node in the system does not only sent data packets to destination but also to all neighbouring nodes which have a job to listen to packets being transmitted from that node and checking for stored signatures of attacks.

Table 2.3 depicts these intrusion detection as well as the mentioned examples.

Table 2-3 Intrusion Detection Scheme Classifications

Detection Scheme	**Examples**
Anomaly Detection Scheme	-Statistical Model Approach -Centralised Approach -Clustering approach -Isolation Table approach
Specification detection Scheme	-Pre-Defined Watchdog -Decentralised approach
Misuse Detection Scheme	-Watchdog approach

2.6.4 Limitations of existing Intrusion Detection Systems for WSNs

Most of the mentioned intrusion detection systems above are experimental proposals and not real implementations (Krontiris, et al., 2009). Most of them mention simulations that have not be provided for, such that everyone can evaluate or implement them, making them difficult to study or test beyond the scope in which they are mentioned. Everyone is left to be satisfied with what the authors or researchers of such methodologies place as proof of their existence. Some methodologies such as Game Theory methodology (Agah, et al., 2004) for intrusion testing seem to be too heavy or suitable for use with wireless sensor nodes. Furthermore, there is still not much being proposed for securing the sensor networks from physical tampering. All that exist is proposed mechanisms on what to do once such a tamper has occurred.

2.7 Wireless Sensor Network Simulation Tools

To test how a WSN works given a certain methodology or how the intrusion detection on the network will work, it is much easier, time saving and economical to use simulators as opposed to destroying a whole set of nodes for a simple test. It makes sense therefore that there is an array of simulators that have been developed and used for testing WSNs with each possessing different merit.

It has been stated that WSN simulator tools can be discrete event simulators, which can simulate lots of jobs running on different nodes or they can be trace driven simulators which are used in real implemented systems (Yu, 2011) .Thus said, given below are some of the WSN simulation tools.

2.7.1 Network simulator version Two (Ns2)

This is an open source simulator written in Object oriented C++, it can be used on the Linux platform. It falls under discrete event simulators. Its disadvantages include the fact that it is poor with graphics and uses a scripting language that is not easy to learn and use quickly (Siraj, et al., 2012).

2.7.2 Omnet++

Omnet is written in C++ and it falls under the discrete event simulator class. It can run on Unix-based operating systems as well as windows operating systems. It also provides a powerful graphical user interface that makes it easy to learn and use (Massin, et al., 2010).

2.7.3 Jsim

Jsim is a network simulator written in Java, which has a strong graphical user interface. It also falls under discrete event simulators. It was not developed for simulation of WSNs but it is now being used in WSN testing. It is also open source software (Sobeih, et al., 2005).

2.7.4 Tossim

Tossim is not a real simulator but an emulator that also falls under the discrete event simulator class, written in Python. It is also open source and has a graphical user interface. It was developed with the aim of supporting use in WSNs (Levis, et al., 2003). Unlike simulators that can be run on computers to check how the network will behave, emulators need to be run in real nodes (Yu, 2011).

2.7.5 Atemu

Atemu is written in C and is an emulator that falls under the discrete event simulator class. It provides a graphical user interface. It was developed for use with WSNs, and is open source (Baras, 2004).

2.7.6 Avora

Avora is a simulator with no graphical user interface. It was developed to simulate WSNs written in Java. It is open source and falls under the discrete event simulators class as well (Titzer & Palsberg, 2005).

All of the above-mentioned emulators and simulators are depicted in Table 2.4.

Table 2-4 Wireless Network Simulators and Emulators

Name	**Programming Language**	**Simulation Class**	**Graphical User Interface**	**Platform**	
NS2	C++	Discrete	No	Linux	
Omnet++	C++	Discrete	Yes	Unix/windows	
Jsim	Java	Discrete	Yes	Windows	
Tossim	Python	Discrete	Yes	Sensor Nodes	
Atemu	C	Discrete	Yes	Sensor nodes	
Avora	Java	Discrete	No	Java Sensors	

2.8 Understanding the ZigBee protocol

ZigBee is the protocol targeted in the realisation of the IDS of this research. Therefore, this subsection is dedicated to giving insight into ZigBee and how it functions.

ZigBee is based on the IEEE 802.15.4, and when broken down, its specification can be identified as 802 which is in place for the networking standard; 15 for wireless network and 4 meaning low data transmission rate as well as low power consumption rate (Vishwakarma, 2012).

The architecture of ZigBee is pretty much all of the 802.15.4 stack layers plus the additional ZigBee layer at the top most.

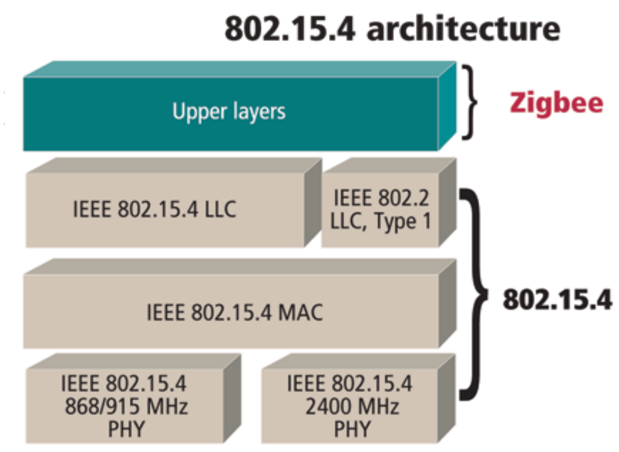


Figure 2-5 ZigBee Protocol Structure, Source: (Dick, 2016)

The physical layer of the IEEE 802.15.4 boasts a of variety of smart, robust driven features such as the receiving energy detection module, clear channel assessment as well as the link quality indicator modules (Kaushal, et al., 2014). The protocol can support up to 65 000 nodes in a network because it uses the 64 bit IEEE addressing as well as a 16 bit short addressing mechanism (Kumar & Gupta, 2013).

The Mac sub layer from IEEE 802.15.4 has a task to control access to the radio using the Carrier Sense Multiple Access with Collision Avoidance. This layer also has authority over the transmission of data packets as it is responsible for acknowledgement and retransmission of data packets and the validation of data frames (Pan, et al., 2011). To achieve the task of transmission tasks, it outlines four frame structures (Bhar, 2015):

1. A beacon frame for transmission of frames.
2. A Mac command frame
3. The acknowledgement frame
4. The data frame

Stacked above the standard layers of the IEEE 802.15.4, in a ZigBee protocol are the layers that define ZigBee, namely the APS layer and the ZigBee Device Objects layer. The APS layer is tasked with managing application end-point addressing in a ZigBee node (Gong, et al., 2014). Given a scenario where more than one application resides on a node, the end-point address is a mechanism for uniquely targeting a particular application on a node in a network. Messages originate from an end-point address of an application on a node, for example, one that could be sensing the amount of light in a particular area, and they end up on the end point address of the destination application.

The network layer, as defined by the ZigBee protocol, is responsible for addressing and routing of data packets in the network and works cooperatively and closely with the Mac layer. It is responsible for starting up of the network, assigning address to nodes in the network, storing or routing sequences in a log table, as well as ensuring security is maintained in the network. The IDS in this research specifically targets this layer, as this is the layer that holds all information related to the actual transmission of data packets in the network (Somani & Patel, 2012).

Consequently, this means that in a ZigBee protocol driven network, applications residing on nodes can be uniquely addressed from 1 to 240, with 255 reserved for broadcasting. Therefore, when sending data from one application to another application residing on a different node, the addressing has to contain both the address of the network that uniquely identifies a node and the end point address of the needed application on the receiving node.

The ZigBee Device Object layer on the other hand is responsible for a number of tasks including the designation of a node type; as to whether it will be a routing node, an end node or a coordinating node. Furthermore, it initialises and participates in creation of a network by a node (Malhotra & Manpreet, 2015).

Table 2.5 shows the performance of ZigBee when compared to other communication protocols, to serve as a justification of its selection in realising the Intrusion Detection System for this project.

Table 2-5 ZigBee in comparison to other Network Structures

	ZigBee	***Bluetooth***	***Wi-Fi***	***Wireless USB***
Range	*10-100 metres*	*10 metres*	*50-100 metres*	*10 metres*
Topology	*Peer-to-peer, mesh, ad-hoc or star*	*Ad hoc*	*Point-to-hub*	*Point-to-point*
Power Consumption	*Low*	*Medium*	*High*	*Low*
Data Rate	*240 Kbits/sec*	*1 Mbits/sec*	*54 Mbits/sec*	*63 Kbits/sec*

From the table above, it is clear that ZigBee betters other common protocol used for wireless communication and therefore a wise choice for the purpose of this project. This is essential because the project deals with low powered devices that need to transfer data in a large area.

2.8.1 ZigBee Flood Attack

ZigBee has been largely developed to be as secure from common intrusion as possible. However it can be forced into a denial of service state or flood through hacking. This hack is initiated by capturing of a ZigBee Node, and re programming it such that it keeps sending hello messages to the central coordinator and re associating itself anew each time with different address. This is done in an attempt to fill up quickly the database containing the address space of all the nodes that make up the network. The aim is to slow down service to other node such as sending and receiving channels and theoretically only one node will be getting served, the node that is under the control of a hacker (Markert, et al., 2011).

2.9 Related Projects

A lot of research has already been done on the need to have Intrusions Detections Systems as a mechanism of ensuring secure networks and the need to build them such that they can be efficient enough to be used in WSNs. This is further amplified by the fact that there are several angles that are being pursued in the development of suitable IDS for each particular network need. For instance, a genetic algorithm approach has been used to develop an IDS and pursuant to the objective, received desired results (Hoque, 2012).

Location aware Trust based Detection and Isolation of Compromised node in wireless sensor networks (Garth, et al., 2011), is another approach to intrusion detection which used a system where reputation-based monitoring is used to carry out detection and isolation of malicious nodes from a network.

It has also been stated that mobile agent based IDS have an advantage over their more static counterparts, such as host based IDS, when it comes to anomaly detection and use of resources (Hakan, 2012; Bhaumik, 2010).

3. Methodology

3.1 Introduction

In this chapter, the design of the intrusion detection system and the methodology chosen by this study is outlined. In section 3.2 the selected quasi-experimental research approach is argued for, while Section 3.5 documents the resources used in this research. Section 3.6 looks into the instrumentation and procedure.

The foremost goal of the study is to create a complete working software program for the detection of threats in ZigBee WSN. This study therefore, apart from coming up with a working solution, is designed around the assumption that, using fuzzy logic for threat detection would result in low false alarms and fast reaction time for detection of intrusions. The result of proving or disproving the research assumption has a direct impact on the construction of proficient intrusion detection software.

There are some standards for measuring the efficiency of an IDS based on how many true detections it manages. True positives (TP) are successful detections of real threats directed at the network. A false positive (FP) occurs when a normal and harmless operation is detected by the IDS as harmful. Conversely, a true negative (TN) exists in a situation where a normal flow of events is seen and regarded as normal by the IDS and a false negative (FN) occurs when a harmful threat is declared as normal by the IDS (Neha, et al., 2016). Figure 3.1 depicts the mentioned scenarios as a matrix.

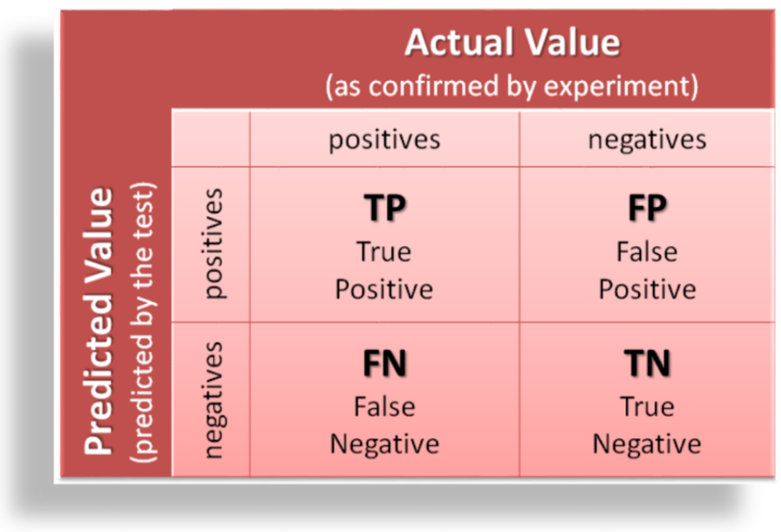


Figure 3-1 Confusion Matrix, source: Mathworks.com

Based on the above information above, we can calculate the rate of each of those parameters in intrusion detection (Neha, et al., 2016).

i. The False Positive rate (FPR)= FP/(FP+TN)

ii. False Negative Rate (FNR)= FN/(FN+TPP)

iii. True Positive Rate (TPR)= TP/(TP+FN)

iv. True Negative Rate (TNR)=TN/(TN+FP)

v. Accuracy =TP+TN/(TP+TN+FP+FN)

vi. Precision=TP/(TP+FP)

Our assumption is to be validated, held true and strengthened if there is an above 90% accuracy rate and a below 5% false alarm rate. These projections are the study's internal set threshold to measure success or failure of the experiment and are not based on any pre-existing standard found in Literature.

Additionally, this research runs on the assumption that not having intrusion detection code directly installed on the sensor nodes would result in the nodes running longer before a need for battery recharge. This is because the nodes are used for monitoring purposes only and no extra calculations needed by the IDS are carried out in the nodes themselves.

Consequently, the IDS is network based because, it is installed only on the computer or server of which the sensors are connected to. Furthermore, giving the IDS some sense of smart thinking in the form of fuzzy logic, or another artificial intelligence notation can give the IDS more accuracy but may be slower due to the increase in calculations performed.

3.2 Research Strategy: Quantitative and Quasi-Experimental

The research is quantitative in nature, as it is based around measuring the precision of the solution domain. The problems are emulated in a controlled environment. Therefore, designing of the research through the experimental methodology route is ideal. However, there is lack of one component of a typical experimental research, being presence of a control group. Therefore, this research was quasi- experimental in nature (Harris, et al., 2006).

3.2.1 A Quantitative study

Quantitative studies are concerned with proving or disproving a set of hypotheses based on numerical data accumulated. The studies involve observing phenomenon or events with the intention of applying statistical methods to aggregate and make logical deductions from the data collected (Suphat, 1996).

Quantitative studies rely heavily on the collection of empirical data. This is often done through manipulations of a set of variables of interest to observe causal relationships between the subject of interest and the effect those variables have upon it (Jerome, 1997). Conversely, these studies can be carried out through observing a phenomenon as it unfolds and recording data (Jessica, et al., 2009).

Typically, when a new method of measurement is used or being developed, a quantitative study is the preferred choice as well as in situations where a new model is being generated (Carrie, 2007). This means quantitative studies are made to answer, empirically, questions posed by a research hypotheses on the occurrence of a phenomena and why it occurs the way it does.

Causal relationships exist in the event where manipulation of one variable directly impacts the phenomena under investigation (Kosuke, et al., 2013). However, it should be noted that, in quantitative studies, it is always emphasised that the existence of correlation between the variables being manipulated and the observed phenomenon does not imply that causality exists between the two (BMJ Publishing Group , 2016). Correlation means that two observed variables have a linear relationship with each other over time.

Additionally, when conducting quantitative research, if it happens that a certain variable of interest cannot be directly measured, proxies are used instead, and these then are classified as quasi-Quantitative research (The Regent of the University of Michigan, 2016).

Quantitative studies are perceived to be in contrast with qualitative studies, which emphasise the collection of data based on how humans perceive the environment around them. These studies are typically devoid of using any statistical methods to make deductions on the data collected. There is minimal empirical angle to a qualitative study. Qualitative studies generally aim to develop theories about an occurrence and these theories are then often tested using quantitative methodologies. In this sense, qualitative and quantitative studies can also complement each other and co-exist as part of one research study (Jennifer, 2013).

