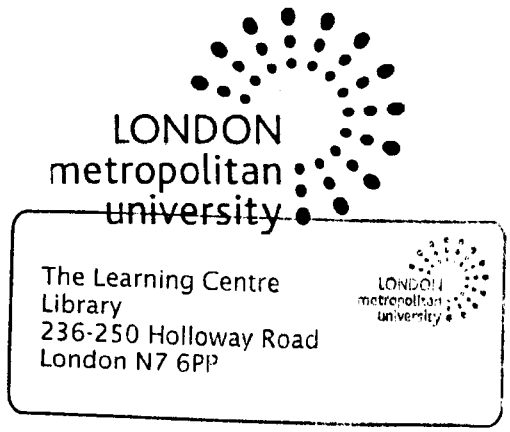


Probe Request Attack Detection in Wireless LANs using Intelligent Techniques

By

Deepthi N. Ratnayake

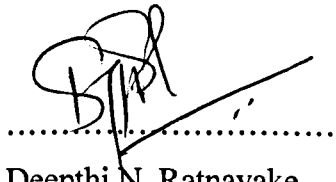
Submitted in Partial Fulfilment of the
Requirements for the Degree
Doctor of Philosophy

Intelligent Systems Research Centre
School of Computing
London Metropolitan University

December 2012

Declaration

I hereby declare that all the work presented in the thesis in partial fulfilment of the requirements for the Degree of Doctorate of Philosophy is original and my own except where otherwise specified.



Deepthi N. Ratnayake

26/04/2013

Date

Abstract

This work demonstrates a new intelligent approach to recognise probe request attacks in Wireless Local Area Networks (WLAN). In WLANs, management frames facilitate wireless stations (STA) to establish and maintain communications. In infrastructure WLANs, any mobile STA can send a probe request management frame when it needs information from an Access Point (AP). AP replies to any probe request from a STA with a known Medium Access Control (MAC) address, with a probe response management frame with capability information, and supported data rates. The next step is to establish its identity with the AP through authentication messages. Once authentication is completed, STAs can associate (register) with the AP to gain full access to the network. Probe request and response management frames are unprotected, so the information is visible to sniffers. MAC addresses can be easily spoofed to bypass AP access lists. Probe requests can be sent by anyone with a legitimate MAC address, as association to the network is not required at this stage. Attackers take advantage of these vulnerabilities and send a flood of probe request frames that can lead to a Denial-of-Service (DoS) to legitimate STAs.

The research investigates and analyses delta-time, sequence number, Signal Strength Indicator (SSI), and frame sub-type of traffic captured on a home WLAN, and uses a feedforward supervised Neural Network (NN) sensor/classifier, with four input neurons, a single hidden layer, and an output neuron, to determine the results. The research also utilises self-consistency test to measure the fitness of the data in the sensor/classifier, and 5-fold cross-validation method to evaluate the sensor/classifier with unseen data. Five Genetic Algorithms (GA) are utilised to optimise the NN using training, validation, and testing sample percentages and number of neurons of the hidden layer. The most optimised NN classifier, with training, validation and test, and sample sizes 40%, 59%, 1%, and hidden neurons 29, produced 100% accuracy on a test sample. The

computer simulation results demonstrate that the classifier classifies frames with 96.5% of overall accuracy.

The most important contribution to knowledge of this research is the introduction of a lightweight, low cost and a realistic solution to detect probe request attacks. This solution is a good candidate for single AP WLANs, to prevent being under attack by inquisitive hackers on a daily-basis. Further, this research has the following key features compared with existing work: The research works with real WLAN traffic as opposed to data from a sample database or synthetic traffic generated by a test-bed, used in many studies so that the solution takes real-world issues of frame transmission into account; This also detects an attack during an early stage of the communication, so that it can prevent any other attacks that an adversary may plan to perform; The user STAs can change their locations as opposed to some experiments with static- STAs that cannot; This solution also works effectively, when the genuine user is offline; The new feature of this approach is capturing the user and attacker-training data separately, relieving the Security Administrators from the tedious task of manually identifying rogue and genuine frames; This also enables efficient housekeeping of training data when replacing or upgrading the STAs in the WLAN.

Dedication

To my little army of nieces and nephews,
Esini, Binovin, Thithira, Ginura, Neyovin, Nevan, and Ayana

Acknowledgements

First and foremost, I would like to offer my heartfelt thanks and gratitude to Prof. Hassan Kazemian and Dr. Adnan Yusuf for being great pillars of this research process. Their kind supervision, valuable suggestions, and intellectual activities, inexhaustible energy to steer forth the research keeping my spirits up during this long journey, without which I could not have achieved what I have. I am truly blessed to have them both as my supervisors. I would also like to thank the staff of School of Computing, Research, and Postgraduate Office, Library, ICT service desk, and Student services for their support during my studies. I like to extend my heartfelt gratitude to Ms. Bo Li, Prof. Azween Abdullah, Dr. Anoosh Nabijou, Dr. Guillame Remy, Prof. Howard D'Abarera, Dr. Malika Perera, Group Captain Wajira Perera, Prof. Bandu Ranasinghe, Prof. Pramila Gupta, Dr. Ochini Madanayake, Dr. May Que, Mr. Niraj Bende and Rev. Fr. Sunil Perera for their kind research guidance and moral support. I also would like to express my sincere thanks and gratitude to fellow research students, especially, Victor Kastano, Samson Habte, Majid Djennad, Khurram Majeed, Teresa Formisano and Nazanin Kermani for making our research time together truly memorable and to my former colleagues at Sri Lanka Air Force for their rock-solid support. Last, but not least, I would like to express my love and appreciation to my dearest husband, Brian Ratnayake for being most understanding, loving, caring, patient, and supportive and to my wonderful parents, other family members and friends, who always supported me lovingly in every endeavour and made me who I am today.

Table of Contents

Figures

Tables

Acronyms and Abbreviations

ACK	Acknowledgement
ACL	Access Control List
ACM	Association of Computer Machinery
AES	Advanced Encryption Standard
AI	Artificial Intelligence
ANFIS	Adaptive Neuro-FIS
ANN	Artificial Neural Network
AP	Access Point
API	Application Programming Interface
ARC414	ARC414 belongs to family of ARC4 also known as ARCFOUR and RC4
ARP	Address Resolution Protocol
ART1	Adaptive Resonance Theory 1
ART2	Adaptive Resonance Theory 2
AS	Authentication Server
ATIM	Ad Hoc Traffic Indication Message
AusCERT	Australian Computer Emergency Response Team
BIP	Broadcast/Multicast Integrity Protocol
BSA	Basic Service Area
BSS	Basic Service Set
BSSID	Basic Service Set ID
BTF	Training Function
CBC-MAC*	Cipher Block Chaining (CBC) with Message Authentication Code (MAC*)
CCA	Clear Channel Assessment
CCMP	CTR with CBC-MAC Protocol
CF	Contention Free
CPE	Cost-Per-Example
CRC	Cyclic Redundancy Check
CRC	Cyclic Redundancy Check
CSV	Comma Separated Values
CTR	Counter Mode
CTS	Clear To Send
DA	Destination Address
DARPA	Defence Advanced Research Projects Agency
DDF	Data Division Function
DNA	Deoxyribonucleic Acid
DoS	Denial-of-Service
DS	Distributed System
DSSS	Direct Sequence Spread Spectrum
EAP	Extensible Authentication Protocol
EC	Evolutionary Computation

EFuNN	Evolving Fuzzy NN
ESS	Extended Service Set
FCS	Frame Check Sequence
FDR	False Discovery Rate
FIS	Fuzzy Inference System
FN	False Negative
FNR	False Negative Rate
FP	False Positive
FPR	False Positive Rate
FRR	Forge Resistance Relationship Method
FSM	Finite State Machine
GA	Genetic Algorithm
GTK	Group Temporal Key
GUI	Graphical User Interfaces
HIDS	Host-based IDS
HMAC-SHA1	Hash-based Message Authentication Code- Secure Hash Algorithm 1
HR	Hit and Run
IBSS	Independent BSS
ICMP	Internet Control Message Protocol
ID	Identification or Identification Data
IDCGA	Intrusion Detection Clustering GA
IDPS	Intrusion Detection And Prevention Systems
IDS	Intruder Detection System
IE	Internet Explorer
IEEE	Institute of Electrical and Electronics Engineers
IGTK	Integrated GTK
IP	Internet Protocol
ISS	Internet Security Systems
IV	Initialising Vector
JOONE	Java Object Oriented Neural-Network Environment
KDD	Knowledge Discovery and Data mining
LAN	Local Area Network
LLC	Logical Link Control
LOF	Location Outlier Factor
MAC	Medium Access Control
Mbps	Mbits/s
MIC	Message Integrity Code
MIC*	Message Integration Check
MLP	Multi Layer Perceptions
MMPDU	MAC Management Protocol Data Unit
MPDU	MAC Protocol Data Unit
MSDU	MAC Service Data Unit
MSE	Mean Squared Error
MSK	Master Session Key
MSOM	Modified Self-Organising Map
NaN	Not-a-Number
NAV	Network Allocation Vector
NIC	Network Interface Cards
NIDS	Network-based IDS
NIST	National Institute of Standards and Technology
NMAP	Network Mapping
NN	Neural Network
NPP	Negative Prediction Precision
OS	Operating System
OSI	Open Systems Interconnection
OUI	Organisationally Unique Identifier
PC	Personal Computers
PCA	Principal Compoenent Analysis

PCAP	Packet CAPture
PF	Performance Function
PHY	Physical layer
PMK	Pair-wise Master Key
PPP	Positive Prediction Precision
PSK	Pre-Shared Key
PS-Poll	Power Save Poll
PTK	Pair-wise Transient Key
QoS	Quality-of-Service
RADIUS	Remote Authentication Dialling User Service
RAM	Random Access Memory
RC4	See ARC414
RMSE	Root-Mean-Square Error
ROC	Receiver Operating Characteristics
RProp	Resilient Backpropagation
RSN	Robust Security Network
RSNA	Robust Security Network Association
RSSI	Received SSI
RTS	Request To Send
SA	Source Address
SFD	Start Frame Delimiter
SLP	Single-Layer Perception
SMART	Specific, Measurable, Attainable, Realistic and Time-bound
SN	Sequence Number
SOMs	Self-Organising Maps
SSH-HACK	Secure Shell – Hacking
SSI	Signal Strength Indicator
SSID	Service Set ID
STA	Stations
SVM	Support Vector Machine
TCP	Transmission Control Protocol
TDOA	Time Difference Of Arrivals
TDR	True Discovery Rate
TIM	Traffic Indication Map
TKIP	Temporal Key Integrity Protocol
TN	True Negative
TNR	True Negative Rate
TP	True Positive
TPR	True Positive Rate
USB	Universal Serial Bus
VM	Virtual Machines
WEP	Wired Equivalent Privacy
WIDPS	Wireless Intrusion Detection and Prevention Systems
WIDS	Wireless Intruder Detection System
WLAN	Wireless Local Area Networks
WM	Wireless Medium
WPA	Wireless Fidelity (Wi-Fi) Protected Access
WPA2	Wi-Fi Protected Access 2

Chapter 1: Introduction

Since Maxwell (1865) proposed the existence of electronic magnetic wave, many inventions followed, utilising radio waves to transfer information through space. Interest in commercial wireless networks emerged mainly in the late 1980s and 1990s. The Institute of Electrical and Electronics Engineers (IEEE) commenced discussions on standardising Wireless Local Area Network (WLAN) technologies in 1991. In 1997, the IEEE ratified the original 802.11 standard that is the foundation of the current WLAN technologies (Coleman & Westcott 2009; IEEE 1997). The current WLAN standard is IEEE 802.11-2012 (IEEE 2012). WLANs enable an information-rich environment through computers and handheld devices with portability and flexibility, increased productivity, and lower installation costs. However, the exposed airwave signals and protocol design flaws have increased risks in information security. Chapter 2 discusses this literature in detail. In spite of years of research and development, there are serious vulnerabilities in WLAN security standards and technologies based on these standards, which lead to security failures. In this thesis, the research seeks to fill a gap of knowledge in the field of WLAN security. This Chapter is organised as follows; Section 1.1 introduces the core research problem, Section 1.2 sets the research goal, Section 1.3 defines research outcomes, Section 1.4 defines the overall scope and delimitations of this research, Section 1.5 summarises the relationship of this research in existing research, Section 1.6 outlines the structure of the thesis, and Section 1.7 lists the conventions used in this thesis.

1.1. Research Problem

The IEEE 802.11 defines communication mechanisms at the Physical Layer (PHY) and Medium Access Control (MAC) sub layer of the data-link layer of the Open Systems Interconnection (OSI) model (Stallings 2009, p. 323). MAC layer communication is based on the exchange of request and response messages, i.e. each request message sent by a station (STA) must be replied with a response

message sent by the Access Point (AP). Probe request flood attacks are designed to take advantage of this request and respond design flaw (Bernaschi, Ferreri & Valcamonici 2008; Bicakci & Tavli 2009). Flooding attacks cause serious performance degradation or prevent legitimate users from accessing network resources. This vulnerability is increased due to the unprotected beacon or probe request and probe response frames which can be read by anyone to learn about the network. The research problem is further discussed in Section 2.5.

1.2. Research Goal

At the highest level, the research goal of this thesis is to fill a gap in existing knowledge by improving the security to detect flooding of probe request management frames that can lead to Denial-of-Service (DoS) attacks in IEEE 802.11 networks.

1.3. Research Outcomes

The overall goal of this research is reached through the following objectives:

a. Investigation of the conceptual framework of IEEE 802.11, its security features, and the critical weaknesses associated, to open to DoS attacks to gain background essential knowledge of IEEE 802.11 and critical weaknesses (Sections 2.2.1 and 2.2.2).

b. Investigation of academic and commercial literature to establish current state-of-the-art solutions for DoS attacks (Section 2.2.3).

c. Critically evaluate the proposed solutions to provide a context for the hypothesis (Section 2.3- Section 2.5).

d. Formulate a hypothesis to resolve the issue of probe request attacks detection in WLANs (Chapter 3).

e. Design, develop, and demonstrate the proposed solution that can detect probe request flood attacks in WLANs (Chapters 4-6).

1.4. Scope and Delimitations

The scope defines the research domain: the areas covered in the research. The characteristics that limit the scope based on conscious exclusionary and inclusionary decisions are delimitations. The broader scope of this research is to propose and validate an intelligent system to recognise probe request attacks in WLANs.

a. This research spans over several state-of-the-art technologies. The research captures data from WLANs, and utilises Neural Networks (NN) to implement and test a probe request attack sensor/classifier. After training or learning, a NN system is able to detect intrusions and deal with varying natures of attacks. NNs have a very high flexibility, hence can analyse incomplete or partial data. NNs are capable of processing non-linear data. NNs do not need too frequent updating, as the generalisation feature enables the NN to detect unknown and variants of known attacks. The research uses Genetic Algorithms (GA) to optimise the NN sensor/classifier. Computerised simulation is used to test the classifier with unseen real-world data. Performance of training, validation, testing and simulation measure, and feedback is given using computerised statistical analysis and computerised simulation.

b. Simulation software available is MATLAB 2008b (MathWorks 2012a) which runs only on Microsoft Windows XP or in higher Windows versions in compatibility mode which some features do not work.

c. The data capture is from a WLAN with single AP, handheld devices, printers and laptop computers with Windows XP OS. The capturing STA is a laptop computer with a Linux based OS - BackTrack 4 (BackTrack 2012). Attackers are laptop computers with Linux based OS - Ubuntu 9 (Ubuntu 2012),

and Windows XP OS. Optimisation and simulations are performed on Windows XP based laptops.

d. The WLAN utilised is a home WLAN. The traffic type and volume may be different to a commercial WLAN situated in a busy town area.

e. The security concepts of WLANs are obtained from IEEE 802.11-2007 (IEEE 2007) and IEEE 802.11w-2009 (IEEE 2009). There are no relevant security developments in IEEE 802.11 since IEEE 802.11w-2009. IEEE 802.11-2012 (IEEE 2012) incorporated subsequent amendments since IEEE 802.11-2007. However, the research refers to IEEE 802.11-2007 and IEEE 802.11w-2009 in this thesis, due to the obvious reason that IEEE 802.11-2012 is released this year and the research will have the tedious task of going through 2793 pages to verify information of existing references.

f. The research assumed that the Network Interface Cards (NIC) based on 'g' and 'n' standards implement the IEEE 802.11 respective standards as it is. However, manufacturers slightly change their specification to improve the performance of their NIC.

g. The research only focuses on developing a sensor/classifier module of a Wireless Intruder Detection System (WIDS).

1.5. Relationship to Previous Work

Probe request attacks belong to the family of DoS attacks. The second Chapter (Chapter 2) of this thesis systematically analyses the previous work related to the thesis. The first section focuses on understanding concepts, techniques and approaches of WLAN security. The latter part includes a large number of seminal works, such as published journal and conference papers, standards and books, and commercial literature offering insights into work for detection and prevention of DoS attacks in WLAN security, their implications, and issues.

1.6. Outline of the Thesis

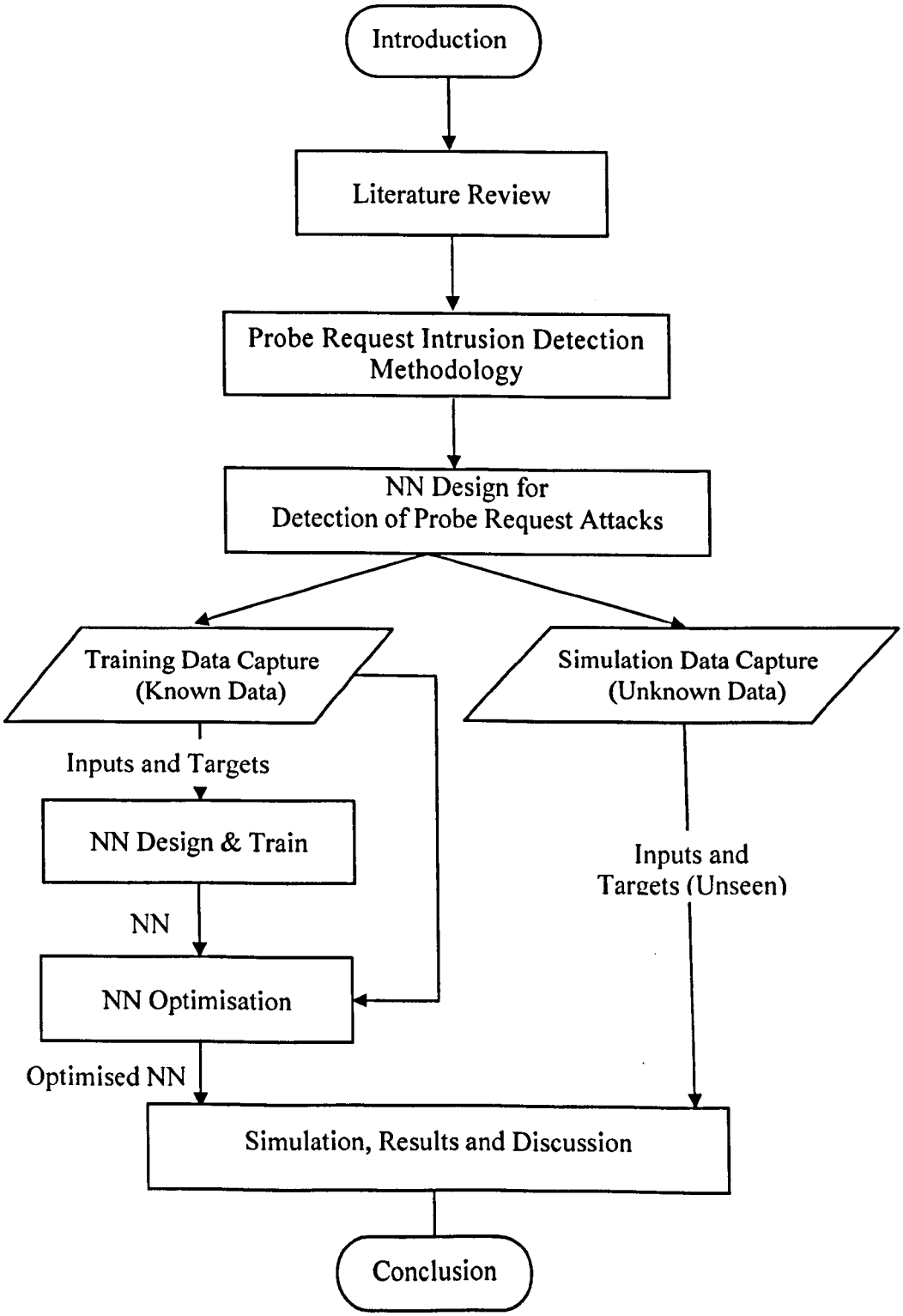


Figure 1-1: Outline of the thesis.

Chapter 2 reviews the literature and defines the research problem. The Chapter, firstly deals with the background knowledge of the architecture of IEEE 802.11 standard, its main security features and their design errors. Secondly, the research investigates commonly known DoS attacks in WLANs and existing countermeasures in general. Thirdly, the research critically analyses some of the popular flooding attack countermeasures, by looking into the methodology used, their strengths and weakness, and possible further improvements in each solution. Then, the research summarises the issues identified by above investigations and defines the research problem: the probe request intrusion detection. Chapter 3 defines the research scope, objectives and formulates research methodology: the methods or approaches and techniques that are used for conduction of research to reach objectives, within the scope specified. Chapter 4 describes the WLAN environment and data capturing process, the design, and implementation of the Artificial Neural Network (ANN) probe request attack sensor/classifier, and the systematic evaluation of performance. Chapter 5 discusses the process of applying GA methods to improve sensor/classifier's performance by optimising data and number of hidden neurons. A systematic evolution is performed executing five GAs, and the best performed NN is selected as the most optimised NN. Chapter 6 discusses simulating of 'probe request attack sensor', using fresh datasets to confirm that the trained NN sensor is functioning as expected. Chapter 7 concludes the thesis summarising the research, discussing the research contribution to the knowledge, implications and impacts of the research to the future research, research issues that limited its design and performance, and finally suggesting future directions to intended researchers who wish to take this research forward.

1.7. Conventions used in this Document

a. A slightly altered version of the scientific format with abstract, introduction, literature review, methods and materials, experiment-results-discussion, and conclusion, is followed.

b. The text is written in British English; however, organisational names are spelled verbatim.

c. Technical English is used with words such as 'dataset', 'test-bed', 'self-consistency', 'cross-validation', 'feedforward' and 'overfit'

d. To improve readability, a list of acronyms and abbreviations is used. These terms are only defined at the first time they are used in the main text.

e. Common information technology terms such as 'IEEE', 'ISO', 'hardware', 'software', 'firmware', 'operating system', 'Windows XP', 'GHz', are used without additional descriptions or references.

f. Chapters are numbered sequentially. Sections up to the 4th level include the Section numbers. Some minor headings are presented in **boldface** lettering.

g. Figures and Tables are numbered chapter wise.

h. Frames from adversaries are referred to as attacker frames, rogue frames, and intrusive frames.

i. Frames from legitimate network users are referred to as user frames and genuine frames.

j. Attack detection is also described in the thesis as identifying or recognising an attack.

k. Artificial Neural Network (ANN) and Neural Network (NN) are of the same meaning.

l. Fuzzy Inference Systems (FIS) and Fuzzy Logic are of the same meaning.

m. Functions (program code) are shown as *'newpr'*.

n. Variables are shown as *sequence no.*

o. Names or words that need emphasis are shown as 'g', 'authors'

p. The thesis refers to others as 'they', 'researchers', 'authors'. Own work is referred as 'this research', 'the research'.

q. Detection modules are known as sensors or sensor agents in the field of computing. However, they are known as classifiers in the field of NN (pattern recognition). Therefore, both terms are used for the NN model as appropriate in the text.

r. Reference list of this document is managed using Endnote Web. The referencing style used is Harvard(Qld).

Chapter 2: Literature Review

2.1. Introduction

Literature review utilises documented information on concepts, principles and trends of technologies and studies, and seeks to identify strengths and weaknesses of theory and practice so that a new design could avoid the currently existing weaknesses, whilst incorporating the strengths where possible. The research is conducted using seven task guideline proposed by Fink (2010, pp. 2-6) namely; Select a research question, select information resources, choose search terms, apply practical screening criteria, apply methodological screening criteria, review using a standardised form for abstracting data from articles, and synthesise results. The broader research question is, 'How can I improve WLAN security?' The main sources of information are academic databases such as Association of Computer Machinery (ACM) Digital Library, IEEE Xplore, and Science Direct. However, due to the wealth of knowledge of attacks available in the web, research also kept its focus on hacking web sites and blogs. The main search tool is Google Scholar. EndNote Web builds the reference list. The key search terms are 'WLAN, IEEE 802.11, DoS attacks, and intruder detection. Information is filtered based on time and current trends of research, and eventually narrowed down to MAC layer related DoS attacks in WLANs.

The Chapter is organised as follows; Section 2.2. establishes the underpinning knowledge of IEEE 802.11 and security features, their design errors, and commonly known DoS attacks effecting IEEE 802.11 networks. Section 2.3, critically analyses related work on flooding attack countermeasures, by looking into the methodology used, their strengths and weakness, and possible further improvements in each solution. Section 2.4 presents common issues identified in existing intrusion detection approaches. Section 2.5 formulates the research problem aroused from the body of knowledge developed in Sections 2.2, 2.3 and 2.4. Section 2.6 concludes the Chapter.

2.2. Background Knowledge

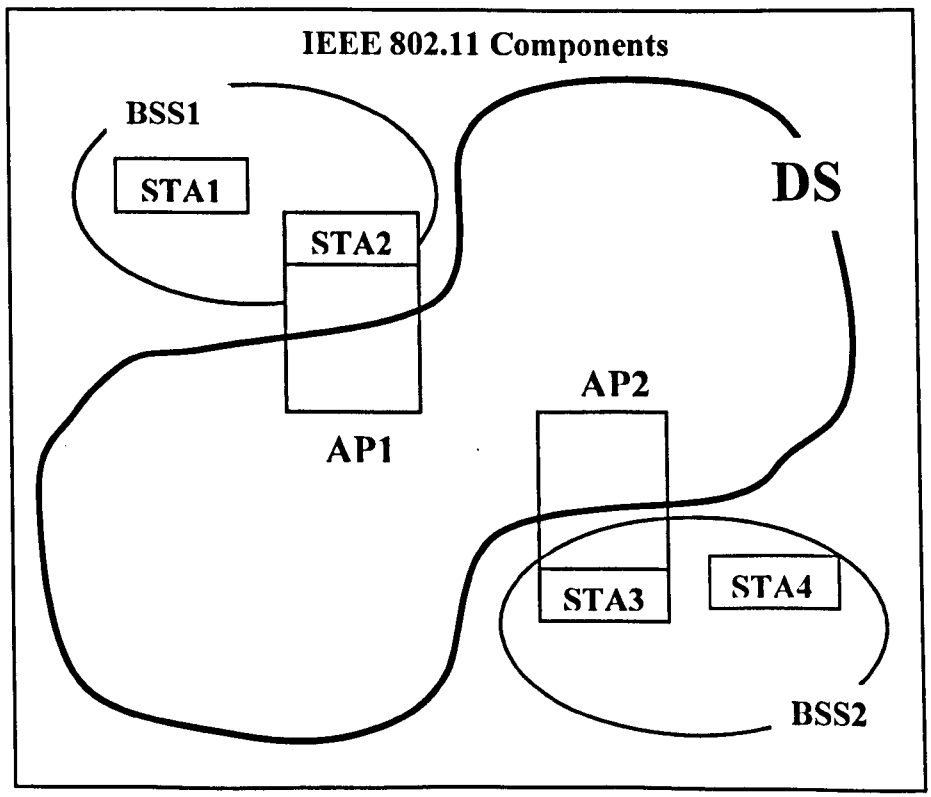
The scope of 802.11, also known as '802.11 legacy' was to develop a MAC and PHY layer specification for wireless connectivity of fixed, portable, and moving STA, which operates in the 5 GHz and 2.4 GHz public spectrum bands (Davis 2004, p. 268). In this section, the research looks into IEEE 802.11 evolution and relevant important characteristics; security features of Wired Equivalent Privacy (WEP), Wireless Fidelity (Wi-Fi) Protected Access (WPA), Wi-Fi Protected Access 2 (WPA2) and IEEE 802.11w; their design errors, and commonly known DoS attacks effecting IEEE 802.11 networks categorised based on OSI reference model.

2.2.1. IEEE 802.11 Standard

IEEE 802.11-1997 was the first wireless networking standard. This base standard has been amended by several sub-letter task groups to versions 802.11a, b, c, d, e, g, h, i, j, k, m, p, r, s, t, u, v, w, y, z. The 802.11i amendment defines a security mechanism that operates between the MAC sub layer and the Network layer. IEEE 802.11-2007 (802.11REVma) is a standard that combined cumulative changes of 802.11a,b,d,e,g,h,i,j multiple sub-letter task groups with the base standard 802.11. IEEE 802.11n, ratified in October 2009 significantly improves network throughput over previous standards. IEEE 802.11w is currently the latest security amendment available for WLAN users, which introduced protection for only authentication, de-authentication, association and de-association management frames. The cumulative amendments since IEEE 802.11-2007 are incorporated in IEEE 802.11-2012. 802.11f was withdrawn and 802.11l, o, q and x are reserved by IEEE for other projects. Other standards in the family are service amendments and extensions or corrections to previous specifications (IEEE 1997, 2004, 2007, 2009, 2012; Stallings 2009, p. 323).

In a wired Local Area Network (LAN), an address is equivalent to a physical location whereas, in a Wireless LAN, an addressable unit is referred to as a STA.

WLAN STA is a message destination, but not a fixed location. The basic building block of an IEEE 802.11 LAN is a Basic Service Set (BSS), which at least consist of only two STAs. When STAs communicate directly, a BSS is called as an Independent BSS (IBSS) or an Ad-Hoc Network. Ad-Hoc Networks are usually created without pre-planning, and can remain connected for only as long as the LAN is needed. Instead of existing independently, STAs in BSS may also form a infrastructure network via an AP. An AP is any entity that has STA functionality that enables communication for associated STAs, via the Wireless Medium (WM). The communication coverage area of the member STAs is referred as Basic Service Area (BSA). Instead of existing independently, a BSS may also form a Distributed System (DS) by connecting to multiple BSSs. The union of the BSSs connected by a DS is known as an Extended Service Set (ESS) (IEEE 2007, p. 27). Figure 2-1 shows an infrastructure network of IEEE 2007.



STA1 , STA2, STA3, STA4 – Stations
AP1, AP2 – Access Points
BSS1, BSS2 – Basic Service Sets
DS - Distributed System

Figure 2-1: Distribution system/infrastructure network/mode.

IEEE 802.11-2007, has four major operational phases: Security policy agreement phase, Extensible Authentication Protocol (EAP) or IEEE 802.1x also known as Remote Authentication Dialling User Service (RADIUS) Authentication Phase, key hierarchy and distribution (4-way and group key handshake), Robust Security Network Association (RSNA) data confidentiality and integrity. During security policy agreement phase, the AP either periodically broadcasts its IEEE 802.11i security capabilities or responds to a STA's probe request message through a probe response message. The STA then selects an AP from the list of available APs and attempts to gain authentication and association with the chosen AP. However, this authentication requires to be supplemented by further mutual authentication. EAP, 802.1x or RADIUS authentication phase does not exist when static Pre-Shared Key (PSK) is used. Otherwise, the STA and Authentication Server (AS) perform a mutual authentication procedure and a common secret key (Master Session Key - MSK) is generated between the STA and the AS. STA and AP utilise the 4-way handshake scheme to; confirm the existence of the Pair-wise Master Key (PMK), verify the selection of the cipher suite, and derive a fresh Pair-wise Transient Key (PTK). PTK is shared between STA and AP. When multicast applications require a Group Temporal Key (GTK), AP generates and distributes a fresh GTK to the STAs. During the RSNA data confidentiality and integrity phase, STA and AP construct a secret transmission channel to accomplish robust data confidentiality by using PTK or GTK (He & Mitchell 2005; IEEE 2007, 2009; Lehembre 2005; Xing et al. 2008). Figure 2-2 is a graphical representation of detailed processes within the four operational phases in IEEE 802.11-2007.

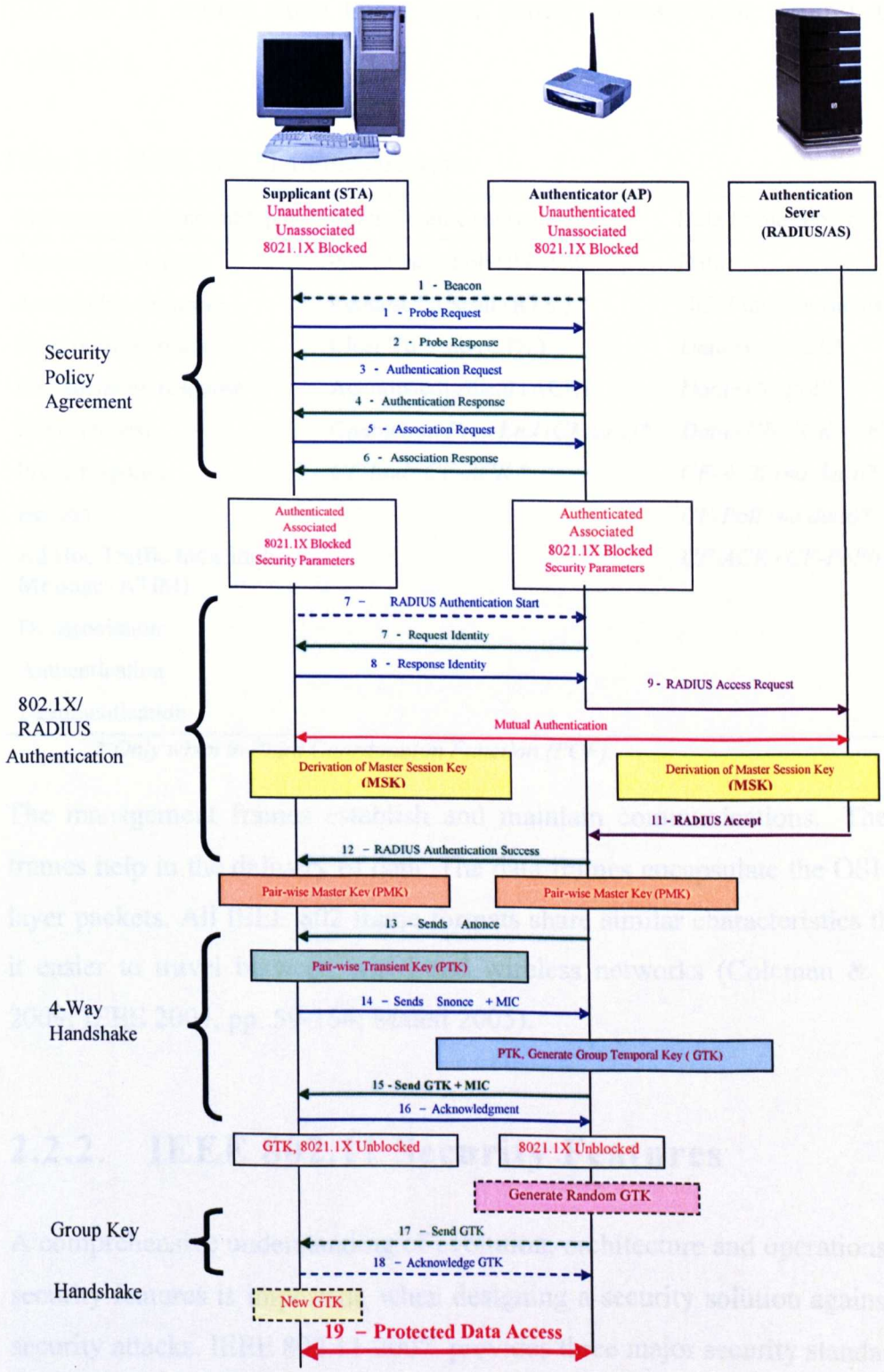


Figure 2-2: Four operational phases and processes of IEEE 802.11-2007

IEEE 802.11 defines three frame types namely management, control and data (Table 2-1).

Table 2-1: IEEE 802.11 frame subtypes.

Management frame subtypes	Control frame subtypes	Data frame subtypes
Association request	Power Save Poll (PS-Poll)	Data
Association response	Request To Send (RTS)	Null Function (no data)
Reassociation request	Clear To Send (CTS)	*Data+CF-ACK**
Reassociation response	Acknowledgement (ACK)	*Data+CF-Poll**
Probe request	*Contention Free End (CF End)**	*Data+CF-ACK+CF-Poll**
Probe response	*CF End+CF-ACK **	*CF-ACK (no data)**
Beacon		*CF-Poll (no data)**
Ad Hoc Traffic Indication Message (ATIM)		*CF-ACK+CF-Poll(no data)**
Disassociation		
Authentication		
Deauthentication		

** Only when in Point Coordination Function (PCF).*

The management frames establish and maintain communications. The control frames help in the delivery of data. The data frames encapsulate the OSI network layer packets. All IEEE 802 frame formats share similar characteristics that make it easier to travel between wired and wireless networks (Coleman & Westcott 2009; IEEE 2004, pp. 59-154; Mateti 2005).

2.2.2. IEEE 802.11 Security Features

A comprehensive understanding of evolution, architecture and operations of IEEE security features is important, when designing a security solution against WLAN security attacks. IEEE 802.11-2007 provides three major security standards based on cryptographic algorithms to protect data traffic: WEP, Temporal Key Integrity Protocol (TKIP), and CTR [Counter Mode] with CBC-MAC* [Cipher Block chaining (CBC) with message authentication code (MAC*)] Protocol (CCMP). Foundation for WEP and TKIP is ARC414 algorithm, and for CCMP, Advanced Encryption Standard (AES). This facilitates the STAs to select the algorithm(s) to

be used for a given association. ARC414 belongs to family of ARC4 also known as ARCFOUR and RC4 (IEEE 2007).

2.2.2.1. Wired Equivalent Privacy (WEP)

WEP is the initial privacy standard (also known as encryption algorithm and privacy protocol), developed in 1999, for IEEE 802.11 networks to provide a shield against casual eavesdropping between the STAs and APs. It proposed three main security goals; confidentiality, access control, and data integrity. By 2001, WEP was extensively criticised for a number of serious weaknesses such as poor key management and easily broken encryption. Since WEP is used at the data link and PHY layers of the OSI model, it does not provide end-to-end security (Arbaugh 2003; Geier 2002b; Karygiannis & Owens 2002). In 2004, Australian Computer Emergency Response Team (AusCERT), issued a report on a DoS vulnerability, which can cause significant disruption to all WLAN traffic within the range, through an exploitation of the Clear Channel Assessment (CCA) procedure, used in equipment running Direct Sequence Spread Spectrum (DSSS) at the network's PHY layer. The attack forces all the WLANs in range using DSSS to defer transmission of data. This vulnerability compromised data by either destroying it or intercepting it. This exposure eventually led the elimination of WEP (Fleishman 2010).

2.2.2.2. Wi-Fi Protected Access (WPA)

In 2001, the IEEE created a new task force, 802.11i, to bring a Robust Security Network (RSN) amendment to the existing wireless LAN standard. IEEE 802.11i is proposed to provide enhanced MAC layer security to IEEE 802.11 networks (IEEE 2004, 2009). In 2003, WPA of the Wi-Fi Alliance group (2002) superseded WEP and announced WPA as an interim standard, a subset of forthcoming IEEE 802.11i based on draft 3 of the IEEE 802.11i amendment. WPA includes an access control protocol and an encryption protocol, IEEE 802.1X and TKIP, respectively. IEEE 802.1X does not provide the actual

authentication mechanisms. However, 802.1X ties an EAP to both the wired and wireless LAN media and supports multiple authentication methods. Therefore, the use of IEEE 802.1X in WPA offers an effective framework for authenticating and controlling user traffic to a protected network. It also supports dynamically varying encryption keys, facilitating the use of session keys since cryptographic keys should change often (Geier 2002a). 802.1X authentication is mandatory in WPA. WPA uses TKIP to improve security of keys used with WEP. TKIP includes four new algorithms to enhance the security of 802.11: Cryptographic Message Integrity Code (MIC), IV (Initialising Vector) sequencing discipline, per packet key mixing function and re-keying mechanism; this effectively prevents replay attacks. TKIP improves packet integrity by adding a Message Integration Check (MIC*) field combined with the cryptographic algorithm, Michael: designed to prevent packet forgeries. MIC also addresses the critical need to change keys and provides a special countermeasure mechanism, which detects any attempt to break TKIP, and as a result blocks the communication with the attacker (Hassinen 2006; IEEE 2004). Depending on the network type, WPA can have two security authentications; WPA-Enterprise verifies network users through a central authentication server called RADIUS on the network (Geier 2002a). WPA-PSK (WPA with PSK) or WPA-Home can be used without a server. WPA-PSK uses a pass-phrase, which is created and entered by the user into any client STA's configuration utility, as well as into the AP (Shinder 2004; Wi-Fi 2003).

As in WEP, WPA's weakness too lies in the key. It is possible for an adversary to passively intercept initial key exchange messages and then use an offline dictionary attack to find the password or pass-phrase. WPA also allows DoS attacks, although it limits the risk by shutting down the network whenever an attempted attack is detected. But this can be manipulated by an adversary by sending large quantities of unauthorised data, triggering a series of shut-downs which leads to non-availability of service to users (Edney & Arbaugh 2004).

2.2.2.3. Wi-Fi Protected Access (WPA2)

IEEE formulated 802.11i in 2003, and officially ratified it in June 2004, as IEEE 802.11i-2004. A special security feature introduced with 802.11i is RSNA that provides a RSN. RSNA defines a number of security features in addition to WEP and IEEE 802.11 authentication. These features include enhanced authentication mechanisms for STAs, key management algorithms, cryptographic key establishment, an enhanced data encapsulation mechanism called CCMP, and, optionally, TKIP (IEEE 2004, p. 23). RSNA dynamically negotiates the authentication and encryption algorithms used for APs and STAs' communications (Shinder 2004). This also provides the flexibility of adding new algorithms when new threats are discovered. IEEE 802.11i standard consists of three main protocols CCMP, TKIP and 802.1X. CCMP is compulsory for RSN compliance (IEEE 2004) and also known as the security protocol built around AES to provide RSN. AES encrypts (encipher) and decrypts (decipher) information, which by all practical means is impossible to perform the transformation, without knowing the key provides the RSN. AES is also capable of working at multiple network layers simultaneously. In addition, 802.11i encrypts the whole data frame with AES. CCMP provides integrity and confidentiality. TKIP in old 802.11 equipments can upgrade with a driver or firmware to provide integrity and confidentiality. 802.1X provides authentication for TKIP and CCMP (Edney & Arbaugh 2004). The Wi-Fi Alliance group (2005) refers to their complete implementation of the approved IEEE 802.11i-2004 specification as WPA2.

However, WPA2 too is unable to protect WLANs from MAC spoofing and rogue STAs, mis-associations, honey-pots, ad-hoc connections, DoS attacks, to name but a few (Vacca 2006, p. 135). Table 2-2 shows a comparison of WEP, WPA and WPA2 security methods (Bulbul, Batmaz & Ozel 2008).

Table 2-2: Summary of WEP, WPA and WPA2 security methods.

Comparison of WEP, WPA and WPA2 Security methods			
Security Protocol	**WEP**	**WPA**	**WPA2**
Confidentiality/Integrity	WEP	TKIP	CCMP
Cipher	RC4	RC4	AES
Key Length	40 or 104 bits	128 bits encryption, 64 bits authentication	128 bits
Key Life	24 bit IV	48 bit IV	
Key Generation	Concentration	Two phase mixing function	Not needed
Data Integrity	CRC*-32	Michael	CBC-MAC
Header Integrity	CRC*-32	Michael	CBC-MAC
Replay Protection	None	Packet Number	
Key Management	None	EAP-based	
Authenticating	Open or Shared Key	IEEE 802.1x or PSK	

** CRC - Cyclic Redundancy Check*

A STA can select the algorithm(s) to be used for a given association (Coleman & Westcott 2009; IEEE 2007). However, none of these cryptographic algorithms protects management frames; beacon, probe request and response, association request and response, authentication request and response messages, and therefore the information can be read by anyone. This makes these frames vulnerable to security attacks, including Probe Request attacks (Bicakci & Tavli 2009; He & Mitchell 2005; Malekzadeh et al. 2007). The Wi-Fi Alliance announced a timetable to eliminate outdated WEP and TKIP security from certified Wi-Fi devices. WEP and TKIP will be completely phased-out by January 2014 (Fleishman 2010). However, unless it is forced to remove them or users recognise their vulnerabilities, legacy devices with WEP and TKIP will still survive in small or domestic networks for many years to come.

2.2.2.4. IEEE 802.11w-2009

IEEE 802.11w-2009 introduced mechanisms for protecting management frames of WLANs. This includes three new security features: data origin authenticity, replay detection, and data confidentiality for selected management frame protection

(Akin 2008; IEEE 2009; Liu, C & Yu 2007). The data origin authenticity mechanism enables a STA to identify which STA transmitted the MAC Protocol Data Unit (MPDU) by analysing received data or protected robust management frame. This feature is required in an RSNA to prevent one STA from masquerading as a different STA (IEEE 2009, p. 4). The replay detection mechanism enables a STA to detect whether the received frame is an unauthorised re-transmission analysing received data or protected robust management frame. This replay protection mechanism is provided for data frames for STAs that use CCMP or TKIP. The replay protection mechanism is also provided for robust management frames for STAs that use CCMP and Broadcast/Multicast Integrity Protocol (BIP). 802.11w also defines the robust management frames to be protected by the management frame protection service. Management frame protection protocols apply to robust management frames after RSNA PTK establishment for protection of unicast frames is completed, and after delivery of the Integrated GTK (IGTK) to protect group addressed frames. Robust management frame protection is implemented by the CCMP and BIP protocols and the security association query procedure. In unicast transmission, a packet is sent from a single source to a specified destination (IEEE 2009, p. 4).

The frames redefined as robust management frames by IEEE 802.11w are only disassociation, deauthentication, and robust action frames, which leave other management frames including probe request or response frames unprotected. Liu and Yu (2008) argued that 802.11w is only effective for low rate deauthentication and disassociation attacks but fails to protect against the flooding attacks because it takes significant resources to authenticate frames. Further, Wright (2006), a senior security architect at Aruba networks explain that as IEEE 802.11w does not provide protection for control frames, that the attacker can exploit WM control techniques to attack with variety of DoS attacks.

2.2.3. WLAN Denial-of-Service (DoS) Attacks

The available WLANs can be easily disrupted or penetrated using readily available hacking tools and technology knowledge freely available due to their inherent technology vulnerabilities and usability weaknesses. DoS attacks, attacks against availability to prevent legitimate users from accessing network resources, are the most commonly available security threat in WLANs (Bicakci & Tavli 2009). However, DoS attacks seem to receive the least amount of attention out of wireless attacks (Coleman & Westcott 2009). This section identifies different types of DoS attacks, their behaviours, and effects on a WLAN.

WLAN security attacks can be mainly categorised as crypto attacks and DoS attacks. Cryptographic attacks create hidden programme codes, or ciphers to challenge the security of cryptographic algorithms, and attempt to decrypt data without prior access to a key. This is a part of cryptanalysis, the art of deciphering encrypted data crypto attacks include unauthorised access, man-in-the-middle, masquerading, eavesdropping, replay, tampering and session hijack (Conrad 2007; Liu, C & Yu 2007). IEEE 802.11 standard only focuses on authentication, confidentiality, and integrity, but does not emphasise the availability as a key objective (Bicakci & Tavli 2009; Chen, Jiang & Liu 2005; He & Mitchell 2005; Liu, C & Yu 2007). This vulnerability eventually led IEEE 802.11 being subject to DoS attacks that cause serious performance degradation or prevent legitimate users from accessing network resources such as the bandwidth; APs, gateways, servers; and other STAs. A potential attacker usually tries to overflow the resources of these three elements to generate DoS attacks (Bicakci & Tavli 2009; He & Mitchell 2005; Khan, S et al. 2008; Me & Ferreri 2005). In a single node flooding attack, a single host is used, while in distributed flooding attack, multiple infected systems are used to carry out flooding busy networks already congested by its legitimate users. This can prevent access for legitimate users to broadband services. In such attacks, an advisory can be well concealed, and may use a public network for flooding to bring about the disaster. IEEE 802.11 is more vulnerable to flooding attacks due to the large-scale community-based coverage and

decentralised architecture in which the attacker can launch attacks from anywhere, at anytime (Khan, S & Loo 2009). Moreover, DoS attacks can be used as part of other types of wireless network attacks. Any application that is latency-sensitive will suffer dramatic problems through DoS attacks. File transfers, voice and video streams, thin-client sessions, and other real-time applications usually break when disrupted for more than 0.5 seconds (Akin 2008). DoS attacks can be categorised in many ways. This research selects the most common and practical method, that is to categorise based on layers of the OSI reference model. However, some attacks spread across multiple layers.

2.2.3.1. Physical Layer Based DoS Attacks

PHY layer DoS attacks are commonly known as Radio Frequency (RF) jamming attacks. The attacks prevent STAs and APs from successfully transmitting or receiving frames in the PHY layer so that frames cannot be passed on to higher layers (Bicakci & Tavli 2009). RF jamming attacks can occur intentionally or unintentionally. Unintentional RF jamming is more common and can be caused by noise from everyday household items such as microwave ovens, cordless phones, or any other appliance that operates on the 2.4 GHz or 5 GHz RF that 802.11 networks also operate on. Since it is a common band, it is not possible to stop someone from transmitting using the same frequency used by wireless networks. Although an unintentional jamming is not an attack, it can cause disruption as much as an intentional jamming attack (Cakiroglu et al. 2006; Coleman & Westcott 2009, p. 397; Compton 2008, p. 21). Some of the jamming attacks that have proven to be effective in recent researches are resource unlimited attack, preamble attack, Start Frame Delimiter (SFD) attack, reactive attack, Hit and Run (HR) attack, symbol attack and monopolising attack (Bicakci & Tavli 2009). The main disadvantage for a jamming attacker is the time, effort, and expense to build a jammer. The device also has to be placed quite close to the network for an efficient attack. A spectrum analyser can detect any type of layer 1 interference (Coleman & Westcott 2009, p. 397).

2.2.3.2. MAC Layer Based DoS attacks

IEEE 802.11 networks can be disrupted selectively or completely, using relatively few packets and lower power consumption due to its MAC protocol design flaws. During a selective MAC layer attack, an adversary can gain access to or attack a selected STA of a WLAN, so that the legitimate STA will be prevented from accessing the network resources. Some of the commonly available selective attacks are deauthentication and deassociation attacks, duration inflation attacks, attacks against 802.11i and attacks against sleeping nodes (Bicakci & Tavli 2009).

Networks with IEEE 802.11i protocol protects only data frames and does not provide any protection to management frames which can be lead to DoS attacks (He & Mitchell 2005, p. 9). Therefore, an adversary can learn the MAC address of a target STA and/or AP by listening to the traffic. From a spoofed MAC address, an adversary can transmit a deauthentication or a deassociation frame to a STA or AP to disconnect the STA or AP from the network temporarily, or permanently, if the attack continued persistently (Bellardo & Savage 2003). IEEE 802.11w-2009, introduced protection for authentication, deauthentication, association and deassociation frames. However, there are suggestions from researchers that it may not completely protect 802.11 from deauthentication and deassociation attacks. This specification does not protect beacon, probe request and response messages (Akin 2008; IEEE 2009; Liu, C & Yu 2008; Wright 2006).

Duration inflation attacks exploit vulnerability arising from the virtual carrier-sense mechanism in control frames. IEEE 802.11 frames consist of a 'duration' field, which specifies the number of microseconds that the channel is reserved. This value is used by the neighbouring nodes to update its Network Allocation Vector (NAV). NAV is a virtual carrier sensing mechanism used with wireless network protocols to prevent STAss accessing the WM and causing contention. NAV is maintained by each STA, of time periods when transmission will not be initiated. A node can transmit when NAV=0 which is used by RTS or CTS handshake to synchronise access. Though it is rarely used in practice, this

mandatory feature was introduced to 802.11 networks, to mitigate collisions from hidden terminals. An attacker can make use of this feature by asserting a large duration field in RTS or CTS or other frames, thereby preventing well-behaved STAs and APs from gaining access to the channel (Bellardo & Savage 2003).

Due to IEEE 802.11i's strict security procedures, if 802.11i protocol fails at some point, STA needs to exchange extra messages to recover, which consumes additional time and memory. An adversary can make use of this vulnerability to launch a DoS attack by periodically making 802.11i protocol to fail. An adversary can also exhaust the memory of a STA by sending a burst of forged first messages in the 802.11i 4-way handshake (He & Mitchell 2005).

The power conservation functions of IEEE 802.11 also lead to identity-based vulnerabilities. Power conservation function allows STAs in to a sleep mode, during which they are unable to transmit or receive, but occasionally can wake-up and poll the AP for any pending traffic. AP buffers inbound STA frames. When AP receives a polling message from the wakened STA, it sends the frames and clears the buffer. By spoofing STA's polling message, an adversary can cause AP to discard the STAs frames while it is asleep. It is also possible to spoof AP's Traffic Indication Map (TIM) broadcast message to deceive the STA that there is no pending data for it, so that STA will immediately revert to the sleep state. Further, key synchronisation information, such as the period of TIM packets and a timestamp broadcast by the AP, are sent unauthenticated and in the clear. An adversary can spoof these management frames and can cause a STA to fall out of synchronisation with the AP, which will lead to wake up at wrong times, and therefore lose packets (Bernaschi, Ferreri & Valcamonici 2008).

In general, attacks to APs can be classified as complete MAC layer attacks. AP attacks are known as more efficient and resource-depletion. An adversary can simply target the AP and exhaust its finite computation and/or memory resources, so that it can no longer give service to any other STA. Some of the common complete MAC layer attacks are probe request, authentication or association

request flood attacks (Bicakci & Tavli 2009). Almost all wireless NICs permit changing their MAC addresses to arbitrary values, through vendor-supplied or open-source drivers, or using an application program. Sending a burst of probe requests, having different source MAC addresses (single or many STAs) to generate a heavy workload on the AP to prevent or reduce its services to genuine STAs, is called probe request flooding (Bicakci & Tavli 2009; Guo & Chiueh 2006; Wright 2003). Probe request attacks is further discussed in Section 2.5

An adversary can launch a DoS attack on the AP by flooding forged association request frames. This will exhaust the EAP identifier space of the EAP packet, which is only 8 bits long (0-255). EAP identifier space is an optional feature, which only required being unique within a single 802.11 association. Therefore, this vulnerability can be addressed by configuring an AP to adopt a separate EAP identifier counter for each association. (He & Mitchell 2005, pp. 9-10). Further, WPA2 enabled APs are more vulnerable to flooding attacks, due to the heavy overheads by the additional security features (Bernaschi, Ferreri & Valcamonici 2008). The literature suggests that DoS attacks in MAC layer are most common and effective. Further, it is also known that major DoS attacks against WLANs compromises management frame vulnerabilities which are not protected by 802.11i or 802.11w protocols (Bicakci & Tavli 2009; He & Mitchell 2005; Malekzadeh et al. 2007).

2.2.3.3. DoS Attacks in Higher ISO Layers

A DoS attack can spread across many layers in an IEEE 802.11 network. Address Resolution Protocol (ARP) is a stateless protocol used to determine the mapping between Internet Protocol (IP) and MAC addresses. ARP has no source authentication. Hence, an adversary can poison a STA's ARP cache by sending a forged ARP reply when the STA is in the same broadcast domain (Fleck & Dimov 2001). Many of other common upper layer DoS attacks perform using the bandwidth limitations of wireless networks, such as performing Internet Control Message Protocol (ICMP) ping or Transmission Control Protocol (TCP)

synchronised flooding from a wired network to exhaust the wireless bandwidth (Bicakci & Tavli 2009). The standard countermeasures against upper layer attacks are filtering and detection. ARP poisoning attacks can be detected by extreme high levels of unsolicited ARP replies. Filtering is generally used for prevention of attacks. Attacks such as ARP poisoning and flooding can be prevented using packet filtering. A firewall between a switch and an AP can filter ARP attacks generated from a wired network. Detection is also useful to assure prevention methods are working. Further, detection is the only protection option that can be used when prevention is impossible (Bicakci & Tavli 2009; Fleck & Dimov 2001).

2.3. Related Work: Detecting and Preventing DoS Attacks

It is a common belief that DoS attacks cannot be prevented other than locating and removing the source of the attack (Coleman & Westcott 2009; Vladimirov, Gavrilenko & Mikhailovsky 2004, p. 396). However, researchers have attempted to provide detection and/or prevention methods. It is learnt that intruders actively or passively monitor the computer network to learn vital network information before an attack. Intrusion detection is the process of identifying an unauthorised user trying to gain access, or has already gained access or compromised the computer network. Intrusion prevention is the process of identifying intrusions and attempting to end them. The purpose of Intrusion Detection And Prevention Systems (IDPS) is to identify possible incidents occurring in a computer and/or network system and attempt to stop incidents and reporting as alerts or response messages (Karygiannis & Owens 2002; Scarfone & Mell 2007). Wireless Intrusion Detection Systems (WIDS) and Wireless Intrusion Detection and Prevention Systems (WIDPS) are specialised systems for wireless intrusion detection and/or prevention (Liao et al. 2012).

The research broadly categorise intruder detection and/or prevention approaches as conventional and intelligent. In general, conventional detection approaches use

statistical, rule based or state based methods on known vulnerabilities or attack signatures. Statistical methods use techniques such as statistics, Bayesian-theory, game theory to build their models and detect attacks by means of a pre-defined threshold such as mean, standard deviation, and probability. In pattern-based methods, acceptable behaviour of a network or STA is captured by a set of rules using if-then or if-then-else rules by employing techniques such as pre-defined or known rules, existing or defined models or profiles, and support vector machines. State based method makes use of Finite State Machine (FSM), which can be described as a mathematical model for a particularly simple type of computation, to detect attacks using techniques such as state-transition analysis, Markov process model and protocol analysis. These approaches consumes time, lacks flexibility to adaptation to environmental changes, and becomes out-dated (Liao et al. 2012; Lunt et al. 1989). WLAN Security researchers are now gradually moving towards Artificial Intelligence (AI) techniques. Textbooks define AI as 'the study and design of intelligent agents' where intelligent agents recognise its environment and take actions that maximise its chances of success. Simply, AI is the science and engineering of making intelligent machines that can replace human brain (Poole, Mackworth & Goebel 1998). Some of the techniques that widely utilised in IDPS are fuzzy logic, ANN, GAs and hybrid systems (Liao et al. 2012). This section reviews some of the popular countermeasures, their weakness, and possible further improvements in each solution.

2.3.1. Conventional Methods

Bicakci and Tavli (2009) present a detailed review of most popular conventional methods of detecting and preventing DoS attacks in MAC layer. Bansal et al. (2008) evaluate some commonly used non-cryptographic methods of MAC spoof detection in WLANs. They identify use of cryptography, sequence number analysis, Time Difference Of Arrivals (TDOA), re-try limits, and NIC profiling; Signal Strength Indicator (SSI) and RF finger printing for detecting and preventing probe request flooding attacks.

For many years, IEEE 802.11 security research mainly focused on data frames and the management frames were left unprotected (He & Mitchell 2005, p. 9). Therefore, an adversary could learn masses of vital information including MAC address of a target STA and/or AP of the network by listening to the traffic which enabled them creating efficient and effective attacks. A recent amendment IEEE 802.11w introduced encryption to authentication and association frames (IEEE 2009). However, this does not protect, beacon, probe request and response frames (Liu, C & Yu 2008), and therefore, information in these frames is sent clear, which is a window still kept open for attackers.

Bicakci and Tavli (2009) suggest that cryptography is the most reliable solution that can be introduced for attacks at MAC layer. They propose that the possibility of securing management and control frames by two parties sharing a long-term secret key may be worth exploring to protect beacon, probe request and response frames. However, introducing cryptography into beacon, probe request and response frames is not straight forward as keying material is not available at this stage. It also has a possibility of becoming a DoS target itself due its own rigid security restrictions. Further, as it is a protocol upgrade, this solution requires a change in the firmware of the network. A hardware upgrade is a costly solution considering the number of APs and wireless STAs that will have to upgrade.

Malekzadeh et al.(2007) propose a security improvement in management frames by a shared key. They use the HMAC-SHA1 (Hash-based Message Authentication Code- Secure Hash Algorithm 1) algorithm to protect management frames. HMAC-SHA1 is already used by IEEE 802.11i for data frame protection, therefore Malekzadeh et al. suggest that it can be implemented with a minor change in the wireless network cards. Although, the concept of utilising HMAC-SHA1 algorithm at management frames is promising, it still suffers from weakness of client-puzzles. Further, this solution requires a change in the NIC.

