جـامـعـة الــشـرق الأوسـط
MIDDLE EAST UNIVERSITY

An Enhanced Resilient Backpropagation Artificial Neural Network for Intrusion Detection System

Prepared by

Zainab Na'mh Abdula Al-Sultani

Supervised by

Prof. Reyadh Shaker Naoum

A Thesis Submitted in Partial Fulfilment of the

Requirements for the Master Degree

In Computer Science

Department of Computer Science

Faculty of Information Technology

Middle East University

Amman – Jordan

April, 2012

Middle East University

AUTHORIZATION STATEMENT

I, Zainab Na'mh Abdula Al-Sultani, authorize Middle East University to supply hard and electronic copies of my thesis to libraries, establishments, or bodies and institutions concerned with research and scientific studies upon request, according to the university regulations.

Name: Zainab Na'mh Abdula Al-Sultani

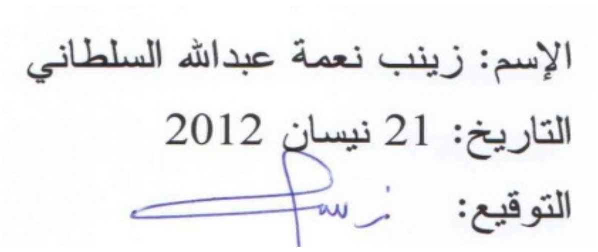
Date: 21 April 2012

Signature: [illegible]

جامعة الشرق الأوسط

التفويض

أنا زينب نعمة عبدالله السلطاني أفوض جامعة الشرق الأوسط بتزويد نسخ من رسالتي ورقيا والكترونيا للمكتبات، أو المنظمات، أو الهيئات والمؤسسات المعنية بالأبحاث والدراسات العلمية عند طلبها.



الإسم: زينب نعمة عبدالله السلطاني

التاريخ: 21 نيسان 2012

التوقيع: [illegible]

Middle East University

Examination Committee Decision

This is to certify that the thesis entitled "An Enhanced Resilient Backpropagation Artificial Neural Network for Intrusion Detection System" was successfully defined and approved on April 21 2012.

Examination Committee Members	**Signature**
Prof. Reyadh Shaker Naoum (Supervisor & Chairman) Professor, Dean of Faculty of Information Technology Middle East University (MEU)	R. NaouM
Dr. Abdulameer Khalf Hussain (Member) Associate Professor, Faculty of Information Technology Middle East University (MEU)	[illegible]
Prof. Alaa F.Sheta (External Member) Professor, Faculty of Information Technology The World Islamic Sciences and Education University (WISE)	Alaa Sheta

DEDICATION

To my lovely Parents
and to my wonderful brother and sisters, Haider, Noor and Isra'a
For their love, support, encouragement and mostly their patience
You are my life support
May God bless you now and forever

ACKNOWLEDGMENT

"In the name of Allah the Most Gracious the Most Merciful". My guidance cannot come except from Allah, in Him I trust; to Him I repent, and to Him I praise and thanks always go.

I would like to express my deepest appreciation to my supervisor Prof. Reyadh Naoum who has the attitude and the substance of a genius, he continually and convincingly push this research to the limit. Without his guidance and persistent help, this research would not have been possible.
It is a pleasure to extend my gratitude towards my university the Middle East University (MEU), for providing adequate resources and supporting their students.
My gratitude towards Prof. Gregory E. Heath for his tremendous help and patience regarding the implementation process. Many thanks and appreciation to my father Dr. Na'mh Al-Sultani and to my both sisters Noor and Isra'a for their great help in the documentation process. I would like to thank my dearest friends and colleges Dina Al-Khateeb, Manal El-Sheikh, Safa Abuhadba, and Umaiya Murad and especially to Amal Shahin and Fadwa Al-Rujoob for their support and encouragement through the entire master period.
And finally I have to say that sometimes our light fades out, but it can be glow again by others who have great impact in our life, my family you are my light flame, thank you for being there when I need you the most.