Accordingly, this study was quantitative, as it deals with analysis of a situation based on empirical numerical data. This study research does not make application of any of the qualitative methodologies of research.

3.2.2 An Experimental study

Researchers who need to substantiate claims they or other researches have made undertake experimental studies (Paul, 2003). Additionally, researchers who need to evaluate models or to give strength or falsify hypotheses also prefer them (Colorado State University, 2016).

Experimental research is facilitated through a variety of methodologies, including but not limited to, controlled experiments, field studies, computer simulations as well as surveys.

3.2.3 Controlled experiments

Controlled experiments as a subset of quantitative studies as well as experimental studies, deal with testing the validity of a hypothesis (Steven, 2002). In a comparative controlled experiment, a hypothesis is made around a baseline and this baseline is the accepted standard of a singularity in existence. The null hypothesis takes the stance that there is no difference between such a singularity and a proposed alternative or vice versa. The experiment is considered a success if the null hypothesis is validated beyond uncertainty or rejected completely (Steven, 2002).

A controlled experiment usually strives collectively for a few common goals. They usually strive to be reproducible, to give the researcher control over variables, manipulate those variables and record the effects of those variables on a situation. Additionally, controlled experiments aim to show a high level of internal and external validity (Victor, 2005).

Internal validity is the extent to which causal conclusions are free from experimental bias, selection bias or confounding and are at a point where no other variables other than the independent variable being tested have an effect on the dependent variable. This is a point in a study where a conclusion can be made with the utmost conviction that the result as observed in the dependent variable can only be caused by the manipulation of the independent variable (Taylor, 2007).

On the other hand, external validity refers to the extent to which generalisations made in an experimental set up can hold true for a wider population. A study is considered a success if it has both high internal and external validity (Taylor, 2007).

This study was conducted using controlled experiment approaches.

3.2.4 Controlled Experimental design

An experimental design can be categorised according to elements of-

i. Dependent versus independent variables of the experiment
ii. Single session versus longitudinal experiments
iii. Within subjects versus between subjects experiments

3.2.4.1 Dependent versus independent variables

The variables in a controlled experiment can be categorised into dependent and independent variables. The independent variables have an impact on dependent variables when manipulated. As a result, dependent variables give off measurable data (Prabhat & Meenu, 2015). For example, in an imaginary experiment where a researcher would wish to know the impact of types of shoes on the performance of an athlete, the shoes are the independent variables and the performance of the athlete is regarded as the dependent variable. Influence of the independent variable is expected to produce a change in the independent variable.

In our research, the inclusion of Fuzzy Logic Systems for detection is regarded as the independent variation while the rate of false positives is the dependent variable.

3.2.4.2 Single session versus longitudinal Experiments Design

Longitudinal studies are most common in qualitative research. The concept of a longitudinal study revolves studying a subject over a long period, from an observational view, without much manipulation of factors or variables. This stands in contracts to elements of an experimental research where tweaking of variables is inevitable. However, the element of observing a phenomenon over a long period of time to record how the independent variables behave can still be applicable in an experimental research (Janet, et al., 2006).

On the other hand, single session experiments are performed once over a short period of time and as the name suggests, typically within a single session. For the purposes of this study, the single session approach was chosen.

3.2.4.3 within subjects versus between subjects Experiment Design

In the within subject design, each participant used in the study is exposed to only one condition, either the condition that favours the null hypothesis or the condition that supports the counter argument to the null hypothesis. The advantages of this approach include the fact that is it easier to implement comparatively, and there is less risk of skill transfer among the participants from one condition to another. However, Inconsistency in this design tends to be high.

Conversely, in a between-subjects-design, all participants are exposed to all conditions of an experiment. It is the most commonly adapted design between the two. Inconsistency is more controlled in this experimental design, which is caused by the apparent skill transfer when participants are exposed to both approaches (Gideon & Jeroen, 1988).

Our experiment for this study false under the between subjects experiment design as all nodes are subjected to similar encompassing conditions of the experiment.

3.3 Appropriateness of the Research Design

As explained in this study, the aim was to build an IDS and to conduct an experiment to determine the best way to build the threat detection module for the IDS. That is, to find out if using fuzzy logic for threat detection is an effective idea. Therefore, the research is quasi-experimental as already explained and single session. There is also an element of randomness, which is also often needed to quantify a study as truly experimental. This comes in the form of simulated attacks to the software, which are randomised in terms of time and frequency.

3.4 Research Design

An experimental study revolves around manipulation and observation of variables as previously mentioned. The variables in this study are the number of false positive, as the dependent variable, and simulated threats as the independent variable.

3.5 Experiment Setting

The experiment is carried out in doors on a simple desktop computer, running Windows 10, 8 gigabytes of RAM and a core i5 processor. There is NetBeans with Java 8 installed for developing the IDE. For fuzzy logic integration into java, the study made use of Jfuzzy lite software. All the sensor nodes are from Libelium and make use of the Xbee data transmitters from Digi, which transmit data through a modified ZigBee protocol (Libelium, 2016).

3.6 Instrumentation and Procedure

The IDS, takes the form of a desktop software application, based on the need to create a network based IDS similar to the ones used in wired networks. Typical studies, in WSN security, usually bind the sensor nodes with the IDS functionality (Anuradha, et al., 2015). The approach used in this research, nonetheless, is selected for the sole intention of trying to find alternative ways to reduce the workload on the sensors.

WSNs have limited processing and power capabilities, and they transmit all their data readings to a sink node which is usually connected to a more complete computer system, with highly accessible and easily renewable source of power. It appears, therefore, only logical to make use of this end node computer to monitor WSNs. Sensor nodes are to be kept purely for sensing environmental data; for any application domain they are intended for. This is done in the hope that sensors retain their battery power longer in the field before needing a battery change.

The IDS software in its entirety consists of three key parts.

1. The interactive graphical user interface.
2. The security algorithm for monitoring and detection of threats
3. An internal database for keeping a log of sensors deployed in the field.

This IDS is designed to work specifically with sensors that make use of the ZigBee protocol for communication and data transfer.

3.7 Software Development

The development of the software takes on the Rapid Application Development (RAD) approach through prototyping (Naz & M, 2015). This is because this is rather a small software that does not need a lot of planning before the implementation is done, making it an ideal platform for rapid application development

4. IDS Design

4.1 Introduction

In this chapter, the design blueprint of the system is outlined in its entirety as well as the algorithm foundation which inspired the algorithm being used in this project. The selection of the design language is made in this chapter. There is also an explanation of how all the parts work together to form one software piece than can also interact with third party applications.

In the following chapters, the design segment will be realised as code and implemented as arranged out in this chapter

4.1 Software Modelling Languages

There are a variety of Modelling notations that can be used to model the behaviour of software. For example there is the Unified Modelling Language (UML), Domain Specific Modelling (DSM) as well as Object Process Methodology (OPM) to name but a few (Garcia, et al., 2014).

Each of the aforementioned modelling languages has their merits as well as weaknesses when used in software design. For example, DSM has the capability to produce executable source code directly from the design models which reduces the coding work to be done by a programmer drastically, and it also produces models of a higher abstraction comparative (Luoma, et al., 2004).There is however, a learning curve that is rather steep and not necessary for such a small scale software project. A more standard approach is needed.

On the other hand, OPM can represent a model of a system as a single graphic composed of visual graphs and textual descriptions (Berger & Dov, 2003).

However, the problem this study encountered with OPM is that the resulting models often have little association with the resulting code and the notation is not common.

Consequently, for the purposes of displaying the IDS components, and how they interact as a single system, UML diagrams are used in this project. UML is a standard of modelling software projects in software engineering principles that is more common as it has been around for quite a long time (Rumbaugh, et al., 2004). The major problem is that most people find UML too complex and a bit cumbersome as it has many different diagrams for every aspect of the software (Reggio, et al., 2013). This problem is overcome by using just a few of the common UML notations in this project.

UML diagrams can be used to depict the interaction between a user and a piece of software, the objects that together build the software as well as the internal behaviour of these objects, within the software piece.

4.2 Modelling Behaviour of the IDS

4.2.1 Use Case Diagram for the IDS

The graphical user interface (GUI) provides a window into the software through which a user perceives the state of the network at any given point in time. It allows a user to check on all sensors that are connected to and sending data in the network, sensors that are healthy as well as sensors that could be deemed unhealthy or dangerous to the network.

The interaction between this GUI and an end user can be represented and modelled using a UML Use Case diagram. A Use Case diagram is a UML diagram that is used to model the behaviour of a system .This Use case diagram is used in a way which models the relationship between an actor and a system, and this actor can be a human or another computer system that interacts directly with a given software system (Rumbaugh, et al., 2004). These Use Cases do not provide a detailed description of the interaction or include elements such as constraints and business rules but rather summarize some of the interaction between a user and a system (Anon., 2015).

The following diagram shows the major interaction instances between a user and the Intrusion Detection System Software.

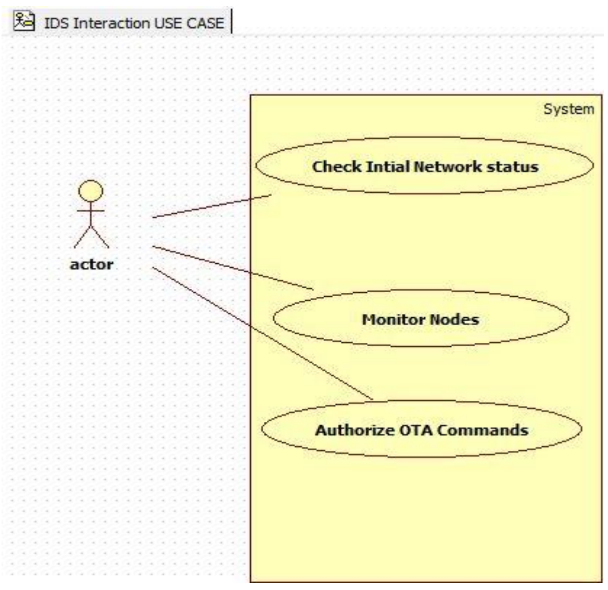


Figure 4-1 IDS Use Case Diagram Depiction

As can be seen in figure 4.1 of the Use Case Diagram, the user can interact with the IDS to view the initial state of the sensor network system. This particular instance of interaction displays to the user, the number of sensors that are sending data toward the computer for any purpose of which they have been deployed to a user. Within this usage parameter, the IDS also shows the identity descriptions of the nodes such as sensor ID and Mac address uniquely for each and every node in the network. These traits are not particularly useful to an end user but are used by the security algorithm within the IDS when detecting for anomalies and when correcting such anomalies from the sensor network.

The second use case scenario shows how a user of the IDS can use the software to monitor ZigBee sensors that are in the field. Essentially, it is the internal algorithm that does the work, but a system user gets to perceive the processes as they occur, such as capturing of identity traits for sensors, storing of active sensors in database, monitoring on assumed rogue nodes as well as actions recommended by the system within.

Finally, the last Use Case scenario depicts that at times the user can interact with the system to give Over the Air (OTA) command authorizations. For example, if one or more nodes have been found to be faulty or proving to be hazardous to the network, meaning that they have been hacked and are acting maliciously, the security algorithm can act autonomously to issue an OTA command to either reboot the suspected node or to shut it down completely based on the level of discomfort it is causing the network. A user of the IDS is given power to authorize the sending of this command based on the suggestion of the algorithm. This interaction is the one represented by the last use case scenario.

4.3 Activity Diagram for the IDS using Fuzzy Logic

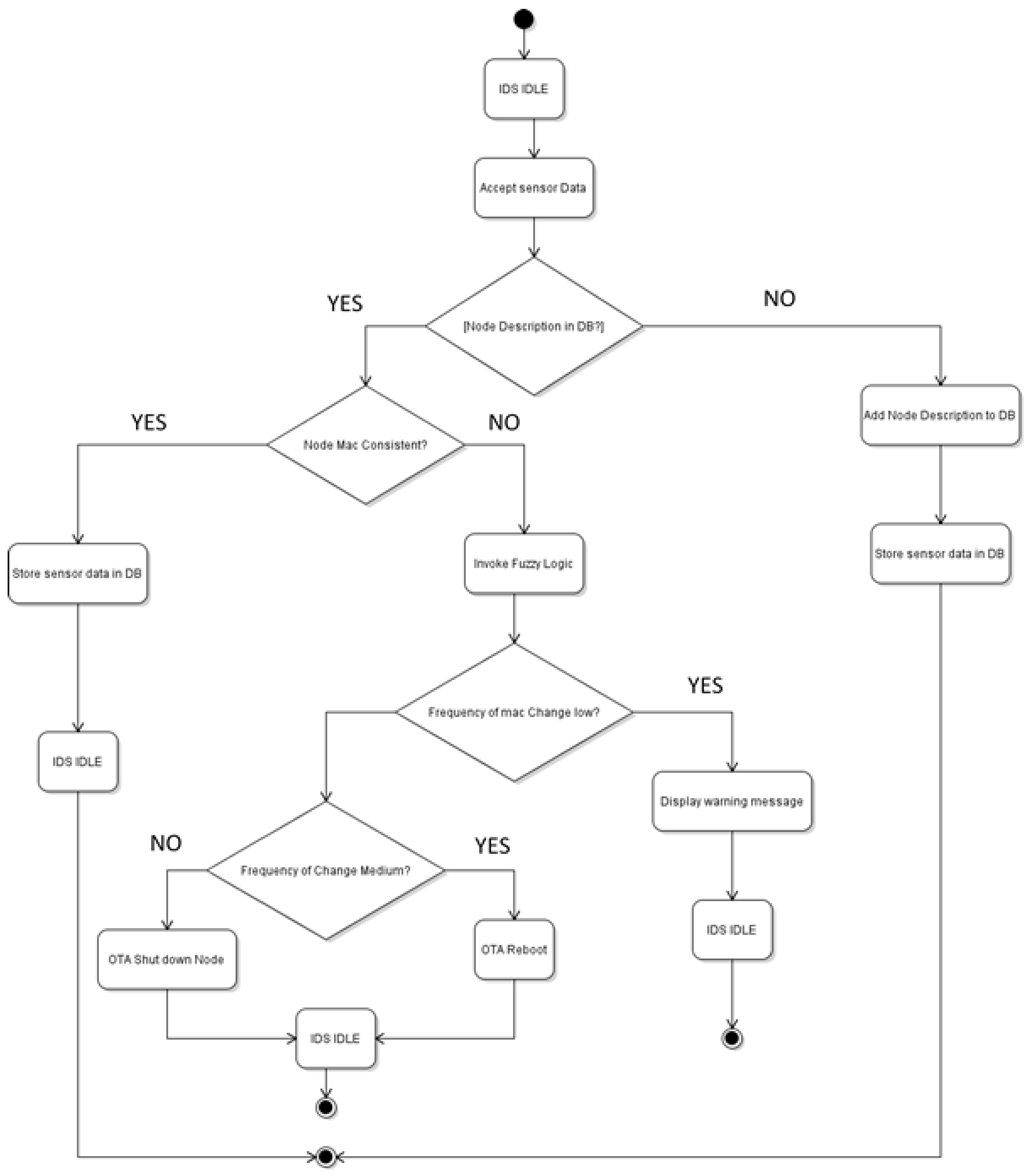


Figure 4-2 The Activity Diagram for The IDS

The activity diagram shows the general flow of activities that happens at each reading of sensor nodes by the IDS. UML Activity diagrams are behaviour modelling diagrams mainly used to show the work flow of a program (Kaewchinporn & Limpiyakorn, 2013). In the IDS, the first process that happens when a node first sends data to the gateway, is the capturing of the node description and storing such information in the database as part of nodes that make up the network.