Another school of thought of preventing DoS attacks in MAC layer is, the use of cryptographic client puzzles, where client must solve a cryptographic puzzle

distributed by the server before gaining access to it. Client-puzzles protect the server resources from depletion as it forces the attacker to spend resources, and limits the attacker's ability to generate successful connections from a STA. Hash functions are the most popular methodology in cryptographic client puzzles (Merkle 1978). Laishun, Minglei & Yuanbo (2010) propose a client puzzle based defence mechanism to resist deauthentication and diassociation DoS attacks, by placing parameters in an unused field of the management frames. The basic vulnerability in client puzzles is the binding between a particular puzzle and the legitimate participants in the puzzle. Therefore, if not managed properly, the attacker can recruit unwitting clients to solve server puzzles (Price 2003, p. 321). Further, client-puzzle approach also has a high involvement with the user in protecting the system. This adds a very high-risk as manipulation by attackers is possible due to user's lack of technology expertise. Current approaches also require high computational power (Abraham & Vincent 2012; Aura, Nikander & Leiwo 2001; Juels & Brainard 1999, 2007; Martinovic et al. 2008). Further investigation is required in this area to construct a low cost, user friendly, but effective cryptographic puzzles.

Further investigations have been performed to prevent specific DoS attacks using non-cryptographic methods. Ferreri, Bernaschi & Valcamonici (2004) suggested that AP's main vulnerability to request flooding attacks resides in frame retransmission, which causes memory buffers exhaustion and freezing AP functionalities. They experimentally showed that decreasing the re-try limit of response frames can protect APs from the burden of having to respond with too many re-tries while on a request flooding attack. One can implement this simple, and low-cost method without any costly and sophisticated software or hardware. However, reducing re-try limit can adversely affect genuine users trying to connect to the AP when the signal power is not very good.

Another method is to define a new interpretation of the duration field used by ACK, RTS and CTS control frames that indicate how long the current transmission will keep the medium busy, to prevent legitimate STAs from

transmitting when attackers send fake frames with a high duration value (Bellardo & Savage 2003). This too can prevent legitimate STAs from transmitting when attackers send fake frames with a high duration value. Bicakci & Tavli (2009) suggested that although these statistical methods are low cost and user friendly, they are least effective primitive methods in preventing DoS attacks.

Detection of spoofed frames plays a major role in detection of DoS attacks. Monitoring sequence number field is a popular application. Sequence number is a MAC frame field. Every MAC frame from a node comes with a unique sequence number. It starts from zero and increments the number every time it sends out a non-fragmented frame and wraps at 4095. Theoretically, a NIC can generate only one set of sequence numbers at a time (IEEE 2004). Guo and Chiueh (2006) introduced an algorithm to detect MAC spoofing using sequence number gaps based on the structure and behaviour of the sequence number field. As this is an uncomplicated lightweight solution, many followed exploring this solution, considering the high overhead that may cause on a STA or AP when implementing a cryptographic solution. Some have argued that if the adversary has the control of the firmware functionality of his wireless card, then the sequence number can be manipulated by synchronising (Bicakci & Tavli 2009). However, the precision and effectiveness of this synchronisation technique is doubtful, as it is impossible to predict behaviour of a user STA, and whether the STA is starting or resetting its NIC card or transmitting data or idling. Further, the effectiveness of a detection system that is based only on the sequence number in real-time applications is questionable, as real WLAN traffic is always unpredictable due to frame delays, frame losses, and Quality-of-Service (QoS) frame re-ordering etc. Guo and Chiueh (2006) experimentally proved that even though the simulation of victim's sequence number is somewhat possible to mislead the monitor node, when the victim is inactive, the attacker would be caught as soon as the victim starts sending out frames.

Researchers also have worked to prevent attacks from forged MAC addresses by validating the sender using reliable PHY properties such as signal strength,

transmitter characteristics and frame statistics, such as delta-time value, without much success of complete detection of MAC spoofing (Faria & Cheriton 2006; Hall, Barbeau & Kranakis 2004; Ureten & Serinken 2005). Madory (2006) introduced a time difference between consecutive frames and a sliding window of received signal strengths for spoof detection. Qing and Trappe (2007) utilised a combination of window of sequence numbers and traffic inter-arrival statistics (FRR - Forge Resistance Relationship Method) to detect spoofing and anomalous traffic in wireless networks using a wireless test-bed named ORBIT. Goel and Kumar (2009) argued that these solutions work only when both the attacker and victim are transmitting and, also may be difficult to differentiate an attacker from a victim, when the victim is offline. They improved Qing and Trappe's (2007) solution by including transmission rate, and by sending a probe request to the AP, after every 9th frame it received. However, this generates 10% additional traffic overhead on the network. Goel and Kumar (2009) also used a test-bed for their experiments and claim to have 96.52 spoof detection percentage where FRR only had 92.67% detection, only 2.71 False Positive (FP) rate where FRR had 81.24%, and 3.48% False Negatives (FN) where FRR had 23.56%.

Faria and Cheriton (2006) proposed detecting identity-based attacks in wireless networks using only signal prints. However, this solution is ineffective when there is a single AP serving all STAs. Moreover, Received SSI (RSSI) measurements by themselves may not distinguish a genuine STA from an adversary if they are too close to each other (Bicakci & Tavli 2009).

Lim et al. (2003) introduced a prototype of a stand-alone WIDS, and claimed to have combined the features of AirDefence, AirMagnet and AirSnare, but was tested only against NetStumber and AirSnort attacks. This solution detects attacks only by counting probe requests received by the monitoring STA per second. When it detects a possible attack, it responds to the attacker with a DoS attack in return. This can lead to attacking a genuine user in scenarios such as; when frames from genuine user of the WLAN is detected falsely as attack frames, when both

genuine user and attacker communicating simultaneously, both will receive responding DoS attacks.

Lazarevic et al. (2003) present a comparative study of anomaly detection schemes and their variations, and developed an Intruder Detection System (IDS) using 1998, Defence Advanced Research Projects Agency (DARPA) intrusion detection evaluation datasets (MIT 2012) and real network traffic. This solution is designed using outlier detection outlier detection approaches such as: mining outliers, nearest neighbour, Mahalanobis-distance based outlier and density based local outlier approach. The solution is tested using both DARPA dataset and network traffic from University of Minnesota. The reason that they introduced a density based location outlier approach using a Location Outlier Factor (LOF) to classify frames as genuine and rogue was because they were unable to classify real traffic for labelling prior training. The Receiver Operating Characteristics (ROC) curve shows LOF approach has a higher detection rate when applied to DARPA dataset. Further, they also claim LOF achieved 89% and 100% detection rates for burst and single connection attacks respectively, when applied to DARPA dataset.

Ataide and Abdelouahab (2010) discuss a range of research architectures and open source and commercially available wireless intrusion and/or detection systems such as AirDefence , AirMagnet, Surveyor, AirSnare, Wireless-Snort , Red-M. Singh and Singh (2012) evaluate Snort-wireless, AirDefense Guard, and Kismet using a Logistic Metrics Scorecard. Potter (2004) describes the features, strengths and weaknesses of AirDefence, Internet Security Systems (ISS), Snort-Wireless, and Kismet-WIDS. He suggests that probably foremost problem for most is the deployment of sensors. Wired IDS sensors can be deployed at checkpoints within a logical infrastructure. WIDS sensors need to be installed based on physical location where sensors must be able to capture radio signals in a free air space. Potter (2004) further points out that WIDS cannot detect when an attacker is quietly (passively) sniffing data for long durations. This is quite dangerous to an enterprise, but nothing a WIDS can do about it. He proposes link-level and network level encryption. He further recommends that proper policies,

procedures, configuration, and auditing must be used in conjunction with a WIDS in order to achieve maximum security. Most of WIDSs describe above use conventional methods. The usage of intelligent methods in commercial WIDSs described in this section, if any, is unavailable to this research.

2.3.2. Intelligent Systems based Methods

Yang et. al. (2004) proposed a distributed and collaborative IDS architecture using multi-sensors that performs specific functions such as network monitoring, decision-making, response action and communication. In the design, they propose sensor agents for network packet monitoring to be installed only in some nodes to preserve total computational power and battery power of mobile hosts. Every node on the WLAN monitors activities internally by a host-monitoring agent at system and application level. Furthermore, every node has decision-making capability on the host-based intrusion threats, whilst only certain nodes are assigned to collect intrusion information and make collective decisions about network-based intrusions. Every node has an action module that is responsible for resolving intrusion situation at a host. All nodes can also exchange information about malicious behaviours on some network segments or on certain hosts and response to intrusions against them. However, this design falls short of an implementation.

Dasgupta et al. (2003) propose a multi-agent architecture using fuzzy decision support system which performs anomaly detection for ad-hoc and infra-structure networks based on sequence number patterns. The design provides a hierarchical security agent framework, which consists of a monitor agent, decision agent, and action agent. The activities of these agents are coordinated through the manager agent. The fuzzy logics enable the use of imprecise and heuristic knowledge to decide the states, as normal, known attacks and unknown attacks. The data used are captured from real ad-hoc and infrastructure networks at normal mode (operations performed under normal mode are not given). Under attack modes are SSH-HACK (Secure Shell – Hacking) and NMAP (Network Mapping) for

password-guessing and scan open ports respectively. Dasgupta et al. (2003) overcome the problem of generating fuzzy rules manually by employing a parameter–free version of GA for generating fuzzy rules. A ten-fold offline testing strategy is employed. The authors illustrate a ROC curve, which shows false alarm rate at 1.0 and above, the detection rate, almost 100%. They present the real-monitoring graphs to show how the confidence level exceeds the predefined threshold (0.5) as the attack progresses during a real-time attack. The authors claim that their long-term goal is to develop a self-adaptive system that will perform real-time monitoring, analysis, detection, and generation of appropriate responses to intrusive activities. The approach is promising, however, it is this research's understanding that the sequence number patterns itself is not a reliable source for anomaly detection as frames can be lost or arrived out-of-order due to other environmental and technical causes. This can further improved by using a set of carefully chosen attributes or using search heuristic such as GA to find most suitable attributes based on the dataset.

Some use a combination of FIS and Artificial Neural Networks (ANN) also commonly known as Neural Networks (NN). This combination is known as Neuro-Fuzzy systems. Pleskonjic (2003; 2007) presents a corporative wireless detection and prevention architecture with power of auto-learning and auto answering to intrusion attacks incorporating NN and Fuzzy logics. FannExplorer is used to develop the two-layer adaptive self-learning NN with 45 inputs and 15 hidden neurons. Batch algorithm, as opposed to incremental learning algorithms, and Sigmoid Symmetric Stepwise activation function is used in the design. Further, the official SNORT rule set with 3454 frames (SNORT 2010) is used for learning and reports a 100% detection rate. This multi-dimensional system claims to be able to defend against new intrusion types, locally and globally, in real-time. However, this system is developed and tested using a sample data base (SNORT 2010). Further, self-learning, 45 inputs, Sigmoid function and co-ordinated processing of data require a considerable amount of computing resources. Authors also intend to test the solution with KDD (Knowledge Discovery and Data mining) cup '99 dataset (KDDCup 1999). However, the research believe, this

solution need to be tested with real-world environment in a real-world data to find its efficiency and effectiveness.

Ataide and Abdelouahab (2010) suggest that ANN-Multi Layer Perceptions (MLP) are currently the most popular method of detecting network attacks. They propose an architecture for IDPS using ANNs. This uses anomalies for analysis and claims to be effective in any occurrence of attacks against the integrity of the system. The proposed architecture is organised into modules: sensor, data, detection, countermeasure, management, and database. A sensor module is designed to capture WLAN traffic and pre-form before sending it to WIDS server. Data is grouped by interval of two seconds, and by the emitting source, to enable detection of change in the behaviour immediately. Detection module is designed using ANN with MLP. NN consists of 21 inputs, 2 hidden layers with 7, 5 neurons respectively, and an output. Training of NN utilises the second order method of Levenberg-Marquardt algorithm. The countermeasure module passes active or passive actions onto actuator module if there is detection. Ataide and Abdelouahab (2010) claim the countermeasure module also is designed using an ANN, but the details of this NN is not given. Management module manages the inter-connected WIDS modules and provides management information to its users. Database module is the repository of information registered by the sensors. The solution is tested with real-time data collected during an hour, with three types of attacks – virtual carrier sense, association flood, and de-authentication. The detection strategy is to observe user's abnormal behavioural changes (anomalies) in the complete network. However, the classifier/sensor is trained only with 14000 registries and tested with 5600 registries. This may not be adequate to develop a high-quality classifier. The authors claim that the maximum error rate is 9.8902%, largest rate of FP is 5.0963%, and average error rate is 0.093%. Further they claim that results of their work is compatible with results of LAN attacks experiment conducted by Moradi and Zulkernine (2004), and improved 5% than the WLAN attacks experiment of Liu et al. (2006). This solution is tested on virtual carrier sense, association flood and de-authentication attacks. IEEE (2009) claims that , IEEE 802.11 design error manipulated by

attackers to create disassociation and deauthetication attacks, is addressed by IEEE 802.11w-2009 standard upgrade, by cryptographically protecting the association request and response, and authentication request and response frames. Therefore, the new NIC cards may not have further vulnerability to association flood and de-authentication attacks. Ataide and Abdelouahab (2010) claim that this solution can detect any unknown attack as the solution is simply based on behavioural changes of the users. However, the research suggests that it can also work in the other way due to the low data sample that has used, i.e. a sudden behavioural change in a legitimate client also can be diagnosed as an attack. Authors also do not explain whether they have taken WLAN factors that generate typical false alarms, such as mobility of the users, user traffic fluctuations in the network, into account. They monitor 21 attributes (details not given) of each message in the NN. Monitoring behavioural changes of the total network for 21 inputs will be quite challenging in a real network, which has a larger number of users. Therefore, real-world implementation of this solution can be very costly and impractical due to the computing power required to process all frames and perform all tasks of its modules. It is also noted, that normal and attack data is collected separately for training and testing, but authors do not explain how it is used for training and testing, as in real-world systems, a data sample consists of both attack and genuine frames.

Ahmad, Abdullah & Alghamdi (2009b) suggest that probing attacks are the basis of other attacks in computer networks. They proposed a cyber probing attack detection system that utilise MLP architecture and resilient backpropagation for its training and testing. This work is an extended version of their work in Ahmad, Abdullah & Alghamdi (2009a). The authors use the complete set of TCP/IP fields or characteristics of the KDD cup '99 database (KDDCup 1999). The NN consists of an input layer with 41 neurons and, 2 hidden layers with 14 and 9 neurons respectively, and an output layer with 2 neurons that classify normal packets from abnormal packets. Selection of the number of layers and its neurons is decided based on Root-Mean-Square Error (RMSE) values of five test cases that trained. However, the research believes that a rigorous method like GA for

selection of number of layers and respective neurons could have provided a more optimised combination. They claim that they used Resilient Backpropagation (RProp) for training due to its speed of convergence (MathWorks 2012f). The design is implemented using a open source software, Java Object Oriented Neural-Network Environment (JOONE) (Sourceforge 2012) and Java runtime environment (Java 2012). The authors claim that the system is tested on seen data as well as unseen data, and when the prototype is applied to IPsweep, Nmap, Portsweep and Satan probing attacks in KDD cup '99 dataset (KDDCup 1999), it reported 98% of detection accuracy. The results are compared with Modified Self-Organising Map (MSOM), Adaptive Resonance Theory 1 (ART1), and Adaptive Resonance Theory 2 (ART2), and authors demonstrate that RProp is more precise and accurate, compared to other approaches. As described above, this solution is to detect cyber-probing attacks and the solution is developed and tested only with frames from a sample database. However, their comparison of training methods makes it interesting to explore with real-time WLAN messages as opposed to offline sample messages that they have used in this experiment. Further, some studies claim that Levenberg-Marquardt algorithm is the most efficient than any other and is widely used in the field of NN, including MATLAB NN toolbox. However, Levenberg-Marquardt algorithm can be very slow to converge correctly on an error minimum. (Hagan & Menhaj 1994; Mathworks 2012; MathWorks 2012h; Wilamowski, Cotton & Hewlett 2007)

Adaptive Neuro-FIS (ANFIS) is another intelligent method that is currently in demand for application of detecting intrusions. Chavan et al.(2004) presented an IDS, using ANNs and FISs. The proposed framework includes a host, internet, and a network-based IDS with an input subsystem that consists of SNORT packet logger (SNORT 2010) and attack signature database created from DARPA intrusion detection attacks database (MIT 2012), a processing subsystem that utilised a combination of misuse and anomaly based detection techniques and NNs. The output subsystem includes reporting mechanisms. Authors claim that this system can learn new types of attacks continuously adding to the previously trained knowledge and new data could be updated in the knowledge base using

simple if-then fuzzy rules. The design, Evolving Fuzzy NN (EFuNN), utilises a Mamdani's fuzzy inference method (Mamdani 1976). The EFuNN uses four membership functions for input variables and evolving parameters (sensitivity and error) for detection of attacks. 89, 115, 123, 134, and 129 rule nodes are generated for Normal, DOS, U2L, U2R, and probes respectively. The decision tree approach is implemented to reduce the number of variables from 41, which is the total number of variables, to 13, 14, 15, 17, and 16 respectively for Normal, DOS, U2L, U2R and probes. Further, the authors compare the training time of EFuNN with an ANN that uses respective input variables, 80 hidden neurons and 1 output neuron, and claim that the EFuNN took few seconds where as an ANN took few minutes to converge. Authors also claim that the system could detect DoS attacks and probing attacks with an accuracy of 98.99% and 99.88% respectively. Reduction of inputs using a decision tree is a positive approach in this experiment towards real world implementation. However, the decision tree only considers features or behavioural patterns of the given frames, which may not represent the full IEEE 802.11 specification, unless the training data is not carefully selected. Therefore, this might lead to not feeding the most decisive inputs to the NN. Furthermore, the authors suggest that EFuNN uses a hybrid mixture of unsupervised and supervised learning techniques with only a single epoch to enable online easy and fast updates. In theory, if a NN is trained to 1000 epochs, the learning algorithm moves through 1000 cycles. Number of epochs are also used as a condition for NN early-stopping to improve generalisation, which response well to unseen data (Carney & Cunningham 1998). However, application of only one epoch limitation is a unique feature that is worth exploring further to analyse its effects on learning and generalisation.

Toosi and Kahani (2007) use a two layer architecture for their IDS. In one layer, they utilise a fuzzy inference module with Mamdani Fuzzy Model, to detect if the input data are normal or intrusive. The other layer consists of 5 ANFIS modules to represent each class of KDD Cup '99 dataset (KDDCup 1999), namely Normal, DOS, U2L, U2R and Probes. They claim that the reason behind their choice is that ANFIS is more appropriate as a binary classifier than a multi-classifier. This

layer classifies the traffic frame into one of five classes. Toosi and Kahani (2007) use all features (41) of KDD Cup '99 dataset (KDDCup 1999), as inputs for their neuro-fuzzy classifiers. This again gives unnecessary overhead on the monitoring STA , as the system has to capture large numbers of frames and process its all features (41 inputs) to detect an attack. Toosi and Kahani (2007) also employ GA approach to optimise the input membership functions of the fuzzy decision making module. The fitness value used is Cost-Per-Example (CPE) calculated based on confusion matrix values and cost matrix values (Sabhnani & Serpen 2003), which is 0.1579. This research combines FIS, ANFIS and GA to develop the solution, which is a promising effort. However, it is trained and tested on various samples of KDD Cup '99 dataset (KDDCup 1999), which they claim to have 5 million records.

Patcha & Park (2007) present an overview of existing solutions and latest technological trends of anomaly detection techniques. They categorise anomaly detection systems into statistical, machine learning and data mining systems and compare them based on the features of the solution and detection methodology applied. They point out that a major challenge in today's IDS is to define what is 'normal' and to find practical solutions.

Liao et al. (2012) presents comprehensive survey of IDSs, and compare them; based on detection approach (statisical, pattern, rule, state and heuristic based), detection methodology (anomaly, signature, protocol analysis), time series (on-line real-time, offline or non-real-time), technology type (host, network, protocol based), detection of attacks (known, unknown, both known and unknown), source of data (audits-trails, application-logs, network packets or connections, events), and charateristics of data (real-time, active, automatically generated, sample etc.). They raise the issue of 'running computational power' required and the coverage of attacks that most of the solutions provide. Liao et al. (2012) also discuss issues of employing Virtual Machines (VM) to develop and test IDSs. This needs a broad discussion, however, the research can agree on the fact that VM can become unstable due to the effect of the DoS attacks.

Bankovic et al. (2007) present a list of machine learning techniques deployed for intrusion detection and their performance on KDD Cup '99 dataset (KDDCup 1999). The most promising results 97.47%, 99% and 99.56% are generated by techniques such as GA, Support Vector Machine (SVM) and GA, and SVM Fuzzy Logic, respectively.

Zhao, Zhao & Li (2005) argue that an IDS which derives rules directly from audit data, may not detect a slightly different attack or cause a fales alarm. They address this issue proposing Intrusion Detection Clustering GA (IDCGA). IDCGA, utlises an algorithm that uses a clustering step to cluster the cases automatically by similarities, and a optimisaton step to optimise the clustering sets to distinguish the normal and intruded action. Zhao, Zhao & Li (2005) used all (41) attributes of KDD Cup '99 dataset (KDDCup 1999), and claim 95% overall accuracy after testing on 5 datasets.

Bankovic et al. (2007) propose a GA based misuse detection system, which is trained and tested using KDD Cup '99 dataset (KDDCup 1999). They implement Principal Compoenent Analysis (PCA) in MATLAB to extract most important features of the data and claim that PCA expedited the data processing and enabled to high level detection rates. PCA select following features; *duration*, *src_bytes*, and *dst_host_srv_serror_rate*. GA, written in C language, generates rules for intrusion detection. The fitness function is based on True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN) values, that is, $(TP/(TP + FN)) - (FP/(TN + TP))$. The scale of fitness value is [-1, 1]. Further, they employ a support confidence framewrok to analyse the type of attack. Detecting the type of intrusion is not crucial for intrustion detection. However, it is important for forensics. Parameters of the GA are; 1000 generations, 500 initial rules, single-point crossover with probabilty of 0.6 and 0.7 in 2 experiments. Mutation rate 0.01 and 0.05 in 2 experiments. GA generated 10 best-fit rules for classification of the intrusion detection. Bankovic et al. (2007) claim that the classifer perform with 98.38-100% acuracy with normal

connections. 92.74%-94.87% with attacks. It also can identify portsweep, smurf and neptune attacks 87.6% accurately. Bankovic et al. (2007) compare results with Gong, Zulkernine & Abolmaesumi (2005) and Lu & Traore (2004) based on the number of features they are using: 7 and 41, and claim that their approach is faster.

2.4. Issues in Intrusion Detection Approaches

The above sections presented the detailed survey on IEEE 802.11 standard and common attacks against WLANs, and analysis of the features, strengths, weaknesses and the results of previous recent research work by individuals and research groups, in relation to other existing methods and standards. In this section, the research summarise the common issues or gaps in the literature that one can fill, in future research.

- Soft computing techniques such as ANN, FIS, and Evolutionary Computation (EC) play a major role in current research due to their capability to overcome many integral weaknesses in conventional IDSs such as adaptability issues, which require frequent updates, high computation power and time. However, soft computing techniques too, suffer from their inherited design weaknesses such as requirement of training data, pre-processing of data, time and computing resources required for training or learning, validation, testing, optimisation and simulation of models.

- Many techniques have been utilised to detect intrusions in both wired networks and wireless networks. However, current issues of IDS such as requirement regular updating, low detection rate, high false positive and negative rates, suitability and adaptability to the application, remains the same. NN is the most popular method among the literature that this researcher came across when searching for WIDS literature. Ahmad, Abdullah & Alghamdi (2010) suggest that ANNs can be considered as the most successful approach for IDSs considering its easiness to implement, and lower computational power required. They further

suggest that NN approach is most appropriate to tackle the current issues of IDSs discussed above. Many authors claim 99-100% accuracy in their results, however, as they stand, these solutions still fail to provide a complete protection in real-world networks.

- Two major categories of ANN are unsupervised and supervised NNs. It is also learnt that unsupervised NNs such as Self-Organising Maps (SOMs) and ARTs are more efficient in flexibility and adaptivity. However, these show poor performance in terms of detection rate, FPs and FNs. Therefore, the literature suggests that when data are available and the research can identify attack and genuine data separately, it is advisable to use supervised NN (Amini, Jalili & Shahriari 2006; Min & Dongliang 2009).

- It is a popular approach for researchers to use existing sample datasets such as KDD Cup '99 dataset (KDDCup 1999), and SNORT (SNORT 2010) or other sanitised or simulated traffic to develop and test intrusion detection proposals. The most common reason for choosing publicly available datasets are; it is rich in variety of genuine and attack traffic, and considers as a benchmark for evaluating security detection mechanisms. 'KDD Cup' is an annual Data Mining and Knowledge Discovery competition organised by ACM special interest Group on Knowledge Discovery and Data mining (KDD). The KDD Cup '99 dataset is created by processing the tcpdump portions of the 1998 DARPA IDS evaluation dataset, created by Lincoln Lab (KDDCup 1999; Lippmann et al. 2000; MIT 2012). DARPA (MIT 2012) normal dataset is a simulated synthetic data, and attack data is generated through scripts and programs. These datasets therefore, do not contain background noise that a real-word dataset consists of (Lazarevic et al. 2003; Lippmann et al. 2000; McHugh 2000). SNORT database (SNORT 2010) on the other hand has not been updated since year 2005. Therefore, the solutions that are developed based on these datasets may not work as efficiently and effectively as they claim to be in real-world environments. This problem also applies to solutions that are based on other sample databases and, synthetic or sanitised traffic.

- Apart from the issues discussed above, traffic generated from test-beds also have the issue of limited environmental conditions as data will be collected from a controlled network by simulating traffic.

- Some solutions identify intrusive behaviours based on the exploration of known vulnerabilities.

- Further, it is observed that some of these solutions are extremely complex and are simulated and tested without considering the practical implementation and computing power they may require. Therefore, these solutions may be limited for academic research world as implementation is too complex or expensive. This issue has also been identified by Liao et al. (2012).

- Every research highlights the importance of real-world network intrusion problems. However, less consideration is given to the most crucial issue ofreal-world data collection and real-word application. Training, validating and testing on real data and simulation on real data is very important in the process of bringing the research into real-world applications. Collection and use of real-world traffic also make the researcher understand the real-world issues that a Security Administrator may encounter whilst implementing a proposed application. However, real-world data collection can lead to biased data being used for training and testing, as there is no standard approach or guidelines for collecting and using traffic of a real network. Furthermore, as the dataset is unique to each experiment, results cannot compare with other research, unless one implements other methods on the same dataset to compare two methods.

- Furthermore, the existing studies do not explain how they recognised genuine and attack frames within the training traffic when real-world data is collected. The research assume that they may have collected attack and normal frames separately, or collect traffic whilst an attacker is available, and analysed manually to label them based on other features.

- Further, most of IDSs propose universal solutions to intrusions. This research agrees with Liao et al. (2012) who suggest that the exisitng IDSs pose challenges to the categories that they claim they belong to and huge computational power they require.

- Many of the existing approaches of intrusion detection have focused on the issues of feature extraction. Selecting input features based on the highest eigenvalue from a limited set of data may lead to loosing many important and sensitive features, which can affect the efficiency and effectiveness of the classifier.

- Almost no research evaluates the results of a detection model's performance to different types of scenarios e.g. when user and attackers(s) at different distances from AP, when only the attacker is present, when there is no attacker, when there are more than one attacker, when there are attackers with similar and different NIC types, and so on. This lack of information can confuse or mislead readers or future researchers, as; although their proposals are excellent in technical and practical aspects, they may not reach the outstanding results that other researches may have published using non-challenging traffic.

- WLAN intruder detection is only possible when an attacker is in active mode. Passive attackers in monitoring mode cannot be detected (Potter 2004).

2.5. Research Problem: Probe Request DoS Attacks

This section defines the research problem that this investigation is intending to solve, with reference to comprehensive analysis performed earlier in this Chapter. It has previously been established that the WLAN intruder detection is only possible when an attacker is in active mode. It has also previously been established that passive attackers in WLANs could not be detected, though, could be prevented by encrypting all frames. Based on the knowledge gained through

the literature review, the research understands that the root cause of many problems in WLANs is the non-availability of a unique identity to a STA. A MAC address, which supposed to be unique, can be easily spoofed using commonly available software. Therefore, AP control lists can be easily by-passed by MAC address spoofing. Probing is the very first active communication that a new adversary would perform. Moreover, it is also the easiest attack anyone can perform without sophisticated equipments or passwords, as probing do not require association to the network. Therefore, this researcher believe that if a WLAN can recognise unauthorised probing or probing floods, it can prevent many future attacks by taking preventive measures. To further understand the specification and behaviour of management frames the research analysed security policy agreement phase, probe request and response frames, and statistical and radio information that can be contained in the capture.

2.5.1. Security Policy Agreement Phase

Figure 2-3 shows the security policy agreement phase of IEEE 802.11-2007. A STA seeking to connect to a WLAN has a choice of either a passive scan or an active scan. In a passive scan, STA listens to successive channels, waiting to hear a beacon frame from an AP, while in an active scan, the STA broadcasts a probe request message on channels that its PHY layer supports, until the STA finds an AP. These frames are unprotected and information passed between the frames can be read using freely available software (He & Mitchell 2005; IEEE 2007).

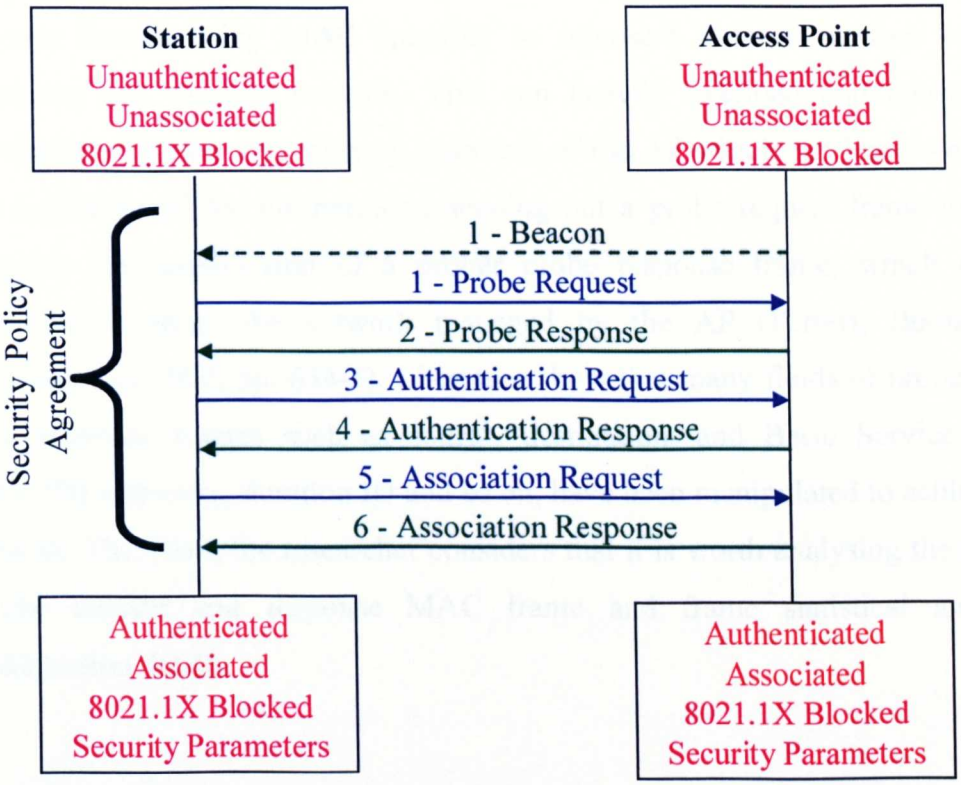


Figure 2-3: Security policy agreement phase of IEEE 802.11-2007.

To enforce reliability, 802.11 compliant STAs expect all transmitted frames to be acknowledged with an acknowledgement frame by the receiving side. If no acknowledgement is received, the transmitting STA performs a number of re-transmissions of the same frame until, the receiver via an acknowledgement frame eventually acknowledges the frame or the number of re-transmissions reaches the retry limit and then the frame is discarded. When the AP responds to frames sent by an attacker STA, it receives no acknowledgement frames back; therefore, it starts a re-transmission cycle for every single frame received. AP uses the buffer space to store repeating response frames waiting to be transmitted. Hence, a simple flood of requests can cause the exhaustion of all internal buffers of the AP, leaving no resources for the management of legitimate communications, thereby resulting, a complete DoS (Ahmad, Abdullah & Alghamdi 2009b; Bicakci & Tavli 2009; Me & Ferreri 2005, p. 5).

The basic theory behind probe request flooding attacks is sending a flood of probe request frames using MAC spoofing to represent a large number of nodes scanning the wireless network. This can heavily overload and consume the computation power and memory resources of the AP which can lead to a DoS to its legitimate STAs. Furthermore, sending out a probe request frame to an AP triggers the transmission of a proper probe response frame, which contains information about the network managed by the AP (Ferreri, Bernaschi & Valcamonici 2004, pp. 634-5). Literature show that many fields of probe request and response frames such as source, destination, and Basic Service Set ID (BSSID) addresses, duration ID and so on, have been manipulated to achieve DoS attacks. Therefore, the researcher considers that it is worth analysing the fields of probe request and response MAC frame and frame statistical and radio information details.

2.5.2. Probe Request Management Frame

As discussed in Section 2.2 above, IEEE 802.11 defines three frame types namely, management, control and data, and each frame type has several frame sub-types (Table 2-1). The MAC frames in the MAC sub-layer are described as a sequence of fields in specific order. Basic components of a MAC frame are header, body, and Frame Check Sequence (FCS). A MAC header comprises of frame control, duration, address, and sequence control information, and, for QoS data frames, QoS control information. MAC frame body contains information specific to the frame type and subtype, and therefore the length varies. FCS contains an IEEE 32-bit Cyclic Redundancy Check (CRC) (IEEE 2007, p. 59). Figure 2-4 depicts the general MAC frame format. The first three fields (Frame Control, Duration ID, and Address 1) and the last field (FCS) constitute the minimal frame format and are present in all frames, including reserved types and subtypes. The fields Address 2, Address 3, Sequence Control, Address 4, QoS Control, and Frame Body are present only in certain frame types and subtypes. The Frame Body field is of variable size. The maximum Frame Body size is determined by the

maximum MAC Service Data Unit (MSDU) size (2304 octets) and any overhead from security encapsulation (IEEE 2007, p. 60).

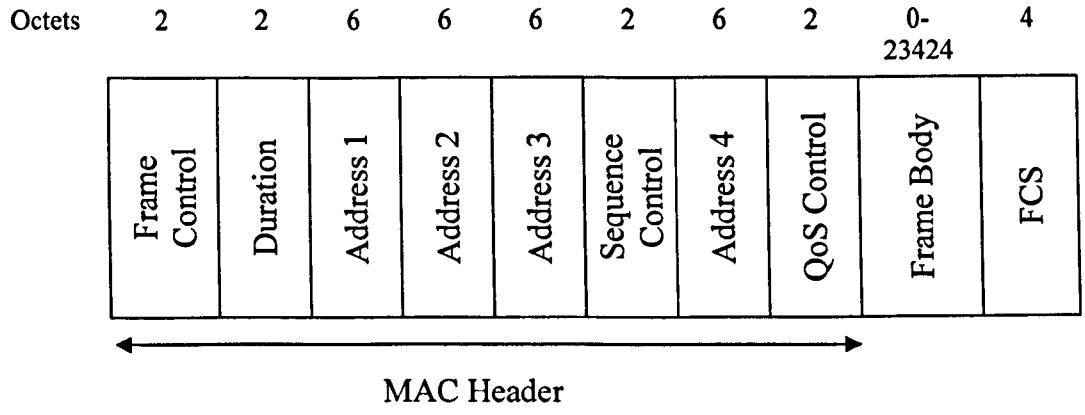


Figure 2-4: MAC frame format.

The frame format of a management frame is independent of frame subtype (IEEE 2007, p. 79). Therefore, the research analysed a management frame to look into construct and behaviour of probe request management frames. A MAC header of a management frame is shown in Figure 2-5.

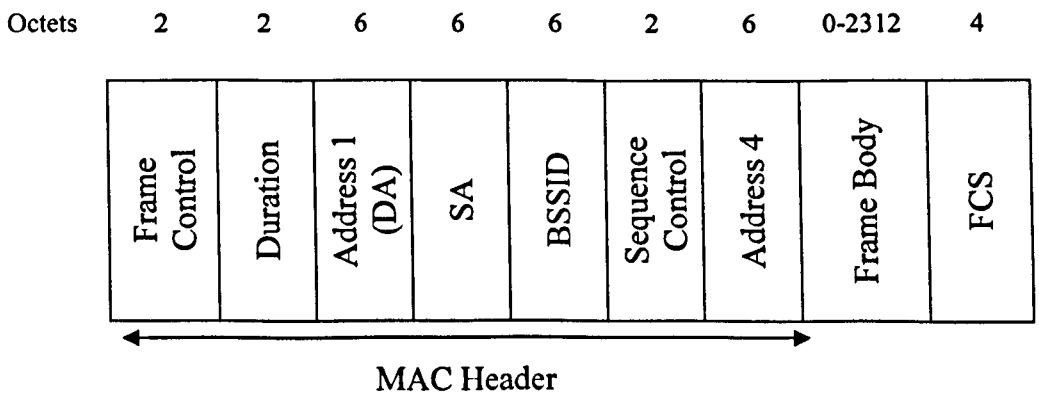


Figure 2-5: Management frame format.

The frame control of MAC header consists of Protocol Version, Type, Subtype, To DS, From DS, More Fragments, Retry, Power Management, More Data, Protected Frame, and Order fields (Figure 2-6).

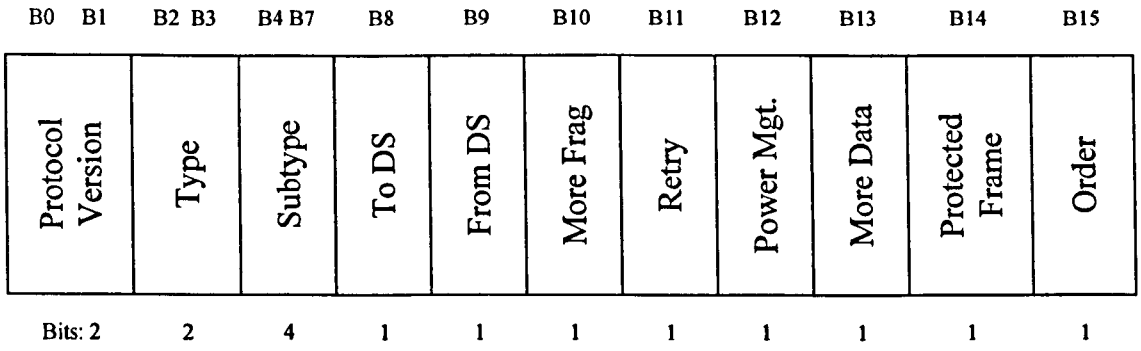


Figure 2-6: Frame control field of a management frame.

Protocol_Version field is 2 bits in size, b0 and b1. For IEEE 802.11 standard, the value of the protocol version is 0, therefore its binary value is'00'. The revision level will be incremented only when a fundamental incompatibility exists between a new revision and the prior edition of the standard. A MAC entity that receives a frame with a higher revision level than it supports will discard the frame without indication to the sending STA or to Logical Link Control (LLC). **Frame type** field is 2 bits in size, b2 and b3. Management frame type is always represented as '00' in binary. **Subtype** field is 4 bits in size, b7 b6 b5 b4. The Subtype value for probe request frame is always '0100'. The Subtype value for probe response frame is always '0101' and b6=1 for both frames which means that the frame Subtype has no Frame Body Field. **To_DS** field is 1 bit in size, b8. It indicates whether the data frame is headed for a DS. DS is used to interconnect a set of BSS and integrated LANs to create an ESS. To_DS value for a management frame is always '0'. **From_DS** field of a MAC frame is 1 bit in size, b9. From_DS value for a management frame is always '0'. **More_Fragments** field is 1 bit in size, b10. Values can be '1' or '0'. Data or management frames that have another fragment of the current MSDU or current MAC Management Protocol Data Unit (MMPDU) to follow, will take the value '1'. **Retry** field is 1 bit in size, b11. It is set to '1' in any data or management type, when there is a re-transmission of an earlier frame. It is set to '0' in all other frames. A receiving STA uses this indication to aid in the process of eliminating duplicate frames. **Power_Management** field is 1 bit in size, b12. The value indicates the mode in which the STA will be after the successful completion of the frame exchange. A value of '1' indicates that the STA will be in 'power save' mode. A value of '0'

indicates that the STA will be in 'active mode'. This field is always set to '0' in frames transmitted by an AP. **More_Data** field is 1 bit in size, b13. The value '1' indicates the STA in PS mode, that is, at least one additional buffered MSDUs or MMPDUs are buffered for that STA at the AP. The More_Data field is valid in directed data or management type frames transmitted by an AP to a STA in PS mode. **Protected_Frame** field is 1 bit in size, b14. This is set to '1' if the frame body contains information that has been processed by a cryptographic encapsulation algorithm. However, as there is no protection, the value is always set to '0' in probe request and response frames. **Order** field is 1 bit in size, b15. The value is set to '1' in any non-QoS data frame that contains an MSDU, or fragment thereof, which is being transferred using the StrictlyOrdered service class. This field is set to '0' in all other frames. All QoS STAs set this subfield to '0'. **Duration** field is 2 Octets in size and contains a real number which specifies the number of microseconds that the channel is reserved for the current transmission. **Destination_Address** (DA) field is 6 Octets in size and specifies an individual or group MAC address(es) that identifies the MAC entity or entities intended as the final recipient(s) of the MSDU. **Source_Address** (SA) field is 6 Octets in size and contains an individual MAC address that identifies the MAC entity from which the transfer of the MSDU was initiated. The individual or group bit is always transmitted as a zero in the source address. **Basic_Service_Set_ID** (BSSID) field is 6 Octets in size and uniquely identifies each BSS. In an infrastructure BSS, the value of this field is set to the MAC address currently in use by the STA or AP of the BSS. The value of this field in an IBSS is a locally administered IEEE MAC address formed from a 46-bit random number generated according to a pre-defined procedure. The individual or group bit of the address is set to 0. The universal or local bit of the address is set to 1. This mechanism is used to provide a high probability of selecting a unique BSSID. The value of all 1s is used to indicate the wildcard BSSID. A wildcard BSSID is not used in the BSSID field except for management frames of subtype probe request. **Sequence_Control** field (Figure 2-7) is 2 octets (16 bits) in size and consists of two subfields, Sequence Number and Fragment Number. The Sequence Number field is a 12-bit field indicating the sequence number of an MSDU or MMPDU.

Each MSDU or MMPDU transmitted by a STA is assigned a sequence number. Sequence numbers are not assigned to control frames, as the sequence control field is not present(IEEE 2007, pp. 60-6).

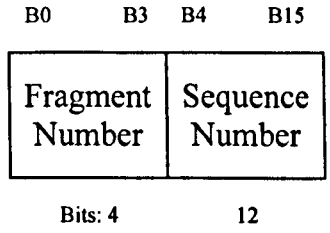


Figure 2-7: Sequence_Control field.

The size of the management frame body can be 0-2312 Octets. A Probe Request frame body contains 5 elements: SSID_Element (octets 2-34), Supported_Rates (Octets 3 - 10), Request_Information (Octets 2-256) - Optional - included if dot11MultiDomainCapabilityEnabled is true, Extended_Supported Rates (Octets 3-257) - Optional - present only when there are more than 8 supported rates. Vendor_Specific (Octets 3-257) - Optional - One or more vendor-specific information elements may appear (IEEE 2007, pp. 67-128). These five elements have a common general format consisting of a 1-octet **Element ID** field, a 1 octet **Length** field, and a variable-length element-specific **Information** field. Each element contains a unique Element ID. The Length field specifies the number of octets in the Information field (IEEE 2007, p. 99).

Service Set ID (SSID) element (octets 2-34): The SSID element indicates the identity of an ESS or IBSS. The unique SSID Element ID is '0'. The size of the SSID information field is 0-32 octets (Figure 2-8). A zero length information field is used within probe request management frames to indicate the wildcard SSID. This enables the STA to associate with any Infrastructure BSS network (IEEE 2007, p. 100).

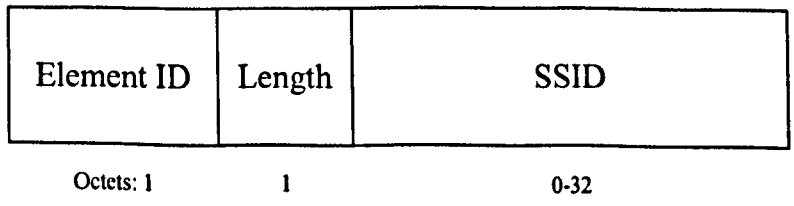


Figure 2-8: SSID element format.

Supported_Rates Element (Octets 3 - 10) and Extended_Supported_Rates Element (Octets 3 - 257): These describe the rates that the WLAN supports. The Supported Rate information in beacon and probe response management frames are used by STAs to avoid associating with a BSS if the STA cannot receive and transmit all the data rates in the BSSBasicRateSet parameter. The unique Supported Rates Element ID is '1'. The size of the Supported Rates field is 1-8 Octets, an Octet per a Supported Rate, with a value in Mbits/s (Figure 2-9). For example, a probe request frame may indicate that only 1, 2, and 5.5 Mbps data rates are available, so that an 802.11b STA would stay within limits and not use 11 Mbps. With this information, STAs also can use performance metrics to decide which AP to associate with (IEEE 2007, pp. 102-11). When number of data rates exceeds 8, the Extended_Supported_Rates element (Figure 2-10) is used to store additional rate information (Gast 2005, p. 93)

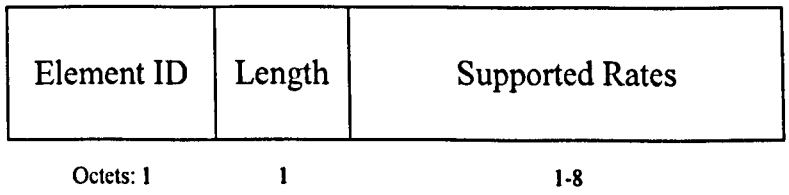


Figure 2-9: Supported_Rates element format.

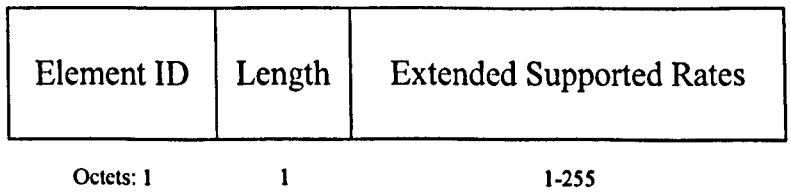


Figure 2-10: Extended_Supported_Rates element format.

Vendor Specific information element (3 - 257): The Vendor Specific information element (Figure 2-11) is used to carry information not defined in IEEE standard within a single defined format, so that reserved information element IDs are not utilised for non-standard purposes, so that interoperability is more easily achieved in the presence of non-standard information. The first three octets of the information field contain the Organisationally Unique Identifier (OUI) that

defines the content of the particular Vendor Specific information element. The unique Vendor Specific information element ID is '221'. The length of the information field (n) is $3 \leq n \leq 255$ (IEEE 2007, p. 128).

Element ID	Length	OUI	Vendor Specific Content
Octets: 1	1	3	n-3

Figure 2-11: Vender_Specific_Information element format.

The OUI field is a public identifier assigned by the IEEE Registration Authority that assigns unambiguous names to objects. It is 3 octets in length. Therefore, the length of the vendor-specific content is n–3 octets. Multiple Vendor Specific information elements can be included in a single frame. Each Vendor Specific information element can have a different OUI value. The number of Vendor Specific information elements that can be included in a frame is limited only by the maximum frame size.

The Frame Check Sequence (FCS) is the last 32 bits in the standard 802.11 frame. This also referred to as the Cyclic Redundancy Check (CRC) as it contains a 32-bit polynomial code checksum for integrity check of retrieved frames (Figure 2-12). When frames are about to be sent, the FCS is calculated over all the fields of the probe request frame header and the frame body field, and appended. When a STA receives a frame it calculates the FCS of the frame and compare it to the one received. If they match, there is high probability that the frame was not changed or damaged during transmission. This also provides a form of security by checking the integrity of the frame before and after transmission. However, FCS does not help in detecting or preventing DoS attacks (IEEE 2007, p. 70).

$$G(x) = x32 + x26 + x23 + x22 + x16 + x12 + x11 + x10 + x8 + x7 + x5 + x4 + x2 + x + 1$$

Figure 2-12: Standard generator polynomial of degree 32.

Figure 2-13 is an illustration of a probe request frame (Gast 2005, p. 107)

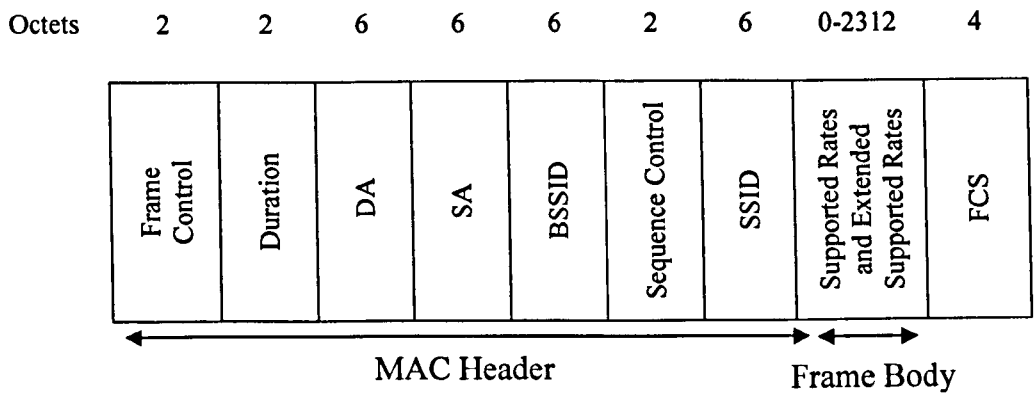


Figure 2-13: Probe request frame

2.5.3. Probe Response Management Frame

As discussed in Section 2.5.1, the MAC layer of the 802.11 protocol communication is based on the exchange of request and response messages, and therefore, each probe request message sent by a STA will be responded with a probe response message sent by the AP. Hence, if no acknowledgement is received due to any reason, the transmitting STA performs a number of re-transmissions of the same frame until the frame is acknowledged by the receiver or the number of re-transmissions reaches the retry limit so that, the frame is discarded (Bicakci & Tavli 2009; Me & Ferreri 2005, p. 5).

The frame format of a management frame is independent of frame subtype and is as shown in Figure 2-5. However, the frame body of each frame sub type varies. Frame body of a probe response message contains the information shown in Appendix A. A STA returns only the information elements that it supports. Therefore, in an improperly formed request information element, a STA may ignore the first information element requested, that is not ordered properly, and all subsequent information elements requested. In the probe response frame, the STA returns the requested information elements in the same order as requested in the Request information element (IEEE 2007, p. 84). The format of the information element is shown in Figure 2-14. The Requested Element IDs are the list of elements that are to be included in the responding STA's probe response frame. The Element ID of this information element is 10. The information element is

variable in length and the length of the information element is indicated in the Length field.

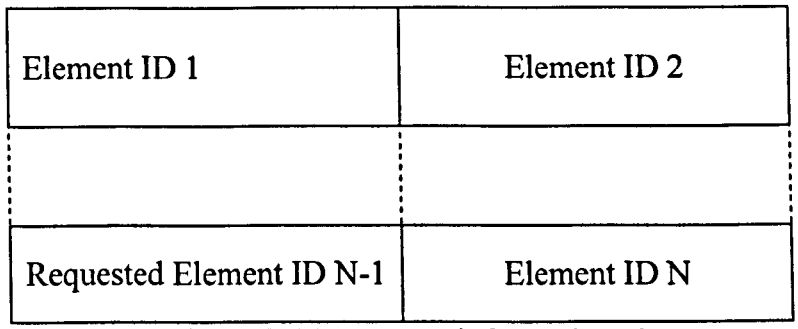


Figure 2-14: Request information element.

Wireshark captures of probe request and response frames are shown in Appendix A.

2.5.4. Statistical Data and Radio Information

Another valuable set of information available to attackers as well as researchers are frame statistical details and radio information generated by the STA that is capturing. Statistical details and radio information gives multitude of operational information. This can be used by an adversary to the learn behaviour of a network to utilise for an attack, or by a Security Administrator to manage and protect a network. PCAP (Packet Capture) protocol helps wireless devices to communicate with each other by converting digital information into radio signals, vice versa. PCAP is used in protocol analysers, network monitors, traffic generators, network testers, Network-based IDSs (NIDS). Linux based version of PCAP is libpcap. Microsoft Windows OSs implement PCAP as WinPcap. Wireshark is a packet analysing software that runs on both Windows and Unix or Linux OSs (Beale et al. 2004; Orebaugh, Ramirez & Burke 2007; tcpdump 2010; WinPcap 2012; Wireshark 2012). Some of the commonly used statistical information are frame arrival time, time delta value, time since reference, frame number, actual packet length, captured packet length, and protocols in frame (Orebaugh, Ramirez & Burke 2007). A detailed list of frame statistical information available for Wireshark users is shown in Appendix C. The Radio tap header contains signal

strength, signal quality, modulation type, channel type, the data rate, channel number. A detailed list of radio information that can be captured using Wireshark is shown Appendix D. In Wireshark frame detail, the 802.11 statistical and radio information is available before the start of the IEEE 802.11 header (Appendix B)

2.6. Summary

IEEE 802.11 standard, WLAN's MAC and PHY specifications for communication introduced in 1999, evolved through the years introducing WEP, WPA and WLA2 security organisations. IEEE-802.11w-2009 is the latest security amendment, which introduced protection for some of the management frames leaving beacon, probe request, and response frames still open and vulnerable for manipulation. The current updated version of IEEE WLAN specification is IEEE-802.11-2012. DoS attacks, is the most common security threat to WLANs. Categorisation of DoS attacks based on some common features also helps finding the correct solution as adversaries conduct similar strategies to perform attacks. This research categorised DoS attacks using OSI layers. The study looked into DoS attack models and their effects on a network and popular and recent DoS attack detection and prevention methods.

IDSs monitor wireless networks to discover any suspicious activity. The monitoring process is performed by analysing wireless network traffic and protocols. IDSs do not respond, but sends alert or response messages to Security Administrators. However, IDPSs provide intrusion prevention in addition to intrusion detection. IDSs employ basic statistical methods to intelligent methods such as soft computing methods. Soft-computing methods are the most successful in current literature. NN is the most commonly used technique for detection where as GA has been utilised for optimisation of inputs, rules and/or models. There is a noticeable trend of using sample databases for development and testing of the solutions, for convenience and to compare results with other literature. However, this overlooks the necessity of looking into the real-world issues and application. Further, some IDSs propose universal solutions to intrusions. However, these

claims are questionable. On the other hand, few of the solutions that utilise real wolrd data explain about the network, but falls short of explaining the scenarios that the data collected for training and testing so the results cannot be compared with similar research.

In spite of years of research and developments, there are severe security failures in WLANs. After analysing the previous research work, progress of IEEE 802.11 sub committees, it is understood that there is a gap of knowledge to develop a new and practical WIDS that could detect probe request attacks on IEEE 802.11 networks. In this context, this research drew attention to probe request attacks belong to the family of DoS attacks. Probing is the first communication that an adversary will perform on a wireless network. Probe requests are management frames that have common features and therefore, adversaries use similar strategies to carryout attacks. Unlike in LANs where STA can be identified based on location, WLAN's STA cannot be uniquely identified due to its mobility feature and unreliability of the MAC address that can be easily spoofed. Detection of spoofed frames plays a major role in detection of attacks including probe request attacks. The research consider that if a WLAN can recognise unauthorised probing or probing floods, it can prevent many future attacks by taking preventive measures. Spoofed STAs can collectively be recognised by MAC frame information, frame statistics and radio information. However, further analysis should be carried out to define most sensitive attributes that can be utilised in an IDS.

With the knowledge gained by this literature review process, in the next Chapter, the research defines the research scope, objectives and formulates research methodology: the methods or approaches and techniques that are used for conduction of research to reach objectives, within the scope specified.

Chapter 3: Probe Request Detection Methodology

3.1. Introduction

Approaches to research include both theory and method (Lazarus 1993). In Chapter 2, the research explored the foundation knowledge of the IEEE 802.11 standard, its security features and common DoS attacks against WLAN, their behaviours and effects on a network. The research also reviewed related scientific and academic work in the area and identified that detection of unauthorised probing or probing floods can also prevent many future attacks by taking preventive measures. Therefore, this research considered to find a method to detect unauthorised probing or probing flooding attacks in WLANs. Research methodology is a way to systematically solve a research problem (Kothari 2009). In this Chapter, the research aims to formulate a hypothesis to resolve the issue of probe request attacks in WLANs, based on the knowledge gained through the literature review. The research describes the scope, objectives and the details and justification of design methods incorporated into this research in Sections 3.2, 3.3 and 3.4 respectively. The adopted methodology is presented in 10 sections: Selection of Intruder Detection Methods, Selection of the Classification Method, Selection of Dataset, Pre-processing of Dataset, Selection of a NN Classifier Design, Performance Measurement, Optimisation, Computer Simulation, Development and Simulation Software, Analysis and Interpretation. Finally, Section 3.5 concludes this Chapter.

3.2. Scope

The scope presents the area of this research's significance, and the extent of its effect, i.e. where, how and under what circumstances this research is applicable, relevant and significant - in other words, what are the research plans to include and exclude in this experimental investigation. The broader scope of this research is to provide

an intelligent system to recognise probe request attacks in WLANs. However, due to resource restrictions such as time and technology, the solution will be designed within below described scope.

a. The sensor will only detect external attackers, i.e. those who do not have the network key.

b. NIDSs capture and analyse network traffic. Capturing software supports capturing traffic on the complete bandwidth. However, a NIC can capture only a single channel at a time. To capture from the entire bandwidth, channel hopping feature required to be enabled, and in the process, will fail to capture all the traffic received, in between channel hopping. The solution for this issue is to use multiple NIC cards that perform co-ordinated capturing so that the STA can capture almost all the traffic that it receives. However, the initial experiments showed high frame losses when capturing on all channels. Further, the STAs in the experiment could not capture efficiently from more than one NIC card due to lack of computing power. Therefore, this research employs only a single network interface card, which is configured to capture only from a single channel.

c. The research also uses a Single AP network known as a BSS (Figure 2-1). Therefore, the proposed design scope excludes DSs.

d. Sensor agent/classifier is only part of a complete IDS.

e. Simulation is performed offline with a collected dataset from the real-network.

f. This thesis will discuss only the design, development and testing the prototype of the sensor/classifier mechanism.

g. The Graphical User Interfaces (GUI) that allow the Security Administrators to interact with the Linux OS and Wireshark capturing software in the capturing STA; to configure the MAC addresses to capture, length of capturing sessions, and output reporting facilities are not included in the scope.

h. This research does not support WLANs with channel-hopping feature.

3.3. Objectives

The research objectives here state what the research aims to accomplish whilst paving the way to formulate the research hypothesis and methods for meeting the objectives. This research project sets objectives that are Specific, Measurable, Attainable, Realistic and Time-bound (SMART) (Dudman 2009). The scope created the boundaries of the research work making it manageable with existing resources. The results will be compared with objectives to measure the success of the research.

The following are the objectives of this research;

a. To design and develop a solution that can detect probe request flood attacks in WLANs.

b. To develop a prototype of a sensor that can detect probe request attacks flood in WLANs using state-of-the art technologies available in the current field of wireless security.

c. To demonstrate that the solution can detect probe request flood attacks efficiently and effectively than the existing solutions using procedures, methods and techniques that have been tested for their validity and reliability.

d. To compare results with existing similar research conducted using; statistical and intelligent systems based methods, online real-time and offline batch frame processing methods, and artificial data and real-word data.

e. To develop a generalising system.

Systematic development and testing of these objectives can be accomplished using a wide variety of methods. These objectives will be reached through research methods described in Section 3.4.

3.4. Research Methods

Research methods define the methods or approaches and techniques that are used for conduction of research.