TABLE OF CONTENTS

TABLE OF TABLES

TABLE OF FIGURES

ABBREVIATIONS

ADR	Attack Detection Rate
AI	Artificial Intelligence
ANN	Artificial Neural Network
AR	Accuracy Rate
BP	Backpropagation
CIA	Confidentiality, Integrity, and Availability
CPU	Central Processing Unit
DARPA	Defense Advanced Research Projects Agency
DBN	Deep Belief Network
DMARs	Data Mining Association Rules
DoS	Denial of Service
DR	Detection Rate
DT	Decision Tree
ERBP	Enhanced Resilient Backpropagation
FAR	False Alarm Rate
FCM	Fuzzy c-means
FL	Fuzzy Logic
FN	False Negative
FNR	False Negative Rate
FP	False Positive
FPR	False Positive Rate
GA	Genetic Algorithm
HIDS	Host-based Intrusion Detection system
ICMP	Internet Control Message Protocol
ID3	Interactive Dichotomizer3
IDS	Intrusion Detection System
IP	Internet Protocol
IPS	Intrusion Prevention System
ISP	Internet Service Provider
kNN	k-Nearest Neighbor
kNN_ERBP	k- Nearest Neighbor and Enhanced Resilient Backpropagation Neural Network
LAN	Local Area Network
ML	Machine Learning
MLP	Multi-Layer Perceptron
MSE	Mean Square Error
Neq	Number of output equation
NIDS	Network-based Intrusion Detection System
NPV	Recall (Sensitivity)
Ntrn	Number of training vectors
Nw	Number of unknown weights
PC	Personal Computer
PPV	Precision (Positive Predictive)

R2L	Root to Local
RAM	Random Access Memory
RPROP	Resilient Backpropagation
SLT	Statistical Learning Theory
SOM	Self Organizing Map
SVM	Support Vector Machine
Tansig	Hyperbolic tangent sigmoid
TCP	Transmission Control Protocol
TN	True Negative
TNR	True Negative Rate
TP	True Positive
TPR	True Positive Rate
U2R	User to Root
UDP	User Datagram Protocol

تحسين شبكات الإنتشار المرتد المرن لإنظمة كشف التطفل

إعداد
زينب نعمة عبدالله

إشراف
أ.د. رياض شاكر نعوم

الملخص

أصبح الحاسوب وأمن الشبكات موضوعا أساسيا خلال العقدين الماضيين وخاصة مع ازدياد اعداد المتطفلين والقراصنة، لذلك تم تصميم انظمة لكشف او/و لمنع المتطفلين. هذه الرسالة تعرض نظام هجين لمنع التطفل بإستخدام خوارزمية تعلم الآلة (k-Nearest Neighbor) وشبكة عصبية متعددة الطبقات مدربة بإستخدام خوارزمية تعليم المرتد المرن المحسن(enhanced resilient backpropagation). النظام المقترح ينقسم إلى خمسة مراحل: مرحلة بيئة النظام، مرحلة خواص البيانات وإعادة معالجتها، مرحلة تصنيف خواص البيانات بإستخدام خوارزمية k-Nearest Neighbor، مرحلة تدريب الشبكة العصبية بإستخدام خوارزمية تعليم المرتد المرن المحسن (enhanced resilient backpropagation) واخيرا مرحلة إختبار النظام الهجين. في المرحلة الاولى من التصنيف تم إستخدام النظيم الأول (First Norm) في خوارزمية k-Nearest Neighbor والذي اعطى نتيجة أفضل من النظيم الثاني (Second Norm). في المرحلة الثانية من التصنيف تم تدريب الشبكة العصبية المتعددة الطبقات بإستخدام خوارزمية تعليم المرتد المرن المحسن. العدد الأمثل للطبقات المخفية ولعدد العصبيات المخفية وعدد مرات تدريب الشبكة تم حسابها لتدريب الشبكة العصبية (enhanced resilient backpropagation). في مرحلة تدريب الشبكة العصبية enhanced (resilient backpropagation) تم إستخدام طبقة واحدة و 34 عصبية مخفية. عامل التعلم الأمثل تم إشتقاقه لزيادة سرعة تقارب أداء الشبكة العصبية. نتائج التجارب أوضحت أن خوارزمية (k-Nearest Neighbor) تمكنت من تصنيف الفئة الطبيعية (normal) أكثر دقة من الشبكة العصبية، ولكن تمكن التعليم المرتد المرن المحسن (enhanced resilient backpropagation) من تصنيف فئات الهجمات بمعدل كشف عالي وبوقت اقل من خوارزمية تعليم المرن الاعتيادي. تقييم النظام تم انجازه بإستخدام قاعدة البيانات NSL KDD99. نتائج التجارب أوضحت بإن معدل الكشف للنظام المقترح حوالي 97.2% مع معدل دقة بلغ 99%. النظام المقترح تم مقارنته مع انظمة كشف التطفل الاخرى التي صممت بإستخدام التعليم الموجه Supervised مثل تعليم المرتد الاعتيادي وانظمة اخرى تم تصميمها بإستخدام التعليم غير الموجه Unsupervised مثل خوارزمية k-Means وشبكة Kohonen (SOM).