The description of the node is the node_ID as well as the Mac_adress of the node. If the description of the node is already a part of the network, the node mac_address is checked for consistency. This measure is done by the security algorithm. Every time a node sends data it is checked if it is transmitting using the same associated and recorded Mac_adress as before. If at any given time these parameters change, the change is recorded under frequency of change. In the case where the frequency of change gets above one, the fuzzy logic module of the algorithm is invoked.

However, if the node is found to be operating under normal conditions, the data that is being transmitted by the node is accepted and stored in the database for third party applications to access. This data is expected to vary based on what the sensors are monitoring.

After each transmission or checking of errors in the transmission signal has been done, the system goes back to an idle mode until the IDS is turned off by the application user.

4.4 Interaction of Objects that make up the IDS

Interaction diagrams are used to depict the way the messages flow in a system or software from one object to another. These diagrams also emphasise the time order in which messages are dispatched between the objects (Alhumaidan, 2012).

There are generally two diagrams under UML for modelling message flow being sequence as well as collaboration diagrams.

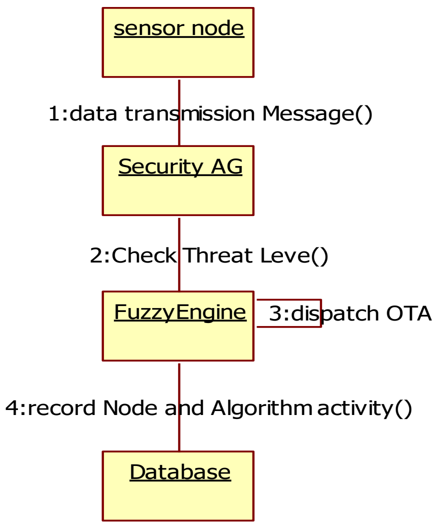


Figure 4-3 Collaboration Diagram for the IDS Objects

The collaboration diagram above depicts the way the objects are organised in their interaction to form one system. Although sensor nodes are not part of the IDS, or algorithm within, they sent the initial message to the network which triggers the IDS and subsequently the security mechanism into action. All of the actions are recorded in the database.

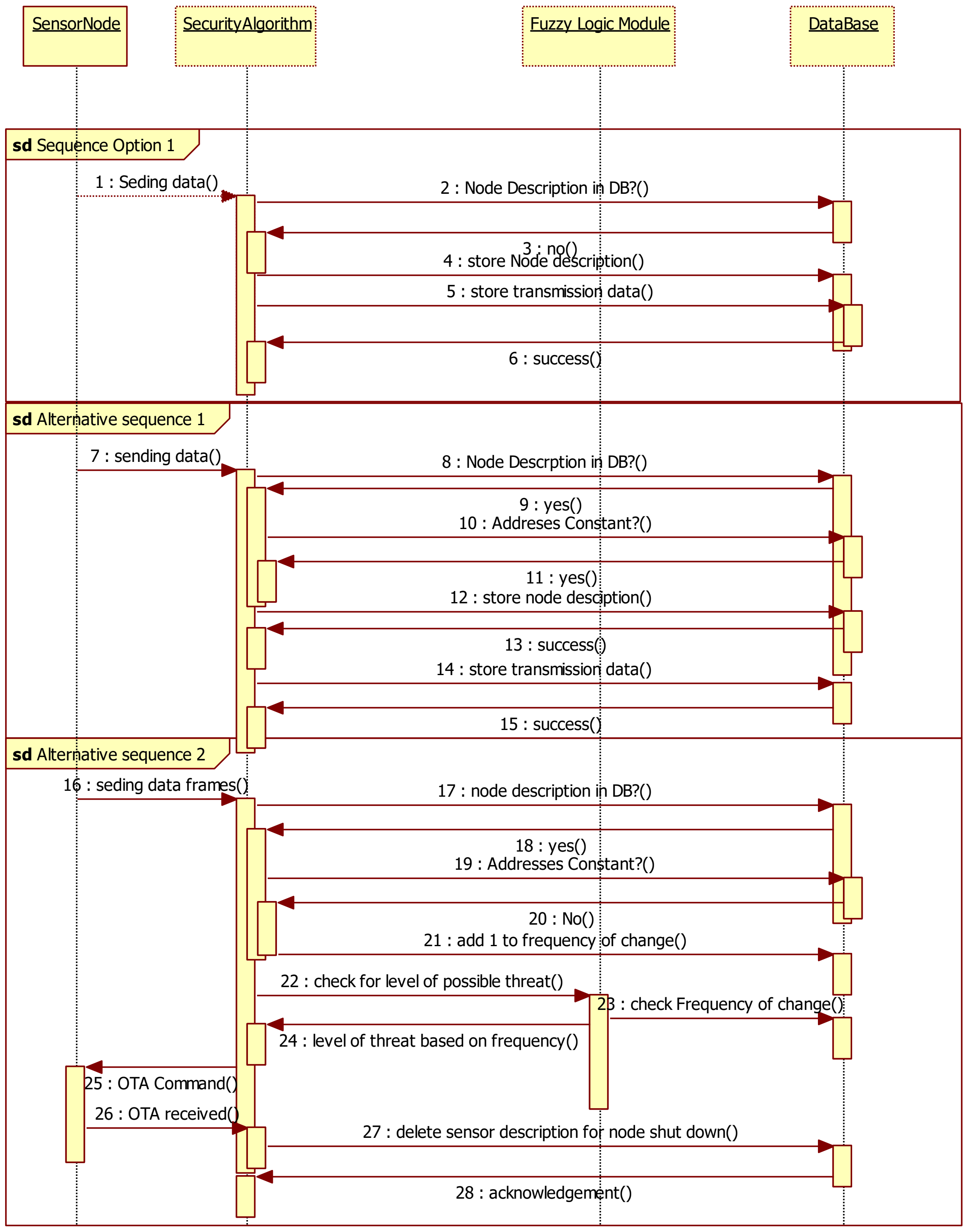

SensorNode
SecurityAlgorithm
Fuzzy Logic Module
DataBase
sd Sequence Option 1
1 : Seding data()
2 : Node Description in DB?()
3 : no()
4 : store Node description()
5 : store transmission data()
6 : success()
sd Alternative sequence 1
7 : sending data()
8 : Node Descrption in DB?()
9 : yes()
10 : Addreses Constant?()
11 : yes()
12 : store node desciption()
13 : success()
14 : store transmission data()
15 : success()
sd Alternative sequence 2
16 : seding data frames()
17 : node description in DB?()
18 : yes()
19 : Addreses Constant?()
20 : No()
21 : add 1 to frequency of change()
22 : check for level of possible threat()
23 : check Frequency of change()
24 : level of threat based on frequency()
25 : OTA Command()
26 : OTA received()
27 : delete sensor description for node shut down()
28 : acknowledgement()

Figure 4-4 Sequence Diagram of the IDS

Figure 4.4, is the sequence diagram of the IDS. It shows the order in which messages flow between objects to form a complete system. The system has three possible scenarios of operation. In Sequence option 1, the diagram depicts a situation in which a node is absolutely new to the network and no known information of the node is in the database. This also constitutes the learning stage of the algorithm in which every node and their initial properties are stored in the database. Since a new node is generally not suspected of threat, its transmission data(information it has gathered and sent toward the network) is accepted and stored in the database to be accessible to third party application that need the data.

Alternative Sequence 1, in the diagram, shows a different scenario in which a node sending data to the network already has its properties recorded in the database. The moment a node sends data and is found to be an existing node in the network, its descriptive data is checked with previously recorded information to check if its descriptive data has remained constant. This is also the work of the security algorithm. This particular sequence of messages shows a scenario where the algorithm, having been satisfied with the state of the node, updates the database on the state of the network, as well as continuing to record the transmission data being ferried from the node.

Finally, in Alternative sequence 2 the scenario depicted is of a node that has had its descriptive data changed since its last known sending session. This change is recorded in the database and messages sent to the fuzzy engine module to investigate the potential threat posed by the node. Based on the evaluation received from the fuzzy logic engine, an OTA command is subsequently sent to the offending node and a message dispatched to the database to delete all information on the node, if and only if, the OTA command was to shut down the node.

4.5 Software Development Process.

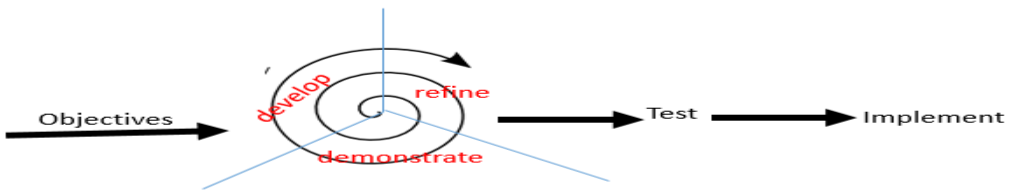


Figure 4-5 RAD Prototyping, Source: Wikipedia by Unknown

Figure 4-5 depicts the processes followed when developing the IDS. The moment the prototype iterations reach a certain satisfaction, as set by the project objectives it would be ready for testing. If the results from testing are good enough, the software could be deemed ready for end user consumption as well as in monitoring a full scale WSN.

4.6 Deployment Diagram for the IDS

A deployment diagram is used when giving a visual scenario of the hardware platform on which the software will run on (Wrycza, et al., 2014). It shows a very highly abstract depiction of the hardware architecture arrangement. In the deployment diagram below, the hardware of interest are the senor nodes as well as the computer or server on which the application is installed on. The diagram shows one node as an example but it was expected that the number of nodes connected would be any number big enough to carry out a specific monitoring purpose. The user interacts with the computer which is running the application and the application communicates directly with the nodes through a com port on the computer.

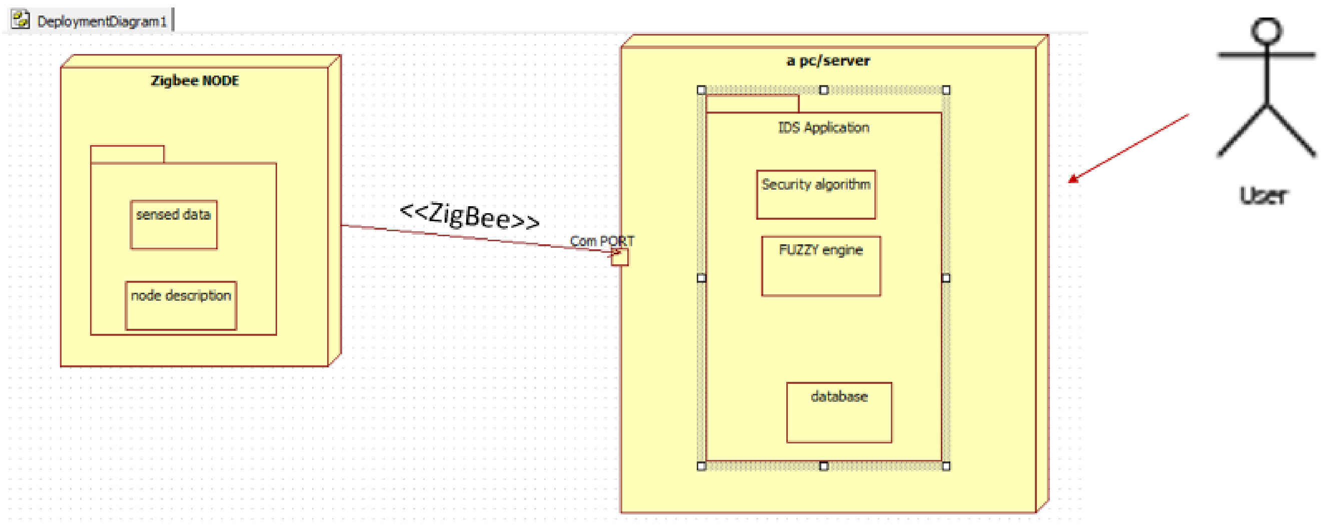


Figure 4-6 IDS Deployment Diagram

4.7 The Security Algorithm Design

The system is built around a security algorithm concept, figure 4.7, first proposed in a paper that focuses on thwarting attacks aimed at ZigBee sensors (Stelte & Rodosek, 2013). The algorithm presented in that paper uses a different approach of monitoring the hash tables of the protocol to check on rogue sensor nodes that have changed their mac addresses. The algorithm has two stages, the learning stage as well as the detection stage.

```
evaluation_time = get_time() + evaluation_step
loop
  receive_byte()
  if $message\_complete()$ then
    evaluate_address()
    update_hashtables() {update short- and long-address
    hashtables}
    if $get\_time() > evaluation\_time$ then
      if $get\_environment() + threshold < size(hashtable)$ then
        drop_communication() {intrusion detected: drop
        message}
      else
        update_environment()
        update_thresholds()
      end if
      evaluation_time = get_time() + evaluation_step
    end if
  end if
end loop
```

Figure 4-7 Security Algorithm for Detecting Flood Attacks in ZigBee WSNs

In this research we have used the template design paradigm of algorithm design, based on the above mentioned algorithm as a template. In this paradigm the overall flow of the algorithm is maintained but certain subclasses or modules or section are modified for a specific purpose such as our approach (Omics International, 2014).

For starters, in this research a database is built into the IDS software so as to keep track of node addresses and to help the algorithm when it tracks down rogue nodes. Additionally, the evaluation is not done at the protocol hash tables but from the database. The aim is still to evaluate a threat before the hash tables crash after being flooded by rogue nodes. These tables keep track of all sensor nodes in the network as well as their associated addresses. A flood, in essence, happens when these tables are being forced to record more addresses than they can hold (Stelte & Rodosek, 2013). This happens in the window period afforded by the protocol during discovery, where a node answers a call from the gateway to send in data. A node replies by giving its mac address and node ID during the association stage.

A rogue node can, at this point, keep answering to the association call, with as many different addresses as possible within a short space of time, leading to a rapid filling up of hash tables and eventual crash of the network.

Lastly, this research takes into consideration that a node can be considered bad based on limited evaluation knowledge and therefore the algorithm is augmented with a fuzzy logic module to lower the possibilities of falsely claiming a node to be under attack.

4.8 Development Milestones

The development of the IDS would have to meet a few milestones before it can be tested, which is in essence another milestone and subsequent implementation.

1. Java coded user Interface showing all buttons working
2. Coded algorithm in background communicates with interface in foreground
3. Connection of the internal database with the User interface
4. Connection of the application with a communication port using java
5. Capturing of the information coming through Com port by the application
6. Separation of information contained in a packet
7. Storing of selected data into different database tables
8. Testing for different scenarios
9. Completion of project

5. Implementation

5.1 Introduction

This chapter looks at the implementation of the WSN security algorithm as part of a desktop Intrusion Detection application as well as the architectural framework of the solution. In section 5.2 the architecture of the IDS is outlined. The IDS algorithm and its enhancements are presented in this chapter as well their analysis. This chapter also documents the process making all the different components of the IDS to function as one unit.

The desktop application software was written purely in JAVA and is adapted from a Java framework meant for com port programming. It monitors sensor nodes deployed as though in field of which, in this case, are still in a controlled indoor environment. These sensors are in turn sending their data readings to a computer through the Libelium ZigBee gateway.