3.4.1. Selection of Intruder Detection Methods

An IDS is a defence system that detects hostile activities in a network. The purpose of an IDS is to identify possible incidents occurring in a computer and/or network system, and to attempt to report them as alerts or response messages (Karygiannis & Owens 2002; Scarfone & Mell 2007). The current intruder detection approaches can be categorised based on detection approaches; misuse and anomaly, and the resources they monitor; host-based and network-based (Depren et al. 2005). NIDSs monitor network behaviour externally, by collecting information from a network's data stream and providing feedback to a Security Administrator. NIDSs are mostly operating on a single channel. However, in ideal conditions, it will monitor the complete bandwidth. Host-based IDS (HIDS), monitor the network behaviour internally by analysing system calls, logs of a network AP, server, and STA to differentiate between permitted actions and those that should be denied (Depren et al. 2005). Kazienko & Dorosz (2003, 2004) provide an overview of IDS architecture, classification methods and techniques. Magalhaes (2003) suggests that HIDS is ideal for a complete solution and NDIS for a LAN solution complementing HIDS. However, NIDSs outperform HIDS as they lie outside and produce no extra workload or traffic on the AP or server, or client. Signature or misuse detection approach is applied when detecting patterns of well-known attacks as it uses priori knowledge on attacks to look for attack traces. The signature or misuse detection approach compares computer or network activities with the stored signatures of known attacks. This is a simple and effective method, as it has simple algorithms and low false alarm rates. However, it needs regular updating of signature database, and therefore a time consuming method. Further, it is ineffective for unknown, evasive and variants of known attacks (Kazienko & Dorosz 2004; Liao et al. 2012). The anomaly detection

approach detects patterns not conforming to a historic norm of a user or STA. Anomaly detection creates knowledge bases of the monitored activities. Therefore, it detects new and unforeseen vulnerabilities. Also, facilitate privilege abuse and less dependent on the OS. The disadvantages of using anomaly detection include substantial false alarm rate and the time and planning required for training the system (Kazienko & Dorosz 2004; Khan, L, Awad & Thuraisingham 2007; Liao et al. 2012). After taking into account the pros and cons of each method, it was decided that the network based anomaly detection approach is most appropriate for this research.

3.4.2. Selection of the Classification Method

The identification of intrusions needs analysing the collected network traffic so that it can classify each frame into a genuine or an intrusive class. Common approaches that are identified from the literature are statistical, rule-based, and state-based, also include intelligent methods such as soft computing including fuzzy logic, neural computing, evolutionary computation (strengths and weaknesses of these methods were discussed in Chapter 2). Soft computing is a human brain inspired approach, and unlike conventional (hard) computing, it is tolerant of imprecision, uncertainty, partial truth, and approximation. The successful applications of soft computing techniques suggest that they play a crucial part in today's analysis approaches. It is also worth noting that NN is the most widely and successfully used technique for intrusion detection. The literature suggests that NNs are good at recognising patterns. NNs can be trained to classify inputs into a set of target categories (MathWorks 2012e). Therefore, this research will be investigating NN for classification of genuine and rogue frames.

NN is an information processing paradigm, inspired by the structure and functionality of human brain neurons. NNs are considered ideal in dealing with real-world tasks, where data are often messy, uncertain or inconsistent and perfect solutions may not exist (Coolen 1998). The capacity to learn by example makes NNs exceptionally flexible and powerful, as there is no need to create algorithms to execute a task. Its parallel architecture facilitates a high response and less

computational times. These make NNs a perfect candidate for WLANs (Stergiou & Siganos 1996). Jha (2009), in his thesis describes a comprehensive list of common applications of NNs for: signal processing, control, robotics, medicine, speech production, vision, business decision making, financial analysis and predictions, data compression, and game playing. For better understanding of NNs, it is always recommended to understand how the human brain neuron is constructed and functions (Coolen 1998; Rios 2010; Veelenturf 1995). NNs are constructed using complex NN algorithms. However, currently available NN development tools come with task oriented GUI, so that the user is guided through to select appropriate network objects, data, and training styles compatible with the problem the user want to solve. Therefore, a designer does not need a profound knowledge about complex NN algorithms in order to design a NN. However, the designer should have a fair knowledge on NN constructs and their functional behaviour to design, to make them more efficient and effective. The research examines vital features involved in NN development and application in Section 3.4.5.

3.4.3. Selection of Datasets

It was previously stated that IEEE 802.11 works with three frame types (Table 2-1) where management frames establish and maintain communications, control frames help in the delivery of data, and data frames encapsulate the OSI network layer packets. It is also stated (Section 2.5.2) that each frame consists of a MAC header, frame body and a FCS. MAC header comprises of frame control, duration, address, sequence control information, and for QoS data frames, QoS control information. Frame body contains information specific to the frame type and subtype. FCS contains an IEEE 32-bit CRC. Further, a STA seeking to connect to a WLAN has a choice of passive scan or active scan and information passed between theses frames are unprotected and traffic monitoring and capturing software like Wireshark can provide MAC information (Appendix B), frame statistics (Appendix C) and radio information (Appendix D:) to the research. (Ahmad, Abdullah & Alghamdi 2009a; He & Mitchell 2005; IEEE 2007).

However, capturing sample data from a real-world WLAN to train a NN is a tedious and resource demanding task, which needs careful planning. Research required selecting most effective attributes to design the sensor from MAC fields, statistical and radio information to prevent the research utilising unnecessary data, which can exhaust resources and can lead to development of an ineffective sensor. Literature show that many fields of probe request frames such as source, destination, BSSID addresses, sequence number, SSI and duration ID, have been manipulated to achieve DoS attacks. A proof-of-concept pilot study is designed to experiment with the probe request and response messages, software and hardware and to simulate and analyse attacks in a controlled environment so that the research can identify the most sensitive and reliable attributes to utilise in the research. The WLAN test-bed with an attacker and a network monitor is presented in Figure 3-1.

An AP with IEEE 802.11g specification, is capable of serving 20-30 nodes comfortably. APs keep a list of legitimate MAC addresses that can access its services to prevent unauthorised access. However, MAC addresses can easily be spoofed using ifconfig, macchanger (Linux) or using SMAC2 (Windows) to pretend it is a legitimate STA. Association to a network is not required to probe and receive a response. Hence, an adversary only requires a legitimate MAC address to send probe requests. Usually, probing is the initial phase of any other attack in computer networks (Ataide & Abdelouahab 2010).

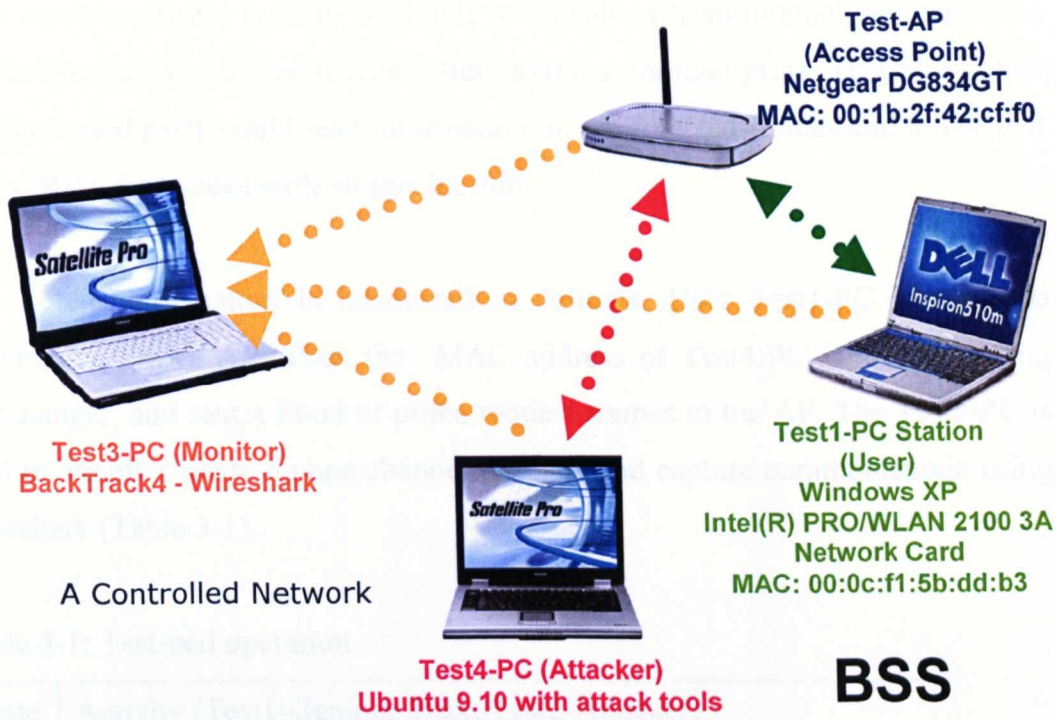


Figure 3-1: Test-bed including an attacker and a network monitor.

The experiment commenced with three Microsoft Windows XP based laptops equipped with internal Intel WLAN cards. However, this researcher soon learned that there are no freely available Windows-based software to monitor and capture management frames. Therefore, the research opted for a monitoring STA installed with BackTrack4. BackTrack4 is an open-source Linux-based OS and contains a large collection of software that can help a security professional or hacker to perform assess or attack a network. Further, commonly available Wireless network adaptors do not support monitor mode and/or monitoring of management frames. Therefore, the monitoring STA is installed with a specialised adaptor that supports promiscuous mode with management frame visibility respectively. Monitor mode also known as promiscuous mode is a mode of network controller that will receive and read all data packets transmitted. NIC in promiscuous mode does not emit any frames. The monitor mode also can be configured to capture frames in one bandwidth or all, by using channel-hopping feature. However, the capture volume depends on the capturing capability of the network card and the processing power of the computer. Therefore, at a very busy location, the network card on monitor mode may not capture all the frames that available in the air. The

open source software is freely available to download from internet, and hardware can be bought from online sources after paying a nominal price. This proves that any interested party could read information on unprotected frames and utilise that information for attacks without much effort.

The Test-bed operation is conducted as follows; User Test1-PC is made to communicate with AP. Then the MAC address of Test4-PC is spoofed using macchanger, and sent a flood of probe request frames to the AP. The Test3-PC is used on monitor mode on one channel to listen and capture communication using Wireshark (Table 3-1).

Table 3-1: Test-bed operation

Time	Activity (Test1-Geniune User; Test2-Attacker)
16.40	Capture started
16.45	Connected (Authentication & Association) test1 to AP
16.47	Test1 started communicating with AP
16.48	Test1 stopped communicating with AP
16.51	Test1 Disconnected (Disassociated) from AP
16.55	Test1 re-connected (Authentication & Association) with AP
17.00	Test2 MAC changed (attacker laptop)
17.01	Test2 continuously sending Probes to AP
17.05	Test1 Disconnected (Disassociated) from AP
17.08	Test1 Shutdown
17.15	Capture Stopped

Figure 3-2 shows the extent of important and sensitive information freely available to an attacker such as MAC fields; AP ID as SSID, network ID as BSSID, DA, SA, Sequence No. (SN), frame type, and frame statistics and signal information during a Wireshark capture.

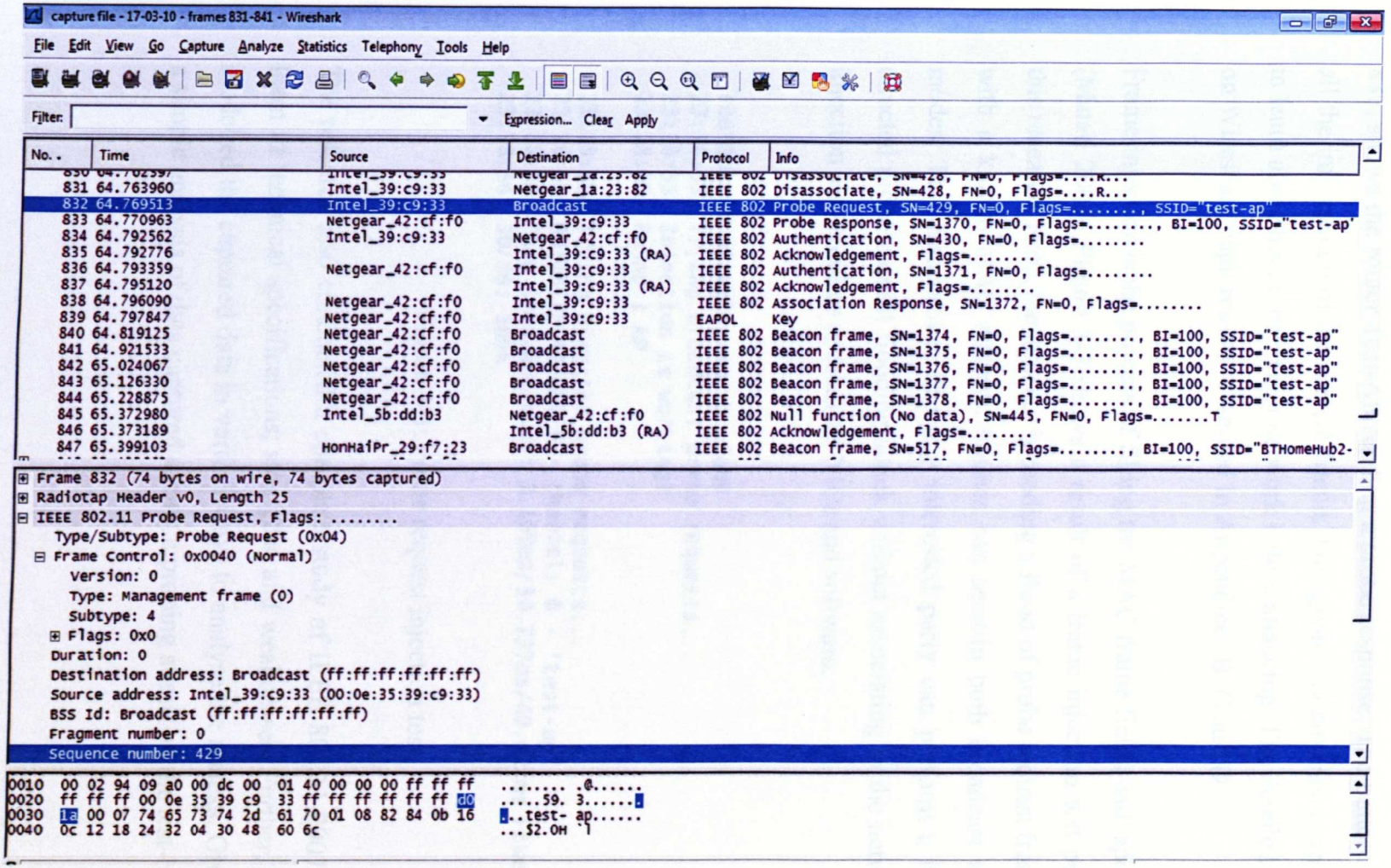


Figure 3-2: List of Management frames captured using Wireshark

In Figure 3-2, at the top pane, frame no. 831 shows Test2-PC disassociating from an AP. Frame no. 832 shows Test2-PC broadcasting a probe request. Frame no. 833, shows the router Test-AP sending a probe response. The bottom pane shows all the information of the current frame. This gives any adversary, an opportunity to learn about his or her target network before attacking. The detailed information on Wireshark captures are presented in Appendices B, C and D.

Frame injection is the process of filling the MAC frame fields with spoofed values (Mateti 2005). Figure 3-3 shows a result of a frame injection test performed by this research. Here, the attacker is sending a flood of probe request frames injected with a known MAC address. Frames can send-in both broadcast and directed modes. This also proves that any interested party can perform a broadcast or directed probe request flooding attack without associating to the network. Frame injection also requires specialised NICs and software.

```
root@bt:~# aireplay-ng -9 wlan0
23:18:53  Trying broadcast probe requests...
23:18:53  Injection is working!
23:18:55  Found 1 AP

23:18:55  Trying directed probe requests...
23:18:55  00:1B:2F:42:CF:F0 - channel: 6 - 'test-ap'
23:18:56  Ping (min/avg/max): 3.169ms/30.737ms/40.422ms Power: -29.77
23:18:56  30/30: 100%
```

Figure 3-3: Probe request injection test.

The research also conducted a complete study of IEEE 802.11-2007 standard to learn its technical specifications, strengths, and weaknesses. Further, the research modelled the captured data in various ways to analyse the attacks. Given below is a simple analysis of data captured during a probing attack on the test-bed.

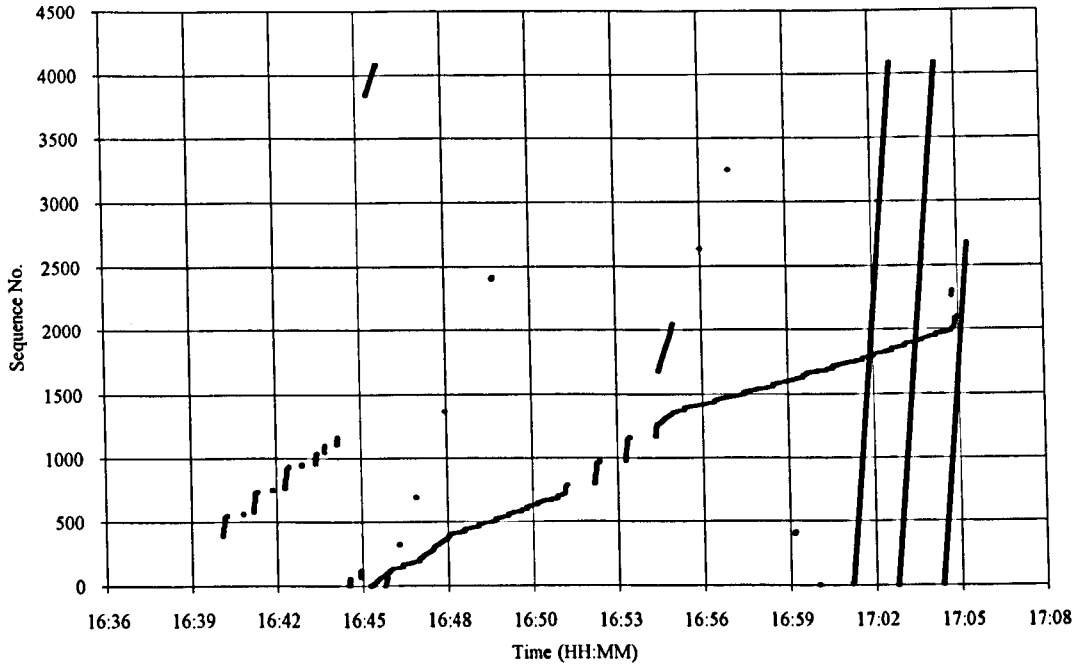


Figure 3-4: Analysis of sequence numbers generated by a MAC address.

Sequence number is a field of a MAC frame. This is a 12-bit counter that starts from zero when a NIC starts or resets, and wraps on 4095 at the overflow. Theoretically, a NIC can generate only one set of sequence numbers at a time (IEEE 2007).

However, Figure 3-4 shows several parallel sequence number patterns generated from the same MAC address. The straight sharp lines (starting at 17.01) are formed due to a high frequency of sequence numbers generated during a spoofed attack whilst other fluctuating and scattered lines are from the genuine user (16.39- 17.08). These additional parallel sequence number patterns may have been generated due to QoS or packet delays (Rumín and Guy 2006).

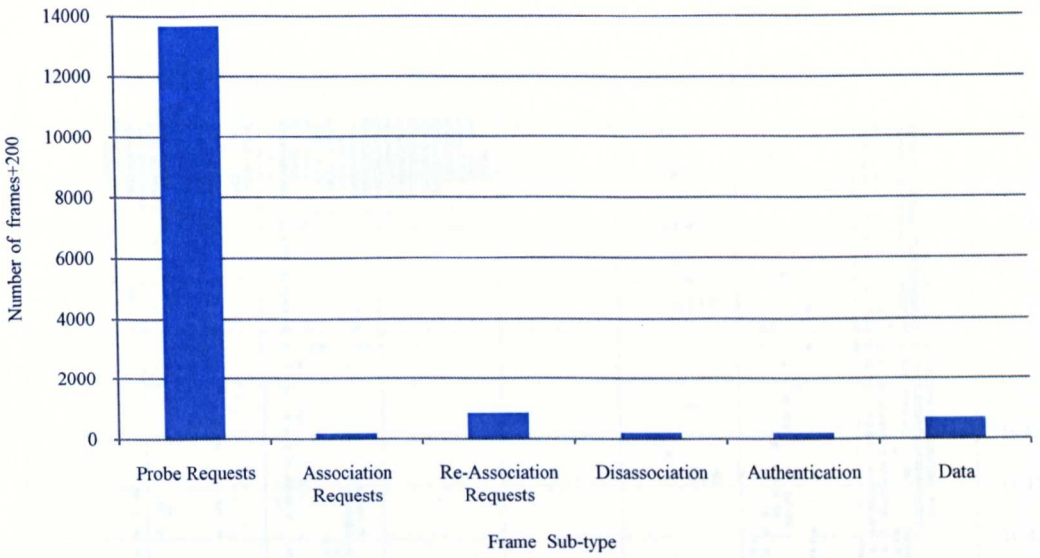


Figure 3-5: Analysis of frame types generated by a single MAC address.

Frame sub-type is also a field of a MAC frame, which identifies the type of the frame (IEEE 2007). Figure 3-5 illustrates a high occurrence of probe request frames, which complements the attack identified by the sequence number analysis in Figure 3-4. Spoofed attacker cannot associate with the network without knowing the network key. Hence, other frame-sub-types have a lower frequency.

Signal Strength Indicator (SSI) also known as Received SSI (RSSI) is one of the radio information of a MAC frame captured by Wireshark. This provides an indication of received transmission power of a NIC, which also gives an indication of the location (Bansal, Tiwari & Bansal 2008). Therefore, SSI patterns are useful in detecting spoofed attacks. In Figure 3-6, from 16.36-17.05, shows the SSI pattern generated by the genuine user. Unusual SSI patterns are generated during the spoofed attack (from 17.01 to 17.15).

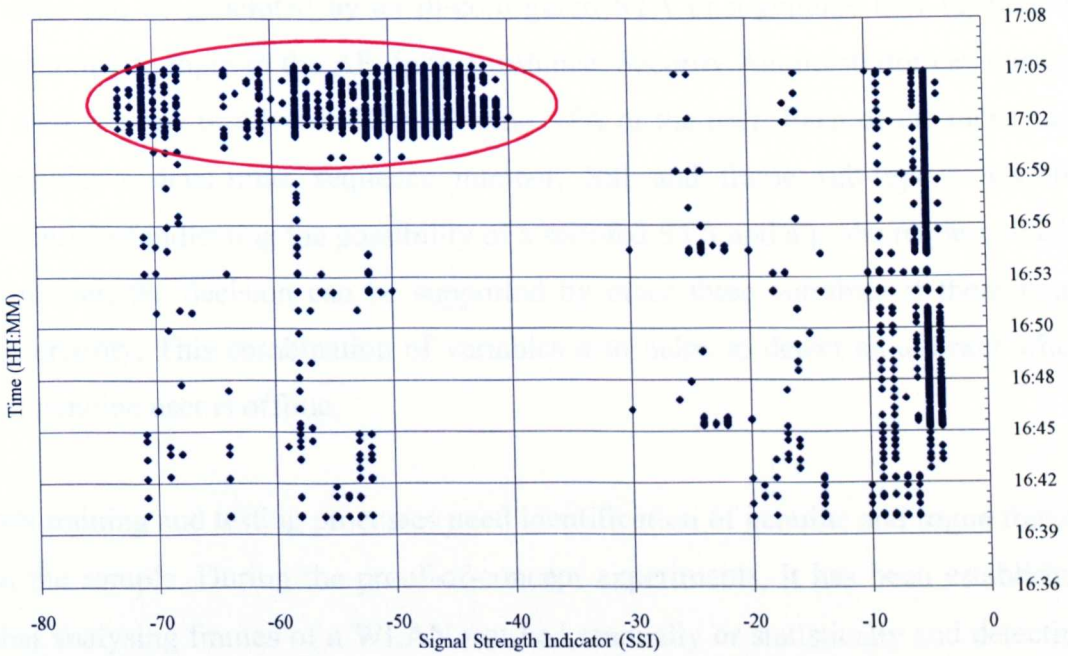


Figure 3-6: Analysis of SSIs generated by a single MAC address.

The delta time value is a frame statistic calculated by Wireshark. Delta time value indicates the time since the previous packet is captured. This gives an indication of the server response time, network round-trip time, and other delays. This is also useful in detecting attacks, as when the AP is under attack, there will be delays in generating frames.

This experimentation provided an opportunity to prove the concepts of IEEE 802.11 standard, and to validate many unrealistic concepts based on extreme theoretical arguments: some argue that attackers can use sequence number synchronising software to generate sequence number patterns to match with the user STA. However, the precision and effectiveness of this technique is doubtful, as they cannot predict the behaviour of the user STA, whether the STA is starting or resetting its NIC card or transmitting data or idling. Some argue that attackers can manipulate SSI (signal strength) by controlling the NIC programmatically or, moving close to the user STA. Signal strength can also simply fluctuate due to environmental factors. These issues are more of theoretical than practical ones as the attacker cannot perform all these activities in a WLAN without exposing as it

is emitting frames through this process. Some argue that excessive probe request frames can be generated by an ill-configured STA or a genuine user repeatedly attempting to login to the AP. In this instance, Security Administrator can correct if there are any problems with a genuine STA or the user. Each of the individual variables - delta-time, sequence number, SSI and frame sub-type - has the potential of indicating the possibility of a spoofed STA and a probe request attack. However, the decision can be supported by other three variables if there is an uncertainty. This combination of variables also helps to detect an attacker when the genuine user is offline.

NN training and testing processes need identification of genuine and rogue frames in the sample. During the proof-of-concept experiments, it has been established that analysing frames of a WLAN test-bed manually or statistically and detecting a rogue frame is possible to some extent due to its controlled nature. However, identifying rogue frames from genuine frames in a real-world WLAN is a difficult task as WLAN traffic pattern is usually unpredictable and it depends on the circumstances such as usage, OS and applications used by its user STA s. Furthermore, it has benn stated previously that monitoring STA can overlook many frames due to its traffic load, or receive them out of order due to packet delay, packet jitter and lost packet or prioritisation services of network traffic such as QoS. Therefore, further experiments are conducted to decide on the data capturing method. Firstly, the research captured genuine and attack traffic separately, and combined it to generate a training dataset. Immediately thereafter, the research added the attack MAC address to the AP's Access Control List (ACL), and performed a probe request attack and captured both genuine and attack frames with same scenarios. The frames from attacker and user were identified by the MAC address, which was not included as a value in the NN training set. The trained model was tested with the first set and second set of frames and the Mean Squared Error (MSE) values were almost identical. However, in real-world scenarios, adding an attacker STA to AP's ACL may not be possible. Therefore, it was decided to capture data separately and combine them for training and testing purposes.

3.4.4. Pre-processing of Dataset

Ideally, traffic filtering is required to capture and save only the required information for the IDS design, training and testing. This research requires only four attributes from each captured frame. NN training requires data to input as numeric values. Therefore, delta-time, sequence number, SSI and frame sub-type will be captured in numerical form. Numerical values for different frame types and sub-types are given in (Orebaugh, Ramirez & Burke 2007, pp. 301-2).

3.4.5. Selection of a NN Classifier Design

The behaviour of a NN classifier is defined by the way its individual computing elements are organised. Therefore, fundamental features involved in NN development and application of a classifier are discussed here. In ANNs, programming constructs a model of some of the properties of biological neurons. As in a human brain neuron, an artificial neuron comprises of many inputs and one output. An operational neuron has two modes. During the training mode, the neuron is trained to activate (fire) or relax (do not fire) for a set of input patterns. During the using mode, the neuron detects taught input patterns and generates its output that was associated with the trained dataset. However, if the input pattern is not in the taught list of input patterns, firing rule decides whether to fire or not (Stergiou & Siganos 1996).

Constructs of a computer-based neuron (k) are synapses (j) with its own weights W_{kj}, a summing junction (transfer function) and an activation function. Further, an externally applied threshold θ_k can lower the net input of the activation function. Conversely, the net input of the activation function can be increased by applying a bias (Haykin 1994, p. 8). Figure 3-7 shows the mathematical model of a nonlinear neuron when a bias is present.

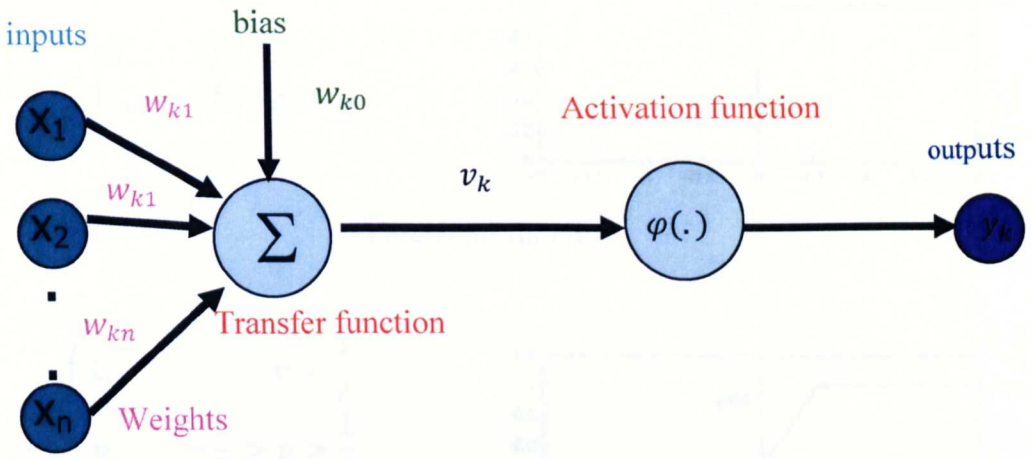
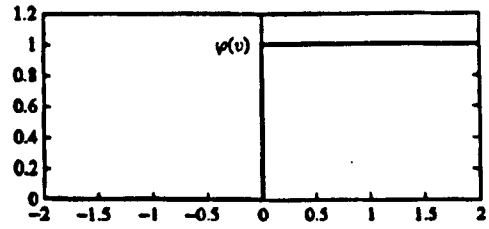


Figure 3-7: The mathematical model of a nonlinear neuron.

The activation function controls the output behaviour of a neuron. Input into the neuron has a weight. This is a floating-point number that will be adjusted when training the network. In Figure 3-7, the input signals are represented as x_1, $x_2 \ldots x_n$ (input vector). The bias (w_{k0}) is an adjustable threshold processor. The bias is also treated as another input. The corresponding weights for the inputs are w_{k0}, w_{k1}, $w_{k2} \ldots w_{kn}$ (vector of weights). At the nucleus, the neuron input is multiplied by its weight. The sum of all new input values gives the activation value (v_k) also known as the total net input. The activation value is a negative or positive floating-point number. The activation function describes the output behaviour of a neuron. The activation function transforms the activation value using a firing rule (Haykin 1994).

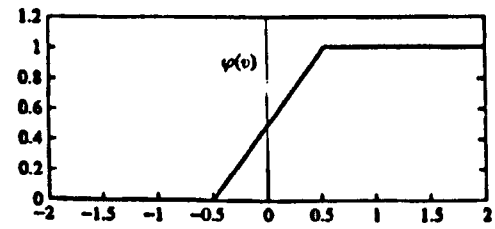
The firing rule is a significant concept in NNs and known for their high flexibility. Firing rule determines whether a neuron should fire or not for an input pattern, including the ones that are not trained (Stergiou & Siganos 1996). In general, there are three basic types of firing rules (Haykin 1994, pp. 10-3) as shown below;

$$\varphi(v) = \begin{cases} 0, & if\ v < 0 \\ 1, & if\ v \geq 0 \end{cases}$$



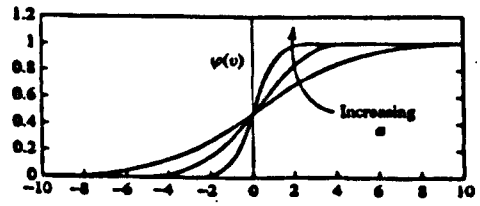
Threshold function

$$\varphi(v) \begin{cases} 1, & v \geq \frac{1}{2} \\ v, & -\frac{1}{2} > v > \frac{1}{2} \\ 0, & v \leq -\frac{1}{2} \end{cases}$$



Piecewise-Linear function

$$\varphi(v) = tanh\left(\frac{v}{2}\right) = \frac{1 - \exp(-v)}{1 + \exp(-v)}$$



Sigmoid function

Figure 3-8: Firing rules.

- Threshold function value = 0, if the summed input is < threshold value (v). Threshold function value = 1, if the summed input is >= to the threshold value (v).
- Piecewise-linear function values = 0 or 1. It can also take values between 0 and 1 depending on the amplification factor in a certain region of linear operation.
- Sigmoid function ranges between 0 and 1. It can also use the -1 to 1 range. The sigmoid curve is frequently used as a transfer function as it introduces non-linearity into the network's calculations by "squashing" the activation level of a neuron into 0 and 1 (Hertz et al. 1990). Two most commonly used sigmoidal activation functions are logistic sigmoid (logsig) and tangential sigmoid (tansig).

The MATLAB NN toolbox pattern classification function *'newpr'* uses NN training function Scaled Conjugate Gradient (SCG) backpropagation algorithm '*trainscg*' with tansigmoid transfer function in both hidden and output layers. The MATLAB NN toolbox function approximation function *'newfit'* uses training function Levenberg Marquardt backpropagation algorithm '*trainlm*' with tansigmoid transfer function in hidden layer, but linear transfer function '*purelin*' in output layer. These functions are further discussed in Section 3.4.9.

Another strong feature in ANNs is the connectionism (or parallel distributed processing). The human brain neurons are connected via extremely complex networks with countless interconnections while ANNs have a much simpler structure. An ANN consists of a group of simple processing units and a large number of weighted connections that communicates with each other. When a processing unit receives input from its neighbours or external sources, it computes an output signal and propagates to other units. At the same time, it also carries out the adjustment to the weights. Many units perform their computations simultaneously, leading to parallel processing (Benediktsson et al. 1997).

As shown in Figure 3-9, there are three types of processing units. The input units receive data from outside the NN; the output units send data out of the NN. The hidden units keep input and output signals within the NN.

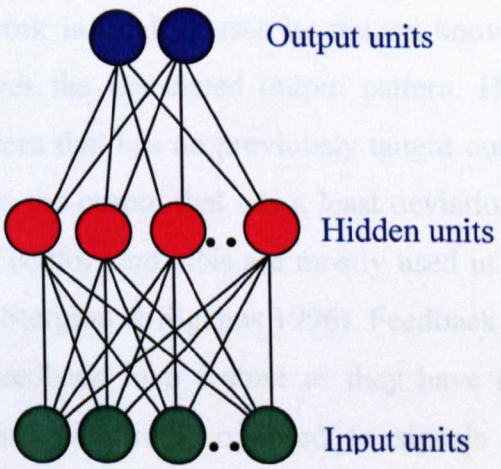


Figure 3-9: Types of processing units grouped by their function.

Hidden units organise their input pattern in their own way. As the weights between the input and hidden units can decide when each hidden unit is active, modifying the weight, a hidden unit can choose its own representation. Each input unit has one-to-many relationship (connections) with hidden units. Each hidden unit also has one-to-many relationship (connections) with output units. Processing units can be updated either synchronously by updating activation simultaneously, or asynchronously by updating activation at a fixed time (Rajasekaran & Pai 2004).

Selecting an NN architecture is another very important decision a NN designer has to make. The designer has to balance the practical implementation, efficiency, and effectiveness of the solution. Therefore, the knowledge in NN architectures and training or learning algorithms is crucial. NN can be broadly categorised by type of network or connecting architecture of neurons and training and learning algorithms (Bishop 1995; Haykin 1994).

Categorisation based on output direction: Feedforward NNs are straightforward networks that associate inputs with outputs, sending signals only in one direction, with no feedback loops. Therefore, the output does not affect the same layer. Feedforward NNs are widely used in pattern recognition. The network is taught to associate the known outputs with its input patterns during the training. Then, when the network is used, it uses its trained knowledge to recognise the input pattern and gives the associated output pattern. However, when the NN receives an input pattern that has no previously taught output association with it, the network calculates the output that has a least deviation from the taught input and output patterns. Feedforward NNs are mostly used in fault diagnosis systems (Haykin 1994, p. 18; Stergiou & Siganos 1996). Feedback networks are extremely powerful owing to feedback loop feature as they have signals flowing in both forward and backward directions, opposed to signals travelling one way in feedforward networks. However, feedback networks can easily turn into enormously complex networks with number of feedback loops easily. The feedback networks are also considered as dynamic as their state changes the

equilibrium point every time a new input is given to the network. Feedback NNs are used in virtual reality systems (Inman 1994, p.190). Figure 3-10 is a pictorial representation of the information discussed above.

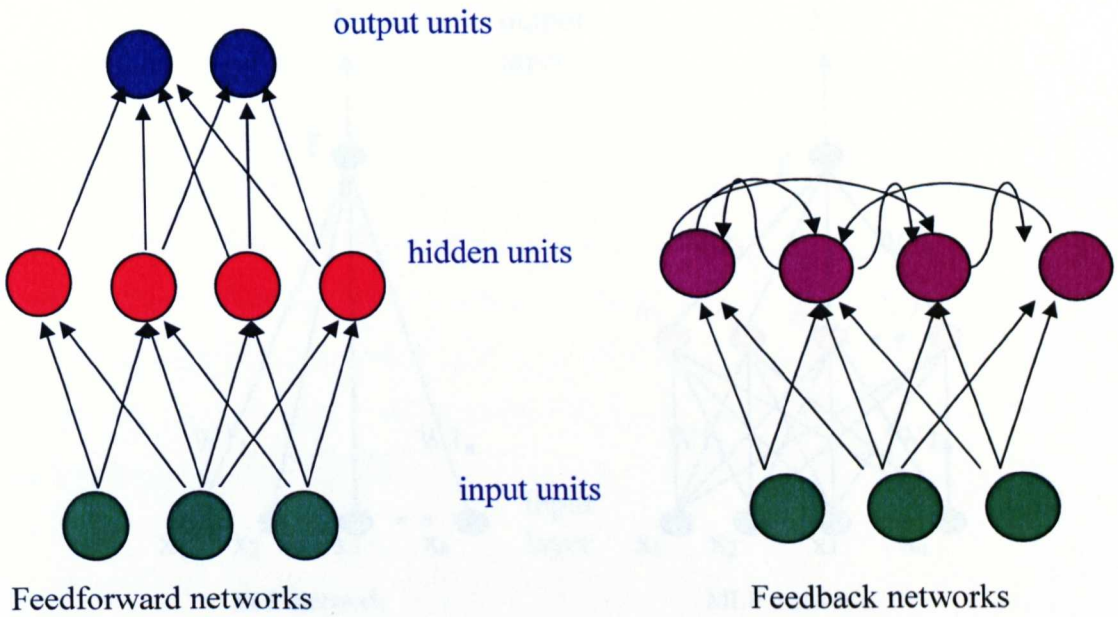


Figure 3-10: Feedforward and feedback networks

Categorisation based on network layers: In layered networks, neurons are structured in the form of layers. In the simplest form of layered network, input nodes layer projects into output nodes layer forming a single-layer network. When counting layers, input layer is not taking into consideration, as it does not perform any computations. Single layer networks are feedforward networks and all units are connected to one another. As shown in Figure 3-11, in a single layer organisation, when appropriate weights ($w_1 \ldots w_n$) are applied to the inputs ($x_1 \ldots x_n$), resulting weighted sum will be passed to a function (f) which produces the output (y). Weights and biases of a single layer network can be trained to produce a correct target vector when presented with a corresponding input vector. Further, a network consists of one or more neurons working in parallel (Haykin 1994, p. 18).

Single layer networks are also known as Single-Layer Perception (SLP) networks, and can be used for classification of linearly separable patterns (Haykin 1994, p.106)

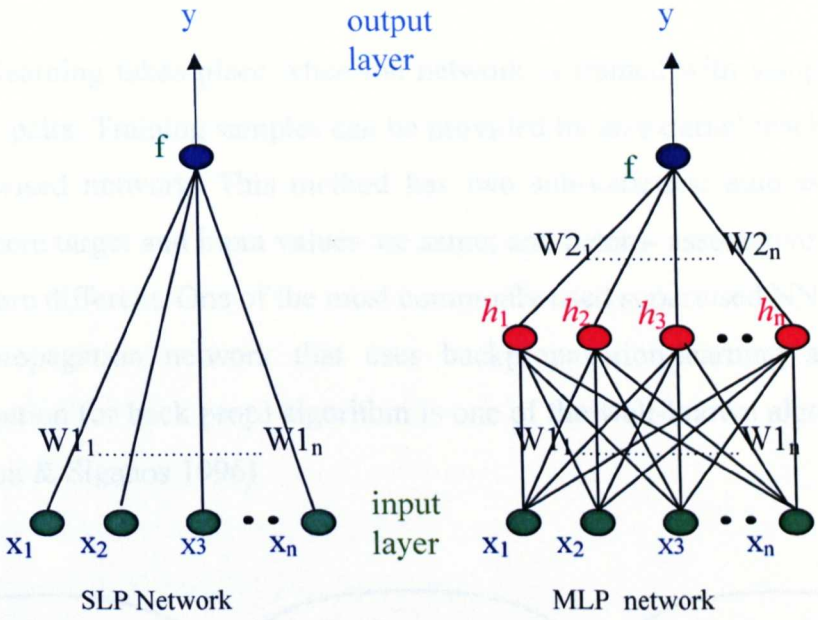


Figure 3-11: Network layers.

Multilayer NN distinguishes itself from the single-layer network by having one or more hidden layers (Figure 3-11). The designer of an NN should consider how many hidden layers are required, depending on target computation capability to be achieved. A multilayer network maps sets of input data onto a set of appropriate output. Except the input layer nodes, all other node layers have a nonlinear activation function. Feedforward multilayer NNs are also known as Multi-Layer Perception (MLP) networks. A MLP network is a modification of the standard linear perception, which can distinguish data that is not linearly separable. MLP utilises a supervised learning technique called backpropagation for training the network (Haykin 1994, p.138). Typically, the multiple layers of nodes are in a directed graph, which is fully connected from one layer to the next. However, networks such as recurrent networks and lattice networks have structure that is more complex.

Categorisation based on training: A NN has to be configured for the inputs to produce the desired set of outputs. There are various methods to perform this task.

Two most common methods are by setting the weights explicitly, using a past knowledge or by training the NN by feeding sample input and respective output patterns and letting it change its weights according to a learning rule. The learning situations can be categorised in to three distinct types as shown in Figure 3-12.

Supervised learning takes place when the network is trained with sample inputs input-output pairs. Training samples can be provided by an external teacher, or by a self-supervised network. This method has two sub-varieties: auto-associative learning, where target and input values are same; and hetero- associative learning, where they are different. One of the most commonly used supervised NN model is the backpropagation network that uses backpropagation-learning algorithm. Backpropagation (or back prop) algorithm is one of the well-known algorithms in NN (Stergiou & Siganos 1996).

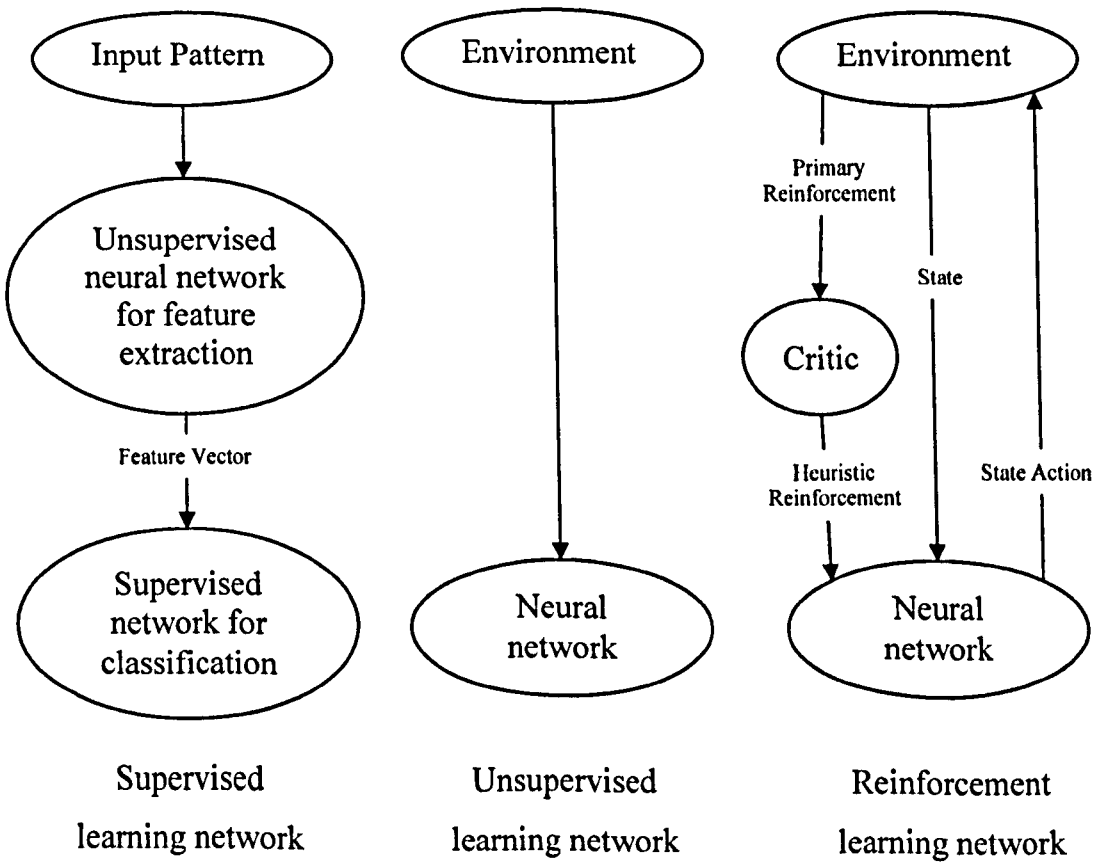


Figure 3-12: Network training.

Unsupervised learning, also commonly known as self-organisation is used in situations where desired target of the network is unknown. During the learning

process, the network is not given a target value, but it is supposed to learn the distribution of patterns and make a classification of that pattern where, similar patterns are assigned to the same output cluster. During training the network performs some kind of data compression such as dimensionality reduction or clustering (Stergiou & Siganos 1996).

Haykin (1994, p. 59) describes reinforcement learning as an online learning of an input-output mapping via trial-and-error process. Reinforcement learning has inherited its features from behavioural psychology. Every action from the learning machine receives a positive (rewarding) or negative (punishing) feedback response from the environment. Then the learning machine adjusts its parameters accordingly, until a stable state occurs, afterwards, there will not be any changes in its parameters. Reinforcement learning lies between supervised and unsupervised learning. The self-organising neural learning may be categorised under this type of learning (Stergiou & Siganos 1996).

The design of the classifier is based largely on the decisions made in Sections 3.4.1, 3.4.2, 3.4.3, and 3.4.4 and knowledge analysed above. The research decided to implement a feedforward NN, with 4 inputs, single hidden layer with 20 neurons, and an output neuron which classify a frame into genuine or attack class. The training algorithm selected is SCG backpropagation that is more suitable for binary classification applications. Twenty neurons is a common value that the literature proposes for NN with single hidden layer. The data and classifier will be validated systematically using self-consistency and 5-fold cross-validation methods. The detailed account of design and evaluation steps is presented in Chapter 4.

3.4.6. Performance Measurement

There is a broad variety of statistical methods used in the literature for measuring the performance of NNs (Demšar 2006; Sokolova & Lapalme 2009). However,

MSE, regression, confusion matrices and ROC curve are the most commonly used methods in the field of intrusion detection using NN.

- The difference between the target and output is known as residual or error, which can be negative or positive. Therefore, the research utilises squared error. Squaring is a way to guarantee that values do not cancel each other during calculations. MSE is the average squared error between the NN's output and the target value of a complete data sample. MSE outputs the error in the same units as the data. Some researchers convert these values to unit-free relative errors, as they are easy to comprehend by business managers (Swamidass 2000, p. 233). However, since the readers of this thesis already have some statistical knowledge, the values will be presented as it is. MSE=zero means no errors. Values closer to zero are better. Some literature suggests that MSE is good for measuring the NN training performance. However, MSEs can be misleading when there are outliers.
- There are many regression methods, which can apply, based on the type of application. Linear regression fits a data model that is linear with the model's coefficients. The most commonly used linear regression is least-squares-fit R2 where the coefficient of determination (R2) ranges from 0 to 1. R2 value closer to one indicates a strong relationship between the two variables. R2 is commonly used in function approximation applications. R2 value 1 means perfect correlation. Statisticians typically use generalised linear models (such as logistic regression) for classification applications. Logistic regression predicts the outcome of a categorical criterion variable based on one or more predictor variables. Although, linear regression is successfully being used in classification applications, many statisticians argue that linear regression should not be used in pattern recognition applications, but logistic regression. Linear regression has a standardised solution and a straight-forward interpretation based on R2 value. In logistic regression, there is no R2 to measure the variance accounted for in the overall model. Instead, a chi-square test is used to indicate how well the logistic regression model fits the data. Multitude of statistical literature is available arguing pros and cons of both the methods (Ciuiu 2007; Freund, Wilson & Sa 2006; Montgomery & Runger 2003; Peng, Lee & Ingersoll 2002; Scott & Freese 2006; Statgun 2007).

- However, the most widely used performance measurement techniques in classification applications are confusion matrices and Receiver Operating Characteristic (ROC) curve (Zaknich 2003). A confusion matrix, 1st introduced by Kohavi and Provost (1998) gives a clear picture of actual and predicted values based on True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) results. In binary classification, a target value 1 and 0 represent the class memberships positive and negative. Output value greater-or-equal to 0.5 indicates the class membership as positive. Output value less-than 0.5, indicates the class membership as negative. Then TP, TN, FP, FN values are computed comparing output value with target value of each frame. Researchers and scientists have later expanded Kohavi and Provost's matrix into extended matrices to present more information concisely. Figure 3-13 shows the MATLAB version of a confusion matrix (MathWorks 2012c).

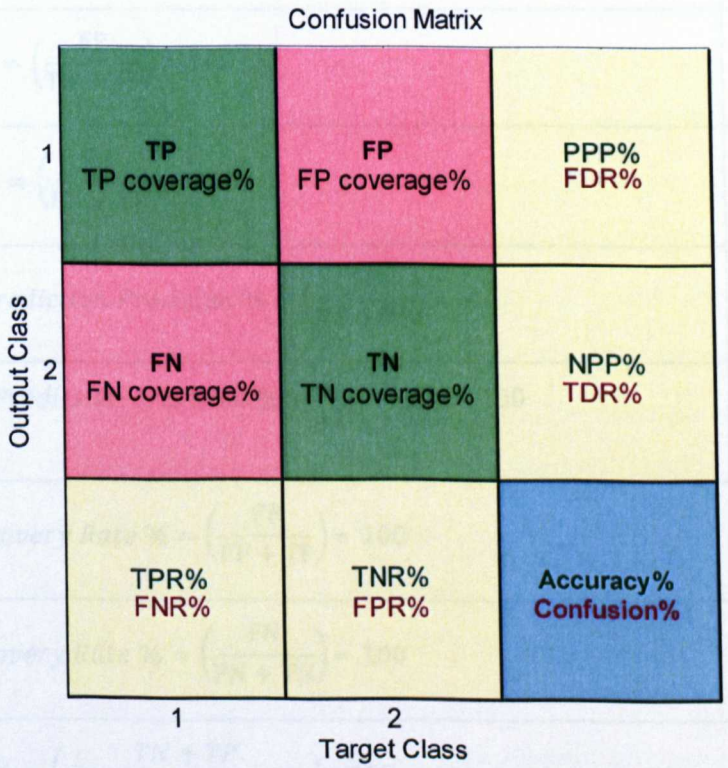


Figure 3-13: An extended confusion matrix.

Table 3-2 shows formulas that derive the information in the confusion matrix. Formulas are obtained from Kohavi and Provost (1998), Benjamini and Hochberg (1995) and Hamilton (2007).

Table 3-2: Confusion matrix formulas.

$TN\ Coverage\ \% = \left(\frac{TN}{TN + FP + FN + TP}\right) * 100$	(1)
$FP\ Coverage\ \% = \left(\frac{FP}{TN + FP + FN + TP}\right) * 100$	(2)
$FN\ Coverage\ \% = \left(\frac{FN}{TN + FP + FN + TP}\right) * 100$	(3)
$TP\ Coverage\ \% = \left(\frac{TP}{TN + FP + FN + TP}\right) * 100$	(4)
$TP\ Rate\ (a.k.a.\ Sensitivity\ and\ Recall)\ \% = \left(\frac{TP}{FN + TP}\right) * 100$	(5)
$TN\ Rate\ (a.k.a.\ Specificity)\% = \left(\frac{TN}{TN + FP}\right) * 100$	(6)
$FP\ Rate\ \% = \left(\frac{FP}{TN + FP}\right) * 100$	(7)
$FN\ Rate\ \% = \left(\frac{FN}{FN + TP}\right) * 100$	(8)
$Positive\ Prediction\ Precision\ \% = \left(\frac{TP}{TP + FP}\right) * 100$	(9)
$Negative\ Prediction\ Precision\ \% = \left(\frac{TN}{TN + FN}\right) * 100$	(10)
$False\ Discovery\ Rate\ \% = \left(\frac{FP}{FP + TP}\right) * 100$	(11)
$True\ Discovery\ Rate\ \% = \left(\frac{FN}{FN + TN}\right) * 100$	(12)
$Accuracy\ \% = \left(\frac{TN + TP}{TN + FP + FN + TP}\right) * 100$	(13)
$Confusion\ \% = \left(\frac{FP + FN}{TN + FP + FN + TP}\right) * 100$	(14)

Coverage is the proportion of a dataset for which a classifier makes prediction. Figure 3-13 shows TP, TN, FP, FN coverage in terms of a rate and a percentage. Table 3-2 shows their percentage calculation using Equations (1) – (14). TP rate is the proportion of positive class members classified correctly. TP rate also known as sensitivity and recall or hit rate. Table 3-2 shows their percentage calculation using equation (5). TN rate is the proportion of negative class members classified correctly. TN rate also known as specificity. TNs are also known as correct rejections. Table 3-2 shows their percentage calculation using equation (6). FP rate is the proportion of negative class members classified incorrectly as positives. FP rate also known as false alarm rate or fall-out. Table 3-2 shows their percentage calculation using equation (7). FN rate is the proportion of positive class members classified incorrectly as negatives. FNs are also known as misses. Table 3-2 shows FN percentage calculation using equation (8). Precision is the proportion of class members classified correctly (or incorrectly) over the total number of instances classified as class members (Keller 2012). Figure 3-13 shows both Positive and Negative Prediction Precision (PPP and NPP) in terms of a rate and a percentage. Table 3-2 shows their percentage calculation using equations (9) and (10). Accuracy is the proportion of class members classified correctly over a dataset where as confusion is the proportion of class members classified incorrectly over a dataset. Figure 3-13 shows both accuracy and confusion in terms of a percentage. Table 3-2 show their percentage calculation using equations (13) and (14). Benjamini and Hochberg (1995) introduced False Discovery Rate (FDR). Although, Kohavi and Provost (1998) do not discuss about false and true discovery rates, scientists and statistics analysis software such as MATLAB have later included it in their presentation of confusion matrixes. Figure 3-13 shows both True Discovery Rate (TDR) and FDR in terms of a percentage. Table 3-2 shows their percentage calculation using equations (11) and (12). A series of FP and TP pairs plots a Receiver Operating Characteristics (ROC) curve. A ROC curve is a visual tool to recognise the positive and negative samples that are incorrectly identified. ROC is given the name as the curve compares two operating characteristics (TPR and FPR) as the criterion changes. When (0.1), the FP=0 and TP=1, which indicates a perfect predictor. Therefore,

more each curve hugs the left and top edges of the graph, the better the prediction it gives. The area beneath the curve can be used as measure of accuracy. ROC curve also encapsulates the all information presented in a confusion matrix and therefore commonly used by researchers to show the consistency of results (Fawcett 2006; Freund, Wilson & Sa 2006; Hamilton 2007; MathWorks 2012g; Montgomery & Runger 2003; Van Trees 2001).

After analysing the knowledge gained from the above study, and the reviewed literature, the research decided to utilise MSE, Confusion value and ROC curve as measurement of performance in this research.

3.4.7. Optimisation

The effectiveness of an IDS is evaluated by its ability to make correct predictions, and therefore, the NN has to be optimised and trained. NN optimisation is the art and science of arranging interconnected factors to the best possible effect. Many computational problems have a need of searching through a large search space for possible solutions, however, conventional methods that are in use including trial-and-error, which are not rigorous enough to arrive at a truly optimal structure. In contrast, GAs outperform traditional optimisation methods, due to its rigorousness in finding optimal parameters. A GA can recognise behaviour of a model for diverse configurations to provide a reliable model through evolution. Therefore, the research considered employing GA to optimise the sensor/classifier.

The initiatives of evolutionary computing is introduced by Rechenberg in his work 'Evolution Strategies'. Evolutionary computing systems are designed to evolve a population of candidate solutions to a given problem, using operators inspired by natural genetic variation and natural selection. GAs are a part of evolutionary computing. A GA can be defined as a search heuristic that is modelled on the principles of natural selection, reproduction by crossover and mutation. A fitness function is applied to evaluate individuals. GA is invented by John Holland, as a theoretical framework for adaptive systems (Mitchell 1996).

However, De Jong (1993) established that population-based GAs using crossover and mutation could successfully be applied to optimisation problems in many areas.

To understand GAs, it is required to be aware of biological terminology of natural evolution. Organisms are made of just one cell or many cells. Each cell has a nucleus containing a unique set of chromosomes. Each chromosome consists of a compact coil of Deoxyribonucleic Acid (DNA) which serves as a 'blueprint' that stores coded information of the organism. A chromosome consists of genes that can also be defined as functional blocks of DNA. Each gene encodes a trait such as colour of eyes, and hair. Settings for a trait such as blue, green, brown are called alleles. Each gene has a particular position in the chromosome called locus. A complete set of chromosomes in a cell is call genome. A particular set of genes in genome is call genotype. An organism's genetic heritage (genetically transmitted genes from one generation the next) is called genotype. The genotype traits that are observable by appearance and behaviour are called phenotype. Some of the phenotypes are height, weight, eye colour, and psychological characteristics such as intelligence, personality, and creativity. However, some organisms with the same genotype do not have the same appearance and behaviour due to environmental factors and developmental conditions. The organisms on earth have been evolving because of processes of natural selection, recombination and mutation. During biological reproduction, genes from parents form in some way to create an offspring that have genes from both parents. This process is also known as recombination or crossover. Occasionally, changes to genes can be occurred by changing the elements of DNA due to errors in copying genes from parents. This is known as mutation. Mutated genes usually do not affect the development of the phenotype but occasionally, it will be expressed in the individual as a completely new trait. The fitness of an organism is measured by success of the organism in its life.

Any GA has at least following elements: population of chromosomes, selection based on fitness value, crossover to produce offspring, and mutation of offspring.

GAs uses combination of biological and computing terminologies. A chromosome is usually an encoded bit string that is used as a candidate solution to a problem. A gene is a single or block of adjacent bits that represent a particular element of the candidate solution. Two chromosomes exchange their genetic material at a locus point during the crossover process. Mutation is performed by randomly flipping a bit at a random locus. Generally, an initial population of chromosomes is created randomly according to search space restrictions. Search space of a GA is the collection of all possible solutions; it can be an infinity for some applications. However, due to resource constraints, GAs restrict the search space to a descent size. The size of the population is maintained the same throughout the GA. Furthermore, there are many schools of thought on population of chromosomes, selection, crossover, mutation probability, and method. Genetic variation arises through mutation and recombination. However, natural selection can remove variation from a population. Unlike natural population, GA populations are finite. Together, these factors lead to a persistent loss of variation, a process referred to as genetic drift (Lande 1976; Nei & Tajima 1981).

In GAs, performance of a chromosome population is evaluated by a fitness value. Some of the popular evaluation criterions found in the literature are MSE, sensitivity, specificity, confusion, and accuracy. Fitness value is more specific to the type of application. Some of the most common applications of GA are optimisation, automatic programming, machine learning, economics, immune systems, ecology, population genetics, evolution and learning, social systems (Mitchell 1996; Obitko 2012).

Table 3-3: Basic steps of a genetic algorithm

Step 1: [Initial Population] Create a population of n chromosomes
Step 2: [Fitness] Evaluate the fitness f(x) of each chromosome x in the population
repeat
Step 3: [Selection] Select two parent chromosomes
Step 4: [Crossover] Crossover over the parents at a locus point to form a new offspring
Step 5: [Mutation] mutate new offspring at a locus point
Step 6: [New Population] Place new offspring in a new population

Step 7: [Fitness] Evaluate the fitness f(x) of each chromosome x in the population until end of number_of_ generations

Reeves & Rowe (2002) suggest that GAs are frequently successful and considered as the most popular in real applications. They also state that probably everybody's GA is unique. Although, GAs can find global optima, when it is applied to different problems, there are other attractors to which they may converge, with a distance from global optimality. Since Holland (1975) and De Jong (1993), researchers and practitioners have recommended many variations of GA recommending population sizes, initialisation methods, fitness definitions, selection and replacement strategies, and crossover and mutation methods. Many have used GA in the field of intrusion detection (Bankovic et al. 2007; Gong, Zulkernine & Abolmaesumi 2005; Haddadi et al. 2010; Hua & Xiaofeng 2008; Li 2004; Pillai, Eloff & Venter 2004; Wu & Banzhaf 2010; Xia et al. 2005; Yao 2010). In reply to Reeve's (1997) feature article on "Genetic algorithms for the operations researcher", Levine (1997) provides practical recommendations that has a clear and reasonable deviation from traditional GAs, which this research utilises as a guideline. In summary, Levine (1997) recommends following; Steady-state reproductive strategy, stochastic universal selection or tournament selection, uniform or two-point crossover, an adaptive mutation rate, hybrid GA which takes problem specific information. Detailed methods and techniques utilised for classifier optimisation using GA are presented in Chapter 5.