ABSTRACT

Over the last two decades, computer and network security has become a main issue, especially with the increase number of intruders and hackers, therefore systems were designed to detect or/and prevent intruders. This research presents a hybrid intrusion detection system models, using k-Nearest Neighbor machine learning algorithm and an enhanced resilient backpropagation artificial neural network. The proposed system is divided into five phases: environment phase, dataset features and pre-processing phase, feature classification k-Nearest Neighbor (kNN) phase, training the enhanced resilient backpropagation neural network phase and testing the hybrid system phase. k- Nearest Neighbor as a machine learning algorithm was used in the first stage of classification using the first norm which demonstrates better results than the second norm. A multilayer perceptron as the second stage of classification was trained using an enhanced resilient backpropagation training algorithm. Best number of hidden layers, hidden neurons and training iterations were calculated to train the enhanced resilient backpropagation neural network. One hidden layer with 34 hidden neurons was used in resilient backpropagation artificial neural network training process. An optimal learning factor was derived to speed up the convergence of the resilient backpropagation neural network performance. The experiments have shown that the hybrid system (kNN_ERBP) was able to classify normal class using the k-Nearest Neighbor, and the enhanced resilient backpropagation on the other hand was able to classify intrusions classes with high detection rate and with less time than the ordinary resilient backpropagation. The evaluations were performed using the NSL-KDD99 network anomaly intrusion detection dataset. The experiments results demonstrate that the proposed system (kNN_ERBP) has a detection rate about 97.2% with an accuracy rate of 99%. The proposed hybrid system (kNN_ERBP) was compared to other Intrusion Detection Systems that was designed using supervised learning such as ordinary backpropagation and unsupervised learning such as k-means and Kohonen self organizing maps.

Keywords: Intrusion Detection System, Artificial Neural Network, Resilient Backpropagation learning algorithm, k-Nearest Neighbor.

CHAPTER ONE

1 INTRODUCTION

1.1 Research Motivation

Attacks on the computer infrastructures are becoming an increasingly serious problem nowadays, therefore several information security techniques are available today to protect information systems against unauthorized use, duplication, alteration, destruction and viruses attacks (Abraham, Grosan & Chen, 2006).

Abraham, Grosan & Chen (2006) defined computer security as the protection of computing systems against threats to confidentiality, integrity, and availability. Security threats come from different sources such as natural forces (such as flood), accidents (such as fire), failure of services (such as power) and people known as intruders. There are two types of intruders: the external intruders who are unauthorized users of the machines they attack, and internal intruders, who have permission to access the system with some restrictions.