5.2 Architecture of the Intrusion Detection System Application

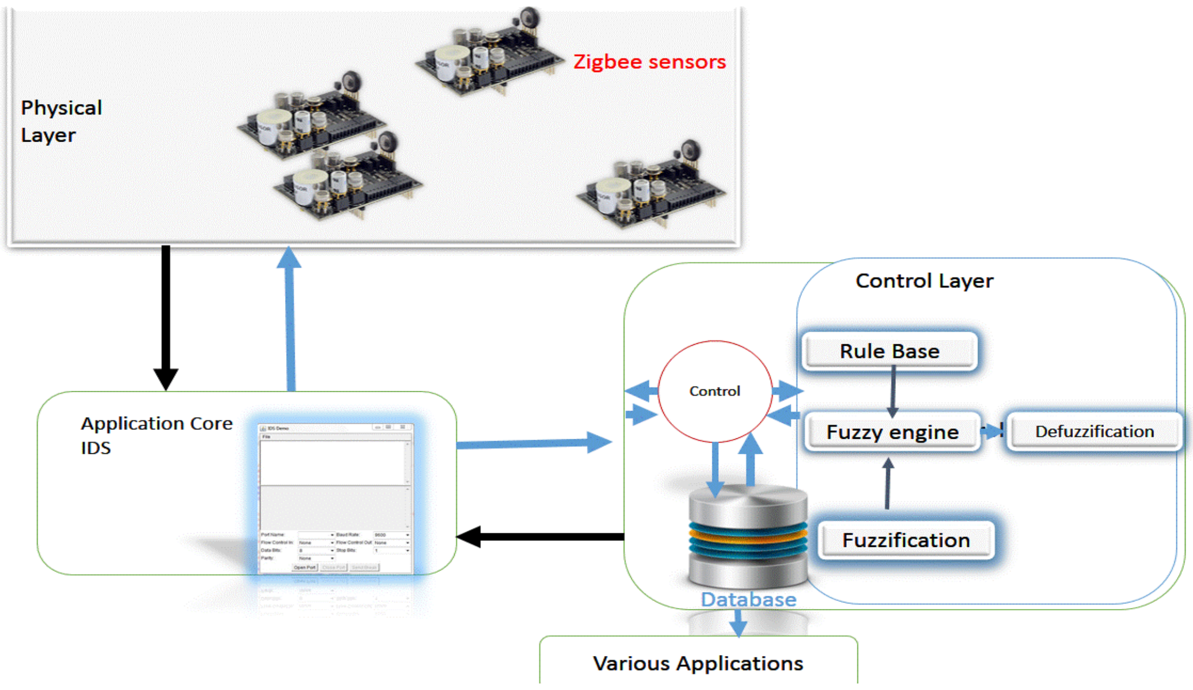


Figure 5-1 IDS Architecture

5.2.1 Physical Layer (Libelium Waspmote ZigBee Sensors)

The physical layer is the point through which data is collected. Sensors transmitting through the ZigBee 802.15.4 communication protocol make up this layer, for the purpose of this research. The ZigBee gateway that collects data from the sensors to the computer is also found at this layer. These sensors have been programmed such that with every data packet frame they sent, for whatever sensed data of interest to an end-user, they also broadcast their mac address as well as their node Id. The sensors are also programed to listen to OTA commands, as well as to broadcast at only a certain baud rate and Pan ID. This is done because they would not be able to find the gateway if it was on a different Pan ID. Additionally, the IDS software would not be able to decode the data packets if it read them at a different baud rates other than the one at which they are sending data. This would cause problems for the control layer as it would not be able to extract desired information from the packet.

5.2.2 Application Layer (IDS)

This particular layer can also be referred to as the presentation layer. In this level a GUI desktop application written in Java is found. This is the Intrusion Detection system software interface. This application also allows the user to make various operative selections such as baud rate, port selection as well as to open or close ports in order to control data flow. As long as a gateway is connected and sensors are deployed, the desktop GUI application enables the user power to monitor and view the data that is being sent by the sensors through the gateway. The GUI also issues warning to the user about any activity going wrong with the sensors, as well as all the actions being taken by the internal system.

5.2.3 Control Layer

At the control layer, data from sensors is monitored and vital information about the sensors stored in the database. This layer houses the security algorithm, which is the control behind the IDS software. This layer is made up of the algorithm as well as the database. Through this layer, other applications that may need to use the sensor data in a different way from the application layer can connect to the database and gain the information they require. Other third party applications are encouraged to use this database because it is often not straightforward and easy for two applications to access the same port simultaneously on one computer.

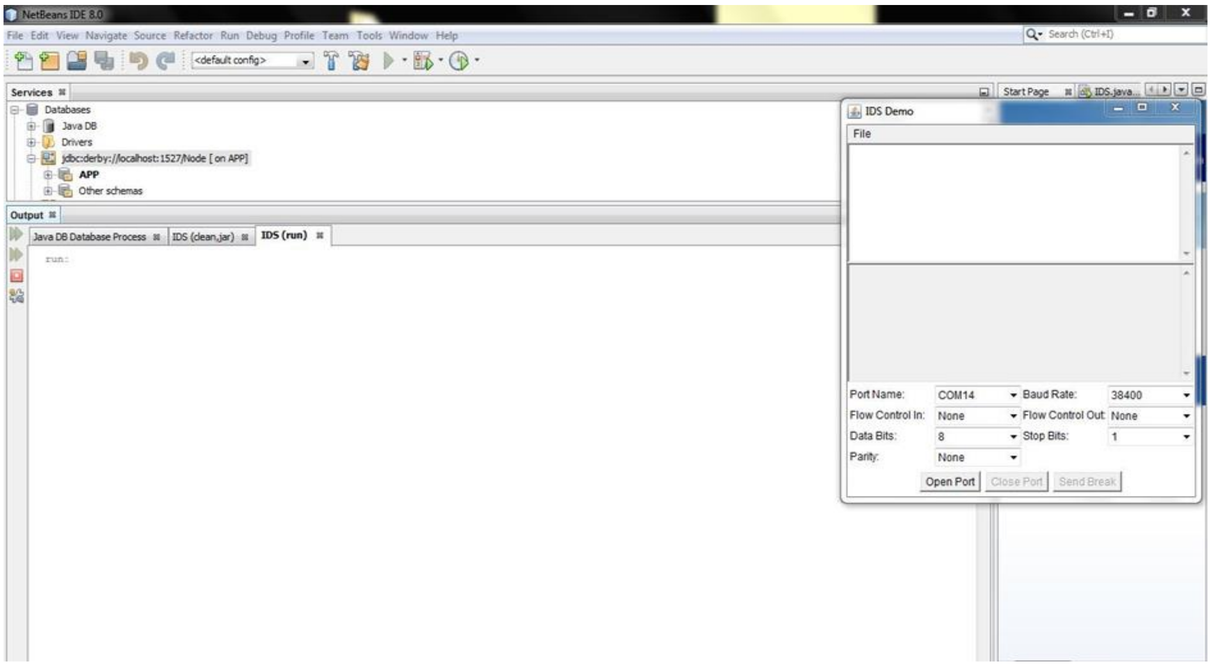
The gateway connects to a port. In the case where a user needs to use another application that reads data from this gateway port while also monitoring the sensors for intrusions, they need to connect the second application to read data from the data storage of the IDS database such that both applications can benefit, even though there is one port of data entry.

In the control layer, after data has been monitored, and in the case of anomalies being present in the data, the fuzzy logic module detects the anomaly and gives the appropriate response of which the application can communicate to the sensors through OTA.

5.3 The Application

The application connects to the same port of which the gateway is connected, and receives the frames as they come in and extracts the information needed in order to find anomalies. The anomaly of interest to this study was the ZigBee flood attack. Signs of its possible occurrences are monitored, from the frames being received. The application is in a perpetual state of checking for this anomaly, for as long as it is in execution.

This interface also allows the network administrator a graphical way of interacting with the network and when authorising some actions to be taken by the IDS, which is in essence, the security algorithm core. The GUI is the representation of this algorithm and not, a different part altogether. This GUI and the algorithm together form the IDS presented as a solution in this research project.

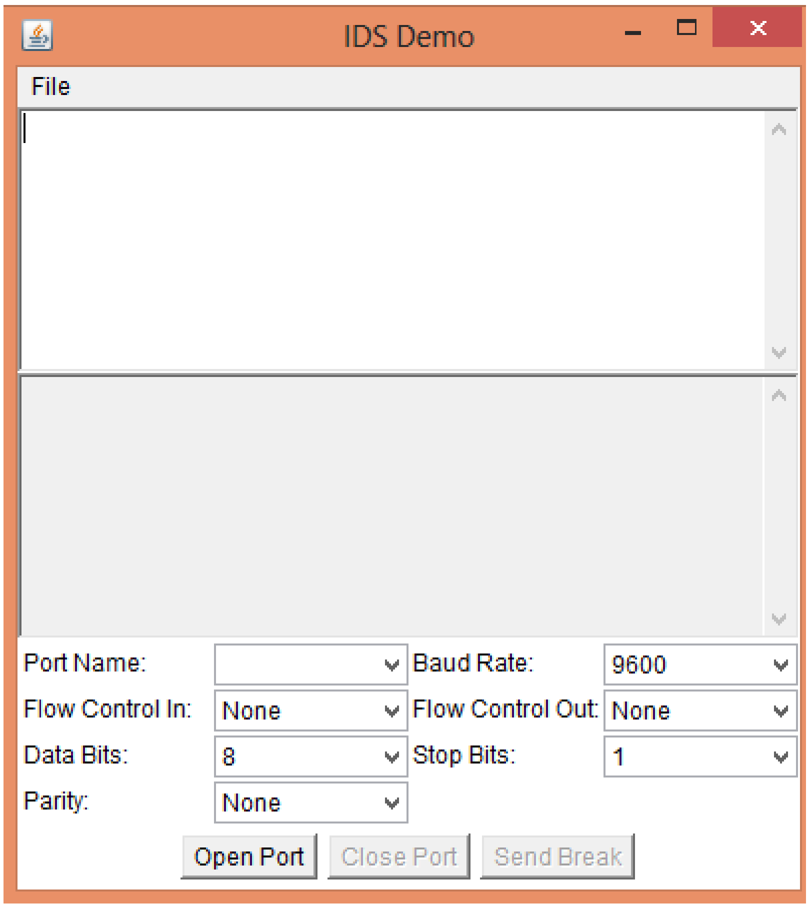


```
run:
```

Port Name: COM14
Baud Rate: 38400
Flow Control In: None
Flow Control Out: None
Data Bits: 8
Stop Bits: 1
Parity: None

Figure 5-2 The IDS Interface Development

The GUI desktop Display, figure 5.3, is for information such as sensor node descriptions, the power level of each node, data carried by the sensors as well as information on sensors that the internal algorithm considers rogue or those that are being cut from the network, during execution. In the background is the security algorithm, which is registering sensor nodes and disqualifying rogue or attacked nodes. Security actions taken by the algorithm are also displayed on screen.



Port Name:		Baud Rate:	9600
Flow Control In:	None	Flow Control Out:	None
Data Bits:	8	Stop Bits:	1
Parity:	None		

Figure 5-3 The IDS interface

The solution is composed up of three parts; a desktop graphical user Interface, the algorithm and an in-build database. The algorithm can in turn be broken into three parts; learning, detection as well as response.

5.4 Learning Segment of the Algorithm

The learning segment of the algorithm provides a simple mechanism through which all the nodes connected to the network are discovered and recorded in the database. At the outset, the ZigBee Wasp Mote gateway sends beaconing signals in a bit to initiate a way for sensor nodes to discover it. For these sensor nodes to be discoverable, they should be on the same network as the gateway. They should also in turn send their association signals through the same Pan Id and baud rate as expected by the gateway.

The leading task of the algorithm is to discover each and every node that successfully communicates with the gateway and is able to send data to the gateway. These sensor nodes that can connect to the gateway have been given association permission by the gateway to send data and are now part of the network. For as long as the network is functional, the algorithm goes into an iterative state of learning the current structure of the network with each data transmission instance.

The learning division of the security algorithm mainly records each and every sensor that is making a fresh connection to the network, in a database. The nodes are stored using their mac addresses as well as the node ID, both of which form the identification trades for the nodes. The algorithm also takes measures to make sure that there are no nodes that are recorded more than once in the database.

5.5 Hierarchy of Work

In chapter 4 the sequence diagram, figure 4.4, depicts the hierarchy of work in the IDS. That is, it shows the sequence in which messages flow in the detection process. In summary, once the application is launched, the coordinator sends a beaconing signal of which nodes respond to with a signal showing that they are interested in joining the network. This is entirely the work of the ZigBee protocol. These nodes are individually programmed such that when association finally happens they send their mac address and node ID address with every packet of message they subsequently sent. These packets therefore contain the information gathered from monitoring a variable of interest as well as the addresses as one message.

This data can be anything the sensors are deployed to monitor, be it temperature, soil moisture or motion detected and are not vital to threat detection. The IDS depends on the addresses being packaged together with the sensor data, for monitoring of threats.

As soon as the coordinator has sent an acknowledgement for all the nodes that answered the beaconing request, the security algorithm starts up to intervene and inspect the signals of each and every node in the transmission cycle, before they can be stored and processed. After the process of inspection is complete, the data part of the transmitted signal is stored in a database, where it can be accessed and used for consumption by a third party application that a user may interact with. At this point, rogue nodes would have been found and cleared by the IDS.

ALGORITHM SEGMENT 1: Node DISCOVERY

Let n be the number of nodes that have answered an association call from the gateway

Let i be the node that is being read at a point in time

```
1    For (int i =1; i<n; i++)
         Receive_frame
2    If  frame_complete
3        Extract node_mac-address
4        Extract node_id
5    Else discard_frame
6    If  mac_address and node_id NOT in Learning_table
7    Update learning_table with new mac_address and node_id
             Monitoring and Detection() //calling the detection module
8    Else
```

```
            discard node_address and node_ID // node is already  registered
            Monitoring and Detection()

9    End if
10   End if
11   End while
12   Drop table learning_database
```

The IDS application is meant to be run each time a user wishes to monitor the sensors as deployed in the field. When the application execution is terminated, the learning segment of the IDS algorithm clears the contents of the database, such that it can learn afresh the environment the next time the application executes. This explains the last line of the algorithm pseudocode, which orders for the contents of the database to be cleared as the usage cycle has timed out. However, if the IDS is run from a server and used for monitoring on a continual time, this last line is not to be called upon, until if ever, the IDS is stopped.

5.6 Monitoring and Detection

The application's main purpose is to detect signs of a flood attack that could be directed at a ZigBee network. Fundamentally, monitoring and detection is at the heart of the entire application. The following is a segment of the algorithm tasked with monitoring all of the nodes that are part of the network as well as detecting the nodes that could provide harm to the integrity of the network, and providing corrective solutions for those particular nodes.

Algorithm Segment 2: Monitoring and Detection

```
1   Receive_frames

2   Extract node_addresses {Mac_address and node_ID}

3   Compare_address with data in Learning_Database

4   If addresses NOT equal

            Calculate frequency_of_change()

            update table frequency in database}

            If   Node_Calculate_frequency ()> 1 then

            Fuzzy_reference () {

                    If attack mild

                            issue warning

                          Update Database

                         Update_environment

                      Else if attack_medium {reset node: drop communication}

                      Else if attack_severe {isolate node: drop communication}

                      } // end of fuzzy module

                      Update_environment

5   else

            Update databases   //receive the sensor data and store in database

            Update_environment
```

```
6   End if
7   End if
8   End if
```

The algorithm starts by inspecting and comparing the sensor description, with respect to Mac address and node ID, to information that is in the Learning table, in the application database. If one or more of the nodes starts changing their association attributes randomly, the frequency of change is calculated and recorded in a separate table in the database. The fuzzy logic module is also invoked to monitor the level of threat as well as to provide a network correction solution.

ALGORITHM SEGMENT 3: FUZZY LOGIC MODULE

```
1. Define Linguistic variables
2. Define membership functions
3. Define rules
4. Use membership function plots to covert crisp input into fuzzy values
5. Evaluate rules in rule base based on fuzzy values
6. Summations of results of every rule inferred
7. Deffuzify output (covert output to non-fuzzy values)
```

The input for the fuzzy logic in the IDS is the measured frequency of change for any sensor that changes its association attributes. This value is mapped onto fuzzy set membership functions. If a node changes its mac address several times, it actually implies that the node is associating itself afresh to the gateway a lot of times.

This can constitute as basis for a flood attack, as one node could request attention from the gateway in as many times as it takes to block the gateway from discovering other legitimate nodes in the time cycle. In the case where there are multiple hops before the message is received at the gateway, it also implies that the sensors relaying the packets to the gateway would be kept awake by one or more rogue sensors and thus filling up their hash tables quicker than they can manage to hold. This is not desirable, as they have limited storage capacity.