3.4.8. Computer Simulation

Computer simulation is an imitation of the functions of a real-world process or system. It is a substitute for experimentation with an actual system that experimentation with it could be disruptive, costly or not possible. Computer simulation is first used as a scientific tool in meteorology and nuclear physics, but now has grown rapidly to become indispensable in a wide variety of scientific disciplines as a tool to create and justify knowledge claims (Banks & Carson 1984; Guizani et al. 2010; Winsberg 2010). A real world IDPS is a fully

functional online real-time system. It can be a software application, a firmware, or a combination of both. However, when simulating, it is essential to decide which key components will be incorporated into the simulator based on its requirement. Each additional component adds complexity to the simulation which may not be feasible due to practical reasons such as non-availability of computing power, expertise required to build a fully-functional software, and time constraints (Wu & Banzhaf 2010). National Institute of Standards and Technology (NIST)'s Guide (Scarfone & Mell 2007) to IDPS, is a popular source of information on IDPS. Architecture of a typical IDPS is shown in Figure 3-14.

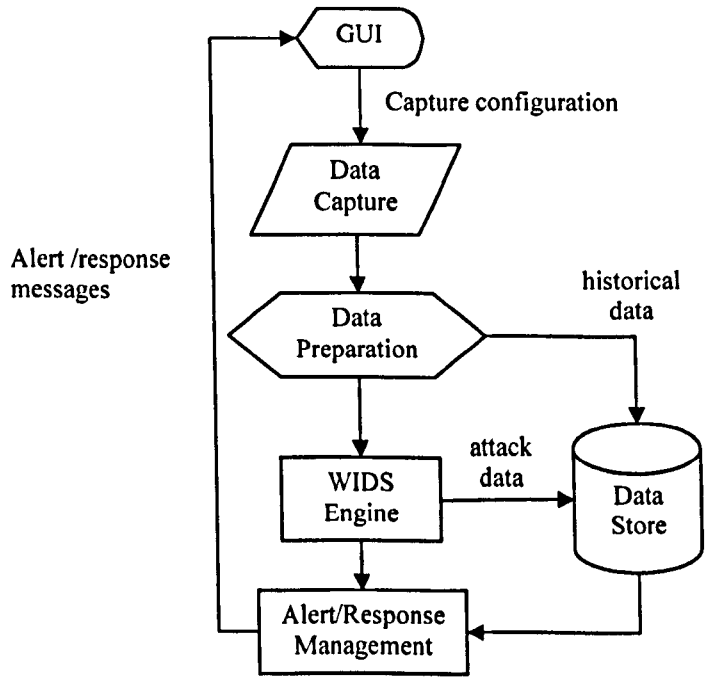


Figure 3-14: Architecture of a typical IDPS.

Graphical User Interface (GUI): A real-world IDS has a GUI to select Network ID (NI), MAC address(s) to be monitored, size-of-capture by type or frames, protocol, modes of reporting processed information and so on. **Data capturing:** Data capturing module interfaces with PCAP protocol that helps wireless devices to communicate with each other by converting digital information into radio signals, vice versa. Security Administrators may require historical data to be analysed in case of an attack or for future decisions. The Security Administrator can control the amount of information that is to be captured and stored based on the traffic volume and resources available. **Data preparation:** A data capture may

include varieties of data types, sizes, noise (garbage) and additional data that may not be required for simulation. Therefore, it may be essential that data be filtered, cleansed, and formatted to suit to the requirement of the IDS engine. **IDS engine:** This is a fully functional sensor or collection of classifiers which actually analyse frames to detect whether they are genuine or not. **Data Storage:** This can be a simple collection of some data and an attack log or a fully functional data warehouse with historical data for reporting and data analysis. **Alert and Response Management:** Alert and response manager interacts with the IDS and the user, and produces many statistical and management information. This research applies this simulation structure for simulation of the trained and optimised classifier in Chapter 6.

3.4.9. Development and Simulation Software

There are many commercial and open source software that researchers utilise to design, develop and simulate prototypes. The research plan requires Linux and Microsoft Windows based Wireshark (Wireshark 2012) for capturing and analysis of traffic, MATLAB for classifier design training, testing and simulation, Java for development of GA, Microsoft Excel and Microsoft Word for data analysis and presentation. The main software tool that is available is MATLAB 2009b provided by the University for the Research. MATLAB technical computing language is a programming environment for algorithm development, data analysis, visualisation, and numerical computation that can employ to solve technical computing problems faster than with traditional programming languages. MATLAB claims that million engineers and scientists in industry and academia are using for wide range of applications with MATLAB add-on products for parallel computing, math, statistics, and optimisation, control system design and analysis, signal processing and communications, image processing and computer vision, test and measurement, computational finance, computational biology, code generation, application deployment, database connectivity and reporting. It also facilitates command-line functions and graphical tools for its tasks, and has the facility for code auto generation (MathWorks 2012a).

NN toolbox: MATLAB NN toolbox provides built in tools for designing, implementing, visualising, and simulating NN for applications such as pattern recognition and nonlinear system identification and control. It supports feedforward and other proven NN paradigms. Some of the MATLAB functions that commonly used in supervised NNs for pattern recognition and classification applications are described as follows:

'newpr' is the default function for creating pattern recognition networks. It creates a feedforward network with a single hidden layer when inputs, targets and number of neurons are given using *'newff'* function. Function *'newpr'*s default transfer functions are *'tansig'* for both hidden and output layers. Sigmoid activation function forces the output values to be normalised between 0 and 1. *'newpr'*s Backpropagation network Training Function (BTF) is *'trainscg'*. It updates weight and bias values according to the SCG method during training. The Performance Function (PF) is *'mse'*, mean squared normalised error performance function that measures the network performance according to average squared error between the network outputs and the targets. Data Division Function (DDF) is *'dividerand'*.

'newff' function is also used for function approximation *'newfit'*, but with default transfer function *'pureln'* for output layer and training function *'trainlm'* that updates weight and bias states according to Levenberg-Marquardt optimisation. MATLAB literature suggests that *'trainlm'* is the fastest training function in the toolbox. However, it also suggests that it performs better on function fitting problems than pattern recognition problems.

dividerand function randomly divides data into three subsets, namely; training, validation and test, using the division parameters gene values *trainVal, validVal,* and *testVal.* The training sample is used to compute the gradient and update the network weights and biases. The error on the validation sample is monitored during the training process. The validation error normally decreases during the

initial phase of training, as does the training sample error. However, when the network begins to overfit the data, the error on the validation sample typically begins to rise. The network weights and biases are saved at the minimum of the validation sample error. The test sample error is not used during training, but it is used to compare different models. It is also useful to plot the test sample error during the training process. If the error on the test sample reaches a minimum at a significantly different iteration number than the validation sample error, this might indicate a poor division of the data sample.

'train' function trains a NN according to *NET.trainFcn* and *NET.trainParam*. *'train'* calles *'trainscg'*, using the training parameter values indicated by *NET.trainParam*. One epoch of training is a single presentation of all input vectors to the NN. Then the NN is updated according to the results of each presentation. Training will stop when the maximum number of epochs, or the performance goal reached, or any other stopping condition of the *NET.trainFcn* occur neurons.

Conjugate gradient algorithms require a line search at each iteration which is computationally expensive, because it requires that the network response to all training inputs be computed several times for each search. However, the SCG algorithm is designed to avoid this resource-consuming line search. However, '*trainscg*' performs more iterations to converge, but the number of computations in each iteration is significantly reduced because no line search is performed. Further, its performance does not degrade quickly as when the error is reduced. The conjugate gradient algorithms have comparatively modest memory requirements.

Properly trained multilayer networks response well to unseen data. Generalisation property makes it achievable by training the network on a representative set of input and output pairs rather than training it on all possible input and output pairs. NN toolbox facilitates generalisation by using regularisation and early stopping. Regularisation is built-in in Bayesian regularisation training function

'*trainbr*'. It minimises a combination of squared errors and weights, and then determines the correct combination to produce a network that generalises well. Early stopping is the default generalisation method used in multilayer feedforward networks, which is maintained via '*dividerand*' data division function.

Function *sim*, simulates a trained network with test data, and generates a vector of errors, which can use to measure the MSE of the dataset using '*mse*' function. Function '*confusion*' takes targets and outputs and produces confusion error rate, and number of TP, TN, FP, and FNs. Functions '*plotperf*', '*plotconf*', '*plotroc*' produce performance, confusion and ROC curve graphs (MacKay 1992; MathWorks 2008, 2012b, 2012e; Møller 1993). Further information is available in prologue comments in the functions.

3.4.10. Analysis and Interpretation

In conventional research, after analysis of data, the explanation of the findings based on the results is call interpretation (Khanzode 1995). This research simulates NN classifier in 10 scenarios to explain results. MSE and confusion matrices can give a clear picture of behaviour of simulated unseen traffic. This research use numerical and graphical representations of MSEs, confusion matrices, ROC curve and accuracy or confusion charts to present results. Following interpretations will be reached based on results;

a. The sensor/classifier's detection rate of known and unknown attacks.
b. Effect of the movement of an attacker and a user on sensor/classifier's detection rate.
c. Effect of the user and an attacker's presence at the same time on sensor/classifier's detection rate.
d. Effect of OSs, NICs and attack tools.

Further details are presented in Chapter 6.

3.5. Summary

The design of any research begins with the selection of objectives, scope and a research methodology. These initial decisions produce assumptions on how science should be conducted, and what constitutes legitimate problems, solutions, and criteria of proof of results. The objective of this research is to design and develop an unbiased and objective solution that can detect probe request flood attacks in WLANs: by developing a prototype of a sensor/classifier that can detect probe request attacks in WLANs, using state-of-the art technologies available in the current field of wireless security, and demonstrate that the solution can detect probe request flood attacks efficiently and effectively, and compare the results with statistical methods and intelligent methods that process frames both online real-time or offline batch using artificial data or real-word data. However, due to resource restrictions such as time and technology, and only a sensor/classifier will be designed only to detect external attackers, on a single channel of a BSS. Simulation will be performed offline with a collected dataset from the real-network. A quantitative research approach has been selected where results will be measured with numbers, and analysed using statistical techniques to determine whether the predictive results are equivalent to expected results.

This Chapter also defined research methods and techniques that will be used for the conduction of research. The research selected a network based anomaly detection approach for intruder detection. A feedforward NN, with 4 inputs, single hidden layer with 20 neurons, and an output neuron which classify a frame, into genuine or attack class. The classifier will be validated systematically using self-consistency and 5-fold cross-validation methods. The classification of frames will be based on MAC frame fields frame sub-type and sequence number, radio information SSI and signal statistic delta-time. These values will be captured in numerical form. Further training, testing and simulation datasets will be captured individually and merged to form a complete dataset of genuine traffic and attack traffic. GA will be utilised to optimise the data and number of hidden neurons of NN and performance will be measured based on confusion error of test set. The

software that this research will use comprise of Microsoft Windows and Linux OSs, Wireshark, MATLAB, Java Development Envirnment, Microsoft Word and Excel . The analysis of results will be performed using MSE, Confusion matrices and ROC curve to reach conclusion about how the classifier faired, during training, validation, testing, and simulation to detect; rate of known and unknown attacks, effect of detection when an attacker and a user move from their habitual locations, effect of detection when the user and an attacker's presence at the same time, and finally the effect of OS, NIC and attack tools.

The next chapter describes the WLAN environment and data capturing process, the design, and implementation of the ANN probe request attack sensor/classifier, and the systematic evaluation of performance of the sensor/classifier.

Chapter 4: Neural Network Classifier Design and Evaluation

4.1. Introduction

This chapter includes the probe request attack sensor design and its performance evaluation. Pattern recognition sensors are also known as classifiers in the field of NNs. The research develops a prototype of the classifier to analyse real WLAN traffic on a home wireless network using supervised feedforward NN, and to classify genuine frames from rogue frames based on methods and techniques defined in Chapter 3. Figure 4-1 is an outline of the steps to creating the classifier: data capturing, preparation, training and validating, and testing. Research applies three evaluation techniques for training, validating, and testing models. The first technique performs training, validation and test processes in parallel with an allocated percentage of data, which all three sets pass through the NN during each cycle known as an epoch until it reaches a stopping condition. The second technique trains and validates using the complete dataset and tests the model using the same data. The third technique is the 5-fold cross-validation method, which segments data into five sets and trains and validates the model using four sets, and tests using the retained set so that each set will be tested against the model trained with other four sets.

The Chapter is organised as follows: Section 4.2 describes the WLAN environment and data capturing process, Section 4.3 explains NN classifier design and training process, Section 4.4 gives a comprehensive account of evaluation of the NN model using self-consistency and 5-fold cross-validation methods, Section 4.5 summarises the experiment results and concludes the chapter.

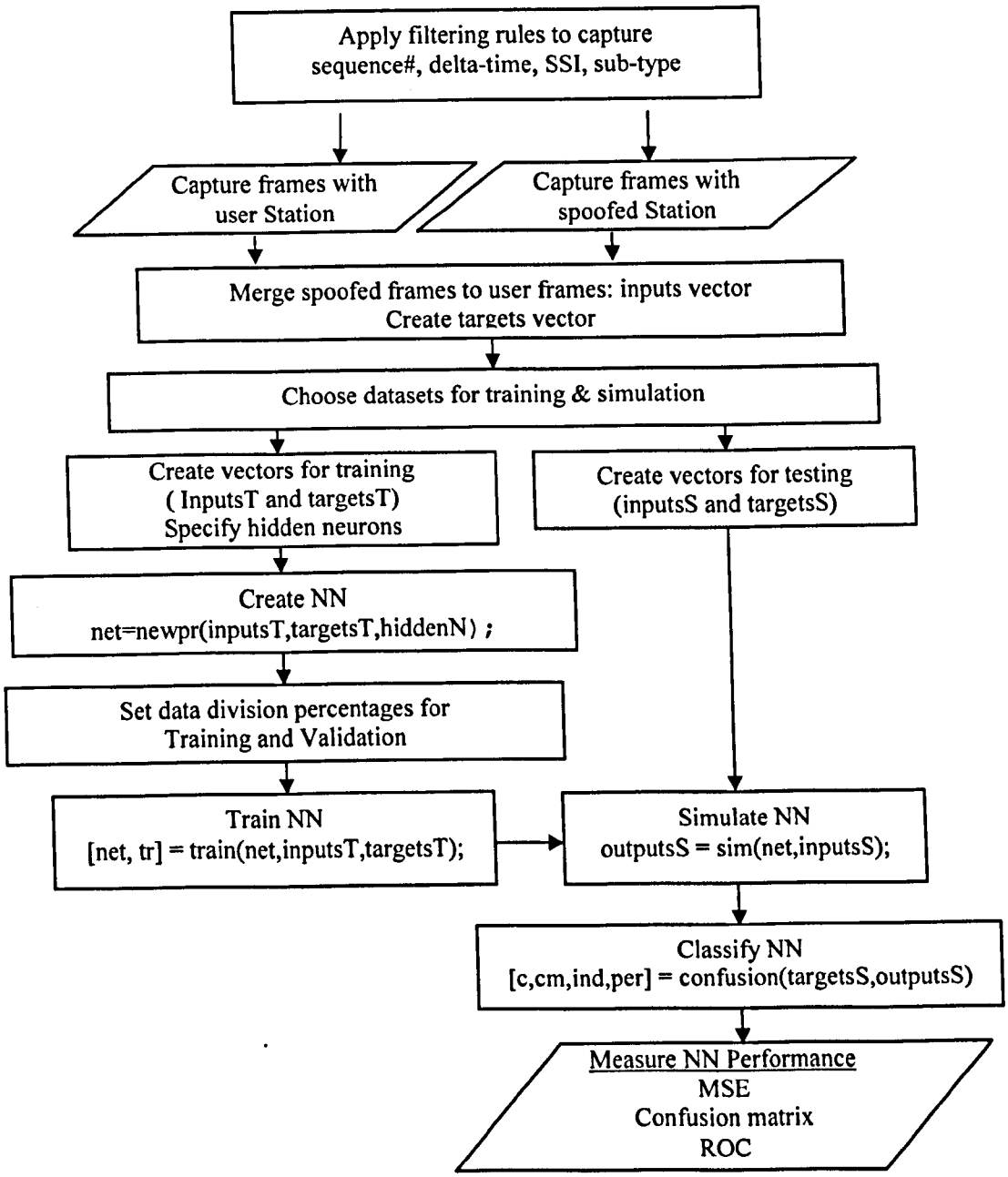


Figure 4-1: Steps of creating the NN classifier.

4.2. WLAN Data Capture and Preparation

This research employs real WLAN traffic as opposed to data from a sample database or synthetic traffic generated by a test-bed used in many studies. A wireless home network is selected for data collection. The wireless home network consists of an AP, and eight user STAs. The user STAs are combination of

Windows XP based Personal Computers (PC) and laptops. However, only one user STA and the AP are utilised for this experiment. Figure 4-2 shows the components of the WLAN relevant to the NN training and evaluation.

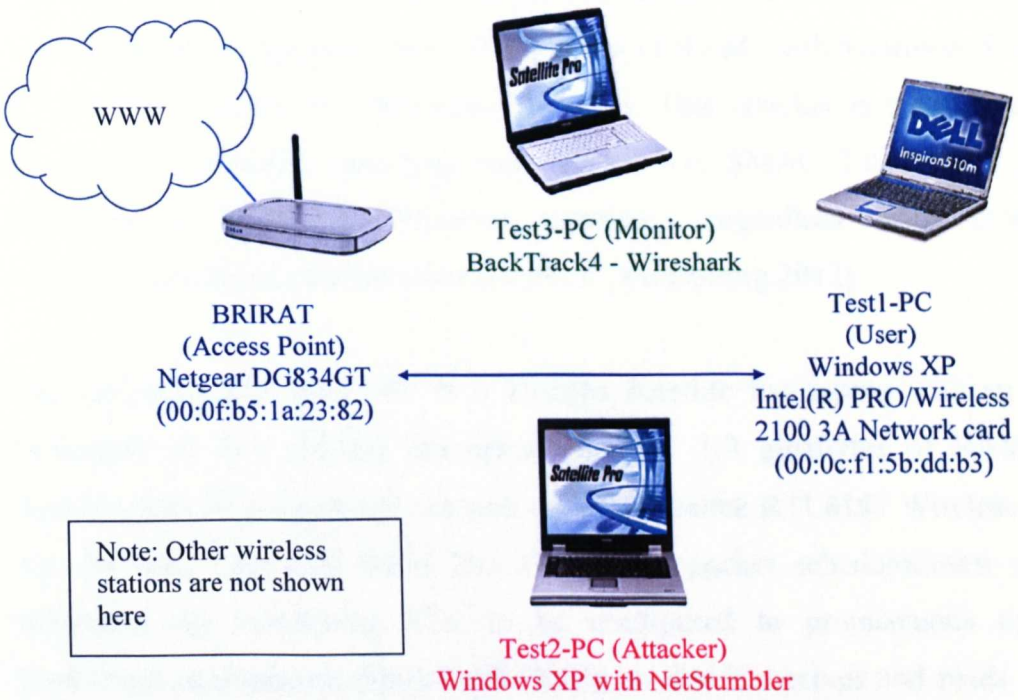


Figure 4-2: WLAN including an attacker and a network monitor.

BRIRAT AP is a Netgear DG834GT router with MAC address 00:0f:b5:1a:23:82. It is configured with WPA2-PSK enabled ACL, so that only the computers with the listed MAC addresses and network key could access the network resources. AP does not respond to computers with MAC addresses not listed in the ACL. Computers without a network key cannot associate with the AP. However, AP replies with probe and authentication responses.

The user STA Test1-PC is a DELL Inspiron 510M laptop with an Intel® Pentium® 1.6 2GHz microprocessor and 1 MB of Random Access Memory (RAM), and runs on Microsoft Windows XP OS. Test1-PC connects to AP using Intel(R) PRO/WLAN 2100 mini PCI NIC with MAC address 00:0c:f1:5b:dd:b3. Microsoft Office 2007 is the main application software. Internet Explorer (IE),

Firefox, AVG, Skype, TeamViewer are some of the other software that is been used.

Attacker Test2-PC is a Toshiba Satellite Pro laptop with an Intel® Pentium® M 740 (2GHz) microprocessor and 1.9 gigabytes of RAM, with Microsoft Windows XP. NIC is Intel(R) PRO/Wireless 2200 BG. This attacker is spoofed using a commercially available spoofing tool, SMAC 2.0. SMAC 2.0 changes MAC addresses in Microsoft Windows systems regardless of whether the manufacturers allow this option or not (KLC_Consulting 2012)

The capturing STA Test3-PC is a Toshiba Satellite Pro laptop with an Intel® Pentium® M 740 (2GHz) microprocessor and 1.9 gigabytes of RAM, with BackTrack4 OS. An external network adaptor, Realtek RTL8187 Wireless 802.11 b/g, 54 MB, Universal Serial Bus (USB) 2.0 packet scheduler/mini adaptor, facilitated the monitoring STA to be configured to promiscuous mode in BackTrack environment (BackTrack 2012), so that it receives and reads all data packets transmitted using Wireshark network analyser (Wireshark 2012). NIC in promiscuous mode does not emit any frames.

Test3-PC is kept on promiscuous mode and Wireshark network monitoring software was used for data capturing. Data capture is performed in two phases. During the first phase (phase_1), genuine frames are captured from user Test1-PC. The user is asked to note the tasks performed during a specific period. During this period, the user Test1-PC accessed internet to browse information, download software, watch a live television channel, listen to a live radio channel and check and send emails. The capturing session is coded as *Capture20* (Table 4-1). The second phase of the capture starts immediately after the first phase. During the second phase (phase_2), the Test1-PC is kept offline. Attacker Test2-PC with its spoofed MAC address is made to send a flood of probe request frames to the AP and make few network key guessing attempts. This capturing session is coded as *Capture22* (Table 4-2).

Table 4-1: User activities during the capturing period.

Starting Frame No.	Action
1	Capture Started
315	Opened IE
409	Googgled and played BBC2 Live and stopped
1901	Googgled and played BBC1 Radio Live
7721	Checked Yahoo email
21474	Sent an email
22840	downloaded a large file
153240	stopped BBC1 radio
154670	download completed
154686	closed all opened windows
154734	Disconnected from AP
154769	Scanned Network
154860	Tried to connect to the network 3 times
155363	Repaired the adaptor
155640	Opened IE
157001	Shut Down
NIL	Stopped Capture

Table 4-2: Attacker activities during the capturing period.

Starting Frame No.	Action
NIL	Capture Started
1	Attacker Started
7	Directed Probing attack started
1577	Directed Probing attack stopped
1710	Network Scanned 3x times
1977	Tried to connect to the network with a guessed network key
1846	Tried to connect to the network with a guessed network key
1871	Tried to connect to the network with a guessed network key
2223	Tried to connect to the network with a guessed network key
2223	Directed Probing attack started
18941	Directed Probing attack stopped
19148	Network Scanned 5x times
19490	Tried to connect to the network with a guessed network key
19551	Turned off the attacker
NIL	Stopped Capture

Frame numbers in Table 4-1 and Table 4-2 are estimated values derived by analysing the captured data and time stamps of recorded user actions. Therefore, there can be gaps or overlaps between each activity. Both user and attacker performed start-up and shutdown procedures, network scans, network connection

and disconnection, and NIC repair. Preliminary checks have been performed to determine that other attackers were not present during the capturing period. Capturing is restricted to IEEE 802.11 WLAN channel number 11 - 2462 MHz. Therefore, statistics of the STA's behaviour on the entire bandwidth is unavailable. A normal Wireshark capture consists of all frames that are received by the NIC of the capturing STA. Each frame contains a combination of MAC frame, radio and statistical data. Therefore, filtering rules are applied to;

- Filter all frames with source address (*wlan.sa*) 00:0c:f1:5b: dd:b3. Phase1 and 2 consisted of 157060 and 19570 frames respectively.

- Filter delta time (*frame.time_delta*), sequence number (*wlan.seq*), SSI (*radiotap.dbm_antsignal*) and frame sub-type (*wlan.fc.subtype*) of each frame.

The research then, exports *capture20* and *capture22* datasets to Comma Separated Values (CSV) files, *capture20.csv* and *capture22.csv* for further processing. The research further appends frames from phase_2 (*capture22.csv*) frames from phase_1 (*capture20.csv*) to use as the input data sample *cap20n22-input.csv.* This input sample consists of 176630 frames generated from MAC address 00:0c:f1:5b:dd:b3. Further more, research generates a target sample *cap20n22-target.csv* with relevant target values: 1 when there is a rogue frame (an attack), 0 when there is a genuine frame (no attack).

4.3. Neural Network Classifier Design and Training

In order to detect probe request attacks, a supervised NN technique is applied as defined in Chapter 3. A supervised feedforward NN with 4 input neurons (*deltaTime, sequenceNumber, signalStrength and frameSubtype*), 1 hidden layer with 20 neurons, and a output neuron which classifies genuine frames (0) from rogue frames (1) is implemented using MATLAB technical computing language version 2008b. The research instigated creating a typical NN classifier, and teaching the pattern to the NN and testing, to see if it has learned the pattern as

follows: Input (*cap20n22-input.csv*) and target (*cap20n22-target.csv)* samples are presented to NN classifier as *inputs* and *targets*. *inputs* is a 176630x4 matrix, representing 176630 samples of 4 elements. *targets* is a 176630 matrix, representing 176630 samples of 1 element. Output layer size is determined from input. As data are captured in numerical form, no conversion was needed before presenting it to NN classifier. Figure 4-3 is the graphical presentation of the two-layer feedforward NN object created using MATLAB '*newpr'* function.

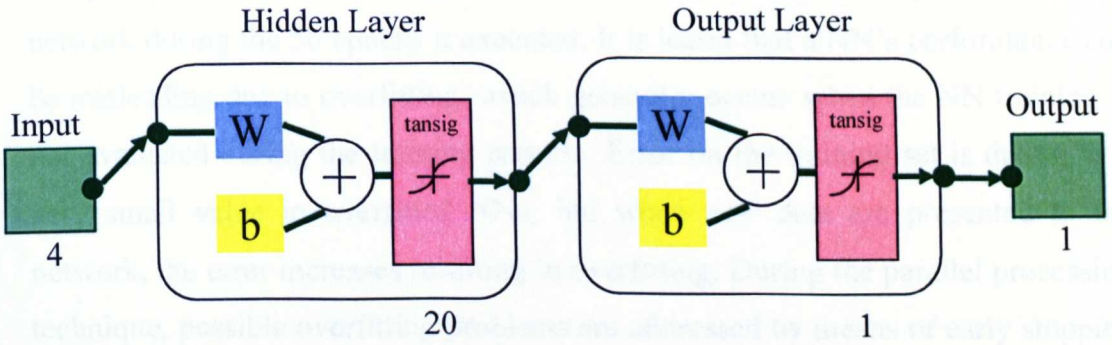


Figure 4-3: Two layer feedforward NN.

The input sample is randomly divided using '*deviderand*' function. 70% of the data are allocated to train the network and 15% each for validation and testing respectively. The network is trained using '*trainscg'* function with SCG backpropagation algorithm. It uses tan-sigmoid transfer function (tansig) in both hidden and output layers. Following training parameters given in Table 4-3 are applied.

Table 4-3: Training parameters (net.trainParam)

Parameter	Param. value	Description
net.trainParam.epochs	100	Maximum number of epochs to train
net.trainParam.goal	0	Performance goal
net.trainParam.time	inf	Maximum time to train in seconds
net.trainParam.min_grad	1e-6	Minimum performance gradient
net.trainParam.max_fail	5	Maximum validation failures

Training stops when any of the conditions in Table 4-3 arise: the maximum number of epochs is reached, the maximum amount of time is exceeded, performance is minimised to the goal, the performance gradient falls below

min_grad, validation performance is increased more than *max_fail* times since the last time it decreased. Test sample has no effect on training and so provides an independent measure of network performance during and after training.

Research applies a parallel processing technique when training and testing the classifier. Here, training, validation and testing are performed in parallel with 70%, 15%, and 15% of data for training, validation, and test respectively. During this process, all three (training, validation and test) datasets are presented to the network during the 50 epochs it executed. It is learnt that a NN's performance can be misleading due to overfitting, which generally occurs when the NN training is not evaluated during the training process. Error on the training set is driven to a very small value in overfitted NNs, but when new data are presented to the network, the error increases resulting in overfitting. During the parallel processing technique, possible overfitting problems are addressed by means of early stopping using intermediate validation during training. After every training cycle (epoch), the network is evaluated on the validation set. When the validation performance failed to improve for six consecutive epochs, the training halts (Table 4-3). In this instance, the training is stopped at 56^{th} epoch as the validation set performance failed to improve, from 50^{th} epoch, for six consecutive epochs. The network's training performance is measured by validation performance. Figure 4-4, Figure 4-5, and Figure 4-6 illustrate the NN training and validation results.

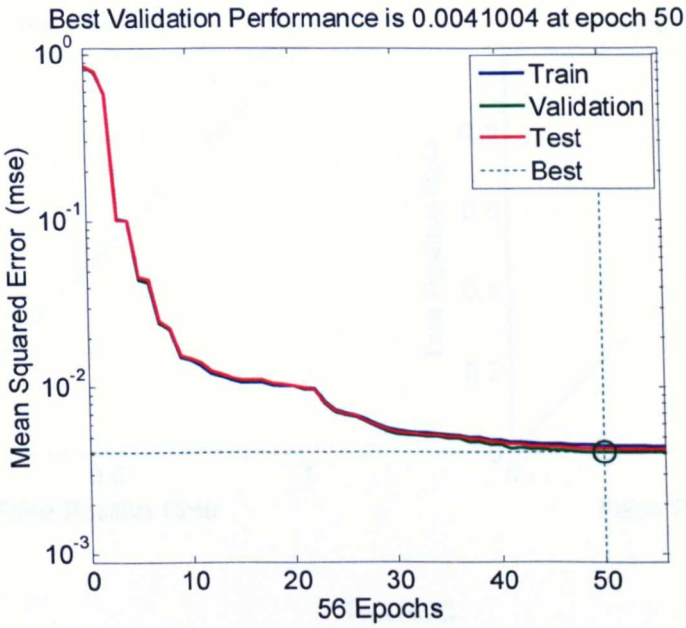


Figure 4-4: MSEs of NN training, validation and test.

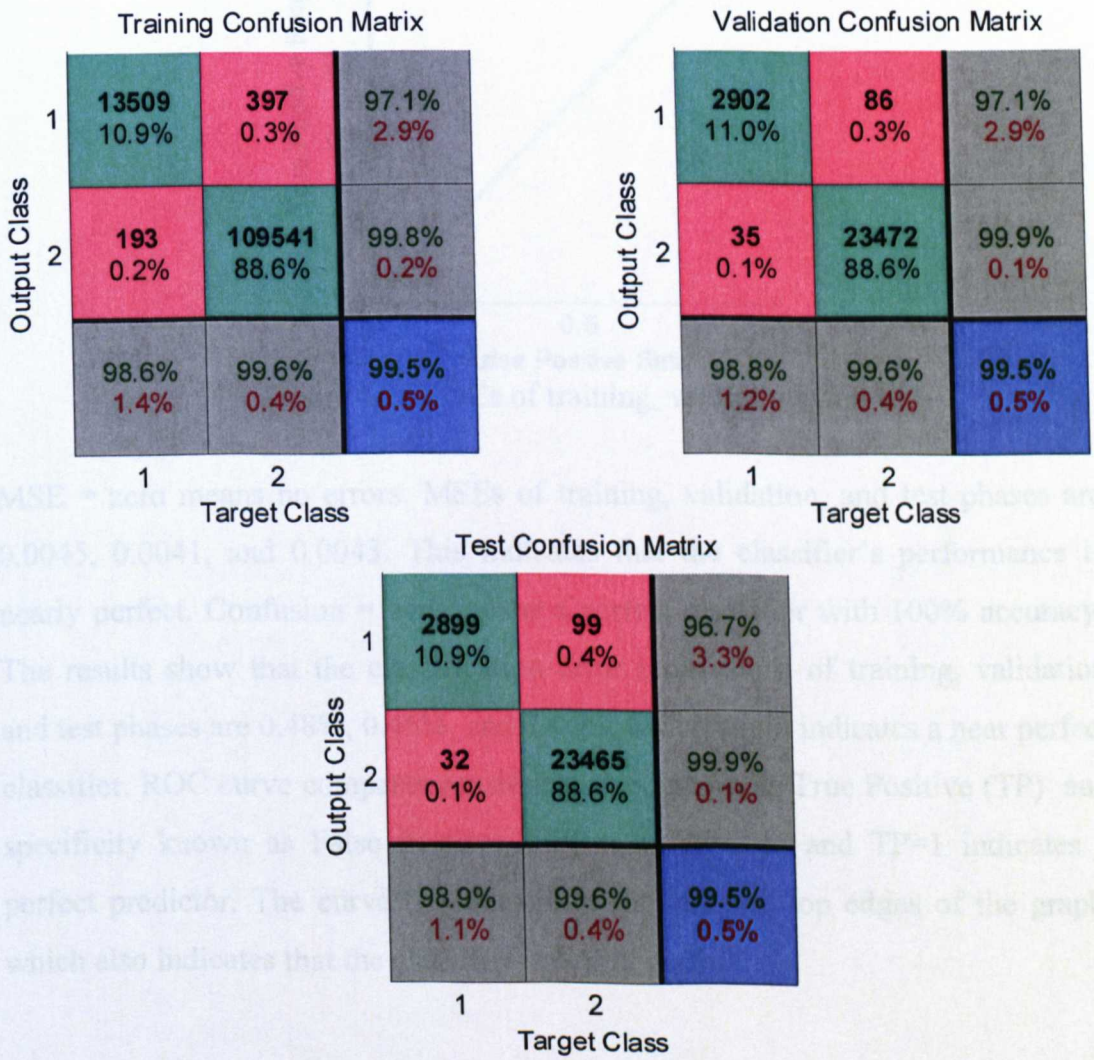


Figure 4-5: Confusion matrices of training, validation, and test.

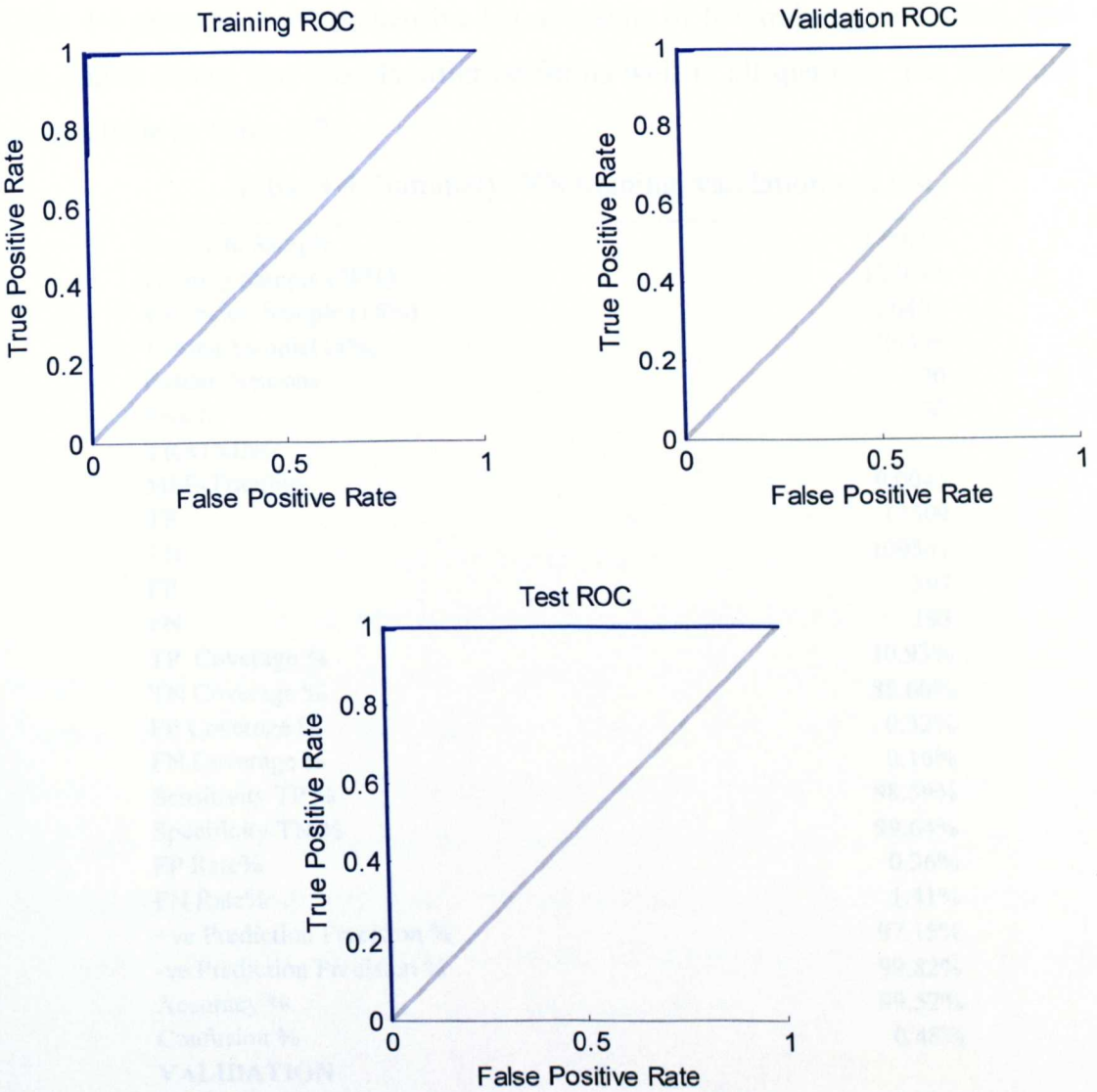


Figure 4-6: ROCs of training, validation, and test.

MSE = zero means no errors. MSEs of training, validation, and test phases are 0.0045, 0.0041, and 0.0043. This indicates that the classifier's performance is nearly perfect. Confusion = zero means a perfect classifier with 100% accuracy. The results show that the classification error (confusion) of training, validation and test phases are 0.48%, 0.46%, and 0.49%, which again indicates a near perfect classifier. ROC curve compares sensitivity also known as True Positive (TP) and specificity known as False Positive (FP) rate. FP=zero and TP=1 indicates a perfect predictor. The curve (in blue) hugs the left and top edges of the graph, which also indicates that the classifier is nearly perfect.

Table 4-4 presents a comprehensive list of results of NN training, validation, and test, which shows that this classifier performs well in all quarters. The formulas are available in Table 3-2.

Table 4-4: Summary: NN training, validation and test.

Complete Sample	176630
Training Sample (70%)	123640
Validation Sample (15%)	26495
Testing Sample(15%)	26495
Hidden Neurons	20
Epoch	50
TRAINING	
MSE-Training	0.0045
TP	13509
TN	109541
FP	397
FN	193
TP Coverage %	10.93%
TN Coverage %	88.60%
FP Coverage %	0.32%
FN Coverage %	0.16%
Sensitivity TP %	98.59%
Specificity TN %	99.64%
FP Rate%	0.36%
FN Rate%	1.41%
+ve Prediction Precision %	97.15%
-ve Prediction Precision %	99.82%
Accuracy %	99.52%
Confusion %	0.48%
VALIDATION	
MSE-Training	0.0041
TP	2902
TN	23472
FP	86
FN	35
TP Coverage %	10.95%
TN Coverage %	88.59%
FP Coverage %	0.32%
FN Coverage %	0.13%
Sensitivity TP %	98.81%
Specificity TN %	99.63%
FP Rate%	0.37%
FN Rate%	1.19%
+ve Prediction Precision %	97.12%
-ve Prediction Precision %	99.85%
Accuracy %	99.54%
Confusion %	0.46%
TEST	
MSE-Training	0.0043
TP	2899
TN	23465

FP	99
FN	32
TP Coverage %	10.94%
TN Coverage %	88.56%
FP Coverage %	0.37%
FN Coverage %	0.12%
Sensitivity TP %	98.91%
Specificity TN %	99.58%
FP Rate%	0.42%
FN Rate%	1.09%
+ve Prediction Precision %	96.70%
-ve Prediction Precision %	99.86%
Accuracy %	99.51%
Confusion %	0.49%

However, it is observed that both MSE and confusion test error are higher than the validation error. This indicates a slight sign of overfitting, which suggests that this classifier may not perform well on unseen data. Furthermore, although early stopping improves generalisation, it is not considered as a well-defined statistical method to evaluate a model's performance. Parallel processing method is a quick and easy way of obtaining a considerably reliable result. However, self-consistency and cross-validation methods are considered systematic evaluation methods in academic research. Therefore, self-consistency and cross-validation tests are conducted to evaluate the model systematically.

4.4. Systematic Evaluation of NN Classifier

Self-consistency and cross-validation are among several methods of estimating how well a trained model will perform in the future. Self-consistency is a method to evaluate the model's performance with seen data. In self-consistency test, the same complete dataset is utilised for training phase (training and validation). Then, the complete dataset is utilised to test the model. The result gives the measure of fitness of data in the model. This also does not require much computation because training, validation and testing are executed only once. However, it can generate a high accuracy rate, as the same dataset is utilised for training and validation, but may not perform well with unseen data. (Kumar, Gromiha & Raghava 2007). Alternatively, k-fold cross-validation methodology

evaluates the model with unseen data. It also minimises the bias present within the random sampling of the data. Here, the original sample is partitioned into k sub-samples. During the evaluation process, the model is trained k times, using k-1 samples and tested with the retained k sample. Some literature suggest that results from each fold to be averaged to generate a single estimation of performance or fitting value, whilst other literature suggest that the best performance to be considered as the fitting value and respective model to be utilised for simulation. 10-fold cross-validation is most commonly used to reduce the wastage of data in circumstances where there is a limited set of data (Moore 2001). The next two Sections discuss the self-consistency and cross-validation test processes and analyse results.

4.4.1. Self-Consistency Test

For training phase, the complete dataset, i.e. *inputs,* the 176630x4 matrix, representing 176630 samples of 4 elements and *targets,* the 176630 matrix, representing 176630 samples of 1 element are used. The dataset is divided randomly, 70% for training and 30% for validation. Other configurations remained same. The results generated by self-consistency training and validation are shown in Figure 4-7, Figure 4-8 and Figure 4-9.

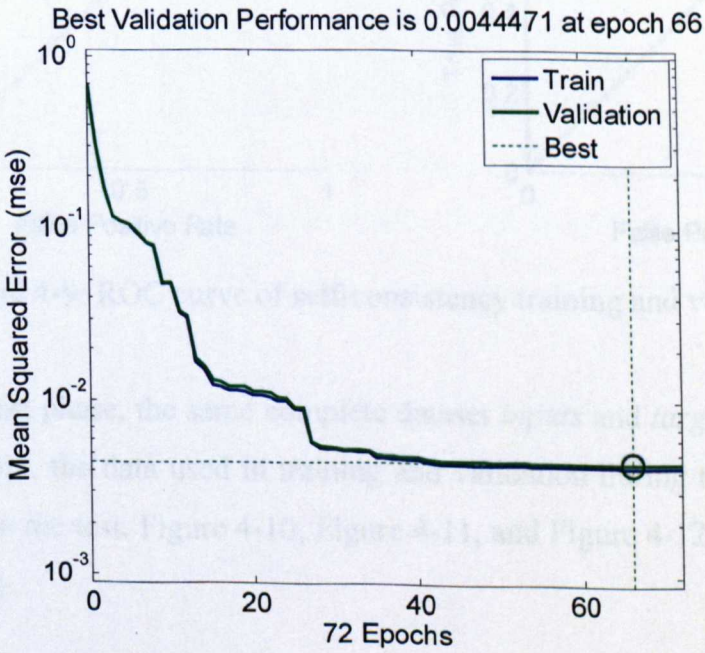


Figure 4-7: MSEs of self-consistency training and validation.

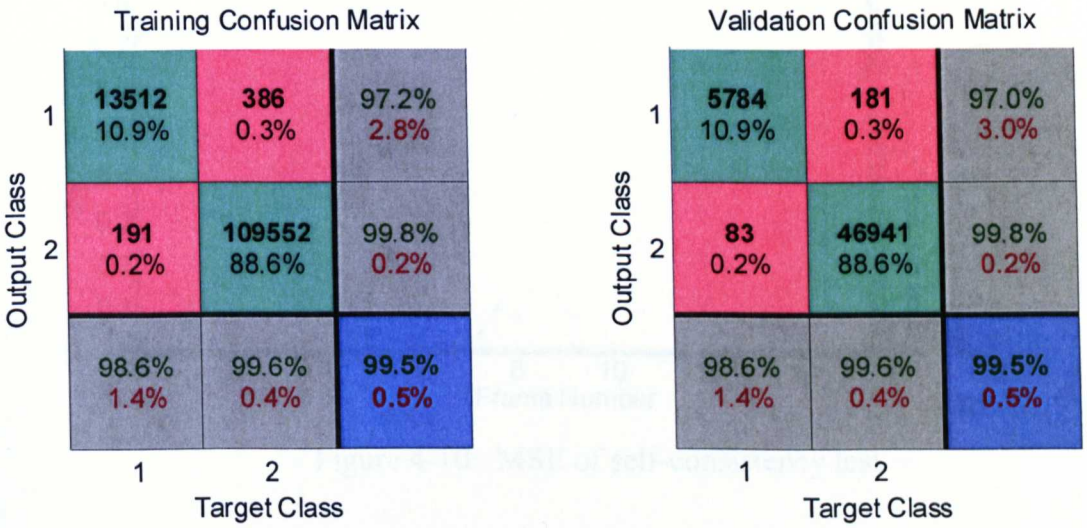


Figure 4-8: Confusion matrices of self-consistency training and validation.

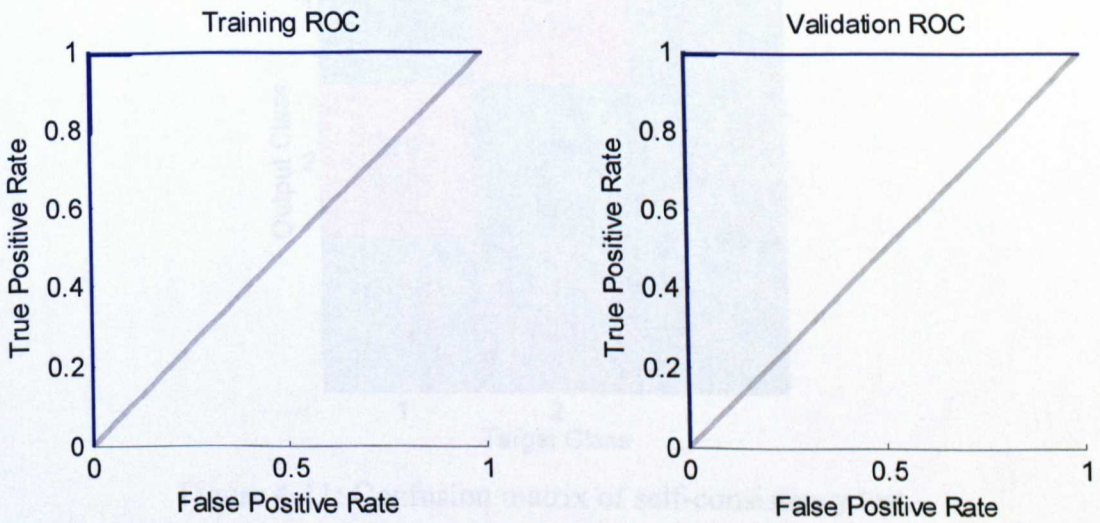


Figure 4-9: ROC curve of self-consistency training and validation.

During the test phase, the same complete dataset *inputs* and *targets*, are fed to the NN. Therefore, the data used in training and validation during the training phase are re-used in the test. Figure 4-10, Figure 4-11, and Figure 4-12 illustrate the NN test summary.

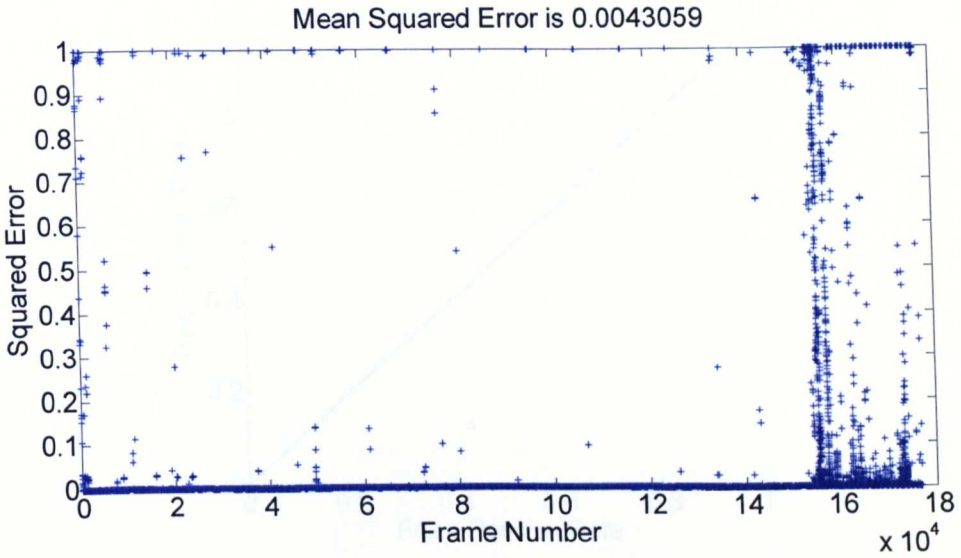


Figure 4-10: MSE of self-consistency test

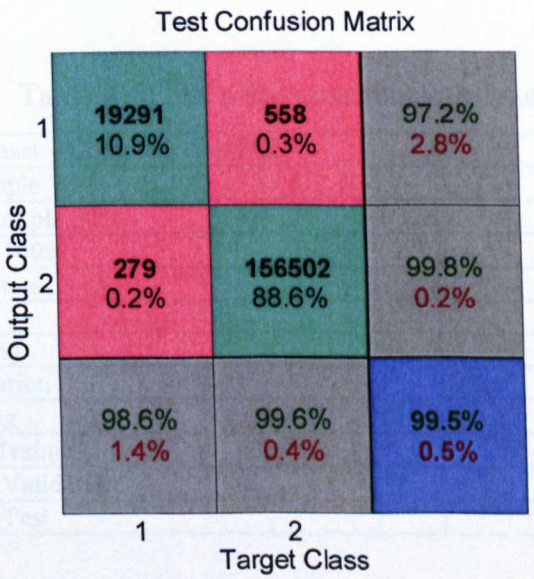


Figure 4-11: Confusion matrix of self-consistency test.

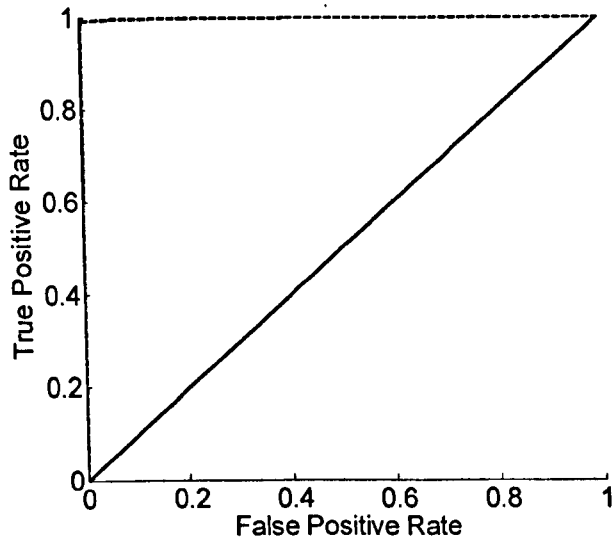


Figure 4-12: ROC curve of self-consistency test.

Table 4-5 summarises the NN self-consistency test results.

Table 4-5: NN self-consistency test summary.

Training Dataset – Complete (FoldAll)	176630 frames
Training Sample 70%	123640frames
Validation Sample 30%	52989 frames
Test Sample 100%	176630 frames
Hidden Neurons	20
Best Epoch	66
MSE-Training	0.0043
MSE- Validation	0.0045
MSE- Testing	0.0043
Confusion -Training	0.47%
Confusion - Validation	0.50%
Confusion - Test	0.47%

The objective of self-consistency test is to assess the fitness of the data in the classifier by testing the classifier against training data it is trained with.
Table 4-5 shows promising results, which indicates that the data is fitting the classifier extremely well. However, this classifier may not produce good results with unseen data. Further analysis of these results are presented in Section 4.4.3.

4.4.2. K-Fold Cross Validation Test

A systematic data division is performed to ensure that all data segments have both genuine and rogue frames. The frames of capture phase_1 (genuine) and capture phase_2 (rogue) are divided into 5 equal segments and combined as shown in (Table 4-6), for the 5-fold cross-validation.

Table 4-6: Data segmentation method.

Data Segments(k+y)	Genuine frames from Normal user (Phase1) : k		Rogue frames from Attacker (Phase2) : y		Running Total
	Start	End	Start	End	
k1+y1	0	31412	0	3914	35326
k2+y2	31413	62824	3915	7828	70652
k3+y3	62825	94236	7829	11742	105978
k4+y4	94237	125648	11743	15656	141304
k5+y5	125649	157060	15657	19570	176630

The cross-validation process is repeated 5 times. Each of the 5 sub-samples is used only once as the validation data, i.e. each time, a single sub-sample is retained to test the model, whilst using the remaining 4 sub-samples as training data (Table 4-7).

Table 4-7: 5-fold NN training data.

Description	Fold1	Fold2	Fold3	Fold4	Fold5
Training Data Segments (k+y)	1,2,3,4	2,3,4,5	3,4,5,1	4,5,1,2	5,1,2,3
Training Sample 70%	98913	98913	98913	98913	98913
Validation Sample 15%	42391	42391	42391	42392	42391

In the following, the research presents the training and validation, and test results of each fold. A comprehensive analysis of results of self-consistency and k-fold test is presented in Section 4.4.3

4.4.2.1. Fold1

For Fold1 training phase, data segments 1-4 are utilised with 70% for training and 30% for validation. Figure 4-13, Figure 4-14, and Figure 4-15 show MSEs, confusion matrices and ROC curve of Fold1 training and validation.

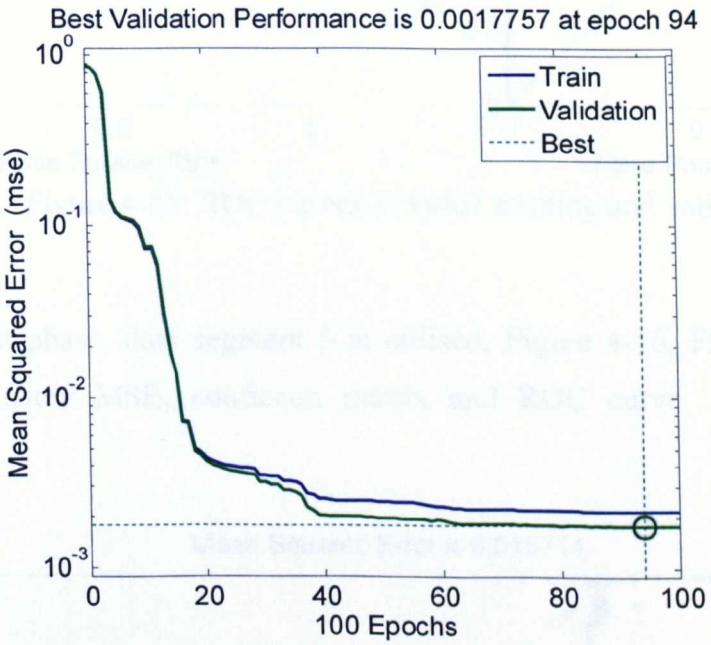


Figure 4-13: MSEs of Fold1 training and validation.

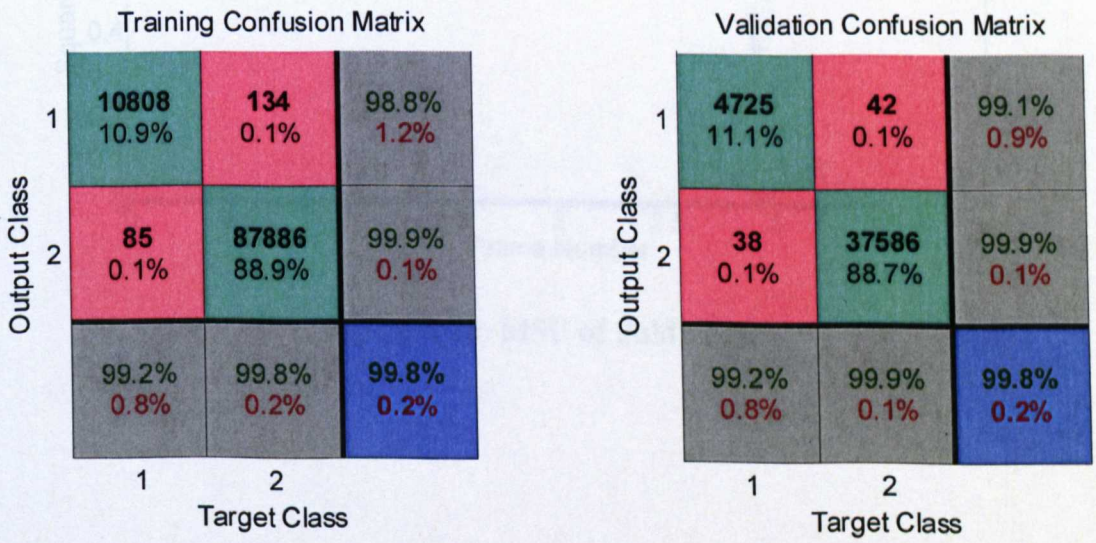


Figure 4-14: Confusion matrices of Fold1 training and validation.

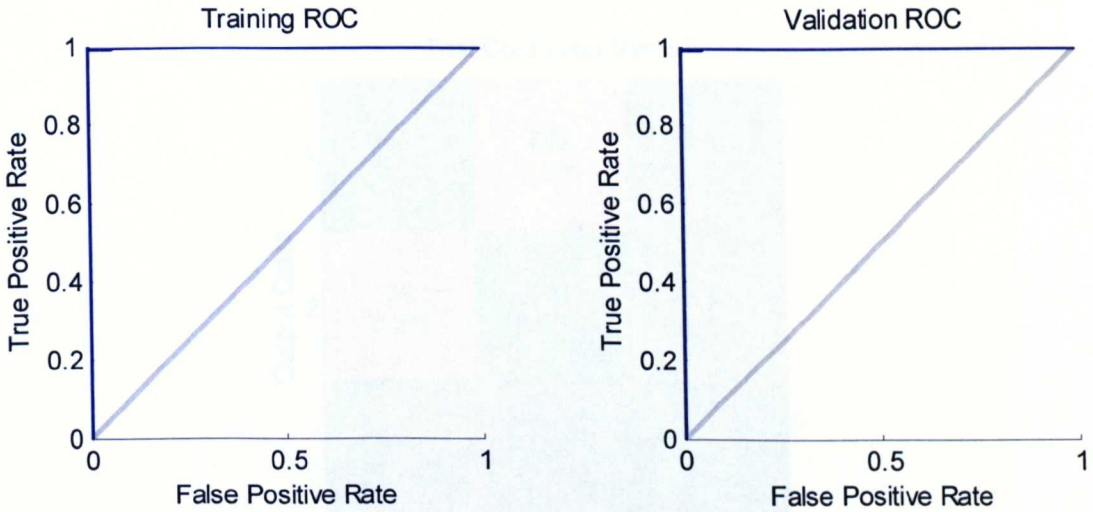


Figure 4-15: ROC curves of Fold1 training and validation.

For Fold1 test phase, data segment 5 is utilised. Figure 4-16, Figure 4-17, and Figure 4-18 show MSE, confusion matrix and ROC curve of Fold1 test, respectively.

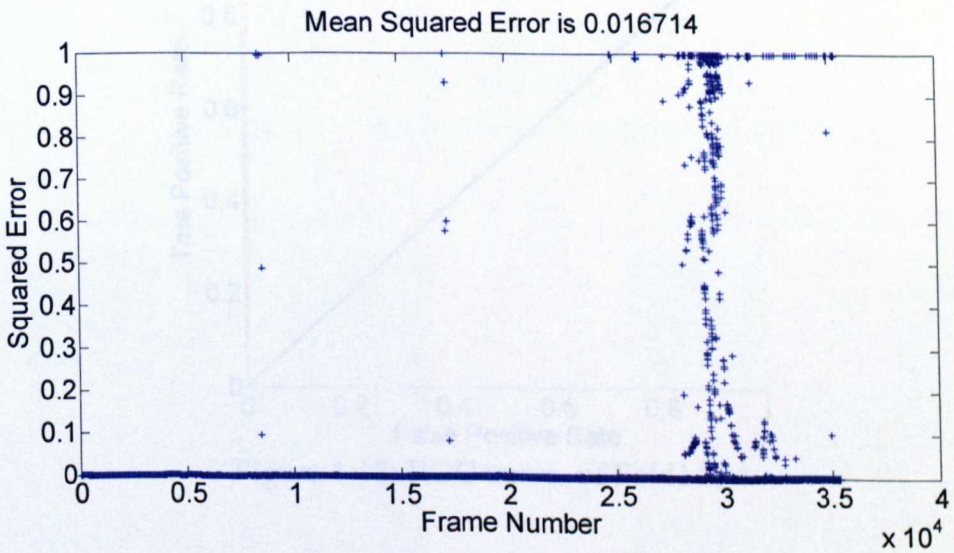


Figure 4-16: MSE of Fold1 test.