Cannady (1998) agrees that detection of computer and network system intrusions has always been an elusive goal for system administrators and information security researchers. The individual creativity of attackers, the wide range of computer hardware and operating systems, and the ever-changing nature of the overall threat to target systems have contributed to the difficulty in effectively identifying intrusions.

Kozushko (2003) clarified that intrusion detection has become the mainstream of information assurance. While firewalls do provide some protection, they do not provide full protection and still need to be complimented by an intrusion detection system. The purpose of intrusion detection is to help computer systems prepare for and deal with attacks.

Verizon Data Breach investigation reports declared that vast majority of data breaches in 2011 were the result of outsiders trying to break-in for malicious purposes (Malware and Hacking). The report insures that external attacks continue to rise as Verizon researcher found that 92 percent of attacks analyzed were external to origin. This is a significant change from previous years. Between 2004 and 2007, 80 percent of the breaches involved outsiders (Rashid, 2012).

The main objective of this research is to detect network-based anomaly intrusions using a hybrid model of machine learning algorithm which is the k-Nearest Neighbor and a neural network which is trained using an enhanced resilient backpropagation.

In this research the multilayer perceptron neural network was trained using an enhanced resilient backpropagation. An optimal learning factor was derived to enhance the convergence speed of the ordinary resilient backpropagation.

This research also aims at verifying the power of supervised learning versus unsupervised learning, by comparing our results against the results of other researchers.

1.2 Contribution

The contribution of this research can be identified as follows:

1. Hybrid system containing two classifiers: k- Nearest Neighbor and enhanced resilient backpropagation neural network.
2. Enhancing the ordinary resilient backpropagation using an optimal factor in the weight-update equation.
3. The neural network was trained using the best number of hidden layers, hidden neurons and training epochs.

1.3 Problem Statement

Moradi & Zulkernine (n.d.) mentioned that one of the most commonly used approaches in Intrusion Detection Systems expert systems is rule-based analysis. Rule-based analysis relies on sets of predefined rules that are provided by an administrator or created by the system.

Unfortunately, expert systems require frequent updates to remain current. This design approach usually results in an inflexible detection system that is unable to detect an attack if the sequence of events is even slightly different from the predefined profile. The problem may lie in the fact that the intruder is an intelligent and flexible agent while the rule based IDSs obey fixed rules. This problem can be tackled by the application of soft computing techniques (ex. Artificial Neural Network) in IDSs (Moradi & Zulkernine, n.d.).

Soft computing is a general term for describing a set of optimization and processing techniques that are tolerant of imprecision and uncertainty. The principal constituents of soft computing techniques are Fuzzy Logic (FL), Artificial Neural Networks (ANNs), and Genetic Algorithms (GAs). The idea behind the application of soft computing techniques and particularly ANNs in implementing IDSs that is capable of disclosing the latent patterns in abnormal and normal connection audit records, and to generalize the patterns to new (and slightly different) connection records of the same class (Moradi & Zulkernine, n.d.).

The main questions in this research are identified as follows:

- Can we classify the data patterns according to their types (normal and type of the attack)?

- Will the enhanced resilient backpropagation be faster and have higher detection rate than the ordinary resilient backpropagation?

- Can we minimize the learning or testing time, maximizing detection rate (DR) and minimizing the false positive rate (FPR) for the hybrid system?

1.4 Methodology

The proposed system contains two approaches in detecting intrusions using the k-Nearest Neighbor as the first approach and the enhanced resilient backpropagation neural network as the second approach. The system is divided into 5 stages: *environment phase, dataset features and pre-processing stage, feature classification k-Nearest Neighbor (kNN) stage, training the enhanced resilient backpropagation neural network stage and testing the hybrid system stage*, see figure 5.1.