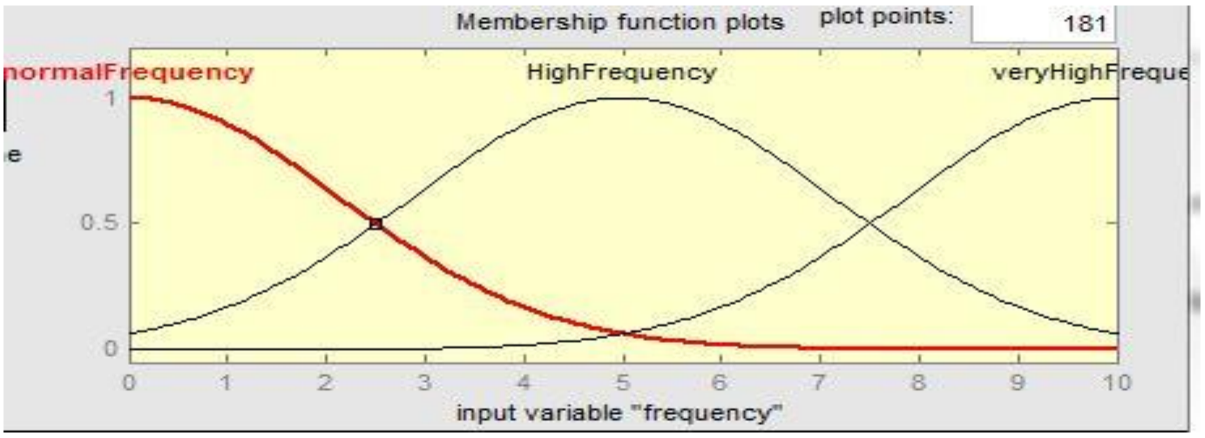


Figure 5-4 Fuzzy Membership Functions

From figure 5.4, the input of frequency, on the x-axis can be mapped into one or more fuzzy membership functions named normal frequency, high frequency of very high frequency that define the level of threat a node can possess. When the frequency is mapped, it can be into one or more sets to a certain degree (Y-axis).For instance, when frequency is zero (0) it is inside normal frequency set to a full degree of one (1). If the reading is equal to one (1), it means that the variable input is a complete member of a particular set. This process of mapping an input variable into fuzzy membership plots is called fuzzification (Sebastian & Philip, 2014).

With fuzzification, a value can fall into more than one membership function to certain varying degree. There is a general formula that is normally used to find the degree into which a value falls inside different membership functions. This formula is explained below;

For input value ***x,*** with ***c*** being the centre of the membership function, σ representing width of the membership function and ***e*** being the exponential function, the degree into which x falls inside each and every membership function, as mapped on the Y-axis of the membership functions, can be calculated as follows.

Equation 5.1 calculating degree of membership

$$\mu(x) = e^{-\frac{1}{2}\left(\frac{x-c}{\sigma}\right)^2}$$

As explained before, **x** can fall into more than one membership set, but the value taken is the largest of the degrees between those two or more membership functions, into which x participates.

The next step in the fuzzy logic module is the definition of rules. For the purpose of this implementation, the rules are found as follows, where threat Zone is the representation of the output membership functions.

1. If (frequency is normal Frequency) then (Threat zone is small Threat) (1)
2. If (frequency is High Frequency) then (Threat zone is medium Threat) (1)
3. If (frequency is very High Frequency) then (Threat zone is high Threat) (1)

After a fuzzy input has been achieved through fuzzification, it is mapped to one or more of the rules in a process known as inference (Shaout, 2014).

Once inference has been done, the output is still a fuzzy value, which should then be defuzzified to produce a crisp output that can be useful in subsequent computations. In defuzzification, output of the inference is mapped onto the output member functions and a defuzzification formula is used to get the crisp output from the output member functions.

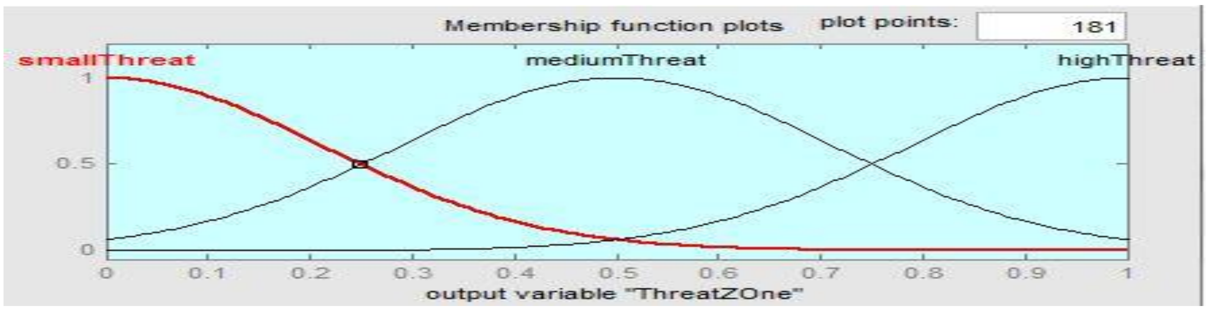


Figure 5-5 Output Member Functions

Figure 5.5 above shows the output member functions of the IDS. The inference value can fall in one or more output member functions and it has to be defuzzified, such as to give an output.

Defuzzification is therefore the process we used to produce crisp output from fuzzy grade input. There are several ways of which this can be achieved with one of the popular ways being the centroid method (Saneifard, 2011). With the centroid method of defuzzification, for example, the centre of area or centre of gravity is calculated. Fuzzy membership function can take on any shape from a trapezoid, triangle or even a Gaussian. For either one chosen, to find the centroid all that is needed is the total area under the shape and the point along the x-axis at which the area would balance.

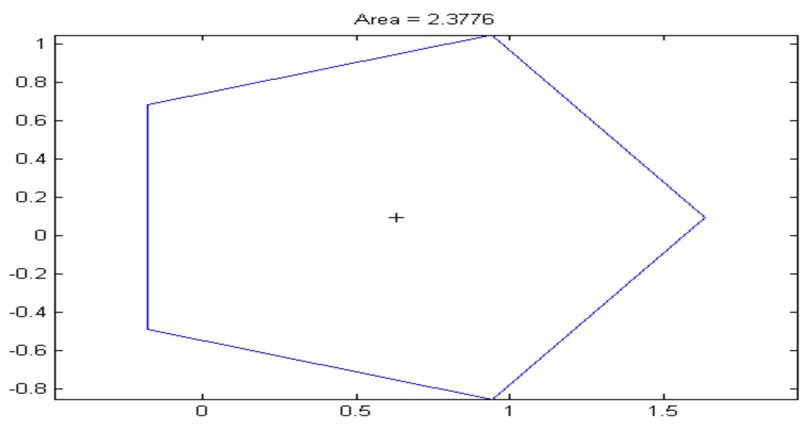


Figure 5-6 Centroid of a Shape, Source: Biomecadio.com

Finally, after the response has been given by the fuzzy logic system, the application is to ignore the node that is acting up, if it assumed not capable of causing any significant harm by the fuzzy logic system. Conversely, a node can also be reset or isolated from the network entirely. To achieve this task, the application sends a command to the node that is affected through use of Over the Air programming.

All the sensors connected to the network and which are communicating with the gateway can also receive commands or new programs from the user through the gateway using OTA. This feature allows for remote sensors to be maintained without physically accessing them (Quadri & Sidek, 2014).

5.7 Algorithm analysis

Asymptotic notation is used in this research to analyse the behaviour of the security algorithm as input eventually grows. This section establishes whether the algorithm will remain robust and fast as input grows. Input in this case refers to the nodes that have answered a beaconing request at a particular time interval and are sending data to the coordinator in a transmission cycle.

The first task to tackle is finding the number of steps the algorithm takes when trying to establish presence or absence, of security breaches given a reading from the sensor. The learning segment of the algorithm always gets called first when a data reading is sent towards the network and it invokes the detection module and subsequently the fuzzy module, if need be.

When studying the learning segment, the first challenge is the main loop. The rest of the algorithm, (learning and detection) run from within this loop. When disregarding the entire code, within the loop, it is found that the loop has a total 4 instructions. At the beginning there is the initialization of ***i*** ***(i=1)*** and the checking, if the loop is still within boundary ***(i< n)***. In this case ***i*** represents the node currently being probed by the algorithm out of a total of all nodes that are part of the transmission cycle. At the end of, each run through the body of the loop, there is another checking of boundary and the increment of ***i (i++),*** which essentially means move to the next node in the sending queue in this current transmission cycle. This gives a total of 4 instructions.

When representing this as function, of the number of nodes that can be present at any time in the network, we get ***f(n) = 2+ 2n***. That is, 2 instructions that do not directly affect n and two more that do affect n directly.

The next step is to find the Big-OH of the algorithm, which represents the worst case scenario that the system can have. To achieve this, the total number of instructions that can ever happen, is counted. This is represented by a case where the instruction steps of the algorithm fire up. These instructions are all the statements in the learning section, detection section as well as the fuzzy algorithm steps. There are, overall, 43 instructions besides those that run the loop. The function at this point can be represented as ***f (n) = 43 + 2 +2n***, which can be simplified to ***f(n) =45 +2n***.

The calculation above shows that there are 45 +2n instructions required when monitoring sensor nodes for security threats in the worst case, where n is the number of nodes sending data towards the gateway at a point in time.

However, this function would only hold true if we make a few assumptions. First, it would have to be assumed that all instructions such as extraction of node addresses in the learning stage, calculation of frequency of change in detection and monitoring as well as definition of fuzzy linguistics in the fuzzy logic module will happen in constant time for every node. It is also assumed that all branching statements defined by “if statements” would also be dealt with in constant time or **Big-Oh (1)** for every node being monitored.

The function will therefore hold, with its approximation of 45 instructions when the main loop is not considered. However, all these instructions differ when implemented in different programming languages and are not largely dependent on the growth of the number of nodes that are sending messages to coordinator.

In complexity analysis, we consider the number of instructions that grow with the input size. Furthermore, asymptotic notation allows us to filter out slow growing instructions or those instructions that do not directly affect the algorithm as the size of the input grows. This means the 45 instructions remain in constant time as the functions is refined even more. What then remains is ***f(n)= 2n*** as it has a directly impact as size grows.

Therefore, using the same logic as applied above the function is reduced to **f(n) = n.** These properties are not defined in strict mathematics but apply when dealing with asymptotic notation.

Therefore, the behaviour of ***f(n)= 45 +2n*** is described by the function ***f(n)= n***. Since this was the worst-case scenario, the notation is represented as ***O (n).*** The security algorithm has a time complexity of ***O(n***) owing to the fact that at the base of it all it employs a single loop, and several constant time instructions. An algorithm running in ***O(n)*** is said to be running in linear time, meaning that as the size of the input increases especially for a really large input factor, the running time also increases linearly with the input (Demaine, 2005).

This also serves as justification for the algorithm as linear complexity time is in most cases, deemed desirable (Rowel, 2016). To put this into perspective, figure 5.7 depicts generic algorithm run times.

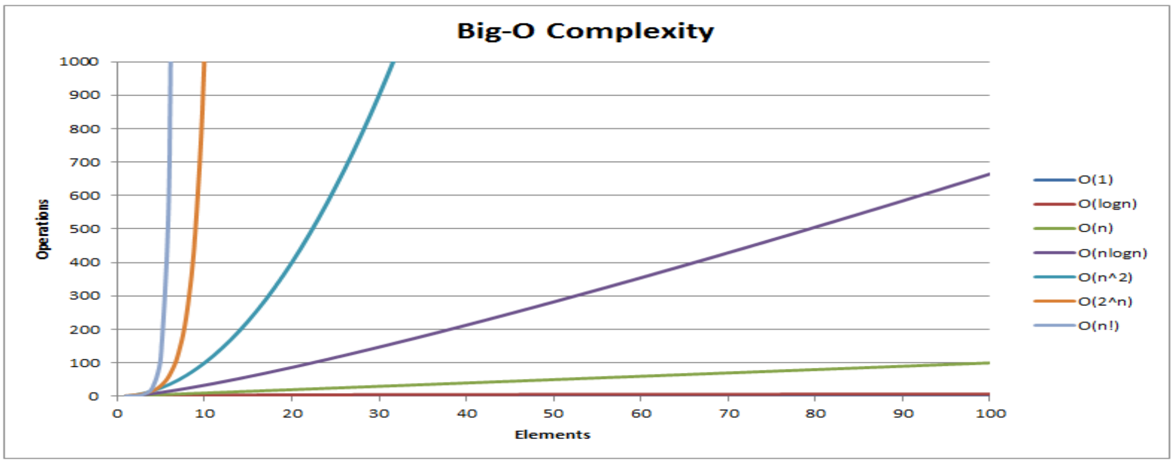


Figure 5-7 Big-Oh Chat, Source: bigohcheatssheet.com

The algorithm can at this point be trusted to remain fast regardless of the number of nodes that form the network and regardless of the underlying computer platform on which it can be installed on. At the same time, it should be pointed out at this juncture that there is no additional software that is required to be in the sensor nodes and this goes a long way in ensuring that the battery life is optimised for sensing and not spend on helping the IDS with detection of threats.

5.8 IDS database

A database system is the third major component of the IDS used in almost every stage of the security algorithm. This database was programmed to not be a standalone database but rather an internal database packaged together with the algorithm, as well as the front end GUI to make one application.

The database has a table used when the algorithm is learning the state of the network. This table is used when making a record of the nodes that have responded to the beaconing request and have sent association signal to the coordinator to show that they are ready to send in data. Every node gets to have its mac_address and its node address recorded once. If a record of a node_ID is already in the database, it is not recorded again.

Additionally, the database has a table used by the detection section of the algorithm, which is only used to record the nodes that have been changing their mac_address since the first time they were recorded. It counts the number of times a mac_address has changed as “frequency of change”. This information is used before the fuzzy logic module can be evoked. This parameter serves as input to the fuzzy logic module. The fuzzy logic module is only called upon by the detection section of the algorithm if this reading has gone above one(1), for any node under inspection.

Finally, there is a table that records the data carried sensors and this table is meant to be accessed by third party applications that need the data for user consumption.

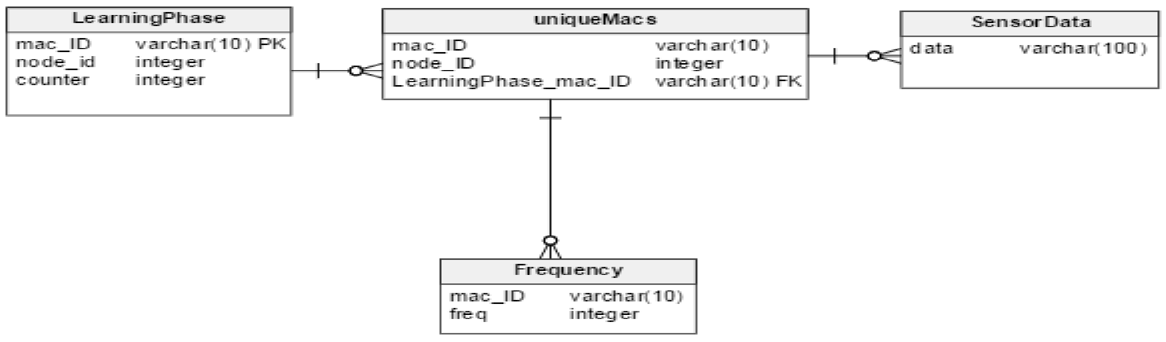


Figure 5-8 IDS Database Entity Relational Diagram

6. Evaluation, Discussions and Conclusions

6.1Testing and evaluation

In this chapter, the IDS with its detection algorithm is tested in a controlled environment to establish if it works as envisioned. The chapter also presents the discussions points gained from the development IDS for the purpose of this dissertation. Future endeavours with regard to the IDS are also outlined in this chapter.