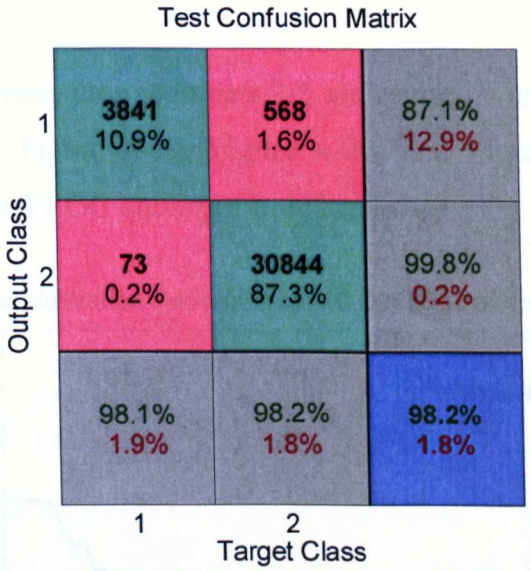


Figure 4-17: Confusion matrix of Fold1 test.

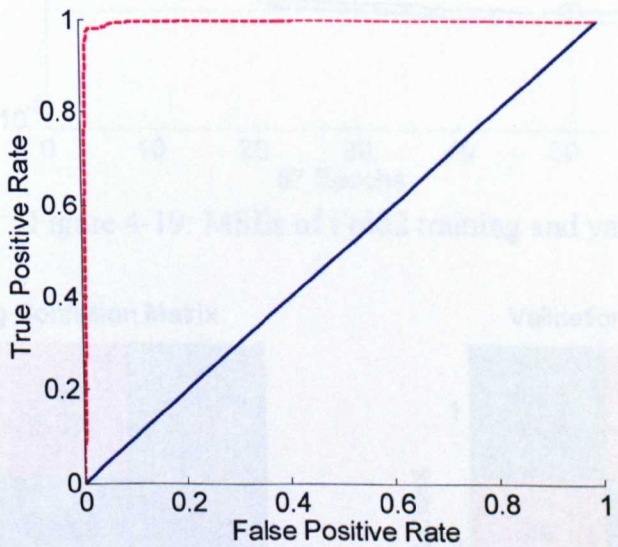


Figure 4-18: ROC curve of Fold1 test.

4.4.2.2. Fold2

For Fold2 training phase, data segments 2-5 are utilised with 70% for training and 30% for validation. Figure 4-19, Figure 4-20, and Figure 4-21 show MSEs, confusion matrices and ROC curves of Fold2 training.

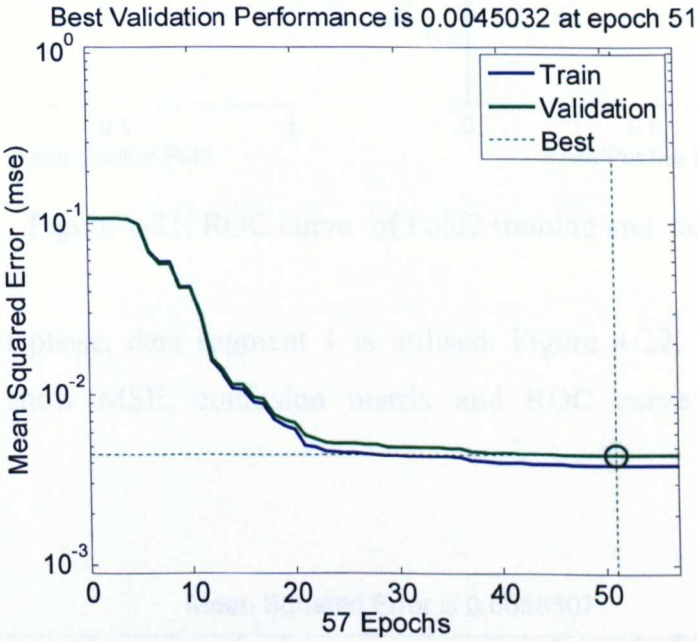


Figure 4-19: MSEs of Fold2 training and validation.

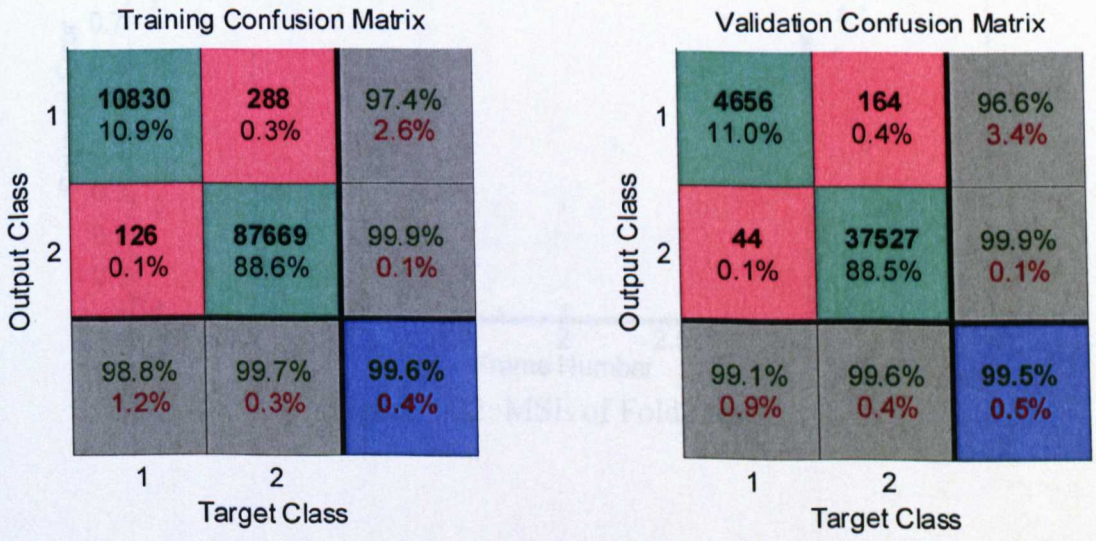


Figure 4-20: Confusion of Fold2 training and validation.

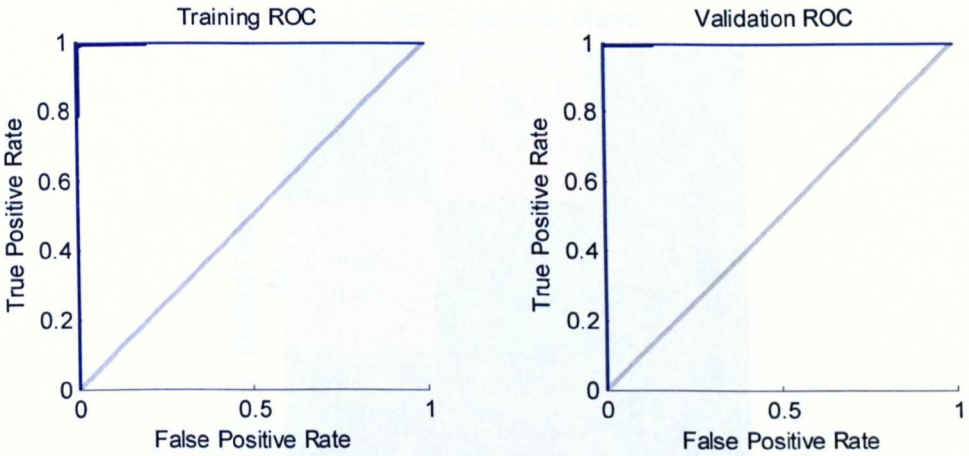


Figure 4-21: ROC curve of Fold2 training and validation.

For Fold2 test phase, data segment 1 is utilised. Figure 4-22, Figure 4-23 and Figure 4-24 show MSE, confusion matrix and ROC curve of Fold2 test respectively.

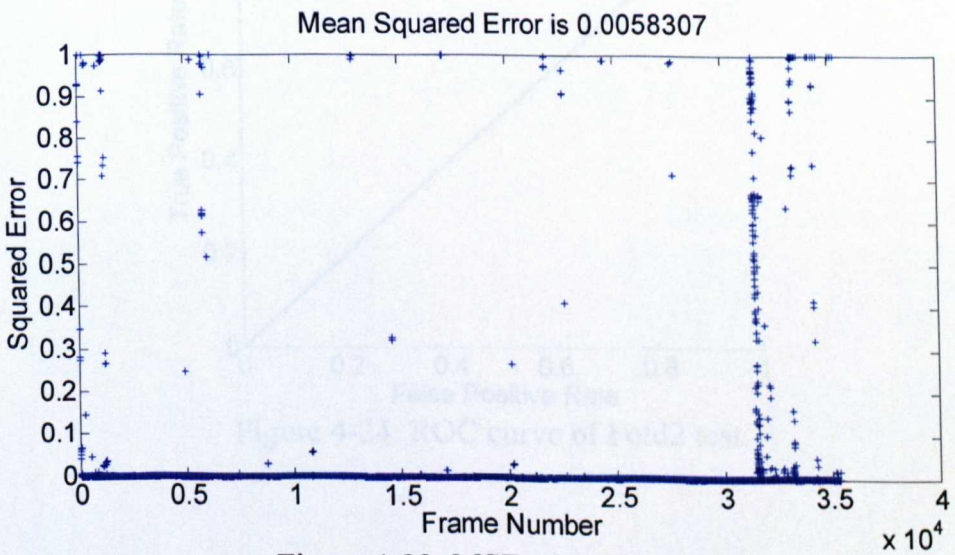


Figure 4-22: MSE of Fold2 test.

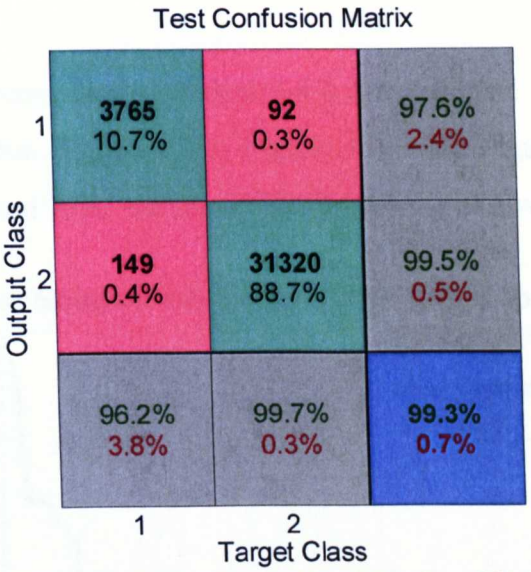


Figure 4-23: Confusion Matrix of Fold2 test.

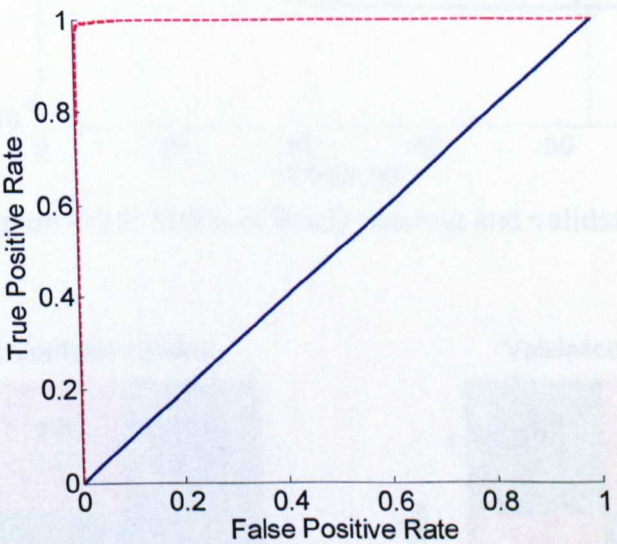


Figure 4-24: ROC curve of Fold2 test.

4.4.2.3. Fold3

For Fold3 training phase, data segments 3,4,5,1 are utilised with 70% for training and 30% for validation. Figure 4-25, Figure 4-26 and Figure 4-27 show MSEs, confusion matrices, and ROC curves of Fold3 training and validation.

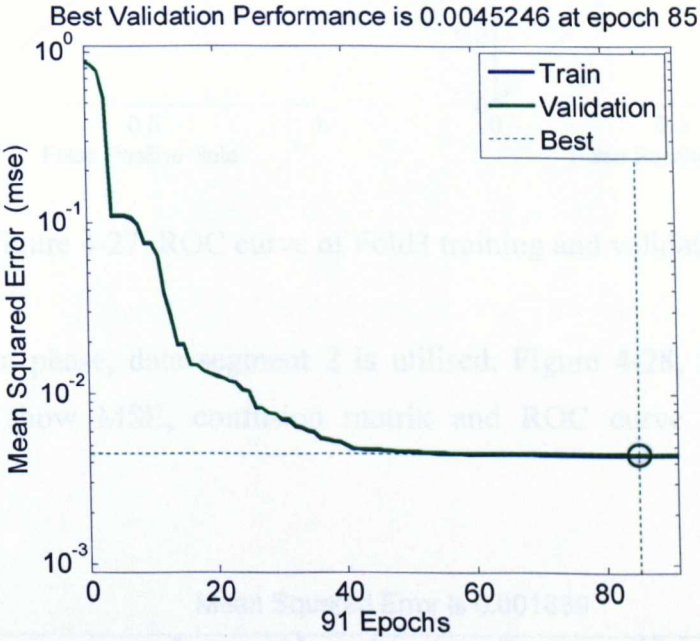


Figure 4-25: MSEs of Fold3 training and validation.

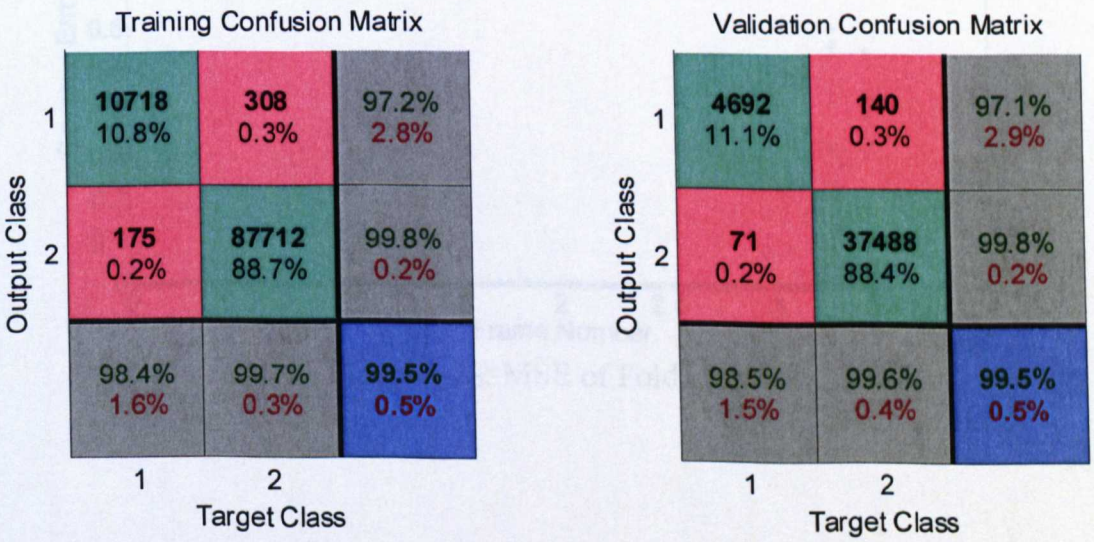


Figure 4-26: Confusion matrix of Fold3 training and validation.

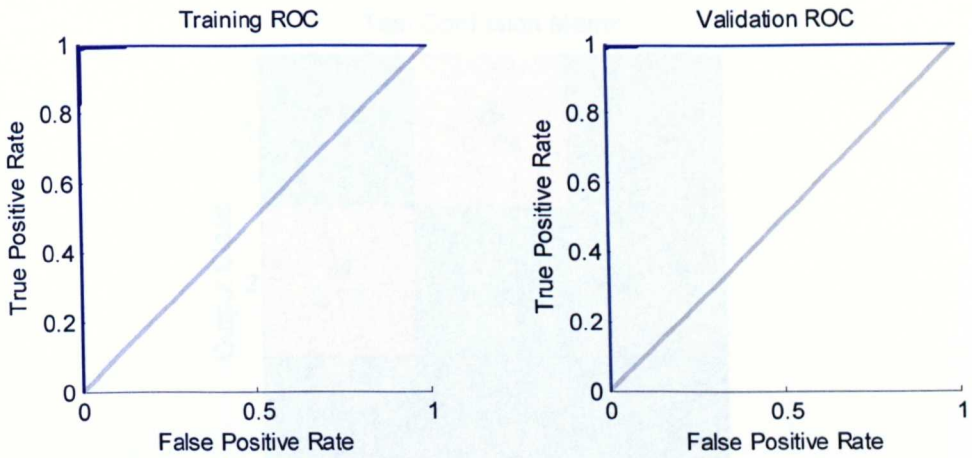


Figure 4-27: ROC curve of Fold3 training and validation.

For Fold3 test phase, data segment 2 is utilised. Figure 4-28, Figure 4-29 and Figure 4-30 show MSE, confusion matrix and ROC curve of Fold3 test, respectively.

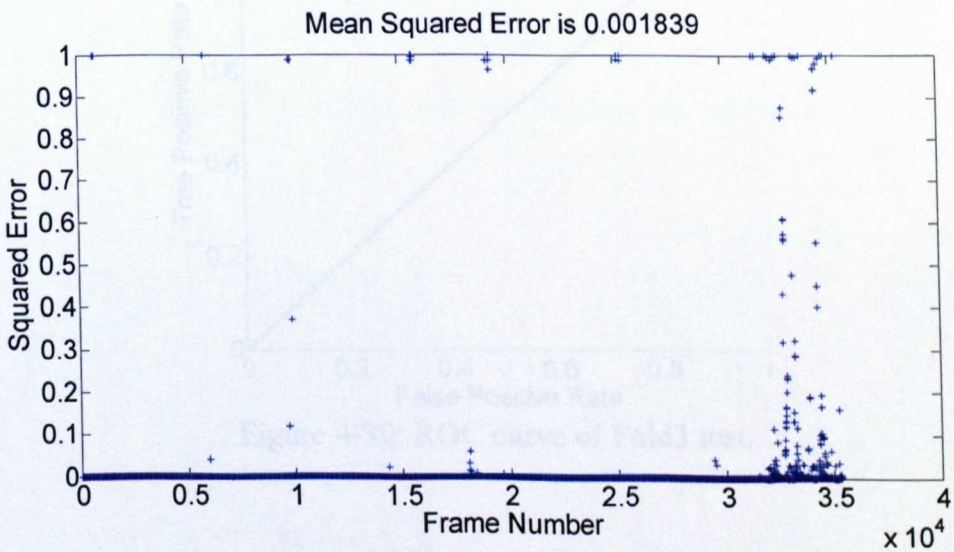


Figure 4-28: MSE of Fold3 test.

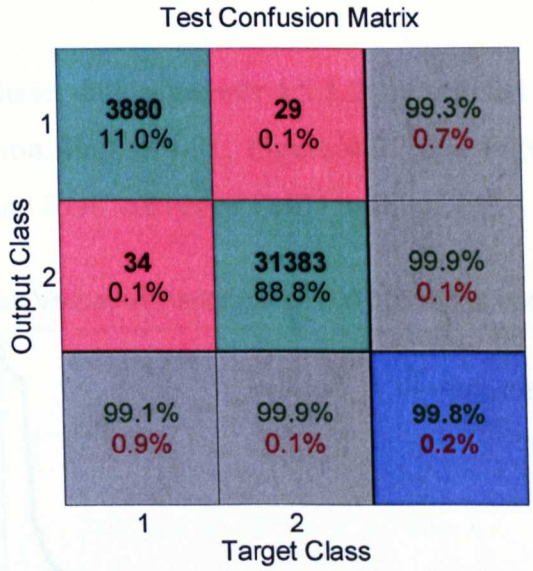


Figure 4-29: Confusion matrix of Fold3 test.

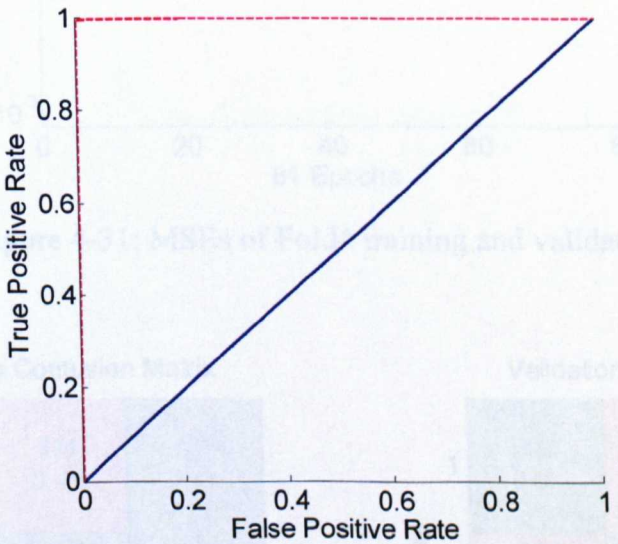


Figure 4-30: ROC curve of Fold3 test.

4.4.2.4. Fold4

For Fold4 training phase, data segments 4,5,1,2 are utilised with 70% for training and 30% for validation. Figure 4-31, Figure 4-32 and Figure 4-33 show MSEs, confusion matrices and ROC curves of Fold4 training.

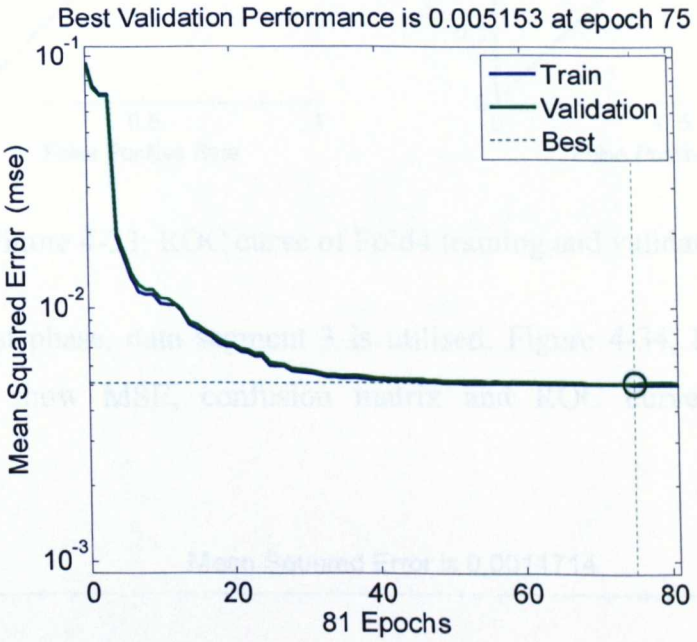


Figure 4-31: MSEs of Fold4 training and validation.

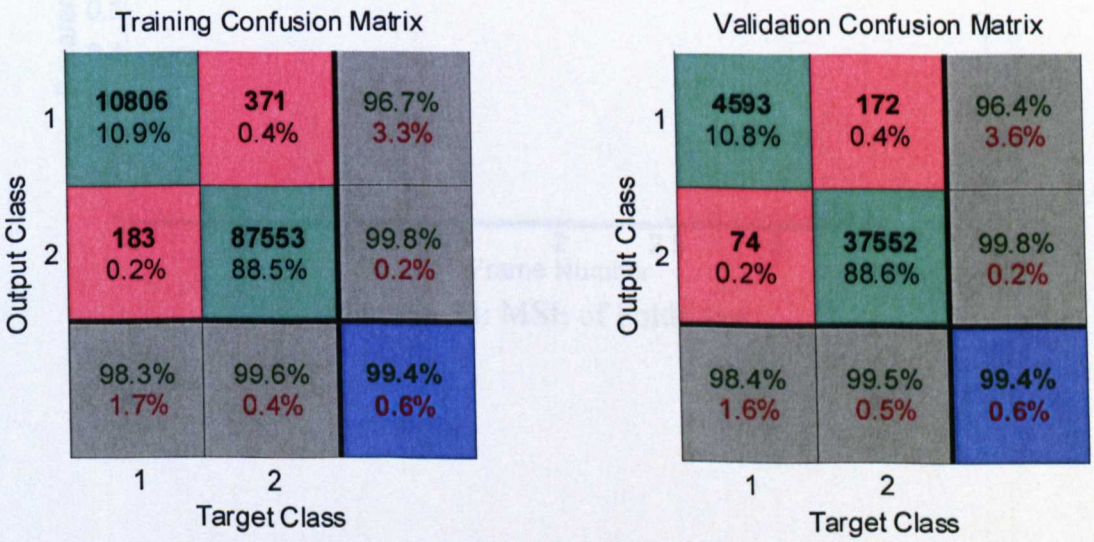


Figure 4-32: Confusion matrix of Fold4 training and validation.

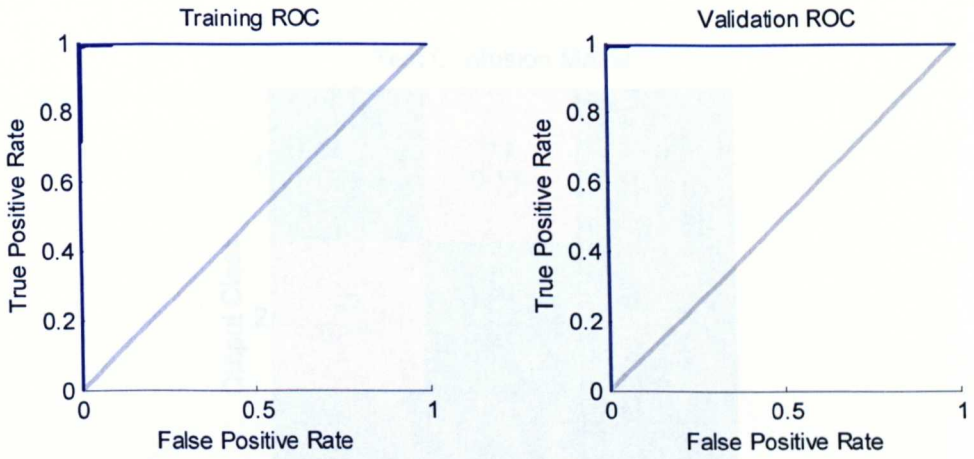


Figure 4-33: ROC curve of Fold4 training and validation.

For Fold4 test phase, data segment 3 is utilised. Figure 4-34, Figure 4-35, and Figure 4-36 show MSE, confusion matrix and ROC curve of Fold4 test, respectively.

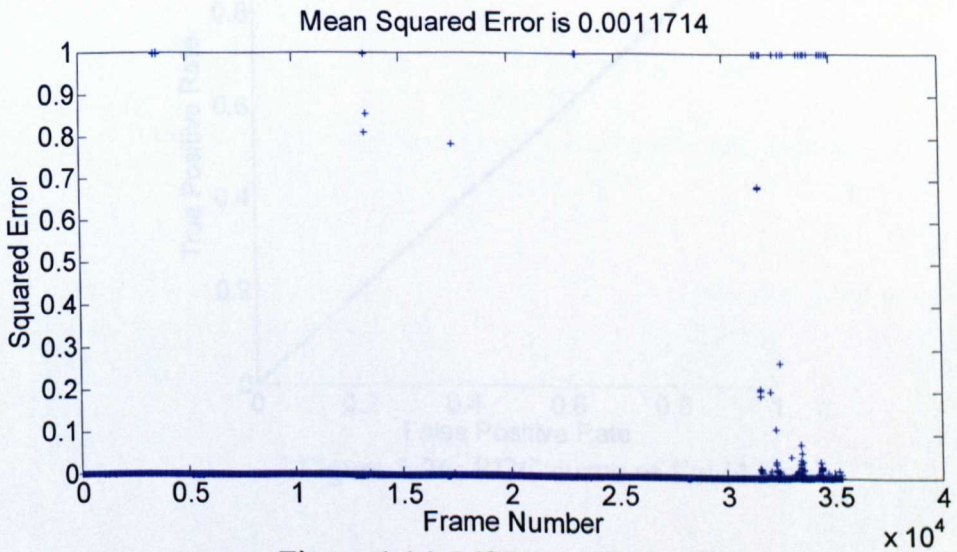


Figure 4-34: MSE of Fold4 test.

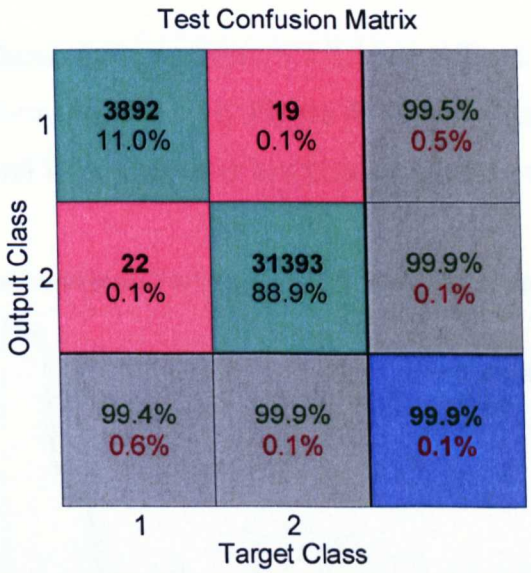


Figure 4-35: Confusion matrix of Fold4 test.

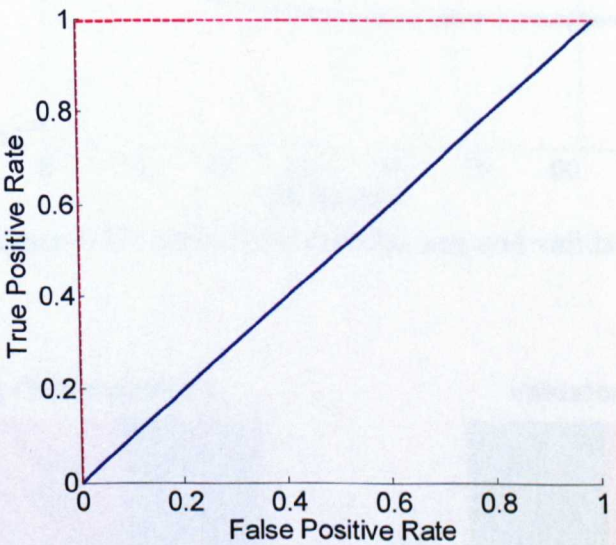


Figure 4-36: ROC curve of Fold4 test.

4.4.2.5. Fold5

For Fold5 training phase, data segments 5,1,2,3 are utilised with 70% for training and 30% for validation. Figure 4-37, Figure 4-38 and Figure 4-39 show MSEs, confusion matrices and ROC curves of Fold5 training and validation.

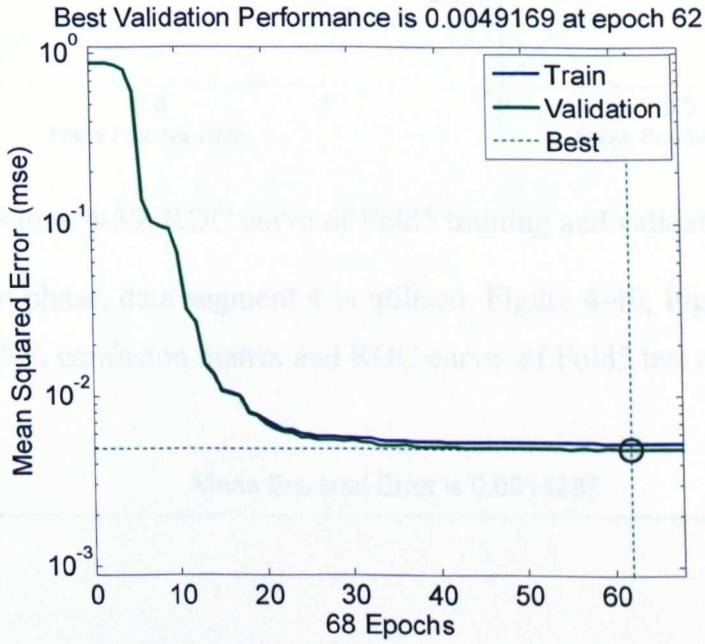


Figure 4-37: MSEs of Fold5 training and validation.

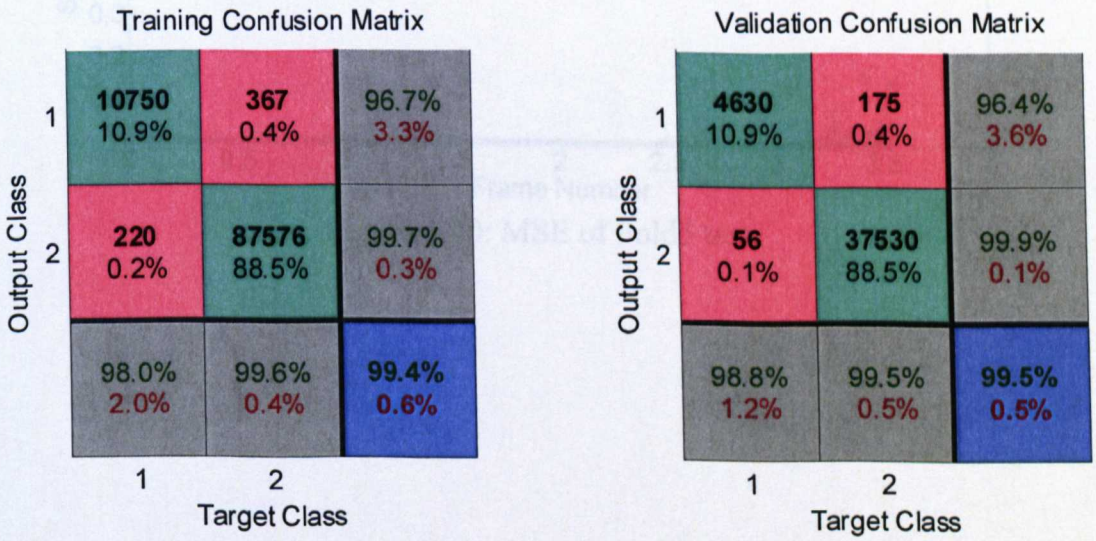


Figure 4-38: Confusion matrix of Fold5 training and validation.

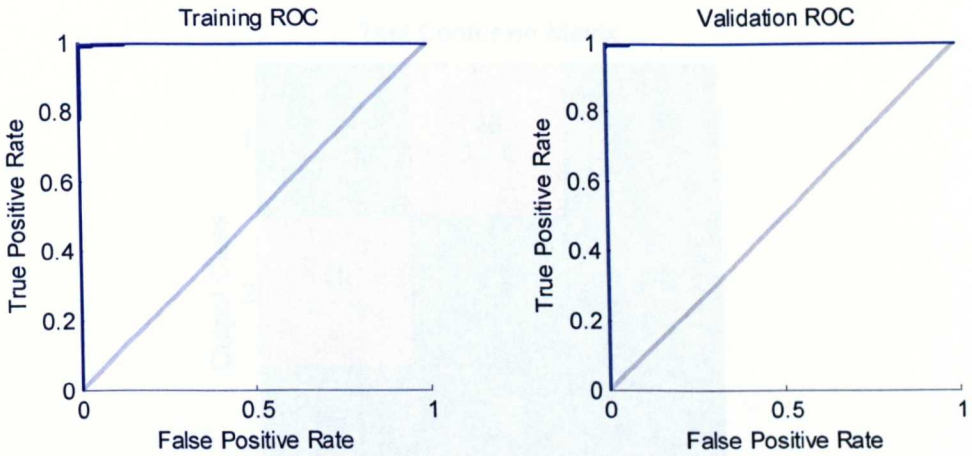


Figure 4-39: ROC curve of Fold5 training and validation.

For Fold5 test phase, data segment 4 is utilised. Figure 4-40, Figure 4-41, Figure 4-42 show MSE, confusion matrix and ROC curve of Fold5 test respectively.

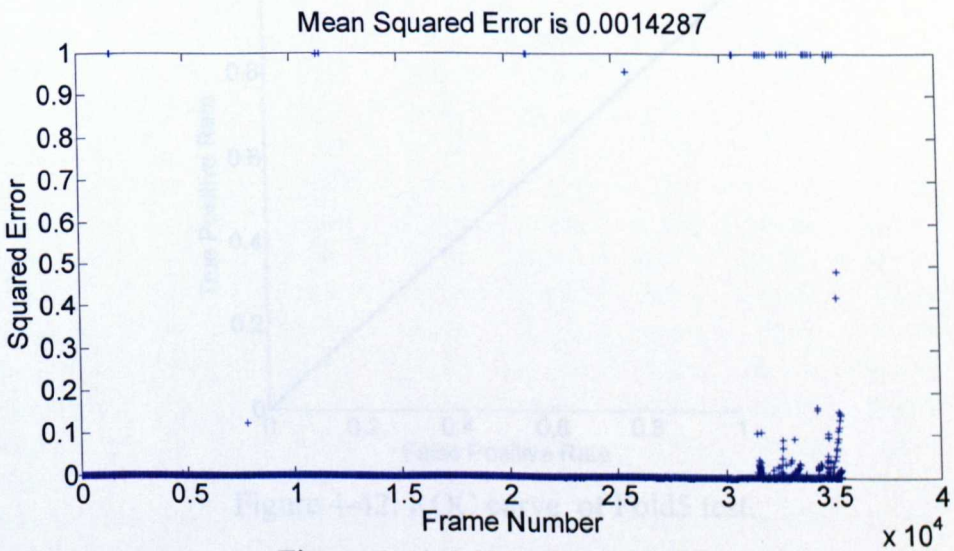


Figure 4-40: MSE of Fold5 test.

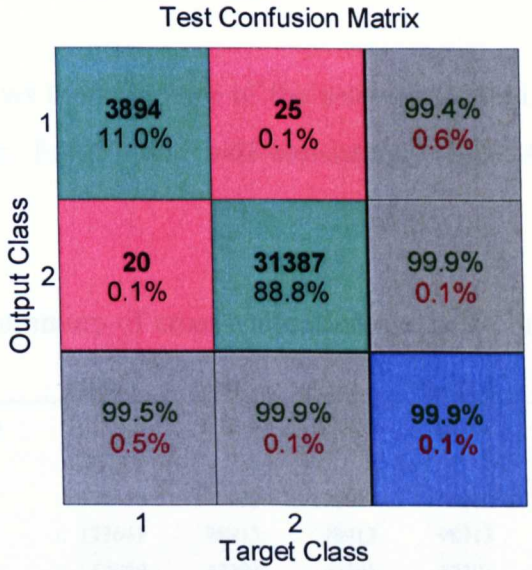


Figure 4-41: Confusion matrix of Fold5 test.

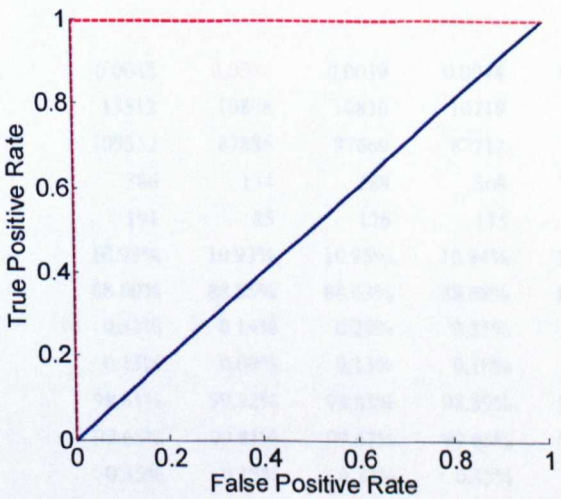


Figure 4-42: ROC curve of Fold5 test.

4.4.3. Analysis of results

Table 4-8 below shows the summary of the tests performed, namely 5-fold cross-validation (Fold1 to Fold5) and self-consistency (FoldAll) tests and results obtained.

Table 4-8: Summary of cross-validation and self-consistency tests.

	FoldAll	Fold1	Fold2	Fold3	Fold4	Fold5
Training Data Segments (ky)	1,2,3,4,5	1,2,3,4	2,3,4,5	3,4,5,1	4,5,1,2	5,1,2,3
Test Data Segment	1,2,3,4,5	5	1	2	3	4
Complete Sample	176630	176630	176630	176630	176630	176630
Training Sample	123641	98913	98913	98913	98913	98913
Validation Sample	52989	42391	42391	42391	42391	42391
Testing Sample	176630	35326	35326	35326	35326	35326
Hidden Neurons	20	20	20	20	20	20
Epoch	66	94	51	84	75	62
TRAINING						
MSE-Training	0.0043	0.0022	0.0039	0.0044	0.0051	0.0053
TP	13512	10808	10830	10718	10806	10750
TN	109552	87886	87669	87712	87553	87576
FP	386	134	288	308	371	367
FN	191	85	126	175	183	220
TP Coverage %	10.93%	10.93%	10.95%	10.84%	10.92%	10.87%
TN Coverage %	88.60%	88.85%	88.63%	88.68%	88.52%	88.54%
FP Coverage %	0.31%	0.14%	0.29%	0.31%	0.38%	0.37%
FN Coverage %	0.15%	0.09%	0.13%	0.18%	0.19%	0.22%
Sensitivity TP %	98.61%	99.22%	98.85%	98.39%	98.33%	97.99%
Specificity TN %	99.65%	99.85%	99.67%	99.65%	99.58%	99.58%
FP Rate%	0.35%	0.15%	0.33%	0.35%	0.42%	0.42%
FN Rate%	1.39%	0.78%	1.15%	1.61%	1.67%	2.01%
+ve Prediction Precision %	97.22%	98.78%	97.41%	97.21%	96.68%	96.70%
-ve Prediction Precision %	99.83%	99.90%	99.86%	99.80%	99.79%	99.75%
Accuracy %	99.53%	99.78%	99.58%	99.51%	99.44%	99.41%
Confusion %	0.47%	0.22%	0.42%	0.49%	0.56%	0.59%
VALIDATION						
MSE-Validation	0.0045	0.0018	0.0045	0.0045	0.0052	0.0049
TP	5784	4725	4656	4692	4593	4630
TN	46941	37586	37527	37488	37552	37530
FP	181	42	164	140	172	175
FN	83	38	44	71	74	56
TP Coverage %	10.92%	11.15%	10.98%	11.07%	10.83%	10.92%
TN Coverage %	88.59%	88.67%	88.53%	88.43%	88.58%	88.53%
FP Coverage %	0.34%	0.10%	0.39%	0.33%	0.41%	0.41%
FN Coverage %	0.16%	0.09%	0.10%	0.17%	0.17%	0.13%

Sensitivity TP %	98.59%	99.20%	99.06%	98.51%	98.41%	98.80%
Specificity TN %	99.62%	99.89%	99.56%	99.63%	99.54%	99.54%
FP Rate%	0.38%	0.11%	0.44%	0.37%	0.46%	0.46%
FN Rate%	1.41%	0.80%	0.94%	1.49%	1.59%	1.20%
+ve Prediction Precision %	96.97%	99.12%	96.60%	97.10%	96.39%	96.36%
-ve Prediction Precision %	99.82%	99.90%	99.88%	99.81%	99.80%	99.85%
Accuracy %	99.50%	99.81%	99.51%	99.50%	99.42%	99.46%
Confusion %	0.50%	0.19%	0.49%	0.50%	0.58%	0.54%
TEST						
MSE-Test	0.0043	0.0167	0.0058	0.0018	0.0012	0.0014
TP	19291	3841	3765	3880	3892	3894
TN	156502	30844	31320	31383	31393	31387
FP	558	568	92	29	19	25
FN	279	73	149	34	22	20
TP Coverage %	10.92%	10.87%	10.66%	10.98%	11.02%	11.02%
TN Coverage %	88.60%	87.31%	88.66%	88.84%	88.87%	88.85%
FP Coverage %	0.32%	1.61%	0.26%	0.08%	0.05%	0.07%
FN Coverage %	0.16%	0.21%	0.42%	0.10%	0.06%	0.06%
Sensitivity TP %	98.57%	98.13%	96.19%	99.13%	99.44%	99.49%
Specificity TN %	99.64%	98.19%	99.71%	99.91%	99.94%	99.92%
FP Rate%	0.36%	1.81%	0.29%	0.09%	0.06%	0.08%
FN Rate%	1.43%	1.87%	3.81%	0.87%	0.56%	0.51%
+ve Prediction Precision %	97.19%	87.12%	97.61%	99.26%	99.51%	99.36%
-ve Prediction Precision %	99.82%	99.76%	99.53%	99.89%	99.93%	99.94%
Accuracy %	99.53%	98.19%	99.32%	99.82%	99.88%	99.87%
Confusion %	0.47%	1.81%	0.68%	0.18%	0.12%	0.13%

Legend: *Lowest Value* *Highest Value*

Table 4-8 shows that the results are very much similar in every test. However, to understand and analyse the behaviours of MSE readings of cross validation and self-consistency test, Figure 4-43 is produced.

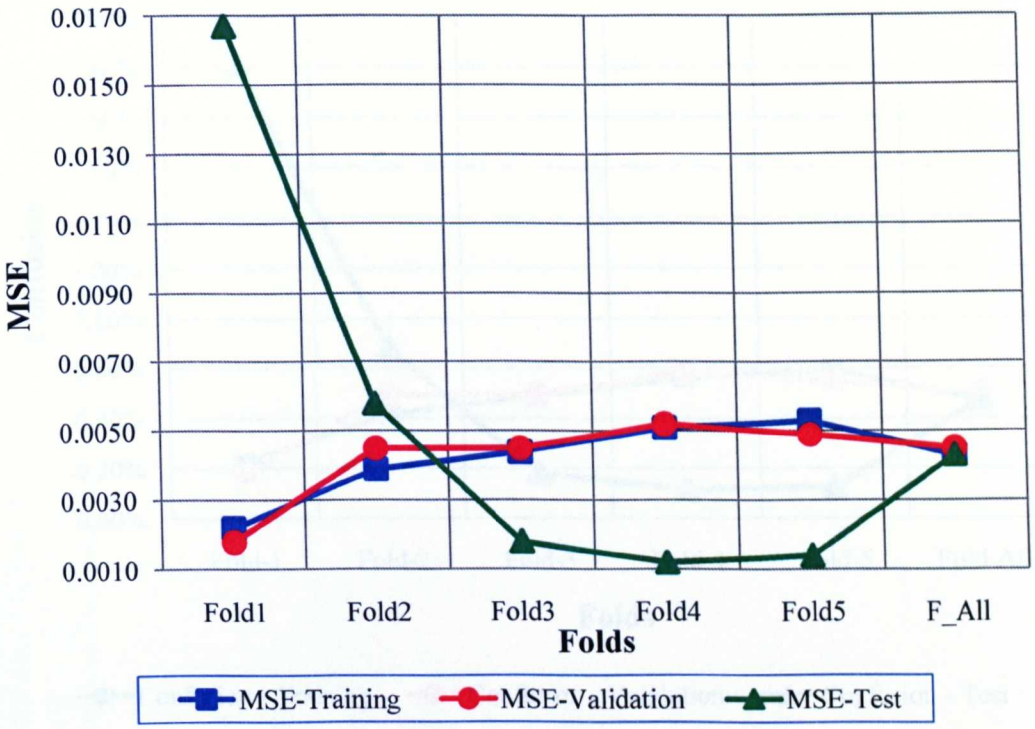


Figure 4-43: MSEs of cross-validation and self-consistency tests.

Fold1 shows the best MSEs 0.0022 and 0.0018 for training and validation respectively. However, Fold1 test records the worst MSE 0.0167. Further, it shows that Fold1 significantly deviates from the rest of the folds. The worst MSE during training is generated by Fold5. Fold4 shows the worst MSE during validation and best MSE during test. Further, it also shows that MSEs of test are higher than the training and validation in Fold1 and Fold2, which indicates an over-fit. However, MSEs do not give a clear picture of how model classified its frames. Therefore, the research analyses classification information, such as sensitivity, specificity, false positive and negative rates, positive and negative prediction precisions, confusion and accuracy rates (Table 4-8). The overall analysis of these results shows, that there are no major deviations in results that is generalised, when calculating confusion or accuracy. Therefore, research utilises confusion as the performance evaluation criterion as shown below;

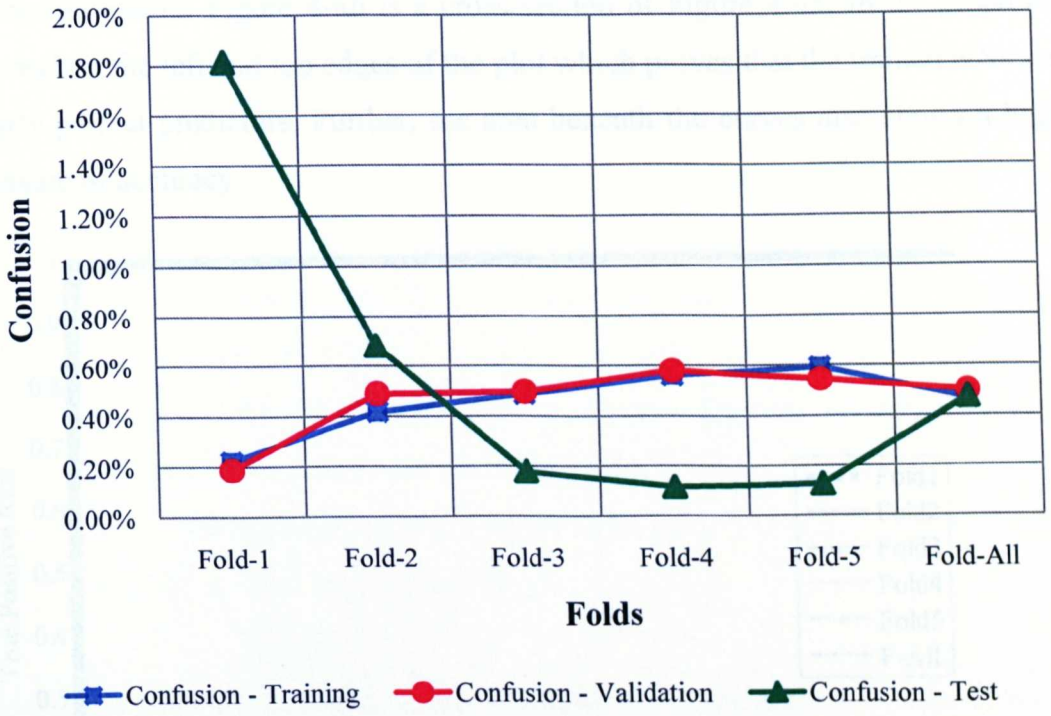


Figure 4-44: Confusions of cross-validation and self-consistency tests.

Figure 4-44 shows that the confusion percentages of the classifiers are extremely low resulting in an accuracy rate ranging from 98.19% to 99.88% during training, validation and test. Fold1 shows the lowest confusion rates 0.22% and 0.19% for training and validation respectively. However, Fold1 test records the highest confusion 1.81%. Further, it shows that Fold1 relatively significantly deviates from the rest of the folds. The highest confusion during training is 0.59%, generated by Fold5. The highest confusion during validation is generated in Fold4. The least confusion during test is 0.12, produced by Fold4. Further, it also shows that confusions of tests are higher than the training and validation in Fold1 and Fold2. Both MSE and confusion values of self-consistency test have a clear least deviation among training, validation and test results: self-consistency test reports MSEs as 0.0043, 0.0045, 0.0043 (Figure 4-43) and confusions, as 0.47%, 0.50%, and 0.47% (Figure 4-44).

The ROC curves in Figure 4-45 is a graphical representation of sensitivity and specificity. It also visually summarises the results of 5-fold cross-validation (Fold1 to Fold5) tests, and self-consistency test (FoldAll) presented in the

confusion matrix. Figure 4-46 is a cross-section of Figure 4-45. In the graph all curves hug the left and top edges of the plot which proves that the trained NNs are nearly perfect predictors. Further, the area beneath the curves also shows a high measure of accuracy.

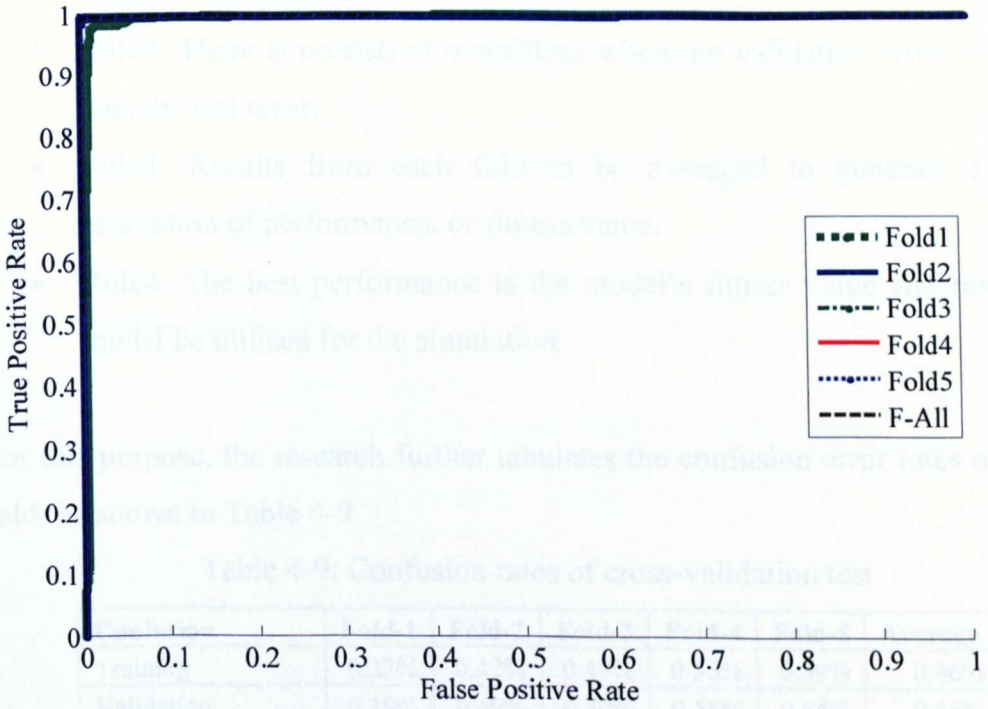


Figure 4-45: ROC curves of Fold1 to Fold5 and self-consistency tests.

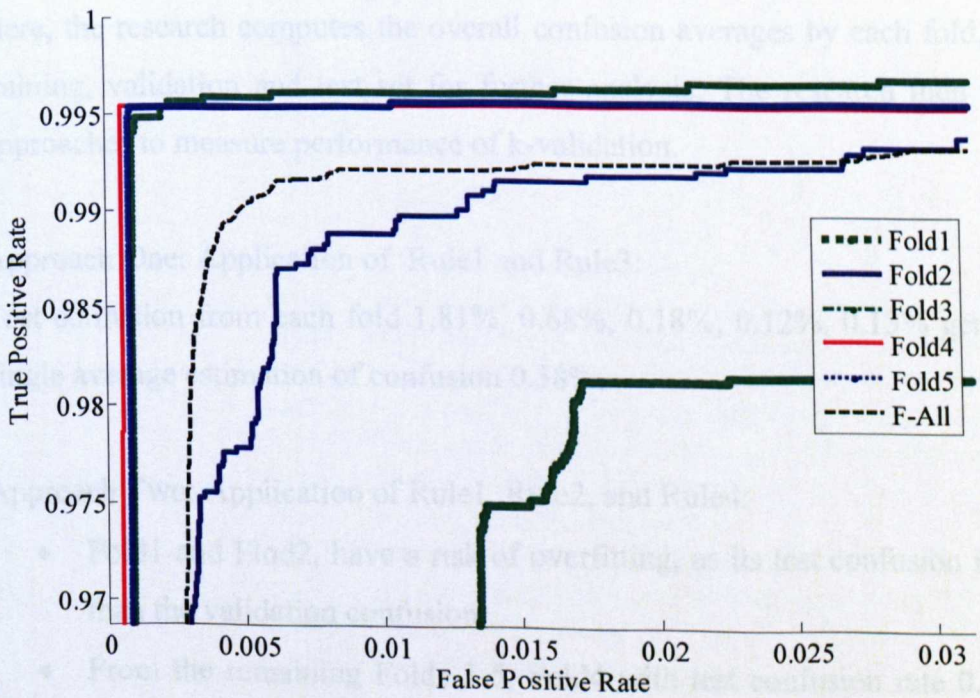


Figure 4-46: A cross section of ROC curves

From the literature, it has been established that the following 4 rules can be applied to qualify a best performing classifier.

- Rule1: Use validation results to measure training performance and test results to measure the performance of the trained classifier on unseen data (MathWorks 2012b).
- Rule2: There is no risk of overfitting when the validation error is greater than the test error.
- Rule3: Results from each fold to be averaged to generate a single estimation of performance, or fitness value.
- Rule4: The best performance is the model's fitness value and respective model be utilised for the simulation.

For this purpose, the research further tabulates the confusion error rates of the 5-folds as shown in Table 4-9.

Table 4-9: Confusion rates of cross-validation test

Confusion	Fold-1	Fold-2	Fold-3	Fold-4	Fold-5	Average
Training	0.22%	0.42%	0.49%	0.56%	0.59%	0.46%
Validation	0.19%	0.49%	0.50%	0.58%	0.54%	0.46%
Test	1.81%	0.68%	0.18%	0.12%	0.13%	0.58%
Average Confusion	0.74%	0.53%	0.39%	0.42%	0.42%	

Here, the research computes the overall confusion averages by each fold, and by training, validation and test set for further analysis. The research then use two approaches to measure performance of k-validation.

Approach One: Application of Rule1 and Rule3:
Test confusion from each fold 1.81%, 0.68%, 0.18%, 0.12%, 0.13% generates a single average estimation of confusion 0.58%.

Approach Two: Application of Rule1, Rule2, and Rule4:

- Fold1 and Flod2, have a risk of overfitting, as its test confusion is greater than the validation confusion.
- From the remaining Folds 3-5, Fold4 with test confusion rate 0.12% has the least confusion, therefore becomes the best-performing model.

- Therefore, Fold4 qualifies to simulate with unseen data.

The research also attempts to provide additional information to support above performance results using the segments of data used in folds (Table 4-6 and Table 4-8). The user and attacker activities during the capturing period (Table 4-10) are analysed with data segments used for NN training, validation and testing.

Table 4-10: User and attacker activities during the capturing period.

Data Segments included	Starting Record No.	Action
User Activities (Capture 20)		
k1	1	Capture Started
k1	315	Opened IE
k1	409	Googgled and played BBC2 Live and stopped
k1	1901	Googgled and played BBC1 Radio Live
k1	7721	Checked Yahoo email
k1	21474	Sent an email
k1, k2,k3,k4,k5	22840	downloaded a large file
K5	153240	stopped BBC1 radio
k5	154670	download completed
k5	154686	closed all opened windows
k5	154734	Disconnected from AP
k5	154769	Scanned Network
k5	154860	Tried to connect to the network 3 times
k5	155363	Repaired the adaptor
k5	155640	Opened IE
k5	157001	Shut Down
	NIL	Stopped Capture
Attacker Activities (Capture 22)		
	NIL	Capture Started
y1	1	Attacker Started
y1	7	Directed Probing attack started
y1	1577	Directed Probing attack stopped
y1	1710	Network Scanned 3x times
y1	1977	Tried to connect to the network with a guessed network key
y1	1846	Tried to connect to the network with a guessed network key
y1	1871	Tried to connect to the network with a guessed network key
y1	2223	Tried to connect to the network with a guessed network key
y1, y2, y3, y4, y5	2223	Directed Probing attack started
y5	18941	Directed Probing attack stopped
y5	19148	Network Scanned 5x times
y5	19490	Tried to connect to the network with a guessed network key
y5	19551	Turned off the attacker
	NIL	Stopped Capture

According to Table 4-9, Fold4 produced the least confusion error rate. Fold4 uses segments 4,5,1,2 to train and validate the network and leaves segment 3 to test the network. Table 4-10 shows that segment 3 consists of k3 and y3 datasets that mainly include user activity *down loading a large file*, and attacker activity - *directed probing attack.* These patterns also include in the training and validation dataset (k4,k5,k1,k2 and y4,y5,y1,y2) which may have helped the classifier to classify test frames efficiently and produce an extremely lower error rate. In contrast, Fold1 produced the most confusion rate. Fold1 uses segments 1,2,3,4 to train and validate the network and leaves segment 5 to test the network. It is clearly visible in Table 4-10 that k5 and y5 datasets includes most vital user and attacker activities which classifier did not learn before, which explains comparatively very high confusion rate. It is also notable that Fold5, which shows the test confusion rate being nearly similar to Fold4, also utilises a similar type of dataset for training and validation, and testing.