In this research, two types of classifiers were used to enhance the performance of the proposed system. Using both approaches the system was able to classify intrusions with high detection rate. The first classifier, the k-Nearest Neighbor was used due to its simplicity and effectiveness when the training dataset is large. The second classifier, the enhanced resilient backpropagation neural network was used for its powerful performance in terms of speed and capacity. Both classifiers were used to classify connections into 5 classes (including normal).

1.5 Thesis Outline

Thesis is organized as follows:

Chapter Two: this chapter will focus on the related works in the field of intrusion detection using either neural networks or other machine learning algorithms. The chapter will discuss also the hybrid system models of supervised and unsupervised training algorithms that have been designed by other researches.

Chapter Three: intrusion detection system will be discussed in details, including the concepts of computer security and firewalls. Intrusion detection system classification (Misuse and Anomaly) will be clarified in terms of advantages and disadvantages. The difference between Host Intrusion Detection System architecture (HIDS) and Network Intrusion Detection System architecture (NIDS) will be discussed. Finally the chapter will mention the network anomaly NSL-KDD dataset which was used for system evaluation.

Chapter Four: Artificial Neural Networks (ANN) is the main subject of the thesis work; therefore this chapter will discuss neural networks including the biological neural and artificial neural network model, advantage and architecture. The backpropagation, resilient backpropagation and the enhanced resilient backpropagation training algorithms will be discussed.

Chapter Five: the thesis methodology will be discussed step by step. The methodology will start with the dataset transformation and standardization ending with system testing. The process of transformation and standardization of the dataset will be discussed. The k-Nearest Neighbor algorithm will be described in detail and the best parameters for neural network training will also be fully discussed.

Chapter Six: the proposed system is tested using the k-Nearest Neighbor and the enhanced resilient backpropagation artificial neural network. The results of the separated models and hybrid models are demonstrated in tables. The best number of hidden neurons is compared to different number of hidden neurons; the results are described in tables. Finally the proposed system will be compared against other systems in terms of the evaluation formulas.

1.6 Published Works

- An Enhanced Resilient Backpropagation Artificial Neural Network for Intrusion Detection System. *International Journal of Computer Science and Network Security IJCSNS*, *12* (3).

- A Hybrid Intrusion Detection System Based on Enhanced Resilient Backpropagation Artificial Neural Network and K-Nearest Neighbor Classifier. *International Journal of Academic Research IJAR*, *4* (2).

- Learning Vector Quantization (LVQ) and k-Nearest Neighbor for Intrusion Classification. *World of Computer Science and Information Technology Journal (WSCIT)*, *2*(4).

CHAPTER TWO

2 LITERATURE SURVEY AND RELATED WORK

2.1 Literature Survey and Related Work

The ability of soft computing techniques for dealing with uncertain and partially true data makes them attractive to be applied in intrusion detection. Some studies have used soft computing techniques other than ANNs in intrusion detection.

2.1.1 Intrusion Detection System based on Clustering & Classification

Nieves (2009) used data clustering for anomaly detection in network intrusion detection system. The author used k-means algorithm to evaluate the performance of an unsupervised learning method for anomaly detection using KDD Cup 1999 network dataset. In this paper the author converted the three symbol columns feature to binary format and the continuous columns were normalized so their maximum were one. Therefore the number of feature columns expanded to 80 features instead of 41. The results of the evaluation confirm a good detection rate about 89% for 5 clusters while maintaining false rate about 4.8%. This method has the advantage of using unsupervised method therefore the false positive rate was reasonable good but on the other hand the system detection rate is not very high in comparing to other systems.

Brifcani & Issa (2011) presented a comparative study where three different classifiers were used, Data Mining Association Rules (DMARs), Decision Trees (DTs) and Artificial Neural Networks (ANNs). A Feed forward neural network was trained using backpropagation algorithm and the type of DT was the Interactive Dichotomizer3 (ID3). Their experiments demonstrate that DMARs gave the worst results in terms of classifications, and their neural network training time took about 23.5 days while ID3