In the previous chapter, it was established that the algorithm alone executes instructions in linear time as the size of the input grows. This theory alone makes the IDS practically commendable for consumption in a WSN setting where a large number of nodes are expected to be a part of the network.

6.1.1 Sensor Node Discovery

The first test phase conducted was to check if all the nodes in the network were discoverable and registered in the database, every time the network was re-started.

This is important because the anomaly detection algorithm only monitors nodes that are deemed part of the network. That is, only the nodes that have been registered into the database.

For this particular assessment, all the nodes were first programmed such that they are sending data at a selected baud rate and are transmitting their addresses together with the data they are reporting back to the gateway.

For this test, three (3) Libelium nodes using ZigBee Xbee transmitters were used and a selected number were reprogrammed to not comply with the set parameters necessary to be a part of the network with each iteration of the experiment over a period ten (10) times. For the purposes of this experiment, the sensors were measuring temperature and the amount of light.

The IDS database was subsequently checked to determine how many nodes had been registered, from those deemed correctly configured.

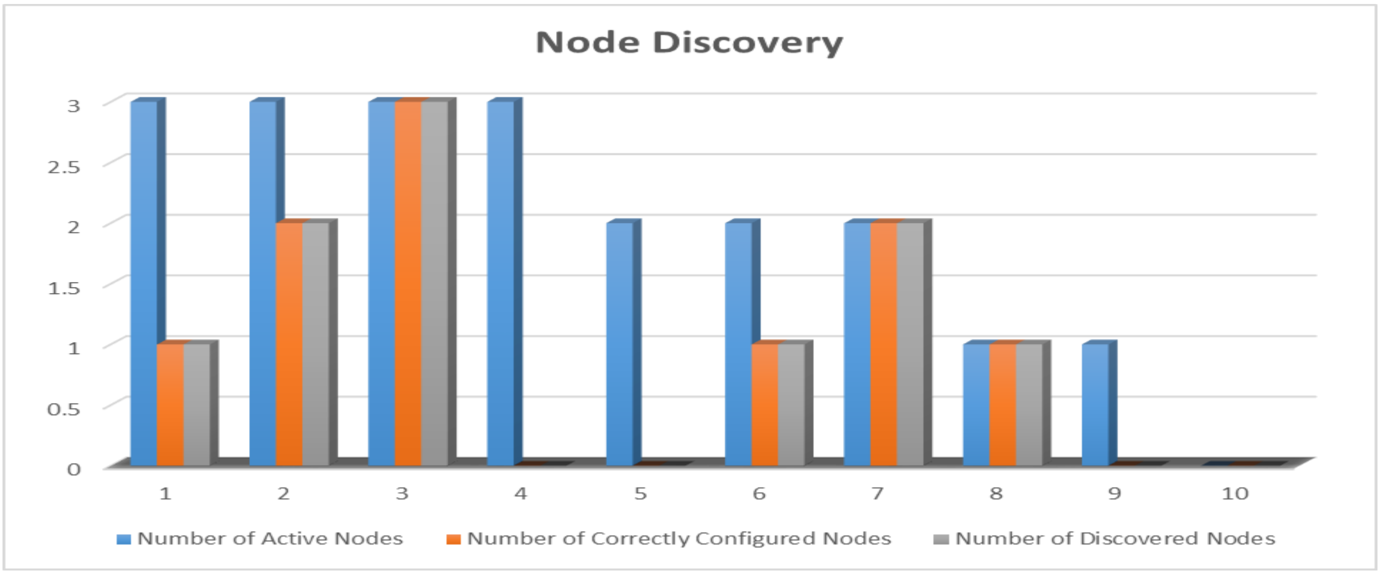


Figure 6-1 Node Discovery Bar Graph

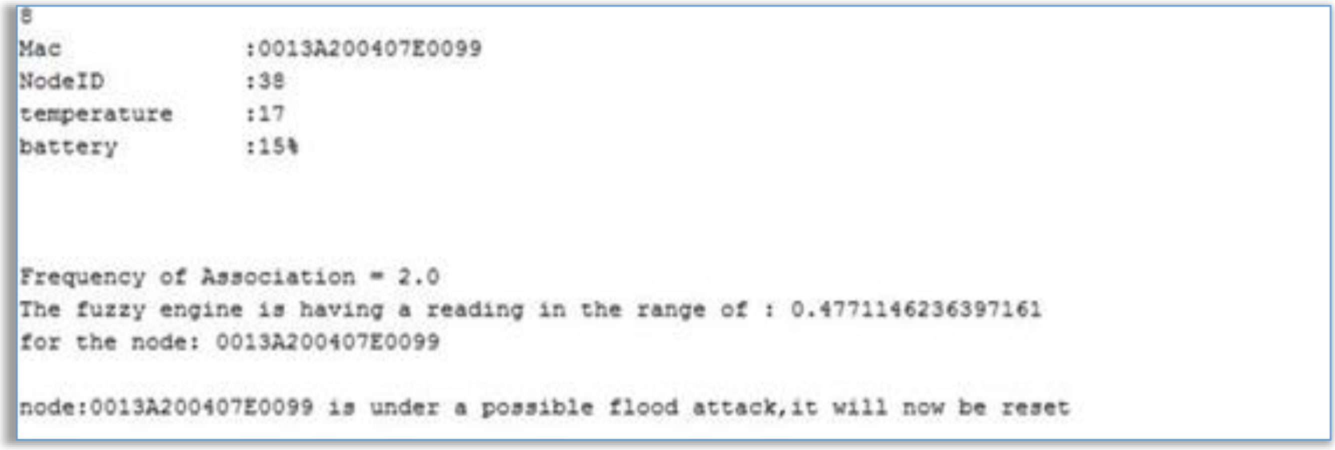
As perceivable from the results in the bar graph above, the algorithm managed to acknowledge every correctly configured node and recorded it as part of the network. No inactive or correctly configured sensor was every recorded or discovered as part of the network.

6.1.2 Anomaly Detection

The second test of the experiment was to test the prowess of the anomaly detection when dealing with simulated ZigBee flood attack. This simple experiment was conducted by picking out one of the nodes and reprogramming it to behave as though it is trying to cause a flooding of the hash tables in the network. This means that it was reprogrammed to change its mac_address intermittently when sending data to gateway.

This was made to be slightly challenging by setting up a random delay period for a node to start behaving maliciously. This was to ensure that a node does not begin to simulate an attack as soon as the node is introduced into the network.

The first cycle of transmission for any node is considered as the learning time period and the algorithm is designed not to monitor for DOS flood on the first transmission cycle of a node as no prior information about the node would be in the database.



```
8
Mac              :0013A200407E0099
NodeID           :38
temperature      :17
battery          :15%



Frequency of Association = 2.0
The fuzzy engine is having a reading in the range of : 0.4771146236397161
for the node: 0013A200407E0099

node:0013A200407E0099 is under a possible flood attack,it will now be reset
```

Figure 6-2 Screen Shot Of the IDS during Rogue Node Detection

The screen shot in figure 6.2 showcases the IDS in action against a rogue node, as can be viewed from the end user GUI. It can be realised at this point that the node has sent data using a different mac address twice, meaning two mac address have been coming from the same source as can be deduced from the connection with the node ID. This in essence means that the node has managed to associate itself twice since its initial recognition as part of the network. It is at this point giving out false information about itself leading the gateway to believe that a total of 3 nodes have associated to the network yet it is all from one source.

Additionally, this means that one node is able to monopolise the sending channel. In the event where there is a hop before reaching the gateway, the node receiving data from this rogue node would eventually crash out due to having its limited hash table space, for storing addresses, being filled up too quickly by one malicious node, associating itself many times to the network.

In the screen shot however, it can be appreciated that the IDS has captured this behaviour as well as the number of times the node has re-associated itself and the fuzzy logic as well has determined the extend at which he node can cause harm to the network. The fuzzy logic embedded in the algorithm also makes a recommendation for the node to be reset. At this point, the IDS issues an OTA to reset the offending node.

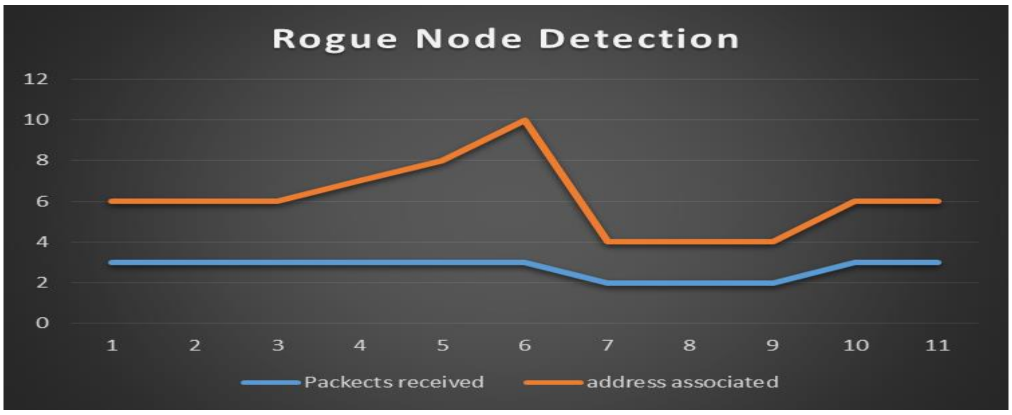


Figure 6-3 Rogue Node Detection

Figure 6.3 above shows a graph that was used to keep track of the number of packets received in a time period compared to the number of associated addresses behind those packets. The time units between 1 and 3 were a period within which the IDS was learning the network from the moment it was launched.

From time sequence 4 to 7 the rogue node started acting up and re associating itself to the network. This was corrected and the chart shows a normal operational period between time sequences 7 to the end of the simulation period.

Overall, for this experiment, one node was re programed to be malicious and re introduced to the network. After a period of time it started acting up but was monitored for a limited time period by the IDS until it was reset and dropped from the network.

In a bit to find as many flaws as possible, this process was then repeated for a minimum of 40 recorded tries, during testing using different sensor nodes as the rogue node for each experiment iteration and at times varying the number of rogue nodes. Each of those times, the rogue node was sniffed out by the IDS as can be seen in the chart below.

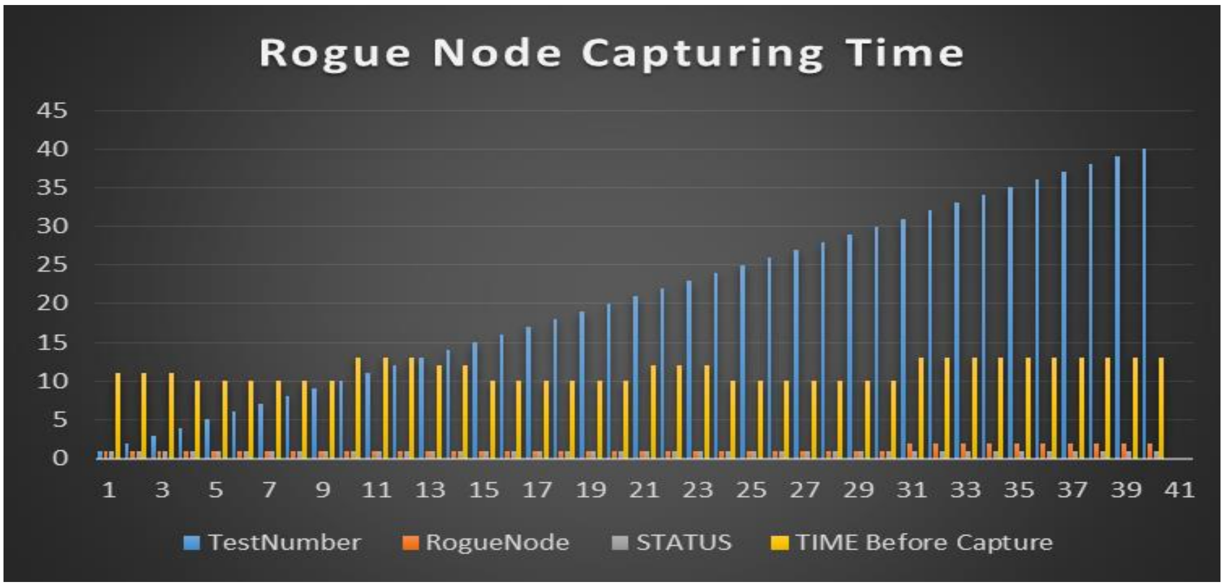


Figure 6-4 Average Time for Rogue Sensor Detection

6.1.3 False Positives

The fact that this was an indoor experiment or that it was a single session experiment and not longitudinal in nature might have hindered the presence of false positives. Figure 6.4 above depicts the actual number of tries this experiment was conducted in order to check whether within every iteration the malicious node could be sniffed out.

Using the formula for measuring the false positive rate (FPR), where FPR=FP/(FP+TN), we find that the rate is 0 for our experiment as 0/(0+39)=0.

On the other hand the true positive rate for threat detection is 1 as 39/(39+0)=1.

6.4 Validating the Research Assumption

It was stated in methodology, how experiments are also conducted to determine whether to accept or reject the null hypothesis. In this study, based on the calculations on the rate of false positives, the research accepts the research assumption initially made in the dissertation.

The study makes a conclusion that using the fuzzy logic as the artificial intelligent notation when developing a threat detection algorithm, can lead to a high efficiency rate, as there was lack of observable false positives during the experiment testing.

6.2 Future Implementation Work

6.2.1 Triple Modular Redundancy Check

Triple Modular Redundancy check is a system where three computer systems, doing the same process send their processing result to a singular voting computer system, which will takes the side of the majority between those three computers and gives it output (Shubham & Mahesh, 2014).

Additionally, Triple Modular Redundancy (TMR) can also be referred to as fault tolerant computing as the system can still work even when one of the computers is dead.

The concept of TMR can be implemented together with the IDS in this research to further enhance the security of data collected. That is, the point at which data is stored for third party usage.

In theory, this would consist of three Xbee Cluster Heads all connected to the sink node and sending all their data to it. The data is checked for signs of intrusions as stated in this study, but the data values from all three Cluster heads are kept in a separate table in the database where the majority value would be considered the correct value to be stored and be read by the application for which the sensors have been deployed.

In essence, there would be three separate networks, each governed by an Xbee Cluster Head, and all three networks are in the same space, measuring one entity and this is to ensure that data passed on for consumption is as consistent as possible.

However, for this concept to work, the sensors in all three clusters would have to be placed relatively close to each other, as too much space would lead to inconsistent data. For example, in a situation where the sensors are measuring soil moisture after a sprinkling session, it might happen that the soil moisture is unevenly distributed and if sensor clusters are comprised of sensors that have too much space in between, all three clusters could end with different values.

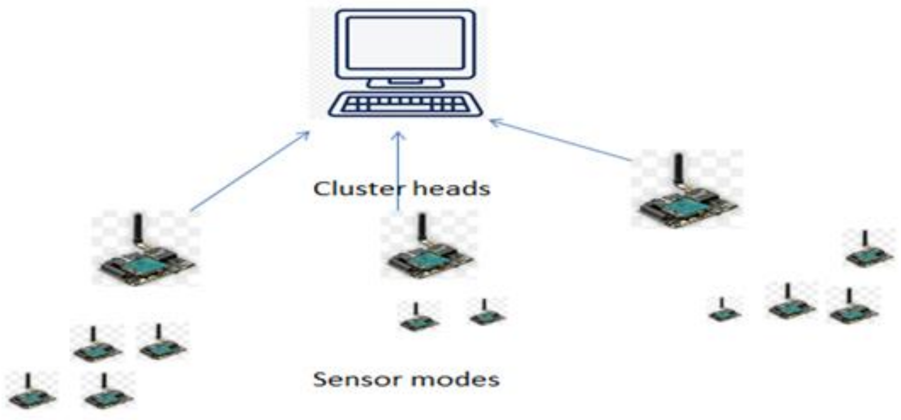


Figure 6-5Triple Redundancy Modular System of Xbee nodes

6.2.2 Additional Communications Protocols

Furthermore, looking forward towards the future, we would like to include more protocols up for detection as opposed to just ZigBee when dealing with WSNs protocols such as Bluetooth and WI-FI. A protocol such as Bluetooth has been shown to have vulnerability issues which can be easily hacked into (Vivekparma, 2016). It would be very good to find ways of using a similar approach to curb this issue. ZigBee was chosen for this research as it proved to be a cut above the rest when used in WSNs.