4.5. Summary

This research is to identify an external attacker by analysing the traffic generated from a user MAC address in a single frequency band of a WLAN. A supervised feedforward NN with four distinct inputs, delta-time, sequence number, signal strength and frame sub-type is applied to identify and differentiate a genuine frame from a rogue one. This research is conducted with real WLAN traffic, and introduced a new and realistic method to capture and analyse data to feed NN. NN is systematically evaluated using self-consistency and 5-fold cross- validation methods. Self-consistency method evaluates the fitness of data in the classifier by testing the classifier with trained and seen data. The 5-fold cross-validation method is employed to evaluate the classifier with unseen data. The 5-fold cross-validation sequentially uses all of the data for training and validation, and testing. The experimental results demonstrate that the proposed NN classifier can detect probe request attacks to a very high degree. The single average estimation of test confusion rate of all classifiers is 0.58%. Fold4 produced the lowest test confusion

rate of 0.12%, therefore selected as the best-performed model, which will be utilised to simulate with unseen data in Chapter 6.

The next Chapter (Chapter 5) discusses the process of applying GA methods to improve sensor/classifier's performance by optimising data and number of hidden neurons. Further, a systematic evolution is performed executing five GAs, and the best performed NN is selected as the most optimised NN.

Chapter 5: Optimisation of NN for Detection of Probe Request Attacks

5.1. Introduction

A supervised feedforward NN classifier to detect Probe Request Flooding Attacks of WLAN is developed in this research. The results converged outstandingly with training, validating, testing sample percentages 70, 15, 15 and hidden neurons 20, as outlined in Chapter 4. The effectiveness of an IDS is evaluated by its ability to make correct predictions. However, during the cross-validation test, the best classifier performed 98.88% accurately with unseen test data. Therefore, this work considered applying an optimisation technique to improve its performance. Section 3.4.7 contains a systematic review of NN optimisation literature. This chapter explains the optimisation of the NN to detect probe request attacks, using GAs using method defined in Chapter 3. Five GAs will be utilised to optimise the data and the number of hidden neurons. The best performed NN, based on least confusion error will be selected as the most optimised NN. The Chapter is organised as follows; Section 5.2 briefly describes the un-optimised NN. Section 5.3 explains NN optimisation process. Section 5.4 gives a comprehensive account of the evolution of GAs and results. Section 5.5 summarises the optimisation results and concludes the Chapter.

5.2. Un-optimised Neural Network

In Chapter 4, this research designed a supervised feedforward NN with four input neurons, one hidden layer, with 20 hidden neurons and a linear output neuron which classifies genuine frames from rogue frames. The design used 70% of the data to train the network and 15% each for validation and testing, respectively. This ratio is adopted, as these are default percentages in the MATLAB toolbox,

and the most commonly used ratio in training, validation and testing NNs in literature. The data sample is divided using random indices so that each sample contains a representation of the complete dataset. SCG backpropagation algorithm is utilised for training. A summary of results presented in Table 4-4, are shown in Table 5-1 below. Performance of the network is determined by the confusion error of the test sample.

Table 5-1: Results of un-optimised default NN.

Complete Sample	176630 frames
Training Sample (70%)	123640 frames
Validation Sample (15%)	26495 frames
Testing Sample (15%)	26495 frames
Hidden Neurons	20
Training Confusion %	0.48%
Validation Confusion %	0.46%
Test Confusion %	0.49%

Although, possible overfitting problems are addressed by means of early stopping using intermediate validation during training, the research recognised that these results indicate a slight sign of overfitting as test errors are higher than validation error (Section 4.3). Therefore, research also performed systematic evaluation of the dataset and models using self-consistency and 5-k cross-validation tests respectively. The results show that Fold-4 classifies its unseen test data 98.88% accurately (Section 4.4). Summary of Fold4 training, validation and test results presented in Table 4-8 is shown in Table 5-2 below:

Table 5-2: Results of un-optimised Fold4 NN.

Complete Sample (segments 1,2,3,4,5)	176630 frames
Training Sample (70% of segments 1,2,4,5)	98913 frames
Validation Sample (30% of segments 1,2,4,5)	42391 frames
Testing Sample (segment 3)	35326 frames
Hidden Neurons	20
Epoch	75
Training confusion %	0.56%
Validation confusion %	0.58%
Test confusion %	0.12%

The overall simulation (simulation is discussed in Chapter 6) results show 4.1% confusion resulting in only 95.9% of overall accuracy from the 10 simulation samples used (Section 6.2.1.)

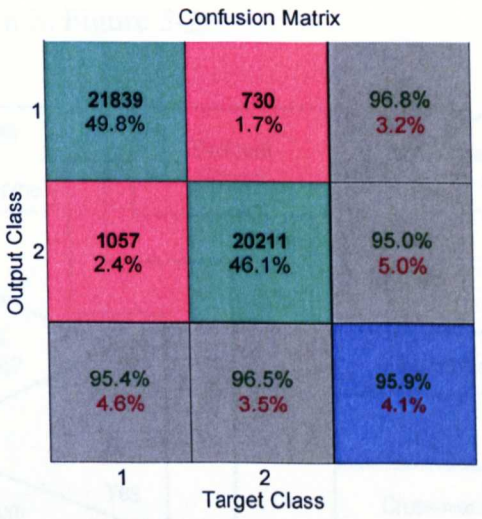


Figure 5-1: NN simulation results of best K-fold: Fold4

However, the intention of this research is to realise a classifier that can produce 100% accurate results.

5.3. Neural Network Optimisation

The literature suggests that NN performance can be improved by initialising the network and training, increasing the number of hidden neurons and layers, using a different training function, or using additional training data (MathWorks 2012d). By considering the literature and results generated previously from this research, the research recognised that optimising training, validation and testing sample percentages, and number of neurons of the hidden layer will generate improved results.

The optimisation system ran on a DELL Inspiron N5050 laptop with 2nd generation Intel® Core™ i5-2450M processors at 2.50GHz and 2.49GHz, 3.41GB of RAM. The OS is Microsoft Windows XP. GA is developed using a combination of MATLAB 2008b and NetBeans environment. Matlabcontrol Java Application Programming Interface (API) is used to interface Java and MATLAB.

NetBeans environment is used for generation of a population of random chromosomes, validation, crossover and mutation. MATLAB environment is used for NN training, calculation of fitness, fitness scaling, and. The flow diagram of the GA applied is shown in Figure 5-2.

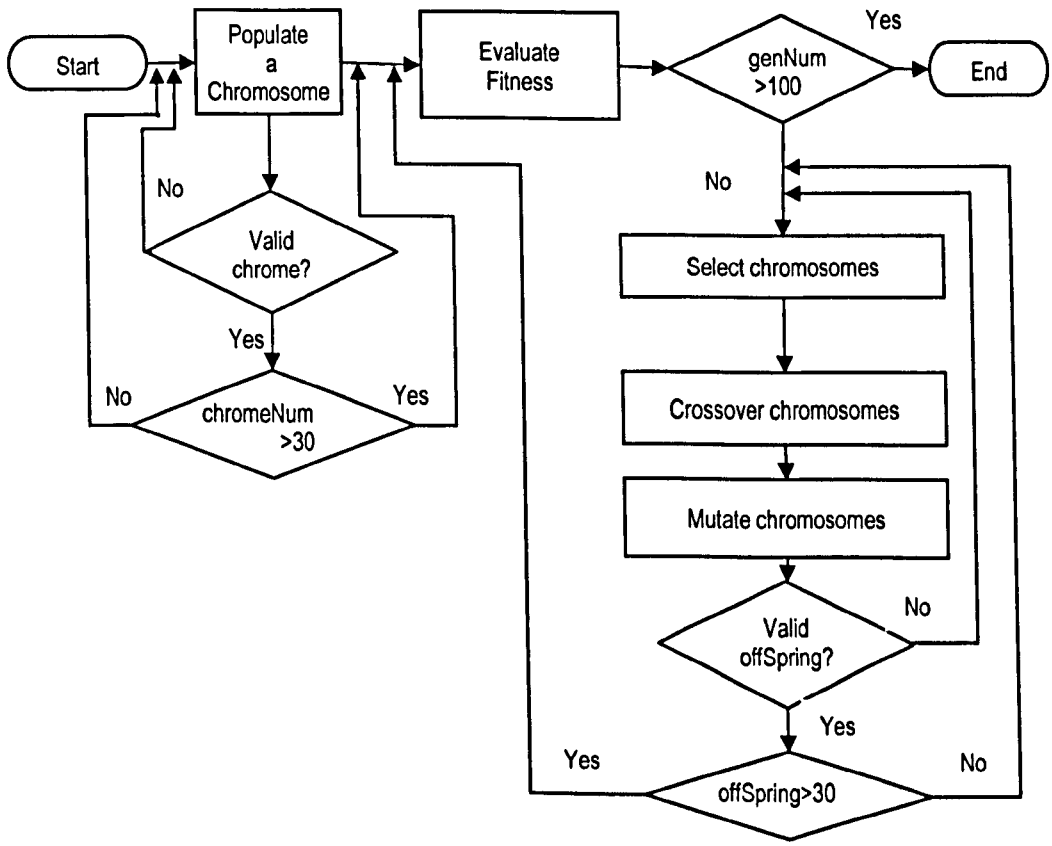


Figure 5-2: Flow diagram of GA

5.3.1. Installation Notes

Literature suggests that GAs can often find good solutions in around 100 generations. Therefore, it is decided to set maximum number of generations to 100. The stopping criterion for the GA is when maximum number of generations is reached (100) or when elite candidate solution did not changed for 25 consecutive generations. Mutation probability and fitness scale can be altered to fine-tune, before the commencement of each GA.

5.3.2. Population

Process population creates a population of a binary vector that represents 4 genes as shown in Table 5-3.

Table 5-3: Genes of a chromosome.

Gene Name	Description
trainVal	% of training sample
validVal	% of validation sample
testVal	% of testing sample
neuL1	number of neurons in hidden layer 1

Each chromosome is designed with four, 7-bit genes to represent *trainVal, validVal, testVal* and *neuL1*. The decimal values that each gene can take is 0-127 i.e. a population can create values between 0-127 for *trainVal, validVal* and *testVal*. However, the NN training function requires *trainVal, validVal* and *testVal* to be non-zero and sum of *trainVal, validVal* and *testVal* to be 100. Therefore, the minimum and maximum values *trainVal, validVal* and *testVal* can take are 1 and 98 (Table 5-4).

As discussed before, GA is an accelerated search of feasible solution space. In nature and in GAs, the key feature for good performance is to sustain the diversity of the population. However, constraints have to be applied to reduce the search space to prevent generating unfeasible or resource exhaustive populations. This usually achieves through penalties, repairs or decoders or new data structures or operators (Michalewicz & Janikow 1991; Michalewicz & Schoenauer 1996; Ponsich et al. 2008). In a previous experiment (Ratnayake, D et al. 2011), it is realised that strict penalty rules restricts achieving diversity as it causes rapid diversity reduction in the population due to high rejection of chromosomes. Therefore, in this research, validation process is designed to decode values, by scaling the absolute values to relative values of 100%, so that the total of *trainVal+validVal+testVal* will always equal to 100.

Literature suggests that larger numbers of neurons and extra hidden layers give the NN more flexibility since the network has more parameters to optimise.

However, a too large hidden layer can cause the problem to be under-characterised if the network has to optimise more parameters than there are data vectors. Further, more neurons and layers increase processing time, and demands high processing power (MathWorks 2012d). Therefore, in this experiment, a single hidden layer and maximum of 32 hidden neurons are used. Here too, validation process is designed to decode values, by scaling the absolute values to relative values, so that number of hidden neurons will always between zero and 32. Thus, GAs create well-dispersed populations, they are biased to create chromosomes with genes that are on the boundaries of the constraints defined in Table 5-4.

Table 5-4: Validation constraints.

Sizes of trainVal, validVal, and testVal should be > 0 and < 98
Total of trainVal, validVal, testVal should be =100
neuL1 should be > 0 and<33

Functions in the population process are *'randomGeneration'* to create the initial population and *'crossOver'* and *'mutation'* to create the populations of the rest of the generations. A *'validation'* function applies validation rules to genes.

Further, current literature also suggests that a large population size may not help to improve performance of a GA quickly. A good population size is considers to be about 20-30. Some literature suggest that 50-500 is the best whilst others suggest that the best population size depends on the size of encoded string: if the chromosomes have 32 bits, the population size should be 32 (Lande & Barrowclough 1987; Obitko 2012). This research uses 4 genes, that are represented by 28 bits. Therefore, the research consideres 30 chromosomes as a good population size. Function *'randomGeneration'* generates 30 random chromosomes within boundaries of Table 5-4. This population limit is maintained throughout GA.

5.3.3. Evaluation

As applied in Chapter 4, here too, confusion value of the test set is used to make a decision on NN performance. Evaluation process is implemented using following MATLAB functions:

- MATLAB '*newpr*' function uses the complete set of inputs, targets, and gene value *neuL1* to create a feedforward backpropagation network, *net*.
- MATLAB function *dividerand* randomly devides data into three subsets, namely; training, validation and test, using the division parameters: gene values trainVal, validVal, and testVal.
- MATLAB function '*train*', trains the network using SCG backpropagation training function '*trainscg*', with default training parameters and generates the network, *train* and training record, *tr*.
- MATLAB '*sim*' function tests the trained network by simulating inputs sample, and generates a vector of outputs which distributed to train, validation and test samples using '*dividerand*' indices.
- MATLAB function '*confusion*' generates confusion error values of training, validation and test.

5.3.4. Selection

There are many methods to select the best chromosomes to be crossed for the next generation such as roulette wheel selection, Boltzman selection, tournament selection, rank selection, steady state selection and some others (Hopgood & Mierzejewska 2009; Obitko 2012). This research uses Stochastic Universal Sampling (SUS) to select chromosomes for recombination. SUS uses a single random value to sample all of the solutions by choosing them at evenly spaced intervals. Therefore, it offers zero bias and minimum spread (Baker 1987; Levine 1997). The research also allows the user to alter percentage of the total population that can be selected to crossover, before running each GA. The selection process is implemented using following MATLAB functions:

- MATLAB function *'fitscalingrank'* takes a vector of fitness values and the number of parents to be selected. It ranks individuals by its position in the vector of sorted fitness values: the rank of the best-fit individual is one; the next best fit is two, and so on. Then it assigns proportional values to individuals, based on the rank of each individual: The scaled value of an individual with rank 'r' is proportional to $1/\sqrt{r}$. The function returns a column vector of scaled values.
- MATLAB function *'selectionstochunif'* then uses the vector of scaled values and the selection percentage to select parent chromosomes to be crossed.
- The concepts of elitism and population overlaps influence are utilised when creating new population, i.e. one elite chromosome, which is the best-performed chromosome from each generation, migrates to the next generation thereby creates a population overlap.

5.3.5. Crossover

Function *'selectionstochunif'* may or may not select the elite chromosome as a next generation parent. However, as discussed above, the selection process passes the elite chromosome to the next generation as its first child. Therefore, the new generation always includes the elite chromosome. Function *'crossover'* randomly choose 2 'chosen parent chromosomes' and crossover until 29 more valid chromosomes generate to complete the next generation. Crossovers are 2 point. The position of the crossover generates randomly.

5.3.6. Mutation

Levine (1997) recommends adaptive mutation rates which many off-the-shelf GA systems supports due to its usefulness in achieving good results. However, this research could not implement this feature in the program due to complexity issues. Therefore, in this program chromosomes to be mutated and position to be mutated are selected randomly based on the percentages defined by the user at the start of

the GA. Mutated child chromosomes are validated using conditions in Table 5-4. Crossing over parents from the list of chosen-chromosomes-to-be-crossed generates new chromosomes to replace any rejected child chromosomes.

5.4. Results of GAs

The GA is executed several times with different parameters. All GAs are executed until one of the stopping conditions is reached. The fitness value is confusion. The elite chromosome with least confusion value is the most optimised NN. However, previous experiments showed that same fitness value could be produced by NNs with different combinations of alleles. Therefore, at the end of each GA, its elite chromosomes will be ordered by; test, validation, train and best epoch respectively. The first chromosome on the list is the best optimised NN. The next sub-sections present five unique GAs that shaped the ultimate objective of this research. The summary of all GAs are presented in Table 5-5.

5.4.1. GA1

The GA1 is executed with 0.5% of single point mutations and 60% of fitness scale. Best and worst confusion values reported from elite chromosomes are 0.25% and 0.41%. GA converged at 49th generation when *trainVal, validVal, testVal* and *neuL1* are 62, 36, 2 and 28 respectively. Confusion value of elite chromosome remained at its best only at 49^{th} generation. Elite chromosome stopped improving from 50^{th} generation. Therefore, GA1 stopped at 75^{th} generation as it reached a stopping condition (Figure 5-3). The behaviour of the elite chromosome's alleles of each generation is shown in Figure 5-4. The search space analysis shows the behaviour of alleles of all chromosomes (Figure 5-5). These clearly show that this GA explores wider range of points until about 50^{th} generation and reduces its diversity as selected chromosomes evolve through the generations. Confusions of best elite chromosome of GA for training and validation are 0.43% and 0.42%. Its best epoch is 68.

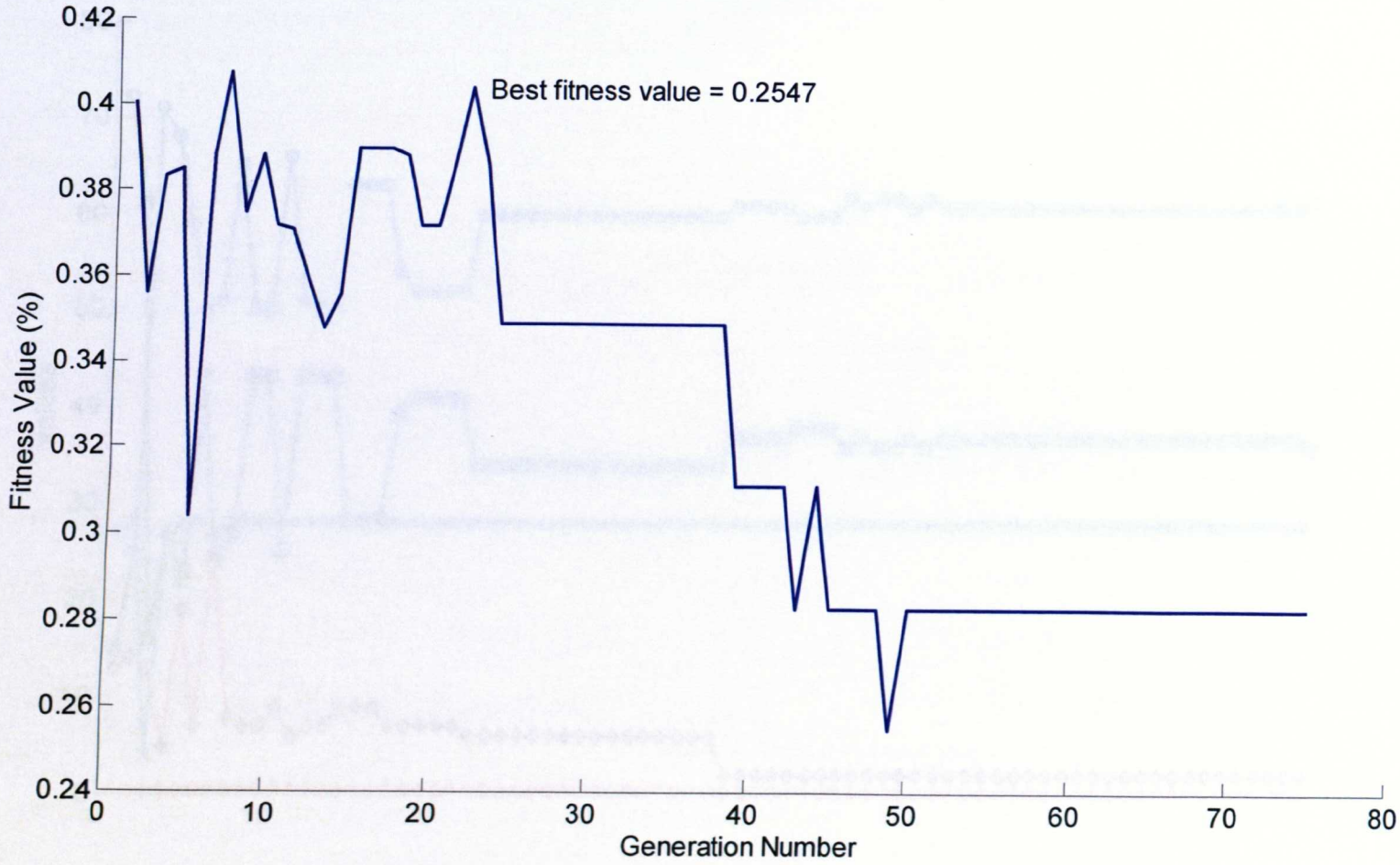


Figure 5-3: Evolution of the elite (GA1).

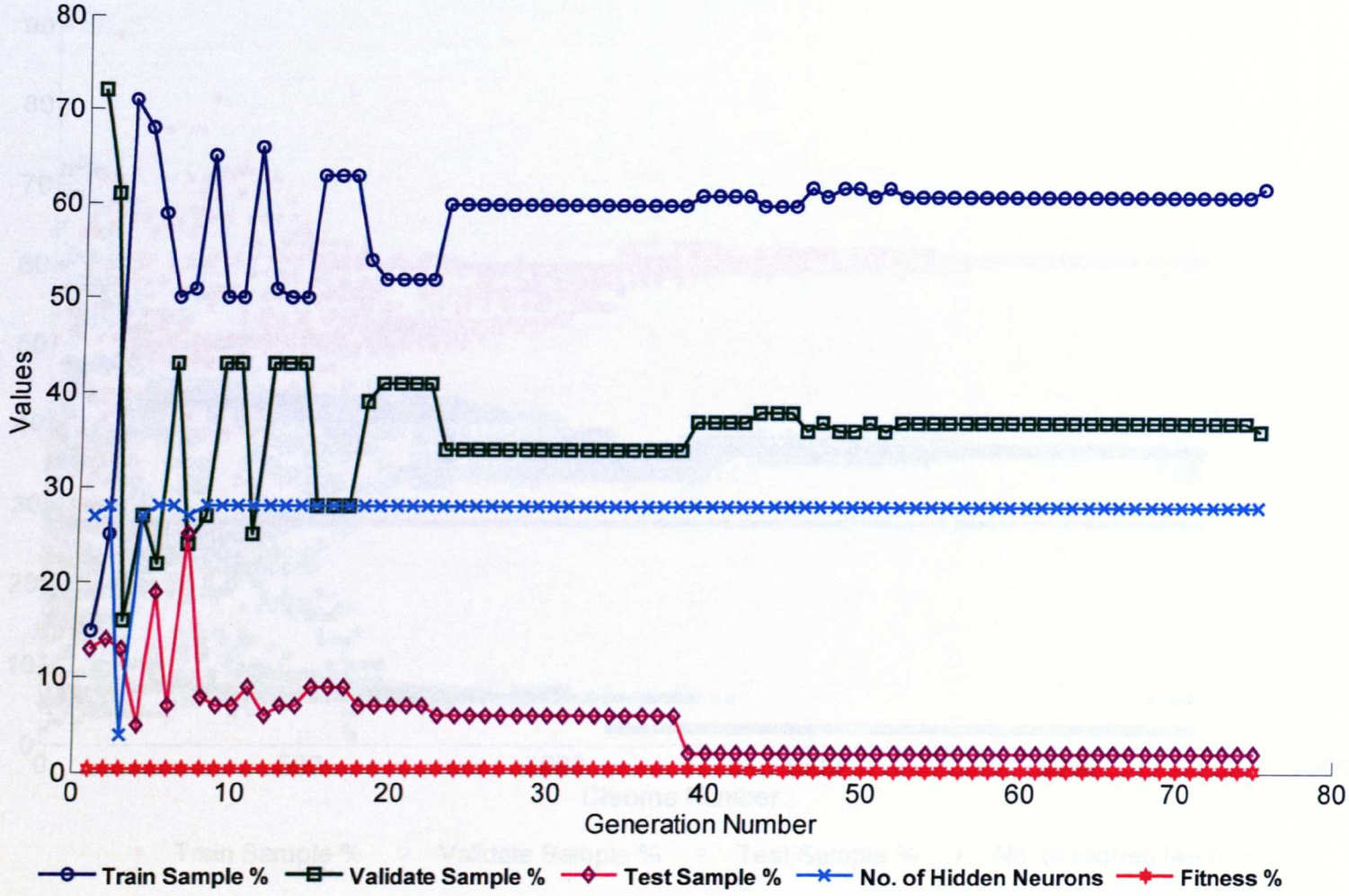


Figure 5-4: Alleles and fitness values of elite chromosomes (GA1).

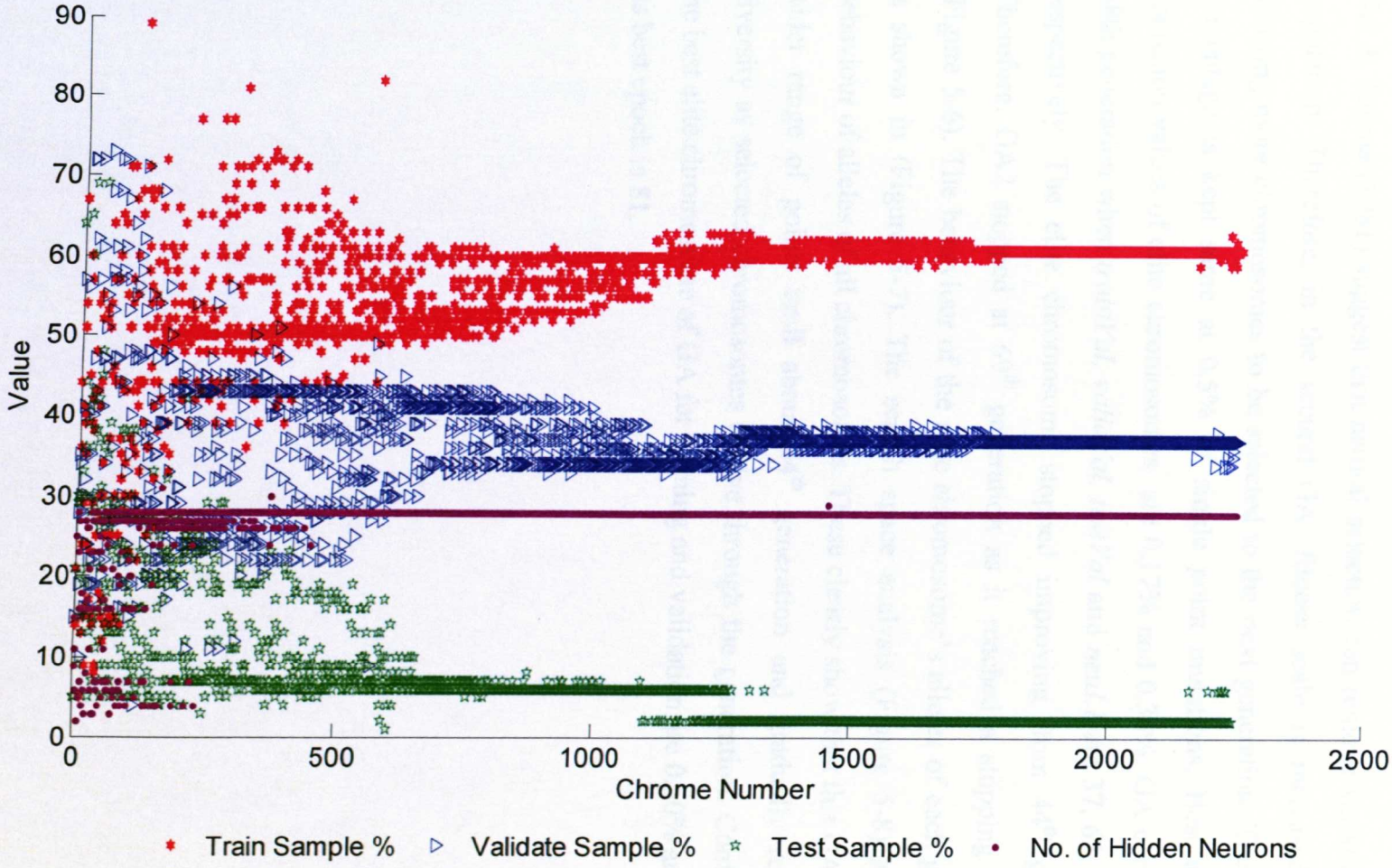


Figure 5-5: The search space of GA1.

5.4.2. GA2

Nei & Tajima (1981) suggest that natural selection can remove variation from a population. Therefore, in the second GA, fitness scale is increased to 80% allowing more chromosomes to be selected to the next generation. The mutation percentage is kept same at 0.5% of single point mutations. Best and worst confusion values of elite chromosomes are 0.17% and 0.38%. GA converged at 48th generation when *trainVal, validVal, testVal* and *neuL1* are 37, 62, 1, and 20 respectively. The elite chromosome stopped improving from 44th generation Therefore, GA2 stopped at 69th generation as it reached a stopping condition (Figure 5-6). The behaviour of the elite chromosome's alleles of each generation is shown in (Figure 5-7). The search space analysis (Figure 5-8) shows the behaviour of alleles of all chromosomes. These clearly show that this GA explores wider range of points until about 44th generation and gradually reduces its diversity as selected chromosomes evolve through the generation. Confusions of the best elite chromosome of GA for training and validation are 0.50% and 0.47%. Its best epoch is 81.

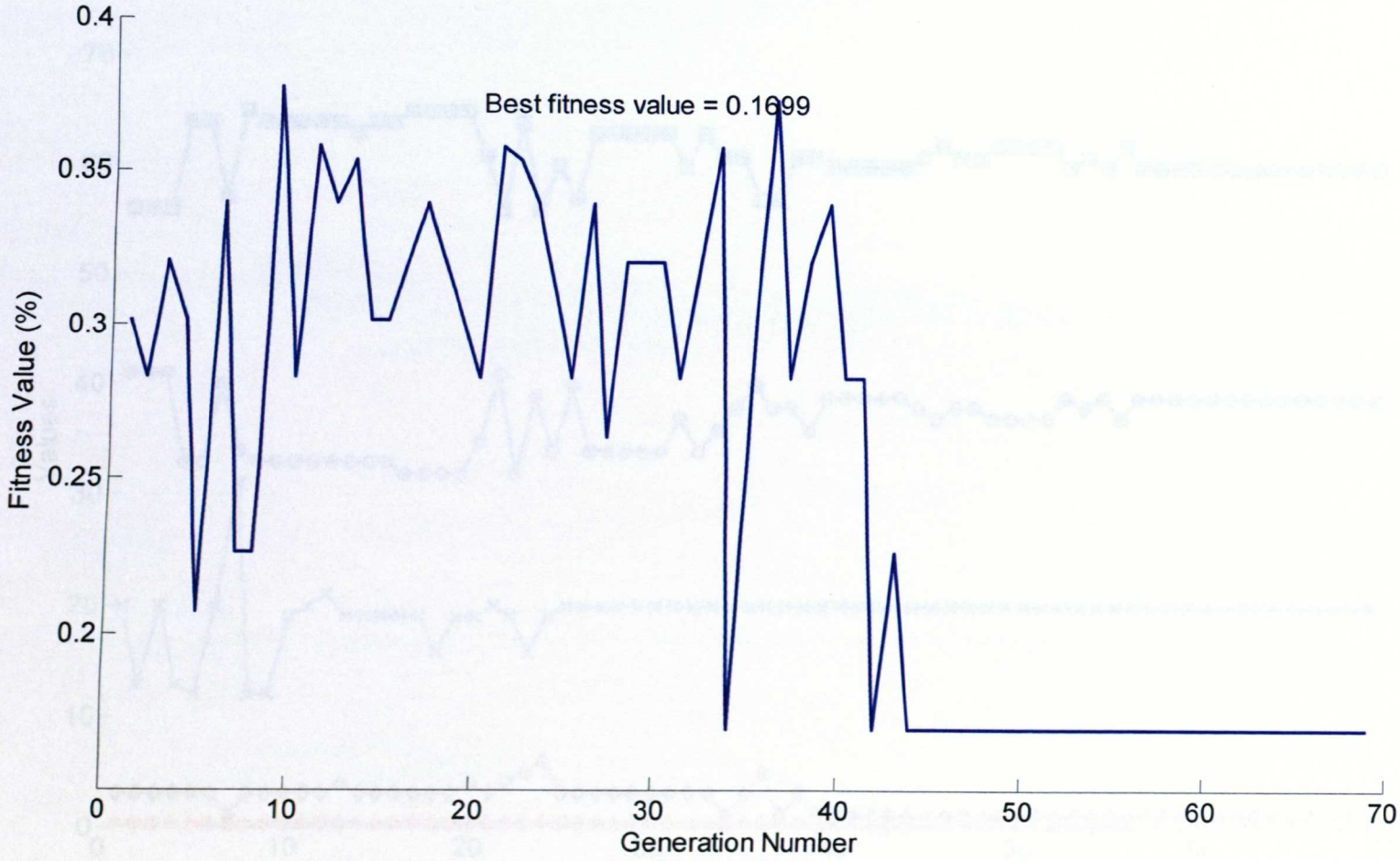


Figure 5-6: Evolution of the elite chromosome (GA2).

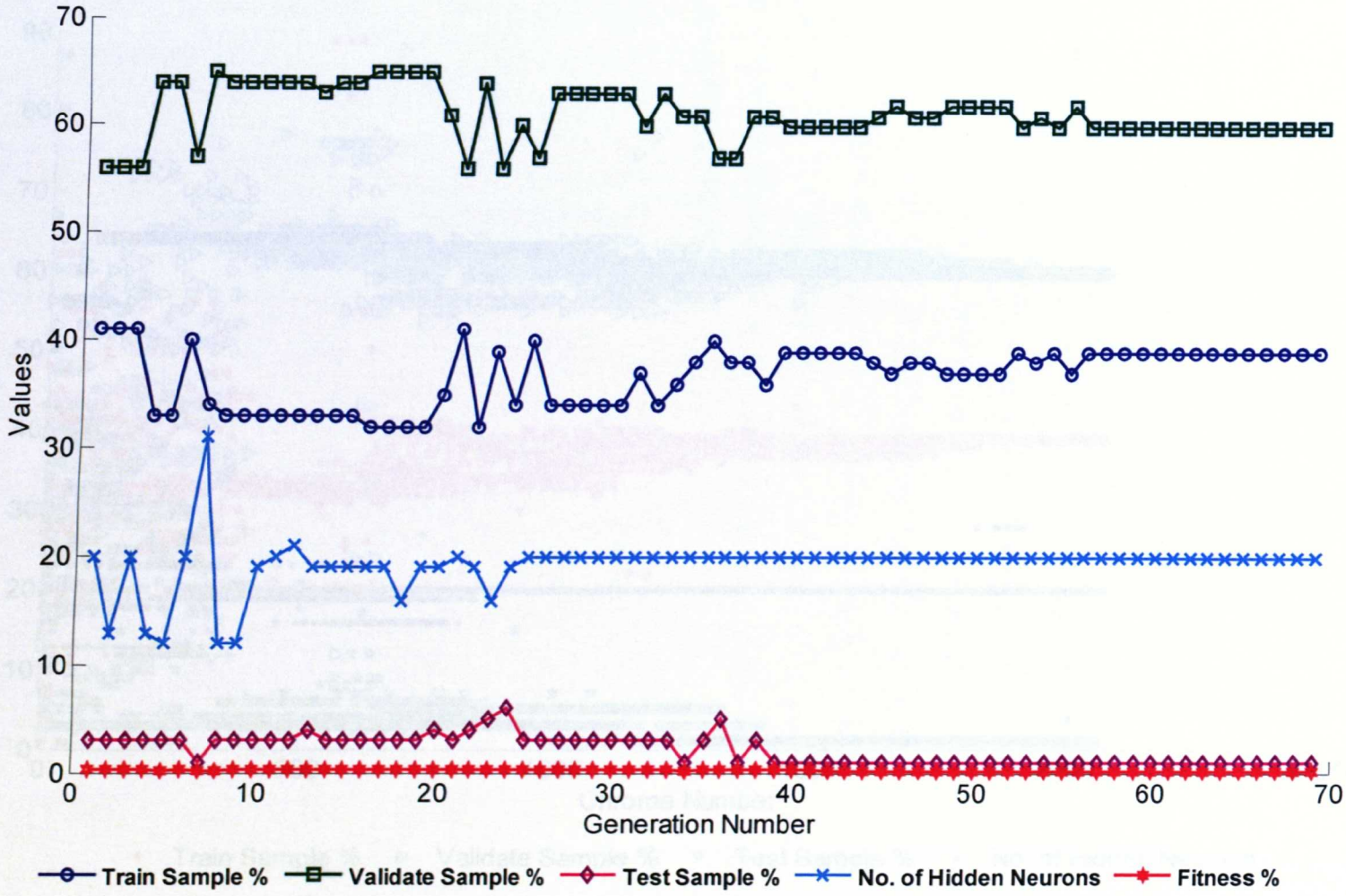


Figure 5-7: Alleles and fitness values of elite chromosomes (GA2).

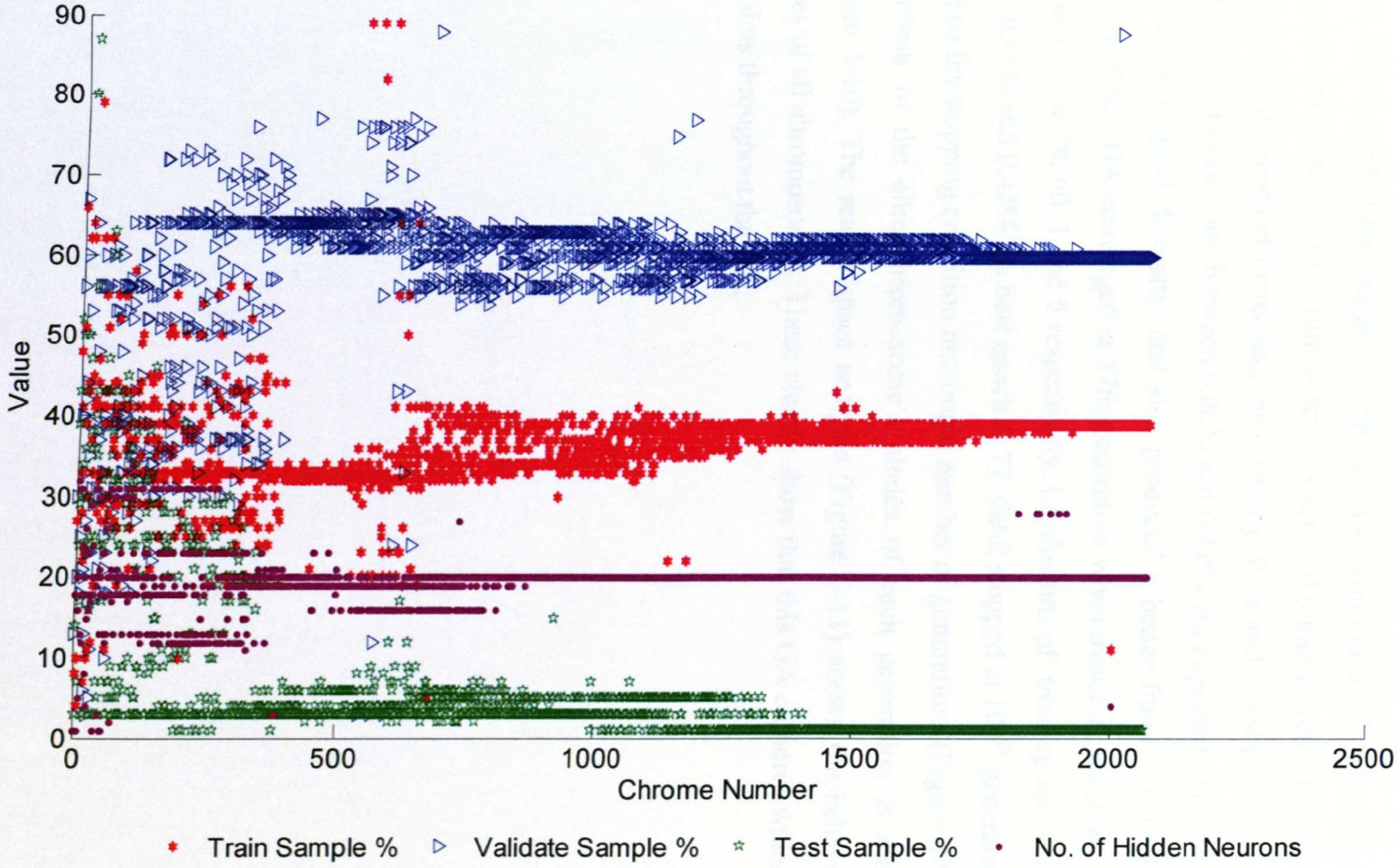


Figure 5-8: The search space of GA2.

5.4.3. GA3

Nei & Tajima (1981) also suggested that genetic variation arises through mutation and recombination. In GA3, fitness scale is set to 0.6 and increased the mutation to 1% in the intention of increasing the diversity. Best and worst confusion values of elite chromosomes are between 0.06% and 0.42%. As expected, this has clearly introduced a good diversity, and also produced a better fitness value than the previous GAs. GA converged at 57th generation when *trainVal, validVal, testVal* and *neuL1* are 39, 60, 1 and 5 respectively. Confusions of training and validation were 0.45% and 0.42%. Its best epoch is 77. GA2 stopped at 100^{th} generation as it reached the stopping condition maximum number of generations (Figure 5-9). The behaviour of the elite chromosome's alleles of each generation is shown in (Figure 5-10). The search space analysis (Figure 5-11) shows the behaviour of alleles of all chromosomes. These clearly show that this GA explores wider range of points throughout the GA.

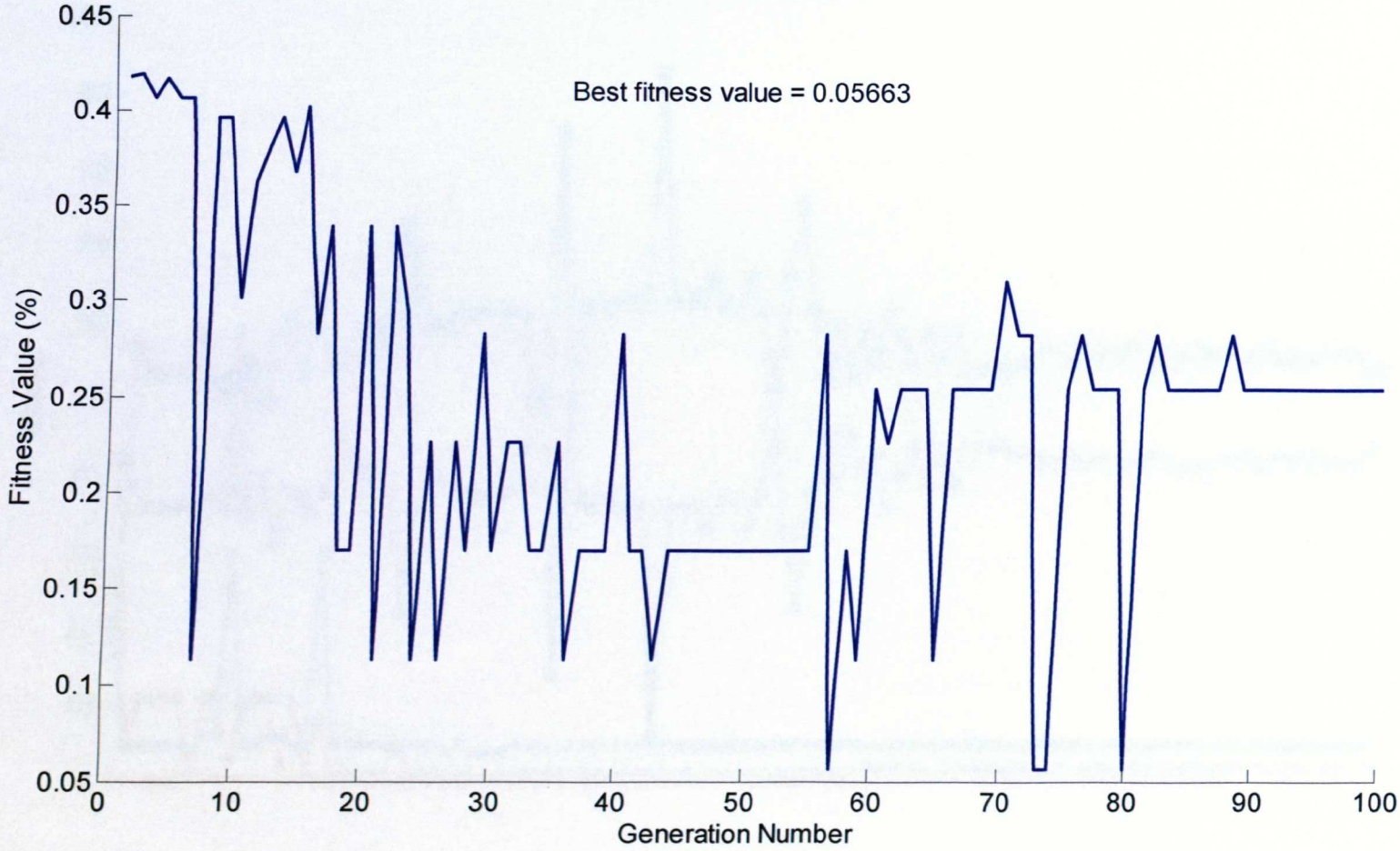


Figure 5-9: Evolution of the elite chromosome (GA3).

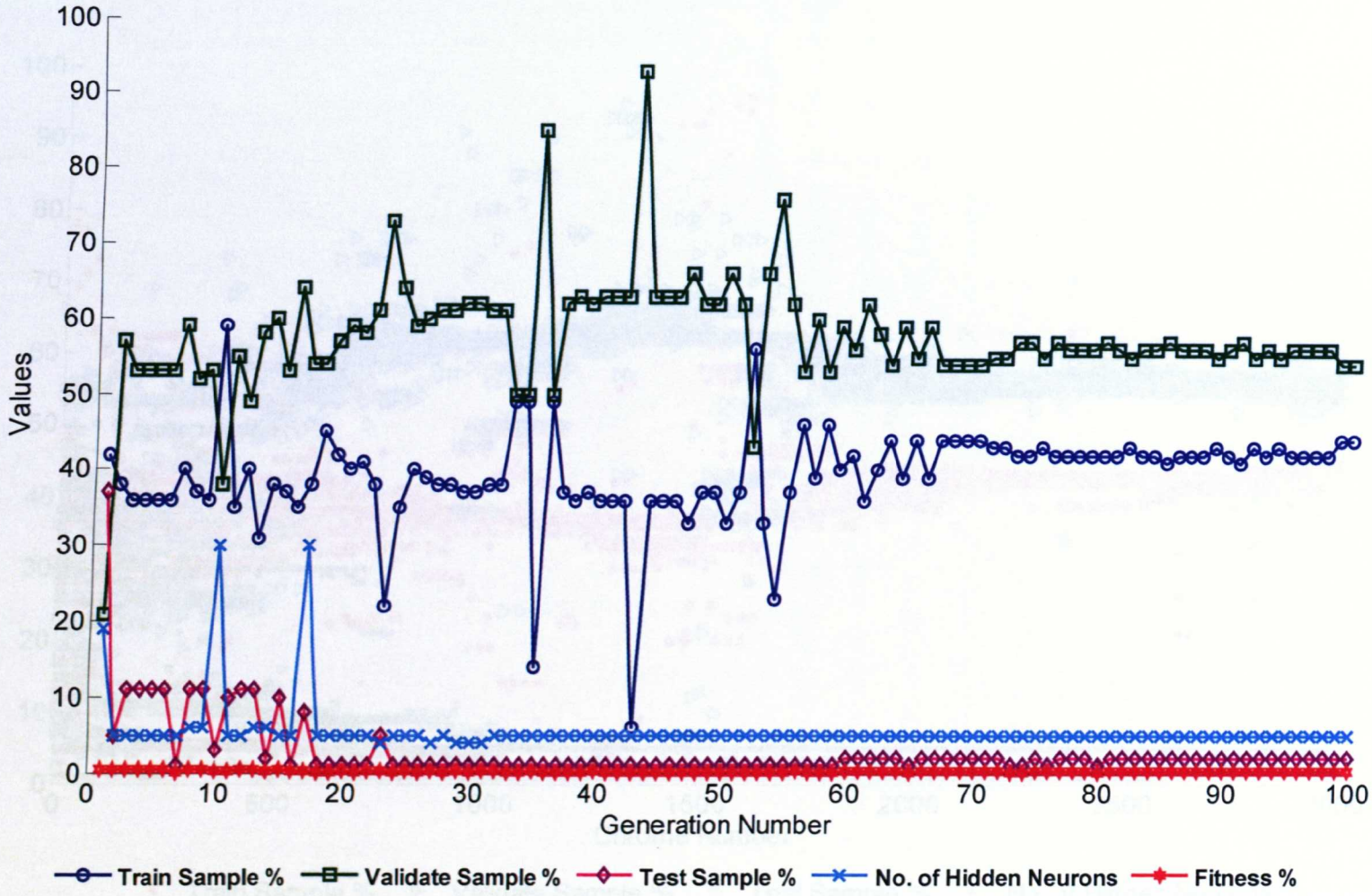


Figure 5-10: Alleles and fitness values of elite chromosomes (GA3).

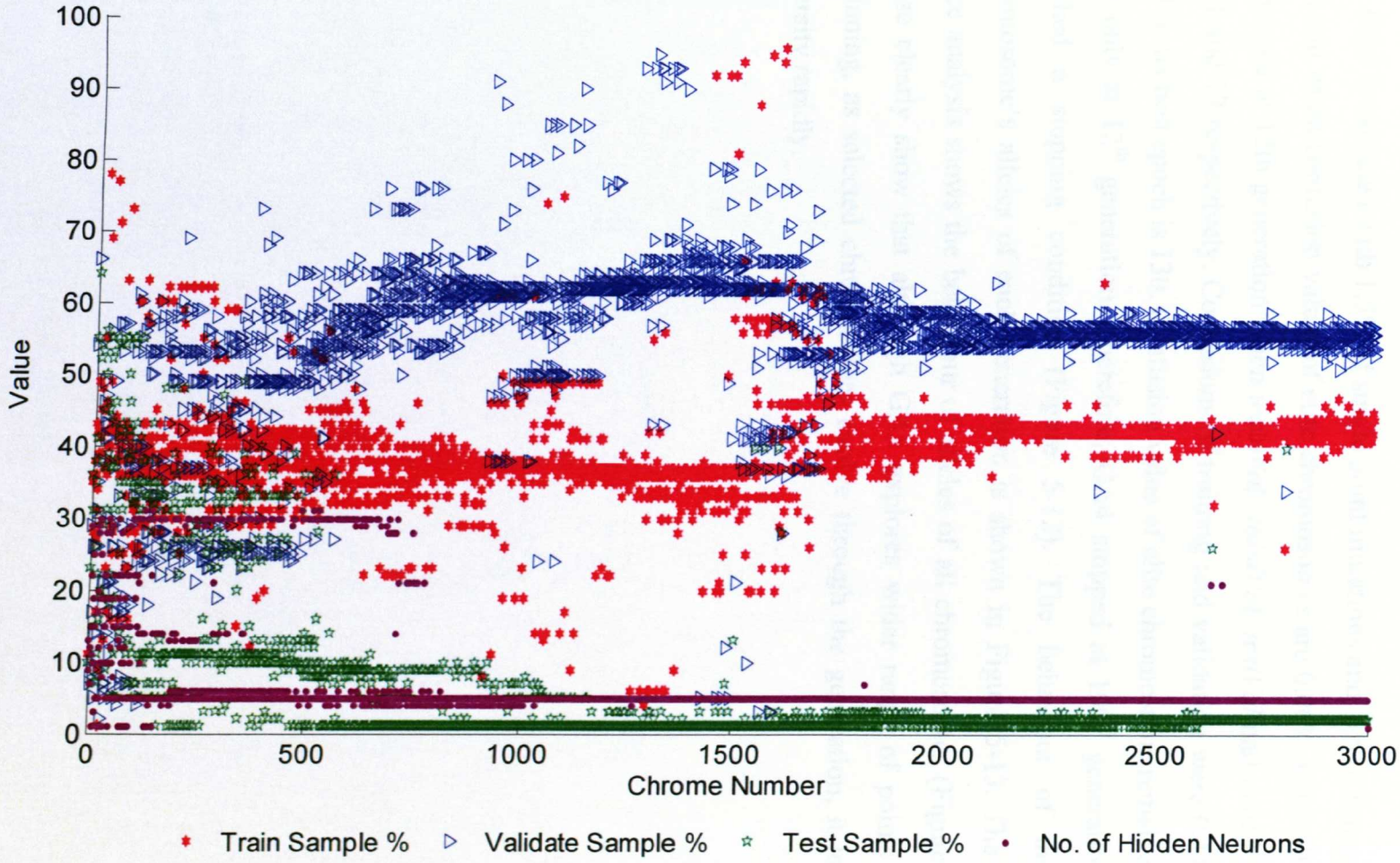


Figure 5-11: Search Space of GA3.

5.4.4. GA4

The GA4 is executed with 1.5% of single point mutations and 0.6 of fitness scale. Best and worst confusion values of elite chromosomes are 0.06% and 0.40%. GA converged at 12th generation when *trainVal, validVal, testVal* and *neuL1* are 50, 49, 1 and 17 respectively. Confusions of training and validation were 0.44% and 0.41%. Its best epoch is 130. Confusion value of elite chromosome remained at its best only at 12^{th} generation. Therefore, GA4 stopped at 100^{th} generation as it reached a stopping condition (Figure 5-12). The behaviour of the elite chromosome's alleles of each generation is shown in Figure 5-13. The search space analysis shows the behaviour of alleles of all chromosomes (Figure 5-14). These clearly show that although GA5 explores wider range of points in the beginning, as selected chromosomes evolve through the generation, it loses its diversity rapidly.

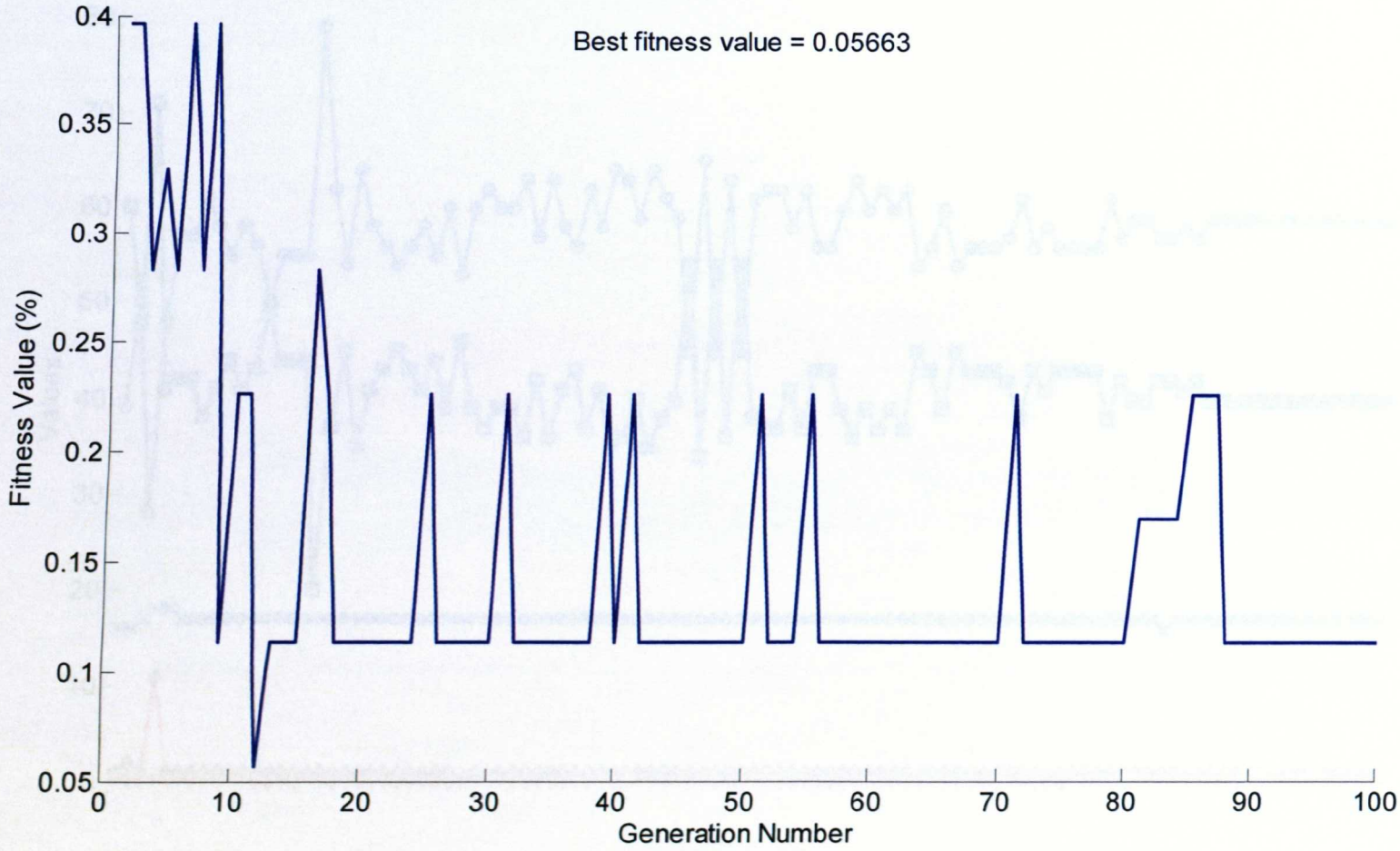


Figure 5-12: Evolution of the elite chromosome (GA4).

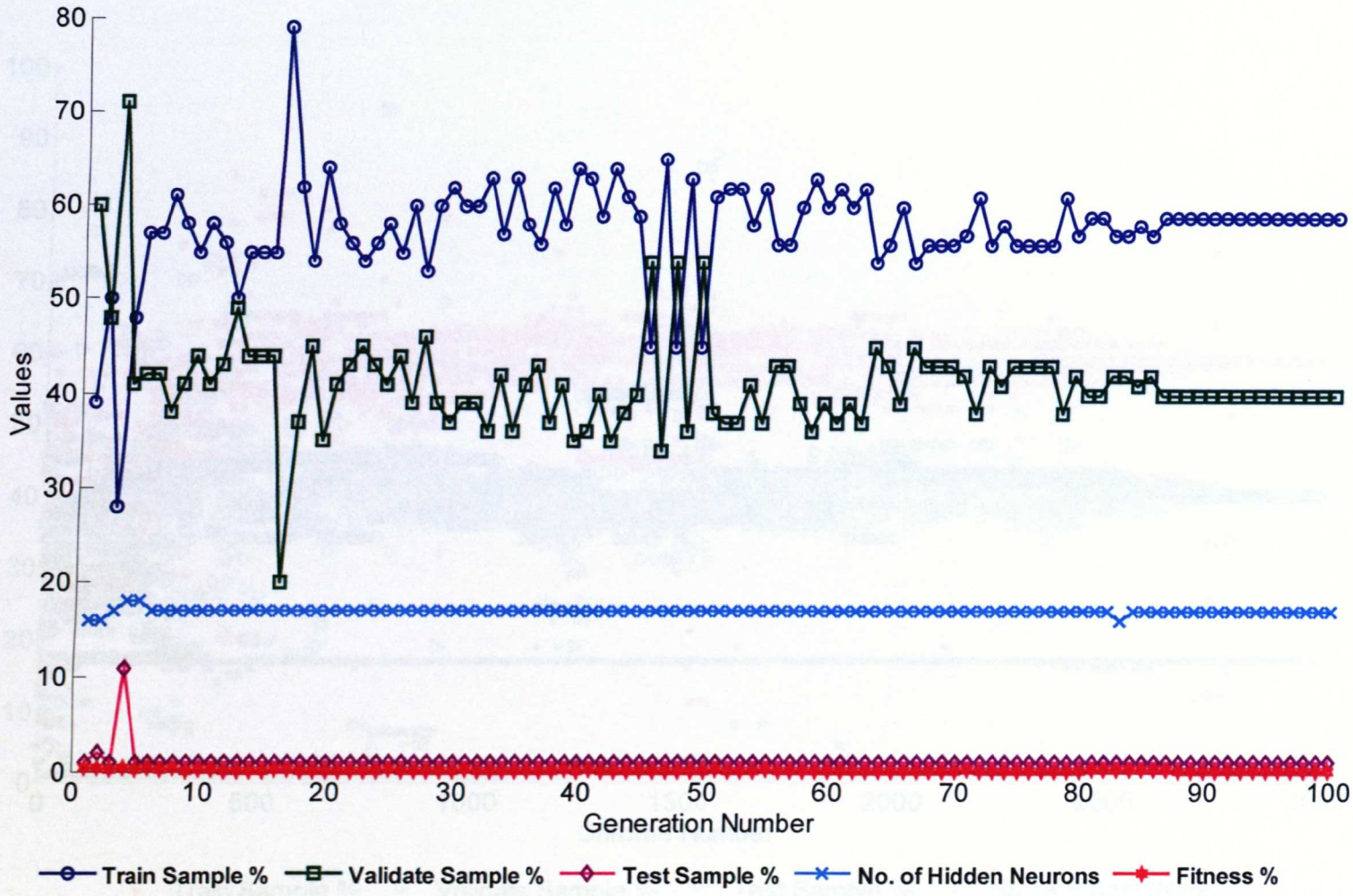


Figure 5-13: Alleles and fitness values of elite chromosomes (GA4).

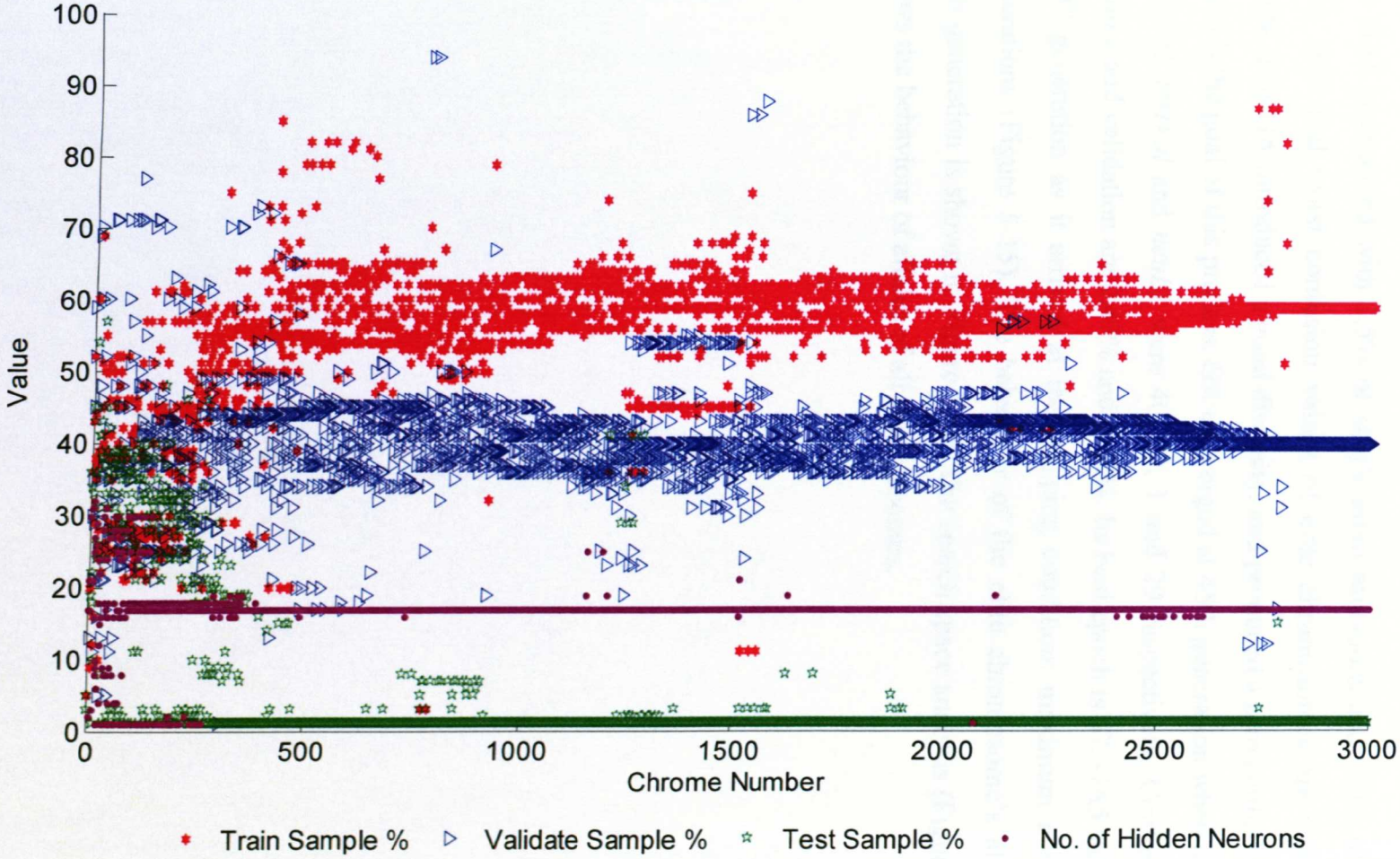


Figure 5-14: Search Space of GA4.

5.4.5. GA5

The GA5 is executed with 1.5% of single point mutations and 90% of fitness scale. Best and worst confusion values of elite chromosomes are 0.00% and 0.41%. This GA introduced a good diversity, and produced a zero confusion value which is the goal of this process. GA converged at 45th generation when *trainVal, validVal, testVal* and *neuL1* were 40, 59, 1 and 29 respectively. Confusions of training and validation are 0.49% and 0.47%. Its best epoch is 67. GA5 stopped at 100^{th} generation as it arrived at the stopping condition: maximum number of generations (Figure 5-15). The behaviour of the elite chromosome's alleles of each generation is shown in Figure 5-16. The search space analysis (Figure 5-17) shows the behaviour of alleles of all chromosomes.

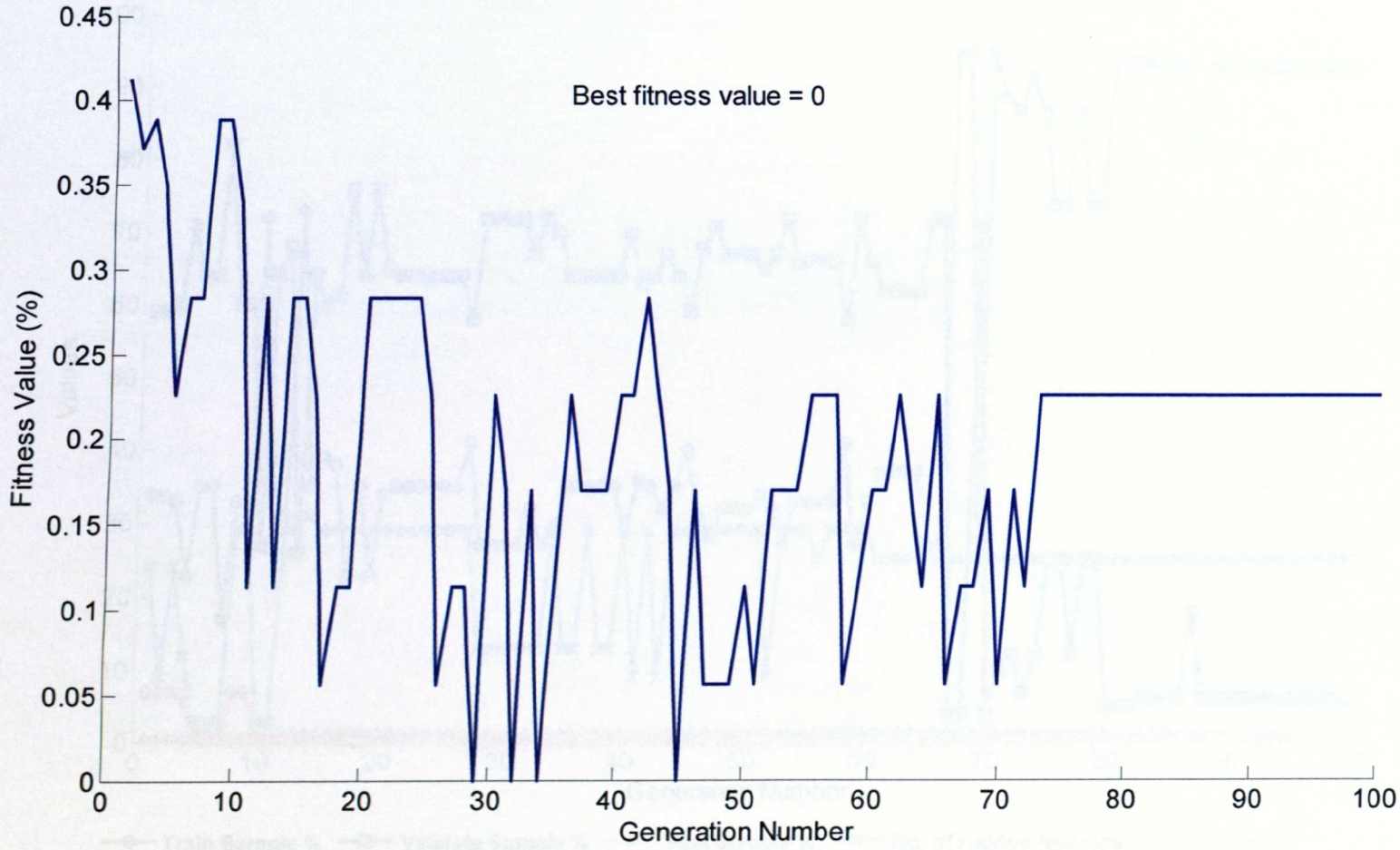


Figure 5-15: Evolution of the elite chromosome (GA5).

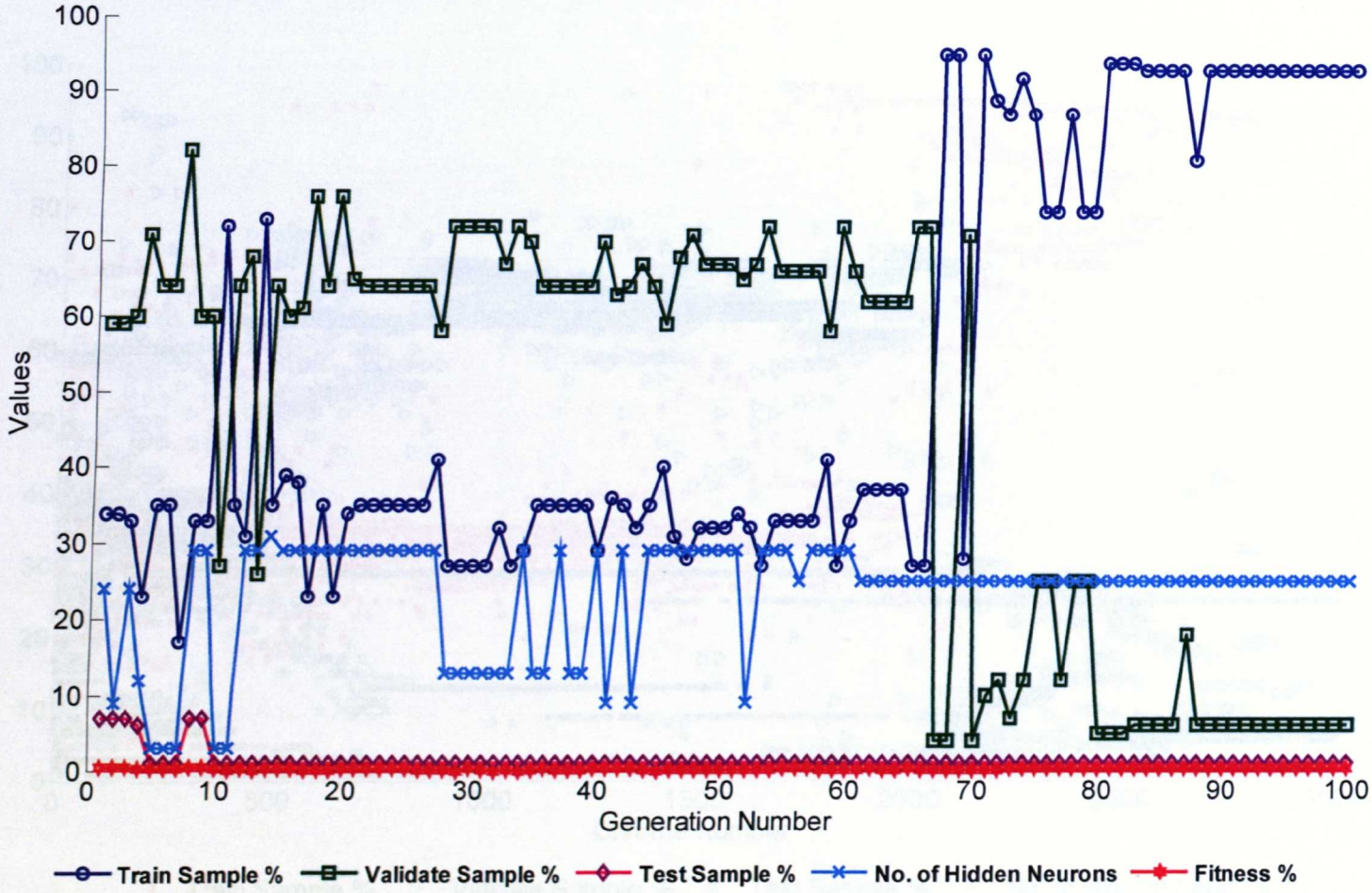


Figure 5-16: Alleles and fitness values of elite chromosomes (GA5).

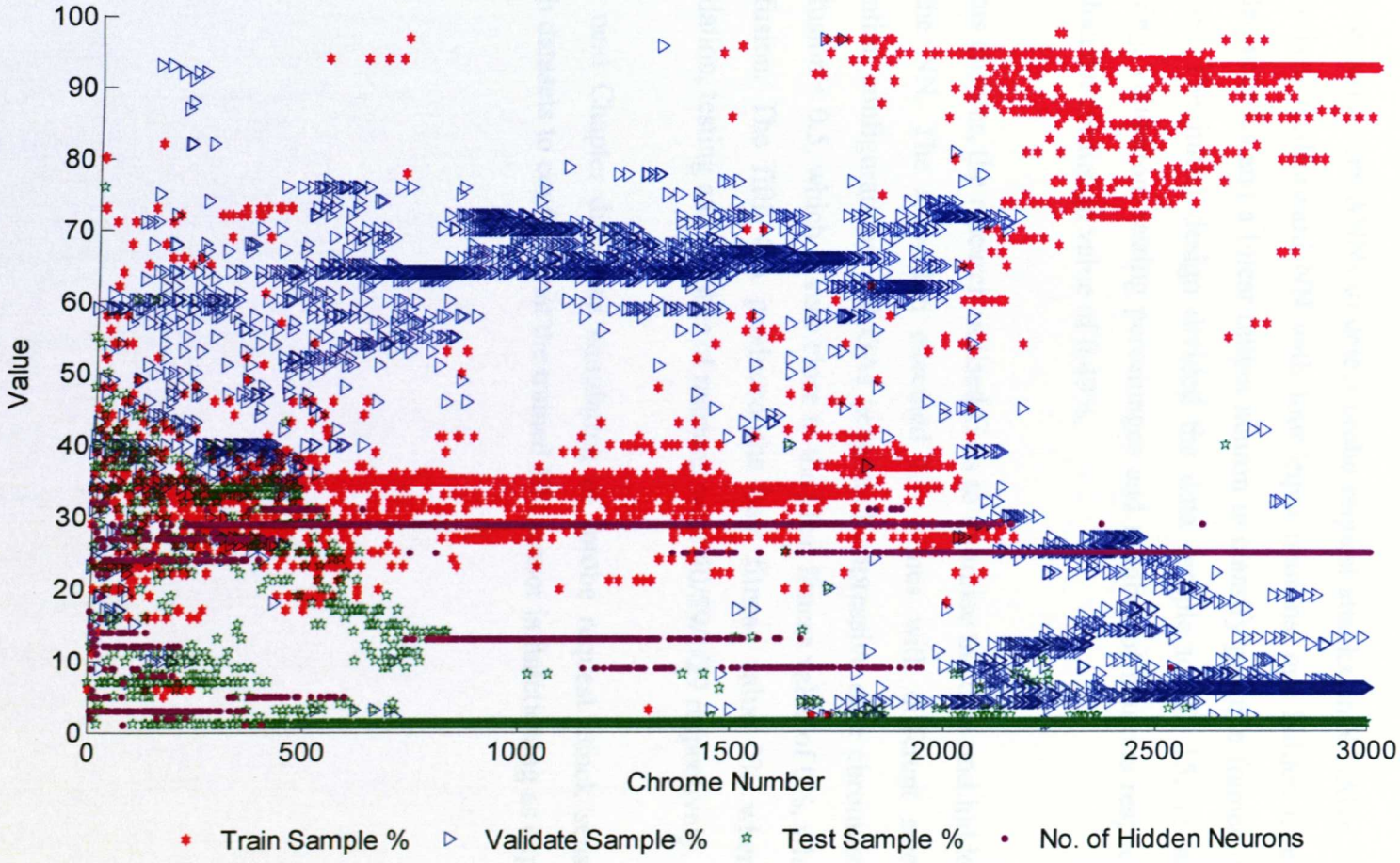


Figure 5-17: Search Space of GA5.

5.5. Summary

This research uses ANNs to detect probe request attacks on WLANs. It utilises a supervised feedforward NN with four input neurons, one hidden layer, with 20 hidden neurons and a linear output neuron to classify genuine frames from rogue frames. The initial design divided the data sample to 70, 15, 15 and 20 for training, validation, testing percentages and number of neurons respectively and produced a confusion value of 0.49%.

In this chapter, the research utilised GAs to optimise the data and hidden neurons of the NN. The algorithm executed five times with different selection and mutation configurations. All GAs generated impressive elite chromosomes with confusion < 0.5, which is very close to the ideal fitness value of 0%, which is zero confusion. The fifth GA produced the best fitness value 0% when training, validation, testing and number of neurons were 40/59/1/29 respectively.

The next Chapter discusses simulating of 'probe request attack sensor', using fresh datasets to confirm that the trained NN sensor is functioning as expected.

Table 5-5: Summary of GAs.

Description	GA1	GA2	GA3	GA4	GA5 Units
Percentage of chromosomes passed to next generation	60	80	60	60	90 %
Probability of mutating a chromosome	0.5	0.5	1	1.5	1.5 %
Stopped at	75	69	100	100	100 generations
Converged generation	49	48	57	12	45 generation
Traits: train, valid, test and neurons	62/36/2/28	37/62/1/20	39/60/1/5	50/49/1/17	40/59/1/29
Best fitness value (test)	0.25	0.17	0.06	0.06	0.00 %
Worst fitness value (test)	0.41	0.38	0.42	0.40	0.41 %
Training confusion	0.43	0.50	0.45	0.44	0.49 %
Validation confusion	0.42	0.47	0.42	0.41	0.47 %
Best Epoch	68	81	77	130	67 epoch

Chapter 6: Simulation Results Discussion

6.1. Introduction

This chapter presents the simulation process of optimised 'probe request attack sensor', using fresh datasets to confirm that the trained NN is operating as expected. Section 3.4.8 describes the framework of a real-world IDS. However, this research is restricted by time and technology. Therefore, the simulation is designed based on available resources. The Chapter is organised as follows: Section 6.2 discusses the simulation process of sensor to detect probe request attacks. Sections 6.3 and 6.4 present the simulation results. Section 6.5 concludes the Chapter.

6.2. Simulation of Classifier

Data capture: Researchers use various forms of simulation data such as: traffic from sample databases, online or offline real-world traffic, or online/offline real-traffic generated from test-beds etc. This research is designed to utilise real-traffic and experience the effects of real WLAN environment whenever possible. Therefore, captures a new set of real-world traffic from the WLAN discussed in Section 4.2, for simulation. In real-world simulation applications, choosing appropriate distributions to represent input data is crucial as; if the distribution of inputs is biased, it will direct to inaccurate conclusions about the system (Guizani et al. 2010). Therefore, after careful considerations,, scenarios shown in Table 6-1 are included in the traffic capture plan.

Table 6-1: Simulation scenarios.

Sim Code	Scenario
Sim51	Unseen dataset from user (Test1-PC)
Sim59	Unseen dataset from user far away from AP (Test1-PC)
Sim54	Unseen dataset from attacker using NetStumbler (Test2-PC)
Sim55	Unseen dataset with attacker at the same location as the user (Test2-PC)
Sim56	Unseen dataset with attacker far away from user (Test2-PC)
Sim57	Unseen dataset with new attacker (Test4-PC) using NetStumbler
Sim58	Unseen dataset with 2 attackers using NetStumbler (Test2-PC and Test4-PC)
Sim60	Unseen dataset with an attacker using a Linux network scanning tool (Test5-PC)
Sim61	Unseen dataset from user and attacker using NetStumbler (known attack) (Test1-PC and Test2-PC)
Sim62	Unseen dataset from user and an attacker using a Linux network scanning tool (Unknown attacker) (Test1-PC and Test5-PC)

Figure 6-1 shows the extended WLAN utilised for simulation. In addition to the attacker Test2-PC utilised in the training capture, two new attackers are utilised for simulation namely: Microsoft Windows based Test4-PC and Linux based Test5-PC. Capturing sessions varied to capture adequate number of frames, approximately 3000 frames per session. Traffics captured from user STAs are normal uncontrolled traffic. However, the research generated the attacks.

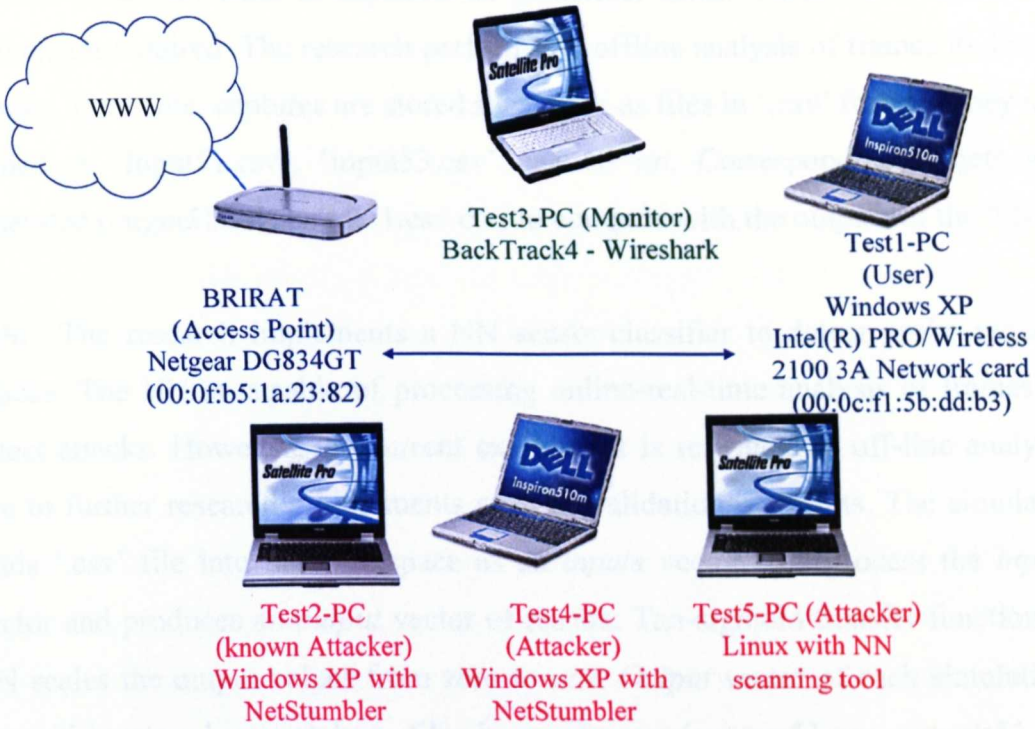


Figure 6-1: Elements of the WLAN for simulation data capture.

The attacker STA Test4-PC is a DELL Inspiron 510M laptop with an Intel® Pentium® 1.6 2GHz microprocessor and 1 MB of RAM, with Microsoft Windows XP OS. NIC is Intel(R) PRO/Wireless 2100 3A. This attacker is spoofed using a commercially available spoofing tool, SMAC 2.0 (KLC_Consulting 2012). Attacker Test5-PC is a Toshiba Satellite Pro laptop with an Intel® Pentium® M 740 (2GHz) microprocessor and 1.9 gigabytes of RAM, with Linux based Ubuntu 9.10 OS. NIC is Netgear WG111T 108 Mbps USB 2.0 Adapter. This attacker is spoofed using freely available macchanger spoofing tool. Frames are captured by the BackTrack4 based monitoring and capturing STA Test3-PC using Wireshark capturing software which has a comprehensive set of GUI and command line tools to configure a capture. Linux uses libpcap protocol to interface with NIC.

Data preparation and storage: Filtering rules are applied to sift only delta time (*frame.time_delta*), sequence number (*wlan.seq*), SSI (*radiotap.dbm_antsignal*) and frame sub-type (*wlan.fc.subtype*) of frames with source address (*wlan.sa*)

00:0c:f1:5b:dd:b3. Data is captured in numerical form. Therefore, there is no conversion required. The research performs an offline analysis of frames to detect attacks. Therefore, captures are stored separately as files in '.csv' format. They are named as 'input51.csv', 'input53.csv' and so on. Corresponding targets are generated (target51.csv, target51.csv etc) to compare with the outputs of the NN.

IDS: The research implements a NN sensor/classifier to detect probe request attacks. The NN is capable of processing online-real-time analysis of frames to detect attacks. However, the current experiment is restricted to off-line analysis due to further research requirements such as validation of results. The simulator reads '.csv' file into the workspace as an *inputs* vector. NN process the *inputs* vector and produces an *output* vector of results. Tan-sigmoid transfer function of NN scales the output values from zero to one. *Output* vector of each simulation, too will be stored separately as files in .csv format (outputs51.csv, outputs51.csv and so on).

Reporting: As discussed in Section 3.4.8, reporting sends reports or alerts of attacks to Security Administrators. In a real-world situation, the probe request detection system receives user frames and/or attack frames continuously and processed results are presented constantly to the Security Administrators. Figure 6-2 presents a sample graph that a Security Administrator will see, when frame capture includes both genuine and attack frames.

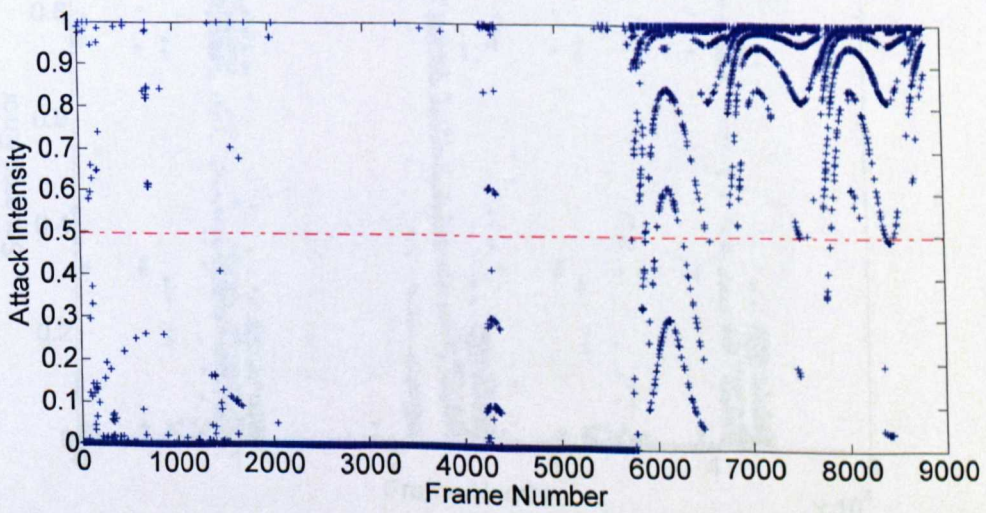


Figure 6-2: Security Administrator's probe request attack monitoring screen.

This figure plots '*output*' values against frame numbers. Output bounds are (0,1). Ideally, the output value should be zero, which means 'no attack'. There are many schools of thought as to how one classifies a frame into an attack or genuine class. This research uses the most common method, that is, frames with 'output' neuron value equal or higher than 0.5 are classified as attack or positive frames (1), whilst others are classified as genuine or negative frames (0). However, in real-world situations, Security Administrators can set the threshold value depending on the degree of sensitivity required. A real-world application also can provide more information on the screen such as the MAC address, time, and other statistics.

Results validation: The result validation compares the actual results with expected results. However, this research cannot verify the accuracy of the detection system from Figure 6-2. Three popular measurements of performance in classification applications are; MSE, confusion matrix, and ROC curve. The research simulated the complete dataset and generated a MSE, confusion matrix, and a ROC curve to investigate the overall performance of the dataset.

6.2.1. Un-optimised Classifier

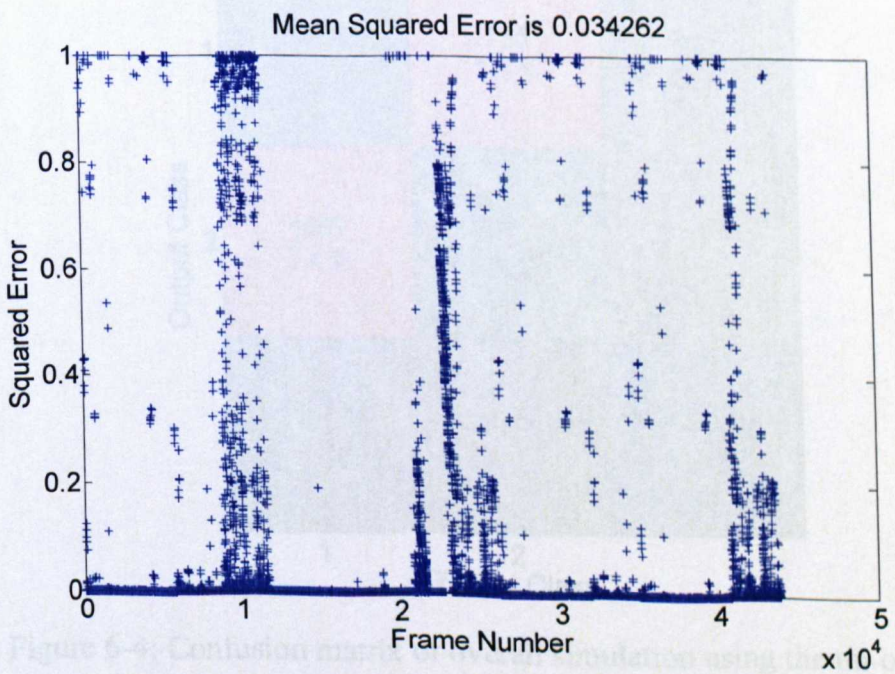


Figure 6-3: MSE of overall simulation with un-optimised classifier.

Figure 6-3 plots the squared difference between the expected value (target) and the actual result (output) of each frame of the complete dataset and produces an error value scaled from zero to one. A frame's error = zero means 'no error', that frame is correctly classified. The MSE of the dataset is 0.034262, which is a value very close to zero that is statistically a good performance. During validation, simulator reads the respective target values from 'target.csv' file which includes ones and zeros to represent attack and no attack respectively into a vector call *targets*. The same threshold value 0.5 applies for classifying frames into member classes. The research classify frames with output value equal or higher than 0.5 as positive frames (1), whilst others classify as negative frames (0). Then TP, TN, FP and FN values are computed comparing output value with target value of each frame. This research uses confusion as an overall measurement of performance of the classifier.

In Figure 6-4, the confusion matrix shows overall results with accuracy of 95.9%, and 4.1% confusion. In Figure 6-5, left and top edge hugging ROC curve confirms consistency of overall results presented in Figure 6-3 and Figure 6-4.

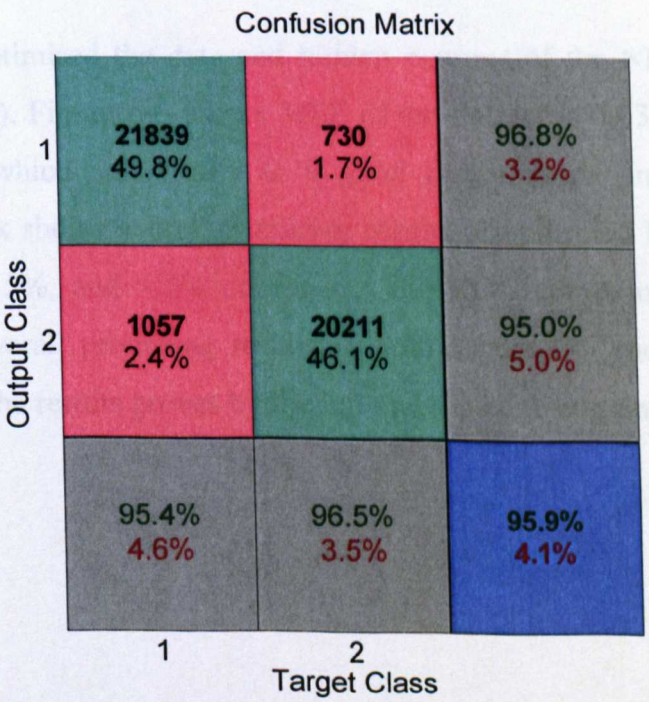


Figure 6-4: Confusion matrix of overall simulation using the un-optimised classifier.

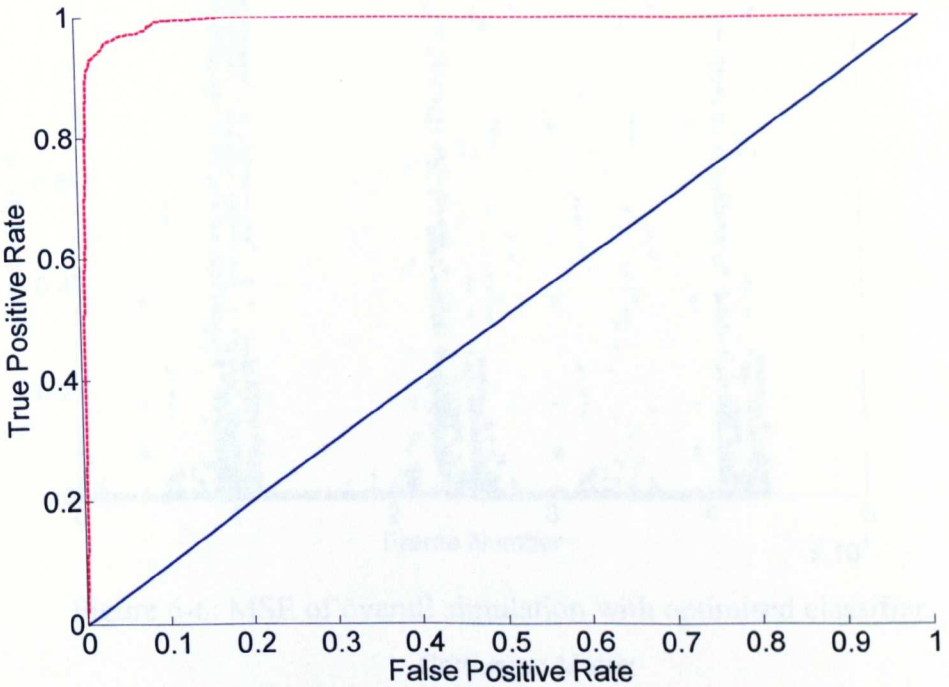


Figure 6-5: ROC curve of overall simulation using the un-optimised classifier.

6.2.2. Optimised Classifier

The research optimised the data and hidden neurons of the NN classifier using GAs (Chapter 5). Figure 6-6 shows MSE of the dataset is 0.03031, a value very close to zero which statistically is a good performance. In Figure 6-7, the confusion matrix shows overall promising results of optimised NN classifier with accuracy of 96.5%, and 3.5% confusion. The ROC curve in Figure 6-8, also confirms the overall promising results in both Figure 6-6 and Figure 6-7. The consistency of the results proves by the left and top edge hugging ROC curve.

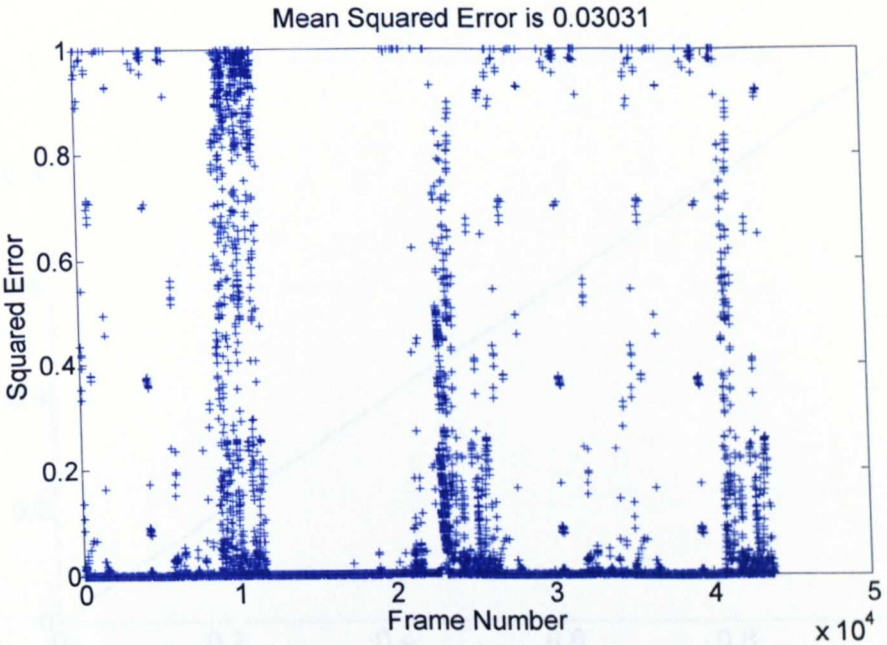


Figure 6-6: MSE of overall simulation with optimised classifier.

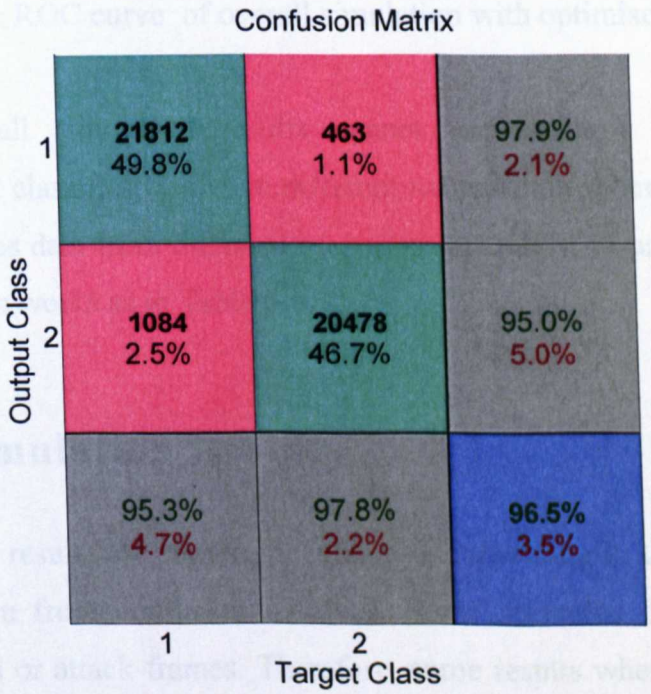


Figure 6-7: Confusion matrix of overall simulation with optimised classifier.

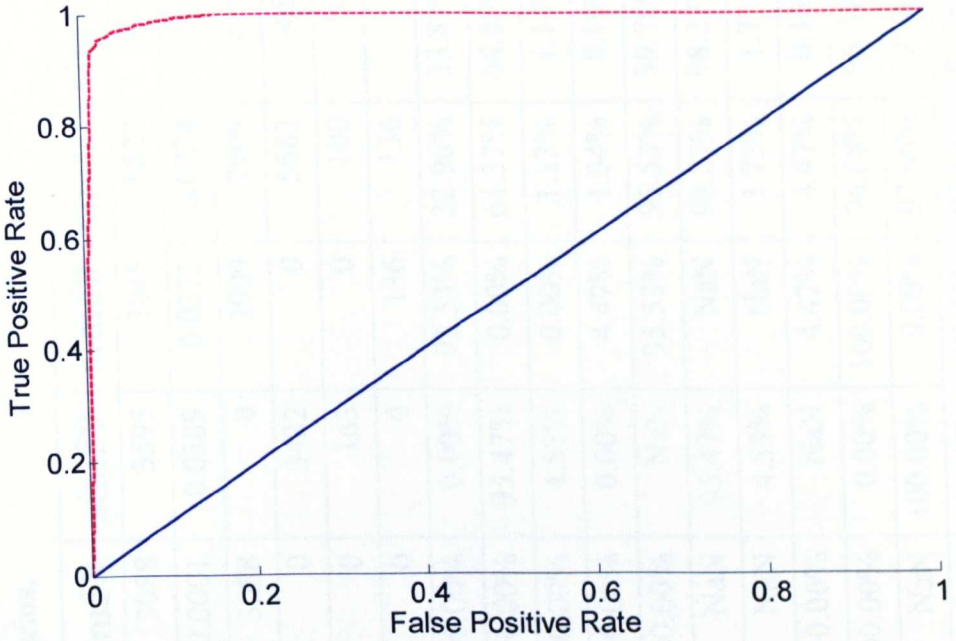


Figure 6-8: ROC curve of overall simulation with optimised classifier.

However, overall simulation results cannot explain how well the sensor understands and classifies frames in different natural atmospheres. Therefore, the research analyses data from different scenarios separately, to understand how the detection system works as in Table 6-1.

6.3. Simulation Results

The simulation results are shown in Table 6-2. Presenting in the table are MSEs and information from confusion matrices. Some scenarios do not have either genuine frames or attack frames. Therefore, some results where output was null, or undefined due to unavailability of genuine or attack frame values, are shown as Not-a-Number (NaN).

Table 6-2: Simulation of real-world scenarios.

	sim51	sim54	sim55	sim56	sim57	sim58	sim59	sim60	sim62	sim61
Complete Sample	5782	2975	2905	2026	2837	3088	3595	3045	8827	8757
MSE-Training	0.0146	0.0026	0.2384	0.0000	0.0000	0.0001	0.0389	0.0372	0.0224	0.0105
TP	0	2966	2111	2837	2837	3088	0	2909	2909	2966
TN	5682	0	0	0	0	0	3432	0	5682	5682
FP	100	0	0	0	0	0	163	0	100	100
FN	0	9	794	0	0	0	0	136	136	9
TP Coverage %	0.00%	99.70%	72.67%	100.00%	100.00%	100.00%	0.00%	95.53%	32.96%	33.87%
TN Coverage %	98.27%	0.00%	0.00%	0.00%	0.00%	0.00%	95.47%	0.00%	64.37%	64.89%
FP Coverage %	1.73%	0.00%	0.00%	0.00%	0.00%	0.00%	4.53%	0.00%	1.13%	1.14%
FN Coverage %	0.00%	0.30%	27.33%	0.00%	0.00%	0.00%	0.00%	4.47%	1.54%	0.10%
Sensitivity TP %	NaN	99.70%	72.67%	100.00%	100.00%	100.00%	NaN	95.53%	95.53%	99.70%
Specificity TN %	98.27%	NaN	NaN	NaN	NaN	NaN	95.47%	NaN	98.27%	98.27%
FP Rate%	1.73%	NaN	NaN	NaN	NaN	NaN	4.53%	NaN	1.73%	1.73%
FN Rate%	NaN	0.30%	27.33%	0.00%	0.00%	0.00%	NaN	4.47%	4.47%	0.30%
+ve Prediction Precision %	0.00%	100.00%	100.00%	100.00%	100.00%	100.00%	0.00%	100.00%	96.68%	96.74%
-ve Prediction Precision %	100.00%	0.00%	0.00%	NaN	NaN	NaN	100.00%	0.00%	97.66%	99.84%
Accuracy %	98.27%	99.70%	72.67%	100.00%	100.00%	100.00%	95.47%	95.53%	97.33%	98.76%
Confusion %	1.73%	0.30%	27.33%	0.00%	0.00%	0.00%	4.53%	4.47%	2.67%	1.24%

Following is the interpretation of the results of Table 6-2.

6.3.1. Detection rate of known and unknown attacks.

Results of simulations sim54, sim56-sim58 show that the sensor accurately detects 99.7% to 100% of known attacks. The results of simulation sim60 show that sensor detects 95.5% of unknown attacks. The classification accuracy of the sensor was 100% when tested with test sample after training and validation (Section 5.4.5). However, whilst the detection rates of known attacks (sim54, sim56-sim58) are 99.7%, 100%, 100%, and 100% respectively, detection of unknown attacks show 4.5% reduction. According to the overall results, the detection rate of known and unknown attacks are promising. However, the sensor is only tested with NetStumbler and Linux scanning tool.

6.3.2. Effect of the movement of an attacker and a user

When NetStumbler attacker is at a general distance or far-away location within the signal range, the detection accuracy is 99.7%- 100% (sim54 and sim56). When NetStumbler attacker is at the same location as the user, the detection accuracy is only 72.7% (sim55). However, this is a nearly impossible scenario in a non-public WLAN. Results of sim51 and sim59 show, that the movement of the user generates approximately 3% (4.53%-1.73%) confusion. When the captured data are analysed, it is observed that when a genuine user scans a network excessively, it could raise a false alarm, because it generates unusually a large number of probe requests. This can occur due to an ill-configured WLAN card, weak signal strength or as in this case, user deliberately scanning the network. This may require security administrator's attention, and can solve within the system by setting a threshold value of warnings to be tolerated per second to suit specific users or network. The results confirm that user's mobility within the signal range does not affect the detection rate very much and therefore, this solution enables the WLAN users to change their location of work in contrast to some experiments which requires user STAs to be static.

When only the attacker is present, The Accuracy% equals the Sensitivity (TP%), and TP Coverage% whilst the Confusion% equals the FN Coverage% and FNR%. Further, the PPP reads as 100% whilst the TN Coverage, the FP coverage and the NPP results are 0%. The Specificity% and FPR% lead to NaN values. Conversely, when only the user is present, the Accuracy% equals the Specificity (TN%) and TN Coverage%, and Confusion% equals the FP Coverage% and FPR%. Further, the NPP reads as 100% whilst the TP Coverage, the FN Coverage and the PPP results are 0%. The Sensitivity% and FNR% lead to NaN values. Therefore, ROC curve graphs cannot be generated in scenarios which either attacker or user is present as ROC curve graphs are generated with specificity and sensitivity values.

6.3.3. Effect of both user and an attacker's presence

The sensor is simulated (sim61) using a random combination of data used for sim51 and sim54 which is an unseen dataset from user and attacker using NetStumbler (known attack). In this scenario, Sensitivity and Specificity of the classifier is 99.7% and 98.2% respectively, which is similar to Sensitivity and Specificity of sim51 and sim54. However, there is a reduction in PPP from 100% to 96.7% and NPP from 100% to 99.8% in sim61. Further, sim61 reports a 1.2% of Confusion, which is a rate remarkably lower than sim51, but slightly higher than sim54. Simulation sim62 utilises a random combination of data used for sim51 and sim60 which is an unseen dataset from user and an attacker, using a Linux network scanning tool (unknown attack). In this scenario too, the Sensitivity (TP%) and Specificity (TN%) of the classifier is 98.3% and 95.4% respectively, which is similar to Sensitivity and Specificity of sim51 and sim60. Again, there is a reduction in PPP from 100% to 96.7% and NPP from 100% to 97.6% in sim61. Further, sim62 reports a 2.7% of Confusion which is slightly less than sim51, and higher than sim60. It is clear that the Confusion percentage slightly increases during an unknown attack. However, this experimentation shows that the sensor could still detect an unknown attack with 97.3% accuracy.

6.3.4. Effect of OS, NIC and attack tools

Although, this research is not formally designed to analyse the effect of OS, NIC and attacker tools on the classifier, the research is keen to know if there is a recognisable effect on any of these features. Therefore, the research tabulates the data available as follows;

Table 6-3: Simulation results with user and attackers that generates data.

Simulation Code	Accuracy %	Sensitivity (TP%)	Specificity (TN%)	User	Attacker
sim51	98.27	NaN	98.27	Test1-PC	Not Available
sim54	99.70	99.70	NaN	Not Available	Test2-PC
sim55	72.67	72.67	NaN	Not Available	Test2-PC
sim56	100.00	100	NaN	Not Available	Test2-PC
sim57	100.00	100	NaN	Not Available	Test4-PC
sim58	100.00	100	NaN	Not Available	Test2-PC and Test4-PC
sim59	95.47	NaN	95.47	Test1-PC	Not Available
sim60	95.53	95.53	NaN	Not Available	Test5-PC
sim62	97.33	95.53	98.27	Test1-PC	Test5-PC
sim61	98.76	99.70	98.27	Test1-PC	Test2-PC

- Simulation codes are defined in Table 6-1.
- The user STA Test1-PC is a DELL Inspiron 510M laptop, runs on Microsoft Windows XP OS with a PRO/WLAN 2100 3A mini PCI NIC.
- Attacker Test2-PC is a Toshiba Satellite Pro laptop, runs on Microsoft Windows XP with an Intel(R) PRO/Wireless 2200 BG NIC, armed with NetStumbler.
- The attacker STA Test4-PC is a DELL Inspiron 510M laptop, runs on Microsoft Windows XP OS, with a Intel(R) PRO/Wireless 2100 3A mini PCI NIC, armed with NetStumbler.
- Attacker Test5-PC is a Toshiba Satellite Pro laptop, runs on Linux based Ubuntu 9.10 OS with a Netgear WG111T 108 Mbps USB 2.0 NIC, armed with Linux scanning tool.

Test2, and Test4 runs on Microsoft Windows XP and attacks with NetStumbler. Attacker Test5 runs on Ubuntu and attacks with Linux scanning tool. sim54, sim57 and sim60 performs in similar environments. However, the results show that the sensor detects attacker frames from Windows STAs more accurately than the Ubuntu STA. Sensitivities of sim61 and sim62 also confirm this trend. One explanation that the research can provide for this result is that the sensor is trained using attack data from a Windows based STA and therefore has a higher competence to detect frames from Windows and NetStumbler based attacks. However, further research is required to justify this rationale. Test1 and Test4 are identical STAs. Therefore, research expected confusion, as theoretically, they should generate similar signal strengths. However, although there is some similarity in the signal strengths, it seems that it does not affect the results as sim58 has generated 100% accuracy.

6.4. Other Simulation Results

Although, the above scenarios generate excellent results, the research extends the investigation further to collect data simultaneously from a user and attacker for simulation. For this purpose, the research creates sim-63 scenario: research disables the ACL of the AP, utilises Test5-PC (unknown attacker) with its original MAC address, using a Linux network-scanning tool, utilises user Test1-PC operating under its normal conditions. This configuration enables AP responding to probe requests from any STA including attacker STA, as AP does not know users from attackers until they authenticate using a network key. Further, as traffic from user and attacker STAs can be identified easily by the MAC address, the traffic can be labelled as rogue and genuine. This enables comparing target and output values of the classifier.

Simulation results of scenario (Sim63): Unseen dataset from user (Test1-PC) and an unknown attacker (Test5-PC) using a Linux network-scanning tool.

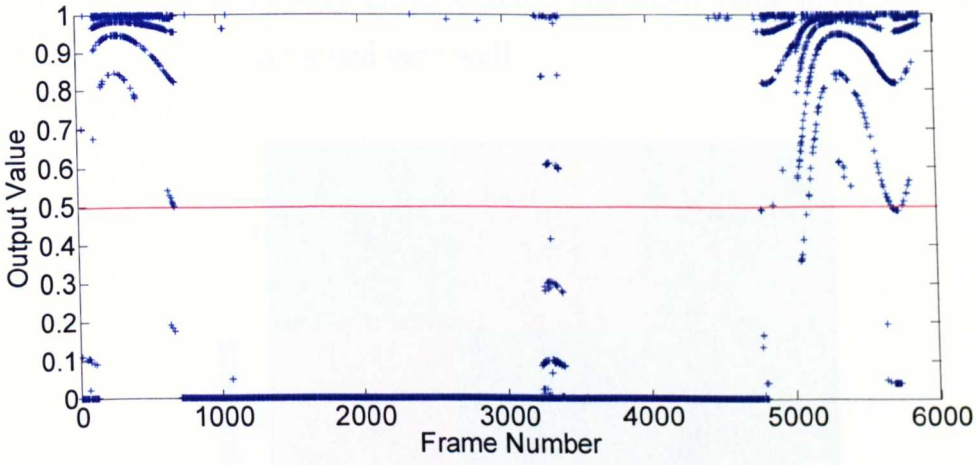


Figure 6-9: Sensor output of Sim63.

Figure 6-9 plots the sensor output value against each frame number. As the threshold value is set to 0.5, all frames with output value greater than 0.5 can be considered as possible attack frames. All frames with output value less than or equal to 0.5, can be considered as genuine user's frames. The research further analyses the output, using standard performance criterion: MSE, Confusion Matrix and ROC curve .

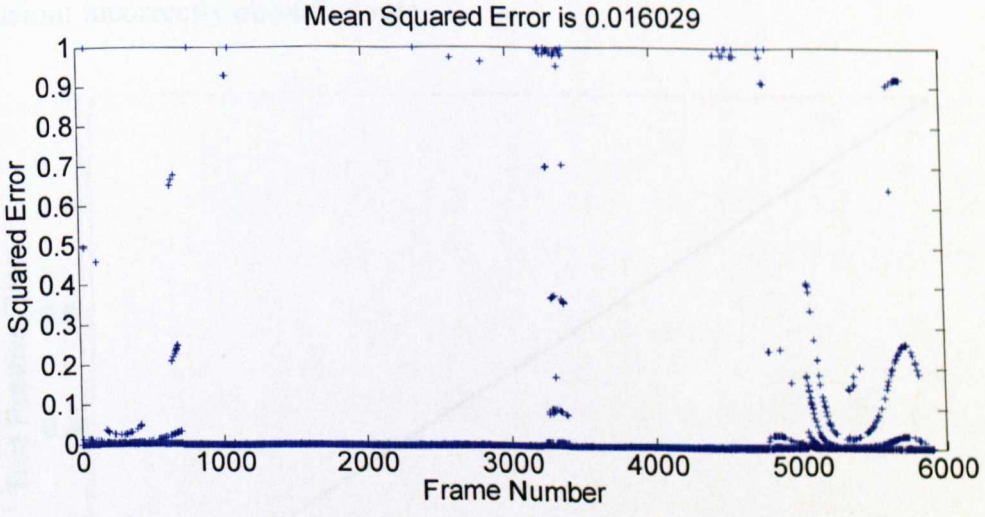


Figure 6-10: MSE of Sim63.

Figure 6-10 presents the squared values of 'error' or 'residual. Error is the difference between the target and output, which indicates the how well or badly the sensor has performed when analysing each frame. Figure 6-10 shows that most of the frames are

with no error or an error very close to zero. The MSE value 0.016029 confirms that overall, the sensor has performed very well.

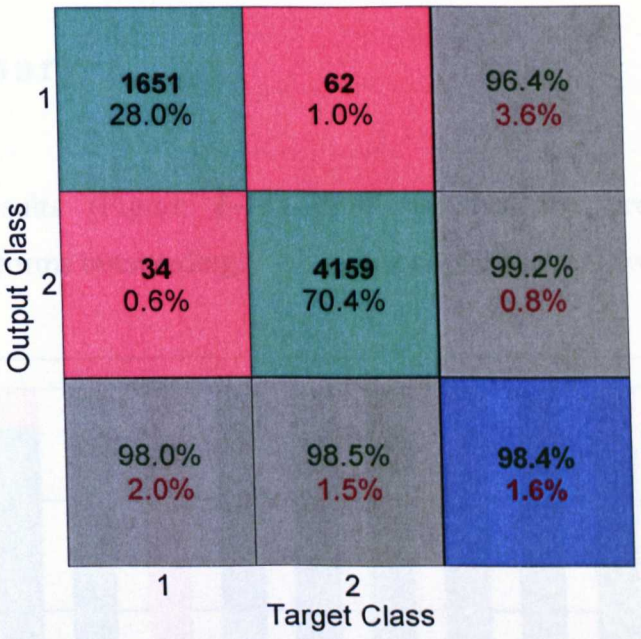


Figure 6-11: Confusion Matrix of Sim63.

The research further analyses the classification performance. Figure 6-11 shows that overall classification accuracy of the classifier is 98.4%, which gives only 1.6% of confusion: incorrectly classified rate.

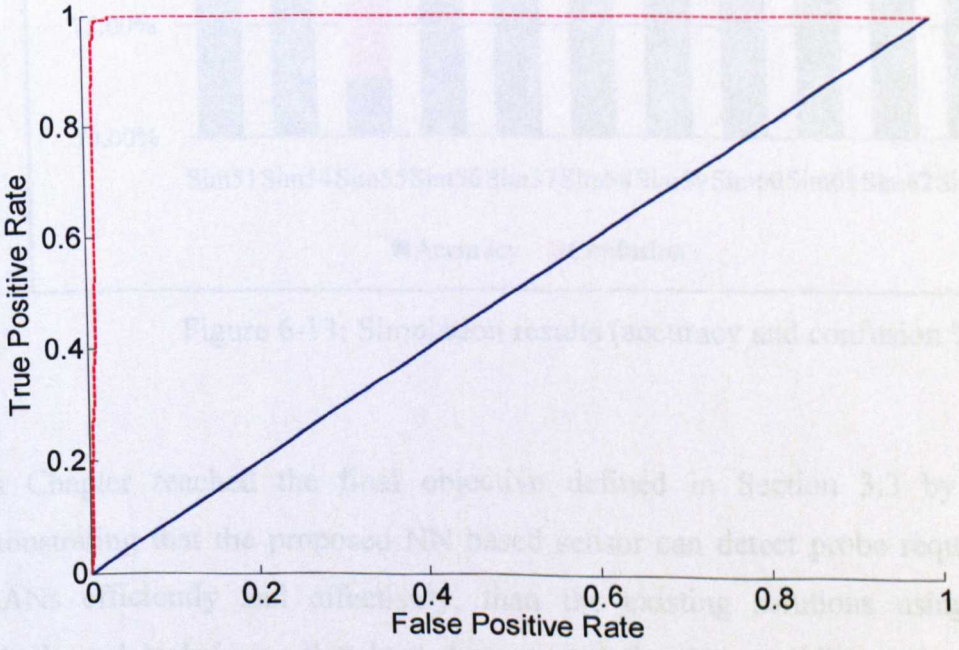


Figure 6-12: ROC curve of Sim63

Figure 6-12 confirms the results and the consistency of classification accuracy, from the left and top edge hugging ROC curve.

6.5. Summary

The simulation results (Figure 6-13) confirms that the probe request attack sensor/classifier performs outstandingly when it is applied to real-world WLAN data.

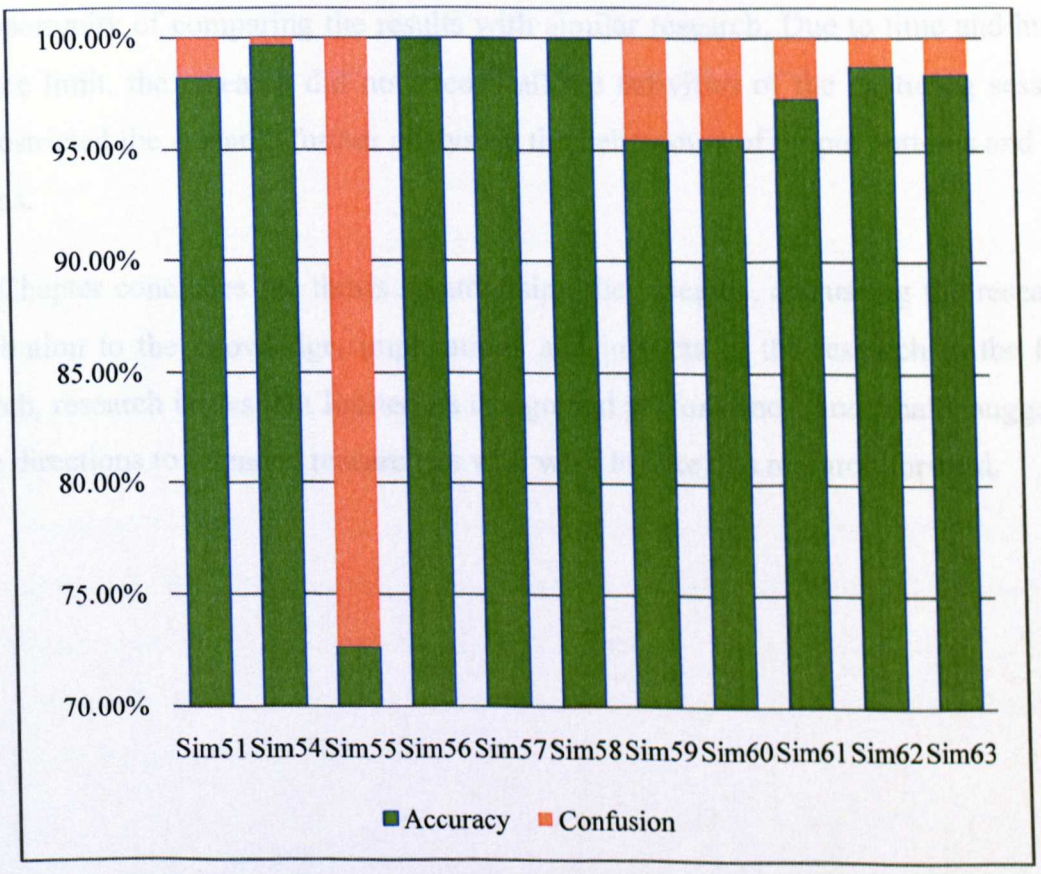


Figure 6-13: Simulation results (accuracy and confusion %).