Additionally, there are not many IDS designed specifically for use with ZigBee and getting a copy of the few that had proposed or designed proved to be a rather difficult task, meaning that it made it difficult to benchmark our approach against other IDS in the same domain

6.2.3 Integration into the Framework

One of the major future milestones of this research will possibly be the inclusion of this IDS into the IoT framework being designed at the Central University of Technology, as already stated in this dissertation (Masinde, 2013). The motivation of this research was ultimately the prospect of being included in the framework.

6.3 Limitation and Challenges Faced

Our Study into the WSN security took considerable time to take off as we could not find suitable sensor modes for all the novel ideas we had in mind such as the use of Java powered mobile agents for WSN security solutions. This is a problem that can be solved once we start developing localised and very flexible sensor modes.

Secondly, the purchasing of good WSN simulators proved almost impossible as some were too exorbitantly expensive, some with dominant mention in general literature were found to be obsolete or not optimised for the scope we were aiming for. On the other hand there were some manufacturers who would not find ways to do business with us as they were all the way in Europe.

6.4 Discussions and Conclusions

In this dissertation paper, an anomaly detection algorithm was adapted, and modified such that it can make use of logic reasoning through fuzzy logic methodologies. This algorithm was then included to be a part of an intrusion detection software.

The research focused on the ZigBee protocol and its vulnerability to flood attacks, during node discovery and association to the network. The research then made a novel initiative of taking the same approach that a typical wired network IDS usually takes, of being actually hosted on a server or central computer system that managed such a network. This means that the IDS was designed to be a computer software not installable in sensor node themselves. It is believed that this leads to longer battery life for nodes in a WSN setting.

Research in WSNs highlighted in Chapter 2, believe in the notion of tying a WSN security mechanism down to the node itself or at least have nodes monitor themselves collectively. This research hypothesizes that the separation of node from IDS mechanism can eventually lead to longer battery life for nodes especially based on the fact that nodes are often used in isolation, for example, when tracking the movement of a wild animal.

Security is very important to a WSN but equally of importance is the need to save power which is only overshadowed by the need to also use sparingly, resources such as memory and processing power of a node. The fact that the sink node is usually connected to a computer with comparatively unlimited memory, power and processing power, the research saw it fit to exploit the advantage of those end node computers and using them to monitor flooding threats in a ZigBee WSNs.

In general, the research introduced an IDS methodology that makes use of a fuzzy logic embedded security algorithm, to monitor for threats in a ZigBee WSN by aggregating, making calculations and assumptions based on the information received from the nodes the moment they send in data from monitoring assignments.

This IDS was fully realised in code based on the design outlined in Chapter 4 of this dissertation. Furthermore, the algorithm was analysed using asymptotic notation to determine how well it fares as the number of node in a network inevitable explode in size. The Asymptotic calculations state that the algorithm will execute in linear time as the size of the network grows. This is actually a good and commendable time complexity for any algorithm, bettered only the time complexity of constant time.

The IDS during simulation was found to be effective in capturing rogue sensor nodes without any problem. The IDS was tested to ascertain its usefulness in a controlled simulation setting. This also gives an idea as to how it will perform when deployed in a real world situation.

The algorithm that makes up the IDS executes in a very promising time complexity meaning it will be very friendly on the RAM of any computer platform it is installed on. The great thing about this approach is that there is no extra code work that is left to be done by the nodes except the little bits of code that help them to broadcast their descriptive information with every data packet send, formatted in a way the IDS can ,make use of.

It is also important to note that a flood attack would not be possible in the first place if this bit of code is excluded by an attacker. This is because of the fact that, any incorrectly configured node is not recognised as part of the network, making this approach again, rather proficient.

The IDS was tested against several simulated attacks and it managed to detect and correct the anomaly.

Finally, all of the project milestones outlined in chapter were completed successfully.

7. References

Abduvaliyev, A. et al., 2013. On the Vital Areas of Intrusion Detection Systems in Wireless Sensor Networks. *IEEE Communications Survey and Tutorials,* 15(3), pp. 1223-1237.

Abirami.K & Santhi.B, 2013. Sybil attack in Wireless Sensor Network. *International Journal of Engineering and Technology (IJET) ,* 5(2), pp. 620-623.

Agah, A., Sajal, K. & Kalyan Basu, D., 2004. *Intrusion Detection In Sensor Networks:A non Cooperative Game Approach,* s.l.: s.n.

Albag, H., 2001. *Network & Agent Based Intrusion Detection Systems,* s.l.: s.n.

Albag, H., n.d. *Network & Agent Based Intrusion Detection Systems,* s.l.: s.n.

Alhumaidan, F., 2012. A Critical Analysis and Treatment of Important UML Diagrams Enhancing Modeling Power. *Intelligent Information Management,* Volume 4, pp. 231-237.

Amiri, F., Yousefi, M. & Lucas, C., 2011. Mutual information-based feature selection for intrusion detection systems. *Journal of Network and Computer Applications,* 34(4), pp. 1184-1199.

Anon., 1997. *Best, Worst, and Average-Case Complexity.* [Online]
Available at: https://www8.cs.umu.se/kurser/TDBAfl/VT06/algorithms/BOOK/BOOK/NODE13.HTM
[Accessed 13 September 2016].

Anon., 2005. *Host versus Network based Intrusion Detection Systems,* s.l.: Sans Institute.

Anon., 2014. *CyberSafe Limited.* [Online]
Available at: https://cybersafe.com/search/node/intrusion%20detection
[Accessed 9 November 2014].

Anon., 2015. *Microsoft Developer Network.* [Online]
Available at: https://msdn.microsoft.com/en-us/library/dd409432.aspx
[Accessed 29 June 2016].

Anon., 2016. *University of Waterloo.* [Online]
Available at: https://uwaterloo.ca/information-systems-technology/about/policies-standards-and-guidelines/security/incident-response-procedure
[Accessed 26 September 2016].

Antonio, J., Latif, L. & SKarmeta, A., 2013. *The Internet of Everything through IPv6:An Analysis of Challenges, Solutions and Opportunities,* s.l.: s.n.

Anuradha, C., Sundararajan, M. & Arulselvi, S., 2015. A comparative Study for Intrusion Detection Systems in Wireless Sensor Networks. *International Journal of Innovative Research in Computer and Communication Engineering,* 3(9), pp. 7932-7936.

Ashoor, A. S. & Sharad , G., 2011. Importance of Intrusion Detection System. *International Journal of Scientific & Engineering Research,* 2(1), pp. 1-5.

Aydin, A., Zaim, H. & Ceylan, G., 2009. A hybrid intrusion detection system design for computer network security. *Computers and Electrical Engineering,* 35(3), pp. 517-526.

Baras, J., 2004. *ATEMU: a fine-grained sensor network simulator,* s.l.: IEEE Explore.

Berger, I. & Dov, D., 2003. *Object-Process Methodology (OPM) vs. UML:A Code Generation Perspective,* Carmel Mountain: University of Haifa.

Bhar, J., 2015. A Mac Protocol Implementation for Wireless Sensor Network. *Journal of Computer Networks and Communications,* Volume 2015.

Bhaskar, K., 2005. *An Introduction to Wireless Sensor Networks,* California: University Of Southern California.

Bhaumik, R., 2010. *Mobile Agent based Architecture for Wireless Sensor Networks,* s.l.: Helsinki University of Technology.

BMJ Publishing Group , 2016. *thebmj.* [Online]
Available at: http://www.bmj.com/about-bmj/resources-readers/publications/statistics-square-one/11-correlation-and-regression
[Accessed 28 October 2016].

Bowden, E., 2007. *Network security Journal-Network Based Inrusion Detection System.* [Online]
Available at: http://www.networksecurityjournal.com/features/network-based-intrusion-detection-systems-031607/
[Accessed 22 October 2014].

Bretano, J. & Snapp, S., 1991. *An architecture for a distributed Intrusion Detetction System,* s.l.: s.n.

Bretano, J. & Snapp, S., n.d. *An architecture for a distributed Intrusion Detetction System,* s.l.: s.n.

Buch, D. & Jinwala, D., 2011. PREVENTION OF WORMHOLE ATTACK IN WIRELESS SENSOR NETWORKS. *International Journal of Network Security & Its Applications (IJNSA),* 3(5).

Butun, I. & Morgera, S., 2014. *A Survey of Intrusion Detection Systems in Wireless Sensor Networks,* s.l.: IEEE.

Cannady, J. & Harrell, J., 2009. *A Comparative Analysis of Current Intrusion Detection Technologies,* s.l.: s.n.

Carrie, W., 2007. Research Methods. *Journal of Business & Economic Research,* 5(3), pp. 65-70.

Chen, R., Hsieh, C. & Huang, Y., 2009. *A New Method for Intrusion Detection on Hierarchical Wireless Sensor Networks,* s.l.: s.n.

Colorado State University, 2016. *Colorado State University.* [Online]
Available at: http://writing.colostate.edu/guides/guide.cfm?guideid=64
[Accessed 1 November 2016].

Cort, A., 2004. *Algorithm based Approaches to Intrusion Detection and Response,* s.l.: SANS Institute.

Da Silva, A., Martins, M. & Wong, H., 2005. *Decentralised Intrusion Detection in Wireless Sensor Networks,* s.l.: s.n.

Dassault Systems, 2013. *The Internet of Things:The past,The present,The Future,* s.l.: s.n.

David, L., Rayes, A. & Morrow, M., 2012. The Internet of Things. *The Internet protocol Journal,* 15(3).

Demaine, E., 2005. *MIT.* [Online]
Available at: http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-046j-introduction-to-algorithms-sma-5503-fall-2005/video-lectures/lecture-2-asymptotic-notation-recurrences-substitution-master-method/lec2.pdf
[Accessed 5 May 2016].

Dick, C., 2016. *ISA.* [Online]
Available at: https://www.isa.org/standards-and-publications/isa-publications/intech-magazine/2004/may/networking-and-communications-zigbee-short-on-power-by-design/
[Accessed 22 September 2016].

Draˇsar, M. & Vykopal, J., 2013. *Flow-based Brute-force Attack Detection,* s.l.: s.n.

Engblom, J., 2002. *Processor Pipelines and Static Worst-Case Execution Time Analysis,* Stockhlom: Elanders Gotab.

Fei, Y., 2011. *A Survey of Wireless Network Simulator Tools,* s.l.: s.n.

Gaikwad, D. et al., 2012. A Proposal for Implementation of Signature Based Intrusion Detection System Using Multithreading Technique. *International Journal Of Computational Engineering Research ,* 2(7), p. 59.

Garcia, et al., 2014. Software process modeling languages: A systematic literature review. *Information and Software Technology,* 2(103-116), p. 56.

Garth, C., Lance, H. & Niki, P., 2011. Location-aware, Trust-based Detection and Isolation of Compromised Nodes in Wireless Sensor Networks. *International Journal of Network Security,* 12(2), pp. 107-117.

Gideon, K. & Jeroen, R., 1988. On Between-subjects design versus within-subject comparisons in Testing Utility Theory. *Organizational Behavior and Human decision Processes,* Volume 41, pp. 233-247.

Gondwal, N. & Diwaker, C., 2013. DETECTING BLACKHOLE ATTACK IN WSN BY CHECK AGENT USING MULTIPLE BASE STATIONS. *American International Journal of Research in Science, Technology, Engineering & Mathematics,* Volume 13, p. 149.

Gong, Q., Li, G. & Pang, y., 2014. Design and Implementation of Smart Home System Based on ZigBee Technology. *International Journal of Smart Home,* 8(6), pp. 143-156.

Gopalakrishna, R. & Spafford, E. H., n.d. *A Framework for Distributed Intrusion Detection using Interest Driven Cooperating Agents,* s.l.: s.n.

Hakan, A., 2012. *Network & Agent Based Intrusion Detection Systems,* s.l.: s.n.

Harris, A. et al., 2006. The Use and Interpretation of Quasi-Experimental Studies in Medical Informatics. *Journal of the American Medical Informatics Associations,* 13(1), pp. 16-23.

Hassan, M. M. M., 2013. Current Studies On Intrusion Detection System,Genetic Algorithm and Fuzzy Logic. *International Journal of Distributed and Parallel Systems (IJDPS),* 4(2), pp. 35-47.

Honus, L., 2009. *Design, implementation and simulation of intrusion detection system for wireless sensor networks,* s.l.: s.n.

Hoque, M., 2012. AN IMPLEMENTATION OF INTRUSION DETECTION SYSTEM USING GENETIC ALGORITHM. *International Journal of Network Security & Its Applications,* 4(2), pp. 109-117.

IBM, 2008. *Security Intrusion detection,* s.l.: IBM.

Inella, P., 2001. *Symantec.* [Online]
Available at: http://www.symantec.com/connect/articles/introduction-ids
[Accessed 9 November 2014].

Innella, P., 2001. *Symantec-An introduction to IDS.* [Online]
Available at: http://www.symantec.com/connect/articles/introduction-ids
[Accessed 22 October 2014].

Islam, S. & AshiqurRahman, S., 2011. Anomaly Intrusion Detection System in Wireless Sensor Networks: Security Threats and Existing Approaches. *International Journal of Advanced Science and Technology ,* Volume 36, p. 4.

Janet, H., Rachel, T. & Sheila, H., 2006. *Qualititative Longitudinal Research: A discussion Paper,* London: London South Bank University.

Jansen, W. A., n.d. *INTRUSION DETECTION WITH MOBILE AGENTS,* Gaithersburg: National Institute of Standards and Technology.

Jennifer, W., 2013. *Mixed Methods:Integrating Quantitative and Qualitative Data Collection and Analysis While Studying Patient-Centered Medical Home Models ,* s.l.: PCMH Research Methods Series.

Jerome, R., 1997. *Using Statistics to Determine Causal Relationships,* s.l.: s.n.

Jessica, I., Ann, B. & Clive, H., 2009. Research Methods – a Case Example of Participant Observation. *Electronic Journal of Business Research Methods,* 7(1), pp. 39-46.

Jiehan, Z., Pakkala, D. & Perälä, J., 2007. *Dependency-aware Service Oriented Architecture and Service Composition,* s.l.: The Computer Societty.

Johnson, R., 2002. *Fuzzy Logic and Fuzzy Logic Sun tracking Control,* s.l.: Calvin College.

Jones, N. & Pevzner, P., 2004. *An Introduction to Bioinformatics Algorithms.* 1 ed. Massachusetts: MIT press.

Jyothsna, P. & Rama, V. V., 2011. A Review of Anomaly based IntrusionDetection Systems. *International Journal Of Computing Applications,* 28(7), pp. 26-34.

Kaewchinporn, C. & Limpiyakorn, Y., 2013. Enhancement of Action Description Language for UML Activity Diagram Review. *nternational Journal of Software Engineering and Its Applications,* 7(2), pp. 255-271.

Kalle, R., Gómez-Herrero, G., Egiazarian, K. & Eriksson, S.-L., 2014. *A general definition of the big-oh notation for algorithm analysis,* Tampere: Tampere University of Technology.

Kaoru, O., Mianxiong, D. & Xiaolin, L., n.d. *TinyBee: Mobile-Agent-Based Data Gathering System in Wireless Sensor Networks,* s.l.: s.n.