This Chapter reached the final objective defined in Section 3.3 by successfully demonstrating that the proposed NN based sensor can detect probe request attacks in WLANs efficiently and effectively, than the existing solutions using procedures, methods and techniques that have been tested for their validity and reliability, and

therefore producing an unbiased and an objective design. However, the research also has some limitations and issues. The sensor/classifier performed disappointingly when the attacker is located at the same place as the trained user. Although, it is a rare scenario, it is not an impossible situation. Therefore, the research requires investigating this issue further. In a real IDS, the capturing and sensor/classification functions may be installed in the same computer. However, the research performs an offline batch operations for data capture, preparation, and simulation. Therefore, there may be practical issues that this research may have overlooked. Working on a real WLAN, and real data may lead to providing more practical and accurate IDSs. However, it ceases the opportunity of comparing the results with similar research. Due to time and human resource limit, the research did not record all the activities of the capturing sessions. This restricted the research further analysing the behaviours of output patterns and error patterns.

Next Chapter concludes the thesis summarising the research, discussing the research's contribution to the knowledge, implications and impacts of the research to the future research, research issues that limited its design and performance, and finally suggesting future directions to intended researchers who wish to take this research forward.

Chapter 7: Conclusion

In WLANs, beacon, probe request and response messages are unprotected, so the information is visible to sniffers. Probe requests can be sent by anyone with a legitimate MAC address, as association to the network is not required at this stage. Legitimate MAC addresses can be easily spoofed to bypass AP access lists. Attackers take advantage of these vulnerabilities and send a flood of probe request frames which can lead to a DoS to legitimate STAs. This research realised in Chapter 2, although, many research solutions and commercial products available for intruder detections, the problem of DoS attacks remains same, largely due to complexity of solutions or not identifying real-world practical implementation issues. This research identifies a probing external attacker by analysing the traffic generated from a user MAC address in a single frequency band of a WLAN. A supervised feedforward NN sensor/classifier with four distinct inputs, delta-time, sequence number, signal strength and frame sub-type is applied to identify and differentiate a genuine frame from a rogue one. This research utilises a user STA of a wireless home network with an AP and seven other user STAs for this research, to enable experimenting with real WLAN traffic. The user STAs are Windows XP based PCs or laptop computers. However, the research introduced a Microsoft Windows based attacker armed with NetStumbler (Test2-PC) for training, and additional 2 attackers (Microsoft Windows based Test4-PC armed with NetStumbler and Linux based Test5-PC armed with Linux network scanning tool) for simulation.

This research implemented a prototype of the proposed design (Section 4.3) firstly (Ratnayake, D et al. 2011), as a function approximation application. The research trained the network using Levenberg Marquardt backpropagation algorithm. MATLAB function *'trainlm'* (MathWorks 2012h) is applied for training. Performance is evaluated using linear regression value R2, and MSE. Section 3.4.6 discusses linear regression. The sensor trained outstandingly producing 0.9804 of overall regression, and MSEs 0.0039, 0.0038 and 0.0037 on training,

validation and test samples. Simulation of classifier in eight scenarios with 1000 frame samples resulted only in an average of 94.5% detection rate. The research also optimised the classifier using GA with fitness function as R2 value (Ratnayake, DN, Kazemian & Yusuf 2012). The best-fit value 0.9782, outperformed similar research where the results have been obtained using synthetic traffic (Gong, Zulkernine & Abolmaesumi 2005; Haddadi et al. 2010; Hua & Xiaofeng 2008; Xia et al. 2005). However, it never reached the R2 value obtained in Ratnayake, Kazemian, Yusuf, & Abdullah (2011). Conventionally, probe request attack detection is a binary classification application where, the output can only have two values 1 (attack) or 0 (no attack). The research applied standard linear regression, and treated the output as if it is binary classifying any value of 0.5 or above treated as a '1', and anything below 0.5, as a '0'. Although standard linear regression has been applied successfully for classification in the past and in current research applications, statisticians argue that linear regression should not be used as binary classification applications, as it violates many assumptions of linear regression (Ciuiu 2007; Peng, Lee & Ingersoll 2002; Statgun 2007).

Secondly, the research implemented the design discussed in Section 4.3, as a binary classification application that this thesis has been discussing in subsequent Chapters. The research trained the network using SCG backpropagation algorithm (MATLAB function 'trainscg'). It uses tan-sigmoid transfer function (tansig) in both hidden and output layers. Instead of linear regression, logistic regression is introduced in statistics to measure binary classification performance. However, the logistic regression is not straight forward as linear regression in decision-making and interpreting. Therefore research decided to measure performance of the classifier using MSE, classification error, and ROC curve. Fitness of data for the classifier is evaluated using self-consistency test. The MSEs and confusion rates of training, validation and test samples are 0.0043, 0.0045, 0.0043, 0.47%, 0.50%, 0.47% respectively (Table 4-5) The left and top hugging ROC curve confirmed the consistency of results (Figure 4-12). Five-fold cross-validation test is conducted to evaluate performance of the classifier with unseen data using five

different sets of training and test data samples. The results are very much alike in every test. However, Fold4 produced the lowest test confusion rate of 0.12% (Figure 4-8). During simulation of Fold4 classifier with unseen real-world data, it classified frames with 95.9% of overall accuracy, generating 4.1% of confusion. To improve the results, the research utilised GA to optimise the data of the classifier, namely; train, validation, test sample percentages, and number of neurons. The most optimised NN classifier, with training, validation and test and sample sizes 40%, 59%, 1%, and hidden neurons 29 produced zero confusion on test sample. The accuracy of training, validation and test is, 99.51%, 99.53% and 100%, respectively. When presented with new unseen real-world data, the optimised classifier produced improved results generating 96.5% of overall accuracy, and generating only 3.5% of confusion. The MSE of overall simulation is 0.03031. The left and top hugging ROC curve confirms the consistency of the results.

7.1. Contribution to knowledge

a. Probing is the first communication that and active intruder will perform with an AP. This sensor/classifier detects probing of adversaries. Therefore, it can detect an attack during an early stage of the communication. This early detection will give an opportunity to take precautions to prevent attacks that are more advanced.

b. The experimental results demonstrate that the NN classifier can detect probe request attacks to a very high degree. The solution also works when network prioritisation services like QoS is enabled, and works well when the genuine user is offline. Furthermore, the solution does not limit the user STA's movement within the network. Monitoring only delta-time, sequence number, signal strength indicator, and frame sub-type considerably reduces the overhead of the monitoring machine whilst producing the results expected.

c. This research's proof-of-concept study discards many arguments against sequence number and SSI. Delta-time, sequence number, SSI, and frame sub-type are nearly impossible to manipulate at any one time. SSI and delta time is computed by the capturing STA thereby manipulation of these two fields at the attackers end is not possible. Frame sub-type and sequence number can be replayed or synchronised by adversaries. However, the precision of these synthetic frames is questionable.

d. The new feature of this approach is to capture the user and attacker-training data separately, label and join them prior to training, so that the security administrator does not have to manually identify rogue and genuine frames. Further, this also makes it easier for housekeeping of training data, as administrators can remove unnecessary parts of training data easily, and add new training data without having to re-capture already available data in circumstances such as replacing or upgrading a STA. Therefore, this is an efficient, lightweight and low-cost solution, compared to solutions currently available, which need super powered capturing STAs.

e. This research broadly categorised the existing research into conventional and intelligent methods. Most conventional methods use statistical, rule based and state-based techniques. Although, many of these solutions are based on real world data, they have a major inherited weakness of lacking the flexibility to environmental changes, which require frequent time-consuming updates where as this research uses intelligent methods that adapt to environmental changes. Existing intelligent methods provide a broad range of techniques. However, many of them are built on sample databases and propose complex techniques to analyse data. This research combined the knowledge from conventional and intelligent methods and developed the classifier utilising state-of-the-art techniques NNs and GA, using real-world data. Research also analyses the results based on different user and attacker scenarios which give a clear idea of performance of the classifier in practical data-to-day situations that a security administrator experience.

7.2. Research Issues

a. One of the biggest limitations of this study was computer resources.

b. This solution is limited to detect probe request attacks only whilst, a real-world network may experience a cocktail of attacks.

c. This solution cannot prevent probing attacks.

d. This solution cannot detect any adversary not emitting any frames.

e. Although this research is intending to detect only probing attacks, it may detect attacks other than probing attacks.

f. Mostly used WLAN cards today have IEEE 802.11g and n standards that have a maximum data transfer rates of 54 and 600 Mbps, respectively. Hence, when the number of participating STAs in the WLAN increases, the number of frames to be captured and processed by the WIDS is also increases. This may slow down the sensor/classifier.

7.3. Implications and Impacts

a. The research analysed the results based on detection rate of known and unknown attacks, effect of the movement of an attacker and a user, effect of both user and an attacker's presence, effect of OSs, NIC and attack tools.

b. Detection rates of known and unknown attacks (sim54, sim56, sim57, sim58, sim60) are promising (95.5%-100%). However, the sensor is only tested with Microsoft Windows based NetStumbler and Linux scanning tool.

c. Probing is an attacker's first communication with network. Therefore detecting or preventing them can deter or prevent anyone who is intending higher attacks.

7.4. Future Work

This work opens a flood of new areas for future research. The following points are presented here with the hope that researchers will further document and explore:

a. The research currently captures the WLAN traffic using a single NIC. However, in a busy network, the capture STA may lose some frames. Capturing with more than one NICs, will enable capturing more frames. This will also enable channel hopping thereby, enabling capture from the entire bandwidth

b. This research captures traffic from a single bandwidth at anyone time. The research can also be extended to capture from multiple STAs and synchronising them. Monitoring the entire bandwidth enables not only monitoring suspicious activities of adversaries, but also enable monitoring to identify misbehaving user STAs: i.e. user STAs that may plan to attack the neighbourhood WLANs.

c. This research is designed to detect rogue probe frames; however, it was observed that it can also detect MAC spoofing. Further research can be conducted to analyse the extent of MAC spoof detection capability of this design.

d. This research simulated only 10 scenarios (Table 6-1). The simulations can be performed with more scenarios with different configuration of computers and different attack tools.

e. This research captures real-time traffic, however, filtering and simulation is performed on an offline-batch. The future research can be extended to perform, online real-time capturing, filtering and simulation.

f. This research trained and tested the sensor/classifier on a single AP WLAN. The research can be extended to train and test on multiple AP WLANs by adding SSID in the list of fields.

g. This research improved performance by optimising data and number of hidden neurons. Literature suggest that NN performance can be improved by

initialising the network and training, increasing the number of hidden neurons and layers, using a different training function, or using additional training data (MathWorks 2012d).

h. The future of IDS lies in data correlation. The IDS of tomorrow will produce results by examining input from several different sources. However, future research should balance the complexity and the practical implementation issues of their solutions.

Appendix A: IEEE 802.11-2007 Probe response frame

Order	Information
1	Timestamp
2	Beacon interval
3	Capacity
4	SSID
5	Supported rates
6	FH Parameter Set - only from STAs using FH PIIYs
7	DS Parameter Set - only from STAs using Clause 15, Clause 18, and Clause 19 PHYs.
8	CF Parameter Set - only from APs supporting a PCF.
9	IBSS Parameter Set - only from STAs in an IBSS.
10	Country - if dot11MultiDomainCapabilityEnabled or dot11SpectrumManagementRequired is true.
11	FH Parameters - if dot11MultiDomainCapabilityEnabled is true.
12	FH Pattern Table - if dot11MultiDomainCapabilityEnabled is true.
13	Power Constraint - if dot11SpectrumManagementRequired is true.
14	Channel Switch Announcement - if dot11SpectrumManagenientRequired is true.
15	Quiet - if dot11SpectrumManagementRequired is true.
16	IBSS DFS - if dot111SpectrumManagementRequired is true in an IBSS.
17	TPC Report - if dot11SpecirumManagementRequired is true.
18	ERP Information - from STAs using ERPs and is optionally present in other cases.
19	Extended Supported Rates - only whenever there are more than 8 supported rates
20	RSN - only from STAs that have dot11RSNA-Enabled set to TRUE.
21	BSS Load - when dot11QosOption-implemented and dot11QBSSLoad implemented are both true.
22	EDCA Parameter Set - when dot11QosOptionimplemented is true.
Last-i	Vendor Specific - One or more, except the Requested Information elements.
Last-n	Requested information elements of the Probe Request frame.

**If the dot11MultiDomainCapabilityEnabled attribute is true, the Probe Response frame will contain a Country information element, and all information elements identified by the Requested Element IDs of a Request information element (IEEE 2007, p. 84) (IEEE 2007, p.84).*

An illustration of a probe response frame is available in (Gast 2005, p. 108).

Appendix B: Wireshark WLAN: Probe Request & Response Frames

Frame 65 (40 on wire, 40 captured)
:
: Frame Statistics information ...
:

:
: Radiotap Header information ...
:
IEEE 802.11
Type/Subtype: Probe Request (4)
Frame Control: 0x0040
Version: 0
Type: Management frame (0)
Subtype: 4
Flags: 0x0
DS status: Not leaving DS or network is operating in AD-HOC mode (To DS: 0 From DS: 0) (0x00)
.... .0.. = Fragments: No fragments
.... 0... = Retry: Frame is not being retransmitted
...0 .... = PWR MGT: STA will stay up
..0. .... = More Data: No data buffered
.0.. .... = WEP flag: WEP is disabled
0... .... = Order flag: Not strictly ordered
Duration: 0
Destination address: ff:ff:ff:ff:ff:ff (ff:ff:ff:ff:ff:ff)
Source address: 00:02:2d:0d:06:c2 (Agere_0d:06:c2)
BSS Id: ff:ff:ff:ff:ff:ff (ff:ff:ff:ff:ff:ff)
Fragment number: 0
Sequence number: 11
IEEE 802.11 wireless LAN management frame
Tagged parameters (16 bytes)

```
    Tag Number: 0 (SSID parameter set)
    Tag length: 8
    Tag interpretation: xyzzyium
    Tag Number: 1 (Supported Rates)
    Tag length: 4
    Tag interpretation: Supported rates: 1.0 2.0 5.5 11.0 [Mbit/sec]

0000  40 00 00 00 ff ff ff ff ff ff 00 02 2d 0d 06 c2   @...........-...
0010  ff ff ff ff ff ff b0 00 00 08 XX YY ZZ ZZ YY 69   ..........xyzzyi
0020  75 6d 01 04 02 04 0b 16                          um......


Frame 66 (63 on wire, 63 captured)
:
: Frame Statistics information ...
:


:
: Radiotap Header information ...
:
IEEE 802.11
   Type/Subtype: Probe Response (5)
   Frame Control: 0x0050
      Version: 0
      Type: Management frame (0)
      Subtype: 5
      Flags: 0x0
         DS status: Not leaving DS or network is operating in AD-HOC mode (To DS: 0
 From DS: 0) (0x00)
         .... .0.. = Fragments: No fragments
         .... 0... = Retry: Frame is not being retransmitted
         ...0 .... = PWR MGT: STA will stay up
         ..0. .... = More Data: No data buffered
         .0.. .... = WEP flag: WEP is disabled
         0... .... = Order flag: Not strictly ordered
      Duration: 258
      Destination address: 00:02:2d:0d:06:c2 (Agere_0d:06:c2)
```

```
Source address: 00:02:2d:2a:1f:d1 (Agere_2a:1f:d1)
BSS Id: 00:02:2d:2a:1f:d1 (Agere_2a:1f:d1)
Fragment number: 0
Sequence number: 263
IEEE 802.11 wireless LAN management frame
  Fixed parameters (12 bytes)
    Timestamp: 0x000000B53A4FC142
    Beacon Interval: 0.102400 [Seconds]
    Capability Information: 0x0001
      .... ...1 = ESS capabilities: Transmitter is an AP
      .... ..0. = IBSS status: Transmitter belongs to a BSS
      ...0 .... = Privacy: AP/STA cannot support WEP
      ..0. .... = Short Preamble: Short preamble not allowed
      .0.. .... = PBCC: PBCC modulation not allowed
      0... .... = Channel Agility: Channel agility not in use
      CFP participation capabilities: No point coordinator at AP (0x0000)
  Tagged parameters (27 bytes)
    Tag Number: 0 (SSID parameter set)
    Tag length: 8
    Tag interpretation: xyzzyium
    Tag Number: 1 (Supported Rates)
    Tag length: 4
    Tag interpretation: Supported rates: 1.0(B) 2.0(B) 5.5 11.0 [Mbit/sec]
    Tag Number: 3 (DS Parameter set)
    Tag length: 1
    Tag interpretation: Current Channel: 1
    Tag Number: 128 (Reserved tag number)
    Tag length: 6
    Tag interpretation: Not interpreted

0000  50 00 02 01 00 02 2d 0d 06 c2 00 02 2d 2a 1f d1   P.....-.....-*..
0010  00 02 2d 2a 1f d1 70 10 42 c1 4f 3a b5 00 00 00   ..-*..p.B.O:....
0020  64 00 01 00 00 08 XX YY ZZ ZZ ZZ ZZ YY 6d 01 04   d.....xyzzyium..
0030  82 84 0b 16 03 01 01 80 06 00 60 1d 00 36 00      ..........`..6.
```

Appendix C: Wireshark IEEE 802.11 Frame Statistics

Field name	Type	Description
frame.cap_len	Unsigned 32-bit integer	Frame length captured
frame.coloring_rule.name	String	Coloring Rule Name
frame.coloring_rule.string	String	Coloring Rule String
frame.file_off	Signed 64-bit integer	File Offset
frame.ignored	Boolean	Frame is ignored
frame.len	Unsigned 32-bit integer	Frame length on the wire
frame.link_nr	Unsigned 16-bit integer	Link Number
frame.marked	Boolean	Frame is marked
frame.md5_hash	String	Frame MD5 Hash
frame.number	Unsigned 32-bit integer	Frame Number
frame.p2p_dir	Signed 8-bit integer	Point-to-Point Direction
frame.pkt_len	Unsigned 32-bit integer	Frame length on the wire
frame.protocols	String	Protocols in frame
frame.ref_time	None	Time Reference frame
frame.time	Date/Time stamp	Arrival Time
frame.time_delta	Time duration	Time since previous captured frame
frame.time_delta_displayed	Time duration	Time since previous displayed frame
frame.time_epoch	Time duration	Epoch Time
frame.time_invalid	None	Arrival Timestamp invalid
frame.time_relative	Time duration	Time since reference or first frame

Appendix D: Wireshark IEEE 802.11 Radio Tap Information

Field name	Type	Description
radiotap.antenna	Unsigned 32-bit integer	Antenna
radiotap.channel	Unsigned 32-bit integer	Channel
radiotap.channel.freq	Unsigned 32-bit integer	Channel frequency
radiotap.channel.type	Unsigned 16-bit integer	Channel type
radiotap.channel.type.2ghz	Boolean	2 GHz spectrum
radiotap.channel.type.5ghz	Boolean	5 GHz spectrum
radiotap.channel.type.cck	Boolean	Complementary Code Keying (CCK)
radiotap.channel.type.dynamic	Boolean	Dynamic CCK-OFDM
radiotap.channel.type.gfsk	Boolean	Gaussian Frequency Shift Keying (GFSK)
radiotap.channel.type.gsm	Boolean	GSM (900MHz)
radiotap.channel.type.half	Boolean	Half Rate Channel (10MHz Channel Width)
radiotap.channel.type.ofdm	Boolean	Orthogonal Frequency-Division Multiplexing (OFDM)
radiotap.channel.type.passive	Boolean	Passive
radiotap.channel.type.quarter	Boolean	Quarter Rate Channel (5MHz Channel Width)
radiotap.channel.type.sturbo	Boolean	Static Turbo
radiotap.channel.type.turbo	Boolean	Turbo
radiotap.channel.xtype.passive	Boolean	Passive
radiotap.datarate	Unknown	Data rate (Mb/s)
radiotap.db_antnoise	Unsigned 32-bit integer	SSI Noise (dB)
radiotap.db_antsignal	Unsigned 32-bit integer	SSI Signal (dB)
radiotap.db_txattenuation	Unsigned 16-bit integer	Transmit attenuation (dB)
radiotap.dbm_antsignal	Signed 32-bit integer	SSI Signal (dBm)
radiotap.fcs	Unsigned 32-bit integer	802.11 FCS
radiotap.fcs_bad	Boolean	Bad FCS
radiotap.fhss.hopset	Unsigned 8-bit integer	FHSS Hop Set
radiotap.fhss.pattern	Unsigned 8-bit integer	FHSS Pattern
radiotap.flags	Unsigned 8-bit integer	Flags
radiotap.flags.badfcs	Boolean	Bad FCS
radiotap.flags.cfp	Boolean	CFP
radiotap.flags.datapad	Boolean	Data Pad
radiotap.flags.fcs	Boolean	FCS at end
radiotap.flags.frag	Boolean	Fragmentation
radiotap.flags.preamble	Boolean	Preamble
radiotap.flags.shortgi	Boolean	Short GI
radiotap.flags.wep	Boolean	WEP

radiotap.length	Unsigned 16-bit integer	Header length
radiotap.mactime	Unsigned 64-bit integer	MAC timestamp
radiotap.pad	Unsigned 8-bit integer	Header pad
radiotap.present	Unsigned 32-bit integer	Present flags
radiotap.present.antenna	Boolean	Antenna
radiotap.present.channel	Boolean	Channel
radiotap.present.db_antnoise	Boolean	DB Antenna Noise
radiotap.present.db_antsignal	Boolean	DB Antenna Signal
radiotap.present.db_tx_attenuation	Boolean	DB TX Attenuation
radiotap.present.dbm_antnoise	Boolean	DBM Antenna Noise
radiotap.present.dbm_antsignal	Boolean	DBM Antenna Signal
radiotap.present.dbm_tx_attenuation	Boolean	DBM TX Attenuation
radiotap.present.ext	Boolean	Ext
radiotap.present.fcs	Boolean	FCS in header
radiotap.present.fhss	Boolean	FHSS
radiotap.present.flags	Boolean	Flags
radiotap.present.lock_quality	Boolean	Lock Quality
radiotap.present.rate	Boolean	Rate
radiotap.present.rxflags	Boolean	RX flags
radiotap.present.tsft	Boolean	TSFT
radiotap.present.tx_attenuation	Boolean	TX Attenuation
radiotap.present.xchannel	Boolean	Channel+
radiotap.quality	Unsigned 16-bit integer	Signal Quality
radiotap.rxflags	Unsigned 16-bit integer	RX flags
radiotap.rxflags.badplcp	Boolean	Bad PLCP
radiotap.txattenuation	Unsigned 16-bit integer	Transmit attenuation
radiotap.txpower	Signed 32-bit integer	Transmit power
radiotap.version	Unsigned 8-bit integer	Header revision
radiotap.xchannel	Unsigned 32-bit integer	Channel number
radiotap.xchannel.flags	Unsigned 32-bit integer	Channel type
radiotap.xchannel.freq	Unsigned 32-bit integer	Channel frequency
radiotap.xchannel.type.2ghz	Boolean	2 GHz spectrum
radiotap.xchannel.type.5ghz	Boolean	5 GHz spectrum
radiotap.xchannel.type.cck	Boolean	Complementary Code Keying (CCK)
radiotap.xchannel.type.dynamic	Boolean	Dynamic CCK-OFDM
radiotap.xchannel.type.gfsk	Boolean	Gaussian Frequency Shift Keying (GFSK)
radiotap.xchannel.type.gsm	Boolean	GSM (900MHz)
radiotap.xchannel.type.half	Boolean	Half Rate Channel (10MHz Channel Width)
radiotap.xchannel.type.ht20	Boolean	HT Channel (20MHz Channel Width)
radiotap.xchannel.type.ht40d	Boolean	HT Channel (40MHz Channel Width with Extension channel below)
radiotap.xchannel.type.ht40u	Boolean	HT Channel (40MHz Channel Width with Extension channel above)
radiotap.xchannel.type.ofdm	Boolean	Orthogonal Frequency-Division

		Multiplexing (OFDM)
radiotap.xchannel.type.quarter	Boolean	Quarter Rate Channel (5MHz Channel Width)
radiotap.xchannel.type.sturbo	Boolean	Static Turbo
radiotap.xchannel.type.turbo	Boolean	Turbo

Appendix E: Publications

Ratnayake, D, Kazemian, H, Yusuf, S & Abdullah, A 2011, 'An Intelligent Approach to Detect Probe Request Attacks in IEEE 802.11 Networks', paper presented to International Conference on Engineering Applications of Neural Networks, Corfu, Greece, 15-19 Sept

Ratnayake, DN, Kazemian, HB & Yusuf, SA 2012, 'Improved Detection of Probe Request Attacks Using Neural Networks and Genetic Algorithm', paper presented to International Conference on Security and Cryptography, part of 9th International joint conference on e-business and telecommunications, Rome, Italy, 24th to 27th Jul.

References

Abraham, AM & Vincent, S 2012, 'Defending DoS Attacks Using a Puzzle-Based Approach and Reduction in Traceback Time towards the Attacker', *Global Trends in Computing and Communication Systems*, vol. 269, pp. 425-33.

Ahmad, I, Abdullah, AB & Alghamdi, AS 2009a, 'Application of artificial neural network in detection of DOS attacks', paper presented to 2nd international conference on Security of information and networks, Gazimagusa, North Cyprus, 6-10 Oct.

---- 2009b, 'Application of artificial neural network in detection of probing attacks', paper presented to IEEE Symposium on Industrial Electronics & Applications (ISIEA 2009), Kuala Lumpur, Malaysia, 4-6 Oct.

---- 2010, 'Comparative Analysis of Intrusion Detection Approaches', paper presented to 12th International Conference on Computer Modelling and Simulation (UKSim 2010), Cambridge, UK, 24-26 March.

Akin, D 2008, *Wireless Security Expo 2008: Hacking & Solutions: 802.11 Protocol Attacks, Deauthentication*, CWNP, Inc, viewed 15 Aug 2009, <http://www.cwnp.com/cwnp_wifi_blog/hacking-solutions-802-11-protocol-attacks-deauthentication>.

Amini, M, Jalili, R & Shahriari, HR 2006, 'RT-UNNID: A practical solution to real-time network-based intrusion detection using unsupervised neural networks', *Computers & Security*, vol. 25, no. 6, pp. 459-68.

Arbaugh, WA 2003, 'Wireless security is different', *Computer*, vol. 36, no. 8, pp. 99-101.

Ataide, RLR & Abdelouahab, Z 2010, 'An Architecture for Wireless Intrusion Detection Systems Using Artificial Neural Networks', *Novel Algorithms and Techniques in Telecommunications and Networking*, pp. 355-60.

Aura, T, Nikander, P & Leiwo, J 2001, 'DOS-resistant authentication with client puzzles', paper presented to 8th International Workshop on Security Protocols, Cambridge, UK, 3-5 Apr.

AusCERT 2004, *AA-2004.02 - Denial of Service Vulnerability in IEEE 802.11 Wireless Devices*, Australian Computer Emergency Response Team, The University of Queensland, viewed 20 April 2009, <www.auscert.org.au/render.html?it=4091,>.

BackTrack 2012, *BackTrack Linux - Penetration Testing Distribution*, viewed 24 Aug 2012, <http://www.backtrack-linux.org/>.

Baker, JE 1987, 'Reducing bias and inefficiency in the selection algorithm', paper presented to 2nd International Conference on Genetic Algorithms on Genetic algorithms and their application, MIT, Massachusetts, 28-31 July.

Bankovic, Z, Stepanovic, D, Bojanic, S & Nieto-Taladriz, O 2007, 'Improving network security using genetic algorithm approach', *Computers & Electrical Engineering*, vol. 33, no. 5-6, pp. 438-51.

Banks, J & Carson, JS 1984, *Discrete event system simulation*, Prentice Hall, Englewood Cliffs, NJ.

Bansal, R, Tiwari, S & Bansal, D 2008, 'Non-cryptographic methods of MAC spoof detection in wireless LAN', paper presented to 16th IEEE International Conference on Networks (ICON), New Delhi, India, 12-14 Dec.

Beale, J, Baker, AR, Caswell, B & Poor, M 2004, *Snort 2.1 Intrusion Detection*, Syngress Media Inc, Massachusetts, United States.

Bellardo, J & Savage, S 2003, '802.11 denial-of-service attacks: Real vulnerabilities and practical solutions', paper presented to 12th USENIX Security Symposium, Washington, DC, USA, 4-8 August, 2003.

Benediktsson, JA, Sveinsson, JR, Ersoy, OK & Swain, PH 1997, 'Parallel consensual neural networks', *Neural Networks, IEEE Transactions on*, vol. 8, no. 1, pp. 54-64.

Benjamini, Y & Hochberg, Y 1995, 'Controlling the false discovery rate: a practical and powerful approach to multiple testing', *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 289-300.

Bernaschi, M, Ferreri, F & Valcamonici, L 2008, 'Access points vulnerabilities to DoS attacks in 802.11 networks', *Wireless Networks*, vol. 14, no. 2, pp. 159-69.

Bicakci, K & Tavli, B 2009, 'Denial-of-Service attacks and countermeasures in IEEE 802.11 wireless networks', *Computer Standards & Interfaces*, vol. 31, no. 5, pp. 931-41.

Bishop, CM 1995, *Neural networks for pattern recognition*, Clarendon press, Oxford.

Bulbul, HI, Batmaz, I & Ozel, M 2008, 'Wireless network security: Comparison of WEP (wired equivalent privacy) mechanism, WPA (wi-fi protected access) and RSN (robust security network) security protocols', paper presented to First International Conference on Forensic Applications and Techniques in Telecommunications, Information and Multimedia, Brussels, Belgium, 21-24 Jan.

REFERENCES

Cakiroglu, M, Ozcerit, AT, Huseyin, E & CETIN, O 2006, 'MAC Layer DoS Attacks in Wireless Sensor Networks: A Survey', paper presented to International Conference on Wireless Networks, Las Vegas, Nevada, USA, 26-29 Jun.

Carney, JG & Cunningham, P 1998, 'The Epoch Interpretation of Learning', *Trinity College Dublin, Department of Computer Science: Computer Science Technical Report*, vol. TCD-CS-1998-09, pp. 1-5.

Chavan, S, Shah, K, Dave, N, Mukherjee, S, Abraham, A & Sanyal, S 2004, 'Adaptive neuro-fuzzy intrusion detection systems', paper presented to International Conference on Information Technology: Coding and Computing (ITCC), Las Vegas, Nevada, 5-7 Apr.

Chen, JC, Jiang, MC & Liu, YW 2005, 'Wireless LAN security and IEEE 802.11 i', *Wireless Communications, IEEE*, vol. 12, no. 1, pp. 27-36.

Ciuiu, D 2007, 'Pattern classification using polynomial and linear regression', paper presented to International Conference Trends and Challenges in Applied Mathematics, Technical University of Civil Engineering, Bucharest, Romania, 20-23 June.

Coleman, DD & Westcott, DA 2009, *CWNA Certified Wireless Network Administrator Official Study Guide: Exam PW0-104*, Wiley Inc., xxxx.

Compton, S 2008, *802.11 Denial of Service Attacks and Mitigation*, SANS Institute InfoSec Reading Room, viewed 5 Sept. 2009, <http://www.sans.org/reading_room/whitepapers/wireless/80211-denial-service-attacks-mitigation_2108>.

Conrad, E 2007, 'Types of Cryptographic Attacks', viewed 3 September 2009, <http://www.giac.org/cissp-papers/57.pdf>.

Coolen, ACC 1998, *A Beginner's Guide to the Mathematics of Neural Networks*, Citeseer, viewed 10 May 2010, <http://www.maths.kcl.ac.uk/~tcoolen/published/1998/summerschool98.pdf>.

Dasgupta, D, Gomez, J, Gonzalez, F, Kaniganti, M & Yallapu, K 2003, 'MMDS: Multilevel Monitoring and Detection System', paper presented to 15th Annual Computer Security Incident Handling Conference, Ottawa, Canada, 22-27 Jun.

Davis, H 2004, *Absolute beginner's guide to Wi-Fi wireless networking*, Que Publishing, Indiana, USA.

De Jong, KA 1993, 'Genetic algorithms are NOT function optimizers', paper presented to FOGA-92, the 2nd workshop on Foundations of Genetic Algorithms (FOGA), Vail, Colorado.

Demšar, J 2006, 'Statistical comparisons of classifiers over multiple data sets', *The Journal of Machine Learning Research*, vol. 7, pp. 1-30.

Depren, O, Topallar, M, Anarim, E & Ciliz, MK 2005, 'An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks', *Expert systems with Applications*, vol. 29, no. 4, pp. 713-22.

Dudman, K 2009, 'Project Management [unpublished lecture notes]', in PhD Workshop: RSDP 006, London Metropolitan University, Workshop held on 29th May.

Edney, J & Arbaugh, WA 2004, *Real 802.11 security : Wi-Fi protected access and 802.11i*, Pearson Education Inc., Boston, MA.

Faria, DB & Cheriton, DR 2006, 'Detecting identity-based attacks in wireless networks using signalprints', paper presented to 5th ACM workshop on Wireless security, Los Angeles, California, 29 Sept.

Fawcett, T 2006, 'An introduction to ROC analysis', *Pattern recognition letters*, vol. 27, no. 8, pp. 861-74.

Ferreri, F, Bernaschi, M & Valcamonici, L 2004, 'Access points vulnerabilities to DoS attacks in 802.11 networks', paper presented to Wireless Communications and Networking Conference, 2004, Rome, Italy, 21-25 Mar.

Fink, A 2010, *Conducting research literature reviews: from the Internet to paper*, 3 edn, Sage Publications Inc., London, UK.

Fleck, B & Dimov, J 2001, *Wireless access points and ARP poisoning: wireless vulnerabilities that expose the wired network*, Bandwidthco Computer Security, viewed 10 Oct. 2010, <http://bandwidthco.com/whitepapers/netforensics/arp-rarp/Wireless%20Access%20Points%20and%20ARP%20Poisoning.pdf>.

Fleishman, G 2010, *Say Goodbye to WEP and TKIP*, viewed 10 Oct 2012, <http://wifinetnews.com/archives/2010/06/say_goodbye_to_wep_and_tkip.html>.

Freund, RJ, Wilson, WJ & Sa, P 2006, *Regression analysis : statistical modeling of a response variable*, 2 edn, Elsevier, Oxford.

Gast, M 2005, *802.11 wireless networks: the definitive guide*, O'Reilly Media.

Geier, J 2002a, *802.1 X Offers Authentication and Key Management*, Wi-Fi Planet, viewed 30 Sept. 2008, <http://www.wi-fiplanet.com/tutorials/article.php/1041171>.

---- 2002b, *802.11 WEP: Concepts and vulnerability*, Wi-Fi Planet, viewed 21 Aug. 2008, <http://www.wi-fiplanet.com/tutorials/article.php/1368661>.

Goel, S & Kumar, S 2009, 'An Improved Method of Detecting Spoofed Attack in Wireless LAN', paper presented to First International Conference on Networks & Communications (Netcom 2009), Chennai, India, 27-29 Dec.

Gong, RH, Zulkernine, M & Abolmaesumi, P 2005, 'A software implementation of a genetic algorithm based approach to network intrusion detection', *Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, and First ACIS International Workshop on Self-Assembling Wireless Networks.*, pp. 246-53.

Guizani, M, Rayes, A, Khan, B & Al-Fuqaha, A 2010, *Network Modeling and Simulation: A Practical Perspective*, John Wiley & Sons Inc, Chichester, UK.

Guo, FL & Chiueh, TC 2006, 'Sequence number-based MAC address spoof detection', *Recent Advances in Intrusion Detection*, vol. 3858, pp. 309-29.

Haddadi, F, Khanchi, S, Shetabi, M & Derhami, V 2010, 'Intrusion Detection and Attack Classification Using Feed-Forward Neural Network', paper presented to Second International Conference on Computer and Network Technology, Bangkok, 23-25 Apr.

Hagan, MT & Menhaj, MB 1994, 'Training feedforward networks with the Marquardt algorithm', *Neural Networks, IEEE Transactions on*, vol. 5, no. 6, pp. 989-93.

Hall, J, Barbeau, M & Kranakis, E 2004, 'Enhancing intrusion detection in wireless networks using radio frequency fingerprinting', paper presented to 3rd IASTED International Conference on Communications, Internet and Information Technology (CIIT), St. Thomas, US Virgin Islands., 22-24, November.

Hamilton , HJ 2007, *Computer Science 831: Knowledge Discovery in Databases*, viewed 2 April 2012, <http://www2.cs.uregina.ca/~hamilton/courses/831/index.html>.

Hassinen, T 2006, *Overview of WLAN security*, viewed 11 October 2008 <http://www.tml.tkk.fi/Publications/C/22/papers/Hassinen_final.pdf>.

Haykin, S 1994, *Neural networks : a comprehensive foundation*, MacMillan College, USA.

He, C & Mitchell, JC 2005, 'Security analysis and improvements for IEEE 802.11i', paper presented to *12th Annual Network and Distributed System Security Symposium.*

Holland, JH 1975, *Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control, and artificial intelligence*, University of Michigan Press, Ann Arbor, Mich.

Hopgood, AA & Mierzejewska, A 2009, 'Transform ranking: a new method of fitness scaling in genetic algorithms', *Research and Development in Intelligent Systems XXV*, pp. 349-54.

Hua, J & Xiaofeng, Z 2008, 'Study on the network intrusion detection model based on genetic neural network', paper presented to International Workshop on Modelling, Simulation and Optimization, Hong Kong.

IEEE 1997, *IEEE Standard for Information Technology- Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.*

---- 2004, 'IEEE Standard for Information Technology- Telecommunications and Information Exchange Between Systems- Local and Metropolitan Area Networks- Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Medium Access Control (MAC) Security Enhancements', *IEEE Std 802.11i-2004*, pp. 0_1-175.

---- 2007, 'IEEE Standard for Information Technology-Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications', *IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999)*, pp. C1-1184.

---- 2009, *IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 4: Protected Management Frames.*

---- 2012, 'IEEE Standard for Information technology--Telecommunications and information exchange between systems Local and metropolitan area networks--Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications', *IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007)*, pp. 1-2793.

Java 2012, *Java Runtime Environment*, viewed 15 Oct 2012, <http://java.com/en/download/index.jsp>.

Jha, P 2009, 'Novel artificial neural network application for prediction of inverse kinematics of manipulator', MTech thesis, National Institute of Technology.

Juels, A & Brainard, J 1999, 'Client puzzles: A cryptographic countermeasure against connection depletion attacks', paper presented to NDSS '99 (Networks and Distributed Security Symposium), San Diego, CA, 4 Mar.

---- 2007, *Cryptographic countermeasures against connection depletion attacks*, 09/496,824, USA, 27 Mar, US7197639 B1, Grant, <http://www.google.com/patents/US7197639>.

Karygiannis, T & Owens, L 2002, *Wireless network security:802.11, Bluetooth and Handheld Devices*, Special Publication 800-48, NIST, Montgomery, U.S.

Kazienko, P & Dorosz, P 2003, *Intrusion Detection Systems (IDS) Part I-(network intrusions; attack symptoms; IDS tasks; and IDS architecture)*, viewed 1 April 2009, <http://www.windowsecurity.com/articles/intrusion_detection_systems_ids_part_i__network_intrusions_attack_symptoms_ids_tasks_and_ids_architecture.html?printversion>.

---- 2004, *Intrusion detection systems (IDS) Part 2-Classification; methods; techniques*, viewed 1 April 2009, <http://www.windowsecurity.com/articles/ids-part2-classification-methods-techniques.html?printversion>.

KDDCup 1999, *KDD Cup 1999: Computer network intrusion detection*, ACM Special Interest Group on Knowledge Discovery and Data Mining, 15 Nov 2012, <http://www.kdd.org/kdd-cup-1999-computer-network-intrusion-detection>.

Keller, F 2012, *Evaluation: Connectionist and Statistical Language Processing*, Universität des Saarlandes, viewed 24 August 2012, <http://www.coli.uni-saarland.de/~crocker/Teaching/Connectionist/lecture11_4up.pdf>.

Khan, L, Awad, M & Thuraisingham, B 2007, 'A new intrusion detection system using support vector machines and hierarchical clustering', *The VLDB Journal—The International Journal on Very Large Data Bases*, vol. 16, no. 4, pp. 507-21.

Khan, S & Loo, KK 2009, 'Real-time cross-layer design for a large-scale flood detection and attack trace-back mechanism in IEEE 802.11 wireless mesh networks', *Network Security*, vol. 2009, no. 5, pp. 9-16.

Khan, S, Loo, KK, Naeem, T & Khan, MA 2008, 'Denial of service attacks and challenges in broadband wireless networks', *IJCSNS International Journal of Computer Science and Network Securit*, vol. 8, no. 7, pp. 1-6.

Khanzode, V 1995, *Research methodology: Techniques and trends*, APH Publishing Corporation, New Delhi, India.

KLC_Consulting 2012, *SMAC 2.0*, viewed 4 June 2012, <http://www.klcconsulting.net/smac/#features>.

Kohavi, R & Provost, F 1998, 'Glossary of terms', *Machine Learning*, vol. 30, no. June, pp. 271-4.

Kothari, C 2009, *Research methodology: methods and techniques*, 2nd revised edn, New Age International, New Delhi, India.

Kumar, M, Gromiha, M & Raghava, G 2007, 'Identification of DNA-binding proteins using support vector machines and evolutionary profiles', *BMC bioinformatics*, vol. 8, no. 1, p. 463.

Laishun, Z, Minglei, Z & Yuanbo, G 2010, 'A Client Puzzle Based Defense Mechanism to Resist DoS Attacks in WLAN', paper presented to International Forum on Information Technology and Applications (IFITA), Kunming, China, 16-18 July.

Lande, R 1976, 'Natural selection and random genetic drift in phenotypic evolution', *Evolution*, pp. 314-34.

Lande, R & Barrowclough, GF 1987, 'Effective population size, genetic variation, and their use in population management', *Viable populations for conservation*, pp. 87-123.

Lazarevic, A, Ertoz, L, Kumar, V, Ozgur, A & Srivastava, J 2003, 'A comparative study of anomaly detection schemes in network intrusion detection', paper presented to 3rd Society for Industrial & Applied Mathematics (SIAM) International Conference on Data Mining San Francisco, CA, 1-3 May.

Lazarus, RS 1993, 'Coping theory and research: Past, present, and future', *Psychosomatic Medicine*, vol. 50, no. 3, pp. 234-47.

Lehembre, G 2005, 'Wi-Fi security–WEP, WPA and WPA2', *Hackin9 (January 2006)*, viewed 20 April 2010, <http://www.hsc.fr/ressources/articles/hakin9_wifi/hakin9_wifi_EN.pdf>.

Levine, D 1997, 'Commentary—Genetic Algorithms: A Practitioner's View', *INFORMS Journal on Computing*, vol. 9, no. 3, pp. 256-9.

Li, W 2004, 'Using genetic algorithm for network intrusion detection', paper presented to United States Department of Energy Cyber Security Group Training Conference, Kansas City, Kansas, 24-27 May.

Liao, HJ, Tung, KY, Richard Lin, CH & Lin, YC 2012, 'Intrusion Detection System: A Comprehensive Review', *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 16-24.

Lim, YX, Schmoyer, T, Levine, J & Owen, HL 2003, 'Wireless intrusion detection and response: a classic study using main-in-the-middle attack', paper presented to IEEE Systems, Man and Cybernetics Society Information Assurance Workshop, New York, USA, 18-20 June.

Lippmann, RP, Fried, DJ, Graf, I, Haines, JW, Kendall, KR, McClung, D, Weber, D, Webster, SE, Wyschogrod, D & Cunningham, RK 2000, 'Evaluating intrusion detection systems: The 1998 DARPA off-line intrusion detection

evaluation', paper presented to DARPA Information Survivability Conference and Exposition, Hilton Head, South Carolina, 25-27 Jan.

Liu, C & Yu, J 2007, 'A Solution to WLAN Authentication and Association DoS Attacks', *IAENG International Journal of Computer Science*, vol. 34, no. 1, pp. 31-6.

---- 2008, 'Rogue access point based dos attacks against 802.11 wlans', paper presented to Fourth Advanced International Conference on Telecommunications, Athens, 8-13 June.

Liu, Y, Tian, D & Li, B 2006, 'A wireless intrusion detection method based on dynamic growing neural network', paper presented to 1st International Multi- Symposium on Computer and Computational Sciences, Hanzhou, Zhejiang, 20-24 Jun.

Lu, W & Traore, I 2004, 'Detecting new forms of network intrusion using genetic programming', *Computational Intelligence*, vol. 20, no. 3, pp. 475-94.

Lunt, TF, Jagannathan, R, Lee, R, Whitehurst, A & Listgarten, S 1989, 'Knowledge-based intrusion detection', paper presented to Annual AI Systems in Government Conference, Washington, DC, 27-31 Mar.

MacKay, DJC 1992, 'Bayesian interpolation', *Neural computation*, vol. 4, no. 3, pp. 415-47.

Madory, D 2006, 'New methods of spoof detection in 802.11 b wireless networking', Master of Science thesis, Dartmouth College.

Magalhaes, RM 2003, *Host-Based IDS vs. Network-Based IDS (Part 2– comparative analysis)*, windowsecurity.com, viewed 5 May 2010, <http://www.windowsecurity.com/articles-tutorials/intrusion_detection/Hids_vs_Nids_Part2.html>.

Malekzadeh, M, Azim, A, Zulkarniam, Z & Muda, Z 2007, 'Security improvement for management frames in IEEE 802.11 wireless networks', *International Journal of Computer Science and Network Security*, vol. 7, no. 6, pp. 276-84.

Mamdani, EH 1976, 'Advances in the linguistic synthesis of fuzzy controllers', *International Journal of Man-Machine Studies*, vol. 8, no. 6, pp. 669-78.

Martinovic, I, Zdarsky, FA, Wilhelm, M, Wegmann, C & Schmitt, JB 2008, 'Wireless client puzzles in IEEE 802.11 networks: security by wireless', paper presented to First ACM Conference on Wireless Network Security, Alexandria, VA, USA, 31 Mar.-2 Apr.

Mateti, P 2005, 'Hacking techniques in wireless networks', in H Bidgoli (ed.), *The Handbook of Information Security*, John Wiley, New Jersey, United States, vol. 3, pp. 991-1001.

MathWorks 2008, 'Faster Training :: Backpropagation (Neural Network Toolbox™) ', *Product Help*, vol. 2008b.

---- 2012a, *MathWorks United Kingdom - MATLAB - The Language of Technical Computing*, viewed 31 August 2012, <http://www.mathworks.co.uk/products/matlab/>.

---- 2012b, *MathWorks United Kingdom - Neural Network Toolbox - Crab Classification Demo*, MathWorks Inc., viewed 1 Sept 2012, <http://www.mathworks.co.uk/products/neural-network/examples.html?file=/products/demos/shipping/nnet/classify_crab_demo.html>.

---- 2012c, *Plot classification confusion matrix - plotperform*, MathWorks, viewed 5 Sept. 2012, <http://www.mathworks.co.uk/help/toolbox/nnet/ref/plotconfusion.html>.

---- 2012d, *Post-Training Analysis (Network Validation) :: Multilayer Networks and Backpropagation Training (Neural Network Toolbox™)*, The MathWorks, Inc., viewed 10 May 2012, <http://www.mathworks.co.uk/help/toolbox/nnet/ug/bss3318-1.html>.

---- 2012e, *Recognizing Patterns :: Getting Started (Neural Network Toolbox™)*, Mathworks Inc., viewed 1 September 2012, <http://www.mathworks.co.uk/help/toolbox/nnet/gs/f9-26592.html>.

---- 2012f, *Resilient backpropagation*, MathWorks Inc., viewed 12 April 2012, <http://www.mathworks.co.uk/help/nnet/ref/trainrp.html>.

---- 2012g, *roc - receiver operating characteristic*, Mathworks, viewed 15 May 2012, <http://www.mathworks.co.uk/help/toolbox/nnet/ref/roc.html>.

Mathworks 2012, *Train the Network :: Multilayer Networks and Backpropagation Training (Neural Network Toolbox™)*, Mathworks Inc., viewed 1 August 2012, <http://www.mathworks.co.uk/help/toolbox/nnet/ug/bss3311-1.html>.

MathWorks 2012h, *trainlm: Levenberg-Marquardt backpropagation*, MathWorks Inc., viewed 02 Nov 2012.

Maxwell, JC 1865, 'A dynamical theory of the electromagnetic field', *Philosophical Transactions of the Royal Society of London*, vol. 155, pp. 459-512.

McHugh, J 2000, 'Testing intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory', *ACM transactions on Information and system Security*, vol. 3, no. 4, pp. 262-94.

Me, G & Ferreri, F 2005, *New Vulnerabilities to DoS Attacks in 802.11 Networks*, Wi-Fi Alliance, viewed 28 July 2009, <http://www.wi-fitechnology.com/Papers+req-showcontent-id-5.html>.

Merkle, RC 1978, 'Secure communications over insecure channels', *Communications of the ACM*, vol. 21, no. 4, pp. 294-9.

Michalewicz, Z & Janikow, CZ 1991, 'Handling constraints in genetic algorithms', paper presented to Proceedings of the fourth international conference on genetic algorithms, San Diego, CA, USA, 13-16 Jul.

Michalewicz, Z & Schoenauer, M 1996, 'Evolutionary algorithms for constrained parameter optimization problems', *Evolutionary computation*, vol. 4, no. 1, pp. 1-32.

Min, L & Dongliang, W 2009, 'Anomaly Intrusion Detection Based on SOM', paper presented to WASE International Conference on Information Engineering, Taiyuan, Shanxi, China, 10-11 Jul.

MIT 2012, *Communication Systems and Cyber Security: Cyber Systems and Technology: DARPA Intrusion Detection Evaluation*, Lincoln Laboratory, MIT, viewed 26 July 2012, <http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/index.html>.

Mitchell, M 1996, *An Introduction to Genetic Algorithms*, A Bradford Book The MIT Press, viewed 10 Jan. 2011, <http://www.boente.eti.br/fuzzy/ebook-fuzzy-mitchell.pdf>.

Montgomery, DC & Runger, GC 2003, *Applied statistics and probability for engineers*, 3 edn, John Wiley & Sons, New York.

Moore, AW 2001, *Cross-validation for detecting and preventing overfitting*, viewed 10 May 2012, <http://www.autonlab.org/tutorials/overfit10.pdf>.

Moradi, M & Zulkernine, M 2004, 'A neural network based system for intrusion detection and classification of attacks', paper presented to IEEE International Conference on Advances in Intelligent Systems - Theory and Applications, Luxembourg-Kirchberg, November 15-18.

Møller, MF 1993, 'A scaled conjugate gradient algorithm for fast supervised learning', *Neural networks*, vol. 6, no. 4, pp. 525-33.

Nei, M & Tajima, F 1981, 'Genetic drift and estimation of effective population size', *Genetics*, vol. 98, no. 3, pp. 625-40.

Obitko, M 2012, *Introduction to Genetic Algorithms*, viewed 15 Dec 2012, <http://www.obitko.com/tutorials/genetic-algorithms/index.php>.

Orebaugh, A, Ramirez, G & Burke, J 2007, *Wireshark & Ethereal network protocol analyzer toolkit*, Syngress Media Inc, Massachusetts, US.

Patcha, A & Park, JM 2007, 'An overview of anomaly detection techniques: Existing solutions and latest technological trends', *Computer Networks*, vol. 51, no. 12, pp. 3448-70.

Peng, CYJ, Lee, KL & Ingersoll, GM 2002, 'An introduction to logistic regression analysis and reporting', *The Journal of Educational Research*, vol. 96, no. 1, pp. 3-14.

Pillai, MM, Eloff, JHP & Venter, HS 2004, 'An approach to implement a network intrusion detection system using genetic algorithms', paper presented to South African Institute for Computer Scientists and Information Technologists on IT research in developing countries, Western Cape, South Africa, 4-6 Oct.

Pleskonjic, D 2003, 'Wireless Intrusion Detection Systems (WIDS)', paper presented to 19th Annual Computer Security Applications Conference, Las Vegas, Nevada, USA, 8-12 December, 2003,.

Pleskonjic, D, Krakovic, D, Matkovic, N, Milutinovic, V, Omerovic, S & Tomazic, S 2007, 'Reduction of False Positive Intrusions by using Neural Nets', paper presented to 8th International Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Services, Serbia, 26-28 Sept.

Ponsich, A, Azzaro-Pantel, C, Domenech, S & Pibouleau, L 2008, 'Constraint handling strategies in Genetic Algorithms application to optimal batch plant design', *Chemical Engineering and Processing: Process Intensification*, vol. 47, no. 3, pp. 420-34.

Poole, D, Mackworth, A & Goebel, R 1998, *Computational Intelligence*, Oxford University Press, New York, USA.

Potter, B 2004, *Wireless intrusion detection*, Wireless Security, viewed Oct 2012 2012, <http://www.itsec.gov.cn/docs/20090507163423280009.pdf>.

Price, G 2003, 'A General Attack Model on Hash-Based Client Puzzles', paper presented to 9th IMA International Conference on Cryptography and Coding, Cirencester, UK, 16-18 Dec.

Qing, L & Trappe, W 2007, 'Detecting spoofing and anomalous traffic in wireless networks via forge-resistant relationships', *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 4, pp. 793-808.

Rajasekaran, S & Pai, GAV 2004, *Neural networks, fuzzy logic and genetic algorithms: synthesis and applications*, PHI Learning Pvt. Ltd., New Delhi, India.

Ratnayake, D, Kazemian, H, Yusuf, S & Abdullah, A 2011, 'An Intelligent Approach to Detect Probe Request Attacks in IEEE 802.11 Networks', paper presented to Engineering Applications of Neural Networks, Corfu, Greece, 15-18 Sept.

Ratnayake, DN, Kazemian, HB & Yusuf, SA 2012, 'Improved Detection of Probe Request Attacks Using Neural Networks and Genetic Algorithm', paper presented to International Conference on Security and Cryptography, part of 9th International joint conference on e-business and telecommunications, Rome, Italy, 24-27 July.

Reeves, CR 1997, 'Genetic algorithms for the operations researcher', *INFORMS Journal on Computing*, vol. 9, no. 3, pp. 231-50.

Reeves, CR & Rowe, JE 2002, *Genetic algorithms: principles and perspectives: a guide to GA theory*, Operations Research/Computer Science Interfaces Series, Springer, New York, USA.

Rios, D 2010, *A neural network tutorial*, viewed 12 April 2010, <http://www.learnartificialneuralnetworks.com/>.

Sabhnani, M & Serpen, G 2003, 'Application of machine learning algorithms to KDD intrusion detection dataset within misuse detection context', paper presented to International Conference on Machine Learning, Models, Technologies and Applications (MLMTA 2003), Las Vegas, Nevada, USA, 23-26 Jun.

Scarfone, K & Mell, P 2007, *Guide to intrusion detection and prevention systems (IDPS)*, Special Publication 800-94, NIST, Montgomery, Maryland.

Scott, LJ & Freese, J 2006, *Regression models for categorical dependent variables using STATA*, 2 edn, Stata Press, College Station, Texas.

Shinder, D 2004, *802.11i, WPA, RSN and what it all means to Wi-Fi security*, TechGenix Ltd., viewed 26 June 2009, <http://www.windowsecurity.com/articles/80211i-WPA-RSN-Wi-Fi-Security.html?printversion>.

Singh, R & Singh, J 2012, 'A Logistic Metrics Scorecard Based Approach to Intrusion Detection System Evaluation for Wireless Network', *International Journal of Computer Networks and Wireless Communications*, vol. 2, no. 3, pp. 293-7.

SNORT 2010, *Snort*, Sourcefire, Inc, viewed 26th July 2012, <http://www.snort.org/>.

Sokolova, M & Lapalme, G 2009, 'A systematic analysis of performance measures for classification tasks', *Information Processing and Management*, vol. 45, no. 4, pp. 427-37.

Sourceforge 2012, *Java Object Oriented Neural Engine*, viewed 10 Feb. 2012, <http://sourceforge.net/projects/joone/>.

Stallings, W 2009, *Business data communications*, 6 edn, Pearson Education International, Upper Saddle River, N.J.

Statgun 2007, *logistic regression tutorial*, StatGun Statistics Consulting, viewed 05 Aug 2011 <http://www.statgun.com/tutorials/logistic-regression.html>.

Stergiou, C & Siganos, D 1996, *Neural Networks*, Imperial College, London.

Swamidass, P 2000, *Encyclopedia of production and manufacturing management*, Kluwer Academic Publishers, Norwell, MA.

tcpdump 2010, *Tcpdump/Libpcap public repository*, Tcpdump, viewed 02 Aug 2012, <http://www.tcpdump.org/>.

Toosi, AN & Kahani, M 2007, 'A new approach to intrusion detection based on an evolutionary soft computing model using neuro-fuzzy classifiers', *Computer Communications*, vol. 30, no. 10, pp. 2201-12.

Ubuntu 2012, *Home | Ubuntu*, Canonical Ltd, viewed 22 Feb 2010, <http://www.ubuntu.com/>.

Ureten, O & Serinken, N 2005, 'Bayesian detection of Wi-Fi transmitter RF fingerprints', *Electronics Letters*, vol. 41, no. 6, pp. 373-4.

Vacca, JR 2006, *Guides to wireless network security*, Springer, New York.

Van Trees, HL 2001, *Detection, Estimation, and Modulation Theory: Part 1*, John Wiley & Sons, New York, USA.

Veelenturf, LPJ 1995, *Analysis and applications of artificial neural networks*, Prentice Hall, Herefordshire, Great Britain.

Vladimirov, A, Gavrilenko, KV & Mikhailovsky, AA 2004, *Wi-Foo: the secrets of wireless hacking*, Pearson Education, Inc., Boston, MA.

Wi-Fi 2002, *Wi-Fi Alliance announces standards-based security solution to replace WEP*, Wi-Fi Alliance, viewed 08 Jun 2008, <http://wi-fi.org/pressroom_overview.php?newsid=55>.

---- 2003, 'Wi-Fi protected access: strong, standards-based, interoperable security for today's Wi-Fi networks', viewed 10 Jun 2008, <http://www.wifialliance.com/OpenSection/pdf/Whitepaper_Wi-Fi_Security4-%29-03.pdf>.

---- 2005, *Frequently asked questions (FAQ): WPA/WPA2 extended EAP*, Wi-Fi Alliance, viewed 8 November 2008, <http://www.wi-fi.org/files/kc_2_Extended%20EAP%20QandA_4-12-05.pdf>.

Wilamowski, B, Cotton, N & Hewlett, J 2007, 'Neural network trainer with second order learning algorithms', paper presented to 11th International Conference on Intelligent Engineering Systems, Budapest, Hungary, 29 Jun-2 July.

WinPcap 2012, *WinPcap · Features*, viewed 8 Sept 2012, <http://www.winpcap.org/misc/features.htm>.

Winsberg, E 2010, *Science in the age of computer simulation*, University of Chicago Press, Chicago.

Wireshark 2012, *Wireshark · Go deep.*, viewed 12 March 2010, <http://www.wireshark.org/>.

Wright, J 2003, *Detecting wireless LAN MAC address spoofing*, viewed 23 April 2009, <http://home.jwu.edu/jwright/>.

---- 2006, '802.11w security won't block DoS attacks:The IEEE standard can only go so far...', *Tech World*, viewed 29 July 2009, <http://features.techworld.com/mobile-wireless/2599/80211w-security-wont-block-dos-attacks/>.

Wu, SX & Banzhaf, W 2010, 'Review: The use of computational intelligence in intrusion detection systems: A review', *Appl. Soft Comput.*, vol. 10, no. 1, pp. 1-35.

Xia, T, Qu, G, Hariri, S & Yousif, M 2005, 'An efficient network intrusion detection method based on information theory and genetic algorithm', paper presented to 24th IEEE International Conference on Performance, Computing, and Communications, Phoenix, Arizona, 7-9 Apr.

Xing, X, Shakshuki, E, Benoit, D, Sheltami, T & Ieee 2008, 'Security Analysis and Authentication Improvement for IEEE 802.11i Specification', paper presented to IEEE Global Telecommunications Conference, New Orleans, LA, 30 Nov-04 Dec.

Yang, H, Xie, L & J., S 2004, 'Intrusion detection solution to WLANs', paper presented to IEEE 6th Circuits and Systems Symposium on Emerging Technologies: Frontiers of Mobile and Wireless Communication, Shanghai, China, 31 May-2 June.

Yao, XH 2010, 'A network intrusion detection approach combined with genetic algorithm and back propagation neural network', paper presented to International Conference on E-Health Networking, Digital Ecosystems and Technologies, Shenzhen, China, 17-18 April.

Zaknich, A 2003, *Neural networks for intelligent signal processing*, Innovative Intelligence, World Scientific, Singapore.

Zhao, JL, Zhao, JF & Li, JJ 2005, 'Intrusion detection based on clustering genetic algorithm', paper presented to International Conference on Machine Learning and Cybernetics, Guangzhou, China, 18-21 Aug.