Kaushal, k., Kaur, T. & Kaur, J., 2014. ZigBee based Wireless Sensor Networks. *(IJCSIT) International Journal of Computer Science and Information Technologies,* 5(6).

Khan, S., Lloret, . J. & Loo, . J., 2014. Intrusion Detection and Security Mechanisms for Wireless Sensor networks. *International Journal of Distributed Sensor Networks,* pp. 1-4.

Khan, W., 2013. Detection and Mitigation of Node Replication Attacks in Wireless Sensor Networks: A Survey. *International Journal of Distributed Sensor Networks,* Volume 2013.

KhorasaniZadeh, H., Mohamed Sidek, Z. & AB Manan, J.-L., n.d. *An overview of Intrusion Detection Systems,* s.l.: s.n.

Koffman, E. & Wolfgang, P., 2010. *Data Structure:Abstractions and Design Using Java.* 1 ed. s.l.:John Wiley & sons.

Kosmerchock, S., 2012. *Wireless Sensor Network Topologies,* s.l.: s.n.

Kosuke, I., Dustin, T. & Yamamoto, T., 2013. Experimental designs for identifying causal mechanisms. *Journal of the Royal Statsical Society,* 176(1), pp. 5-51.

Krishnan, M., 2014. *Intrusion Detection in Wireless Sensor Networks,* s.l.: s.n.

Krontiris, I. et al., 2009. Cooperative Intrusion Detection in Wireless Sensor Networks. *Lecture Notes in,* 54(32).

Krontiris, I., Dimitriou, T. & F, F., 2007. *Towards Intrusion Detection In Wireless Sensor Networks,* s.l.: s.n.

Kumar, A. & Chhabra, C., 2014. Intrusion detection system using Expert system (AI) and Pattern recognition (MFCC and improved VQA). *International Journal of Advance Research in Computer Science and Management Studies,* 2(5), pp. 145-151.

Kumar, A. & Gupta, S., 2013. Study on ZIGBEE Technology. *INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY,* 2(10).

Lee, A., Loo, J. & Lasebae, A., 2012. 6LoWPAN: a study on QoS security threats and countermeasures using intrusion detection system approach. *INTERNATIONAL JOURNAL OF COMMUNICATION SYSTEMS,* pp. 1-25.

Letou, K., Devi, D. & Singh, J. Y., 2013. Host Based Intrusion and Prevention Systems. *International Journal of Computer Applications,* 69(26), pp. 28-33.

Levis, P., Lee, N., M, W. & D, C., 2003. *TOSSIM: accurate and scalable simulation of entire TinyOS applications,* New York: ACM.

Levitin, A., 2003. *Introduction to the Design and Analysis of algorithms.* 1 ed. Philadelphia: Addison-Wessly.

Liao, H.-J., Lin, C.-H. & Lin, Y.-C., 2013. Intrusion Detection System:A comprehensive view. *Journal of computer and Netwrok applications,* 36(1), pp. 16-24.

Libelium, 2016. *Waspmode 802.15.4 Networking Guide,* s.l.: Libelieum Comunicaciones Distribuidas S.L..

Libelium, 2016. *Waspmode 802.15.4 Networking Guide,* s.l.: Libelieum Comunicaciones Distribuidas S.L..

Li, Z., Gao, Y. & Chen, Y., 2005. *Towards a High-speed Router-based Anomaly/Intrusion Detection System,* s.l.: s.n.

Loo, C., Lekcie, C. & M, P., 2006. Intrusion detection for routing attacks in sensor networks. *International Journal Of Distributed Sensor Networks,* Volume 2, pp. 313-332.

Lopez Research, 2014. *Building Smarter Manufacturing With The Internet of Things,* San Fransisco,CA: cisco.

Luoma, J., Kelly, S. & Tolvanen, J., 2004. *Defining Domain-Specific Modeling Languages: Collected Experiences,* Jyväskylä: s.n.

Malhotra, J. & Manpreet, 2015. Simulation Analysis of Tree and Mesh Topologies in Zigbee Network. *International Journal of Grid Distribution Computing,* 8(1), pp. 81-95.

Markert, et al., 2011. *Attack Vectors to Wireless Zigbee Network Communications- Analysis and COuntermeasures,* s.l.: Furwangen University of Applied Sciences, Germany.

Masinde, M., 2013. *Generic architecture for IoT services: an integrated drought early system for Sub-Saharan Africa.* s.l., s.n.

Masinde, M. & Bagula, A., 2015. A Calibration Report for Wireless Sensor Based Weatherboards. *Journal of Sensor and Actuator Networks,* 4(1), pp. 30-49.

Massin, R., Bergot, L., Fracchia, R. & Martret, C., 2010. OMNeT++-Based Cross-Layer Simulator for Content Transmission over Wireless Ad Hoc Networks. *EURASIP Journal on Wireless Communications and Networking,* Volume 50.

Mitchel, R. & Chen, I.-R., 2014. A survey of intrusion detection in wireless network applications. *Computer Communications,* 1(42), pp. 1-22.

Mitrokotsa, A., 2008. *Intrusion Detection Techniques in Sensor Networks,* s.l.: IOS Press.

Mount, D., 2003. *Design and Analysis of Computer Algorithms,* College Park: University Of Maryland.

Naz, R. & M, K., 2015. Rapid Applications Development Techniques: A Critical Review. *nternational Journal of Software Engineering and Its Applications,* 9(11), pp. 163-176.

Neethu, B., 2013. Adaptive Intrusion Detection Using Machine Learning. *IJCSNS International Journal of Computer Science and Network Security,* 13(3), p. 118.

Neha, G., Komal, S. & Sharma, A., 2016. Reducing False Positives in Intrusion Detection System. *International Journal of Computer Science and Information Technologies,* 7(3), pp. 1600-1603.

Nikita, T., Komal, S., V, D. & Deepti, P., 2016. A Review of Traffic Management System Using IoT. *International Journal of Modern Trends in Engineerring,* 3(4).

Ning, P. & Jajodia, S., 2002. *Intrusion Detection Techniques,* s.l.: s.n.

Ning, P. & Jajodia, S., 2002. *Intrusion Detection Techniques,* s.l.: s.n.

Obaid Amin, S., Shoaib Siddiqui, M. & Seon Hong, C., 2009. RIDES: Robust Intrusion Detection System for IP-Based Ubiquitous Sensor Networks. *Open Access Sensors Journal,* 9(1), pp. 3447-3468.

Omics International, 2014. *Omics Group Internaational.* [Online]
Available at: http://research.omicsgroup.org/index.php/Algorithm_design
[Accessed 11 January 2017].

Onat, I. & Miri, A., 2005. An intrusion detection system for wireless sensor networks. *Wireless and Mobile Computing,Networking and communications,* Volume 3, pp. 253-259.

Padmavathi, G. & Shanmugapriya, D., 2009. A Survey of Attacks, Security Mechanisms and Challenges in Wireless Sensor Networks. *(IJCSIS) International Journal of Computer Science and Information Security, ,* 4(1).

Pahlevanzadeh, B., Hosseini Seno, S. A., Kadhum, M. & Budiarto, R., 2008. *A Cluster-Based Distributed Hierarchical IDS for MANETs,* s.l.: s.n.

Pal Singh, V., Jain, S. & Singhai, J., 2010. Hello Flood Attack and its Countermeasures in Wireless Sensor Networks. *IJCSI International Journal of Computer Science Issues,* 7(3), pp. 23-27.

Pan, Q.-Y., Wu, J., Wang, Y.-H. & Ni, J., 2011. A Effective Method for Properties Access in MAC Layer of ZigBee. *Scientific Research,* Volume 3, pp. 149-152.

Pasupathi, N. & Nishanth, B., 2014. Artificial Intelligence In Power System. *IOSR Journal Of computer Engineering.*

Paul, W., 2003. *Experimental Design & Methodology,* s.l.: George Mason University.

Pleskonjic, D., 2003. *Wireless Intrusion Detection Systems(WIDS),* s.l.: s.n.

Prabhat, P. & Meenu, P., 2015. *Research Methodology: Tools and Techniques.* 1st ed. Moldova: Bridge City.

Prapulla, G, S. & C, T., 2015. Smart Refrigerator Using Internet of Things. *Journal of Multidisciplinary Engineering Science and Technology (JMEST),* 2(7).

Priyanka, S., 2013. Incremental Intrusion Detection System for Wireless Sensor Networks. *International Journal of Emerging Trends & Technology in Computer Science,* 2(6), pp. 322-326.

Probst, C. W., 2011. Guest editorial: Addressing Insider Threats and Information Leakage. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications,* 2(1), pp. 1-3.

Quadri, S. & Sidek, O., 2014. An Introduction To Over The Air Programming In Wireless Sensors. *Internal Journal Of Computer Science and Network Solutions,* 2(2), pp. 33-49.

Ranjan Srivastava, P. & Kim, T.-h., 2009. Application of Genetic Algorithm in Software Testing. *International Journal of Software Engineering and Its Applications ,* 3(4), pp. 87-94.

Ravi, P., 2014. An analysis of a widely used version of the CUSUM tracking signal. *JOURNAL OF THE OPERATIONAL RESEARCH SOCIETY,* Volume 65, p. 1189–1192.

Reggio, G., Leotta, M. & Ricca, F., 2013. *What are tthe used UML diagrams? A preliminary survey,* Genova: s.n.

Rowel, E., 2016. *Big Oh Cheats Sheet.* [Online]
Available at: http://bigocheatsheet.com/
[Accessed 8 May 2016].

Rumbaugh, J., Jacobson, I. & Booch, G., 2004. *The Unified Modelling Reference Manual.* 2nd ed. Boston: Addison-Wesley.

Rupert, B., 2009. *Protecting against Insider Attacks,* s.l.: Sans Insitude.

Saneifard, R., 2011. A Method For Deffuzification Based On Centroid Method. *Turkish Journal Of Fuzzy Systems,* 2(1), pp. 36-44.

SANS Institude, 2001. *Understanding Intrusion Detection Systems,* s.l.: SANS Institute InfoSec Reading Room.

Satish, D., Vaidehi, M. & Nithya, N., 2009. *Severity Prediction of Drought in a Large Geogrpahical Area Using Distributed Wireless Sensor Networks,* Patiala: s.n.

Scrucca, L., 2013. GA: A Package for Genetic Algorithms in R. *Journal of Statistical Software,* 53(4), pp. 1-34.

Seal, V., Raha, A. & Maity, S., 2012. A SIMPLE FLOOD FORECASTING SCHEME USING WIRELESS SENSOR NETWORKS. *International Journal of Ad hoc, Sensor & Ubiquitous Computing,* 3(1), pp. 1-16.

Sebastian, S. & Philip, J., 2014. Fuzzification of Fuzzy Sets. *Journal OF Chemical, Biological and Physical Sciences,* 4(3), pp. 2519-2523.

Sen, J., 2010. An Agent-Based Intrusion Detection System for Local Area Networks. *International Journal of Communication Networks and Information Security (IJCNIS),* 2(2).

Shaffer, C., 2009. *Data structures and Algorithm Analysis.* 3rd ed. Blacksburg: Prentice Hall.

Shaout, A. M., 2014. Employee Perfomance Appraisal System Using Fuzzy Logic. *Internal Journal of Computer Science and Inromfation Technology,* 6(4).

Sharma, T. & Sinha, K., 2011. Intrusion Detection Systems Technology. *International Journal of Engineering and Advanced Technology (IJEAT),* 1(2), p. 33.

Shubham, A. & Mahesh, K., 2014. Fault Tolerant and Correction System Using Triple Modular Redundancy. *International Journal of Emerging Engineering Research and Technology,* 2(2), pp. 187-191.

Silva Girão, P. & Enache, G. A., 2007. *WIRELESS SENSOR NETWORKS: STATE OF THE ART AND FUTURE TRENDS,* Madeira: s.n.

Siraj, S., Gupta, A. K. & Badgujar, R., 2012. Network Simulation Tools Survey. *International Journal of Advanced Research in Computer and Communication Engineering,* 1(4).

Sobeih, A. et al., 2005. *J-Sim: A simulation and emulation environment for wireless sensor networks,* s.l.: IEEE Wireless Communications magazine.

Soliman, H., 2012. A comparative performance evaluation of intrusion detection techniques for hierarchical wireless sensor networks. *Egyptians Informatics Journal,* Volume 13, pp. 225-237.

Somani, N. & Patel, Y., 2012. ZIGBEE: A LOW POWER WIRELESS TECHNOLOGY FOR INDUSTRIAL APPLICATIONS. *International Journal of Control Theory and Computer Modelling (IJCTCM,* 2(3).

Stankovic, J. & Wood, A., 2011. *Realistic application for wireless sensor networks,* s.l.: s.n.

Stelte, B. & Rodosek, G., 2013. *Thwarting Attacks on ZigBee – Removal of the KillerBee Stinger,* Neubiberg: Universit¨at der Bundeswehr M¨unchen.

Steven, R., 2002. *Experimental Research Methods,* s.l.: s.n.

Strikos, A., 2007. *A full approach for Intrusion Detection in Wireless Sensor Networks,* Stocklhom,Sweden: s.n.

Suphat, S., 1996. *Fundamentals of quantitative research,* s.l.: Chulalongkorn University.

Surraya, K., Usman, M., Hussain, K. & Zafar, R., 2010. *Energy-Efficient Intrusion Detection System for Wireless Sensor Network Based on MUSK Architecture,* s.l.: s.n.

Surraya, K., Usnam, M. & Alwabel, A., 2012. *Mobile Agent Based Hierarchical Intrusion Detection System in Wireless Sensor Networks,* s.l.: s.n.

Taylor, C. & Meldrum, D., 2000. *ALGORITHM DESIGN, USER INTERFACE, AND OPTIMIZATION PROCEDURE FOR A FUZZY LOGIC RAMP METERING ALGORITHM:A TRAINING MANUAL FOR FREEWAY OPERATIONS ENGINEERS,* Washington: s.n.

Taylor, S., 2007. *Internal and External Validity in Clinical Research,* s.l.: s.n.

The Regent of the University of Michigan, 2016. *Child Care and Early Education Research Connections.* [Online]
Available at: http://www.researchconnections.org/childcare/research-glossary#Q
[Accessed 28 October 2016].

Titzer, B. & Palsberg, J., 2005. *Avrora: scalable sensor network simulation with precise timing,* s.l.: IEEE press.

Tovey, G., 2002. Tutorial On computational Complexity. *Interfaces Informs,* 32(3), pp. 30-61.

Victor, B., 2005. *The Role of Controlled Experiments in Software Engineering Research,* s.l.: s.n.

Vishwakarma, D., 2012. IEEE 802.15.4 and ZigBee: A Conceptual Study. *International Journal of Advanced Research in Computer and Communication Engineering,* 1(7).

Vivekparma, 2016. *techPP.* [Online]
Available at: http://techpp.com/2010/06/30/7-most-popular-bluetooth-hacking-software-to-hack-your-mobile-phone/
[Accessed 29 JUNE 2016].

Vyavhare, A., Bhosale, V., Sawant, M. & Girkar, F., 2012. Co-operative Wireless Intrusion Detection System Using MIBs From SNMP. *International Journal of Network Security & Its Applications (IJNSA),* 4(2), pp. 147-153.

Weber & Richard, M., 2016. Internet of Things Becomes Next Big Thing. *Journal of Financial Service Professionals,* 70(6), pp. 43-46.

Wrycza, S., Marcinkowski, B. & Macœlankowski, J., 2014. Applications of Implementation Diagrams in System Infrastructure Modeling. *Zarz¹dzanie i Finanse Journal of Management and Finance,* 12(3).

Yoav, S., n.d. *An Overview of Agent Oriented Programming,* s.l.: s.n.

Yu, F., 2011. *A Survey of Wireless Sensor Network Simulation Tools,* s.l.: s.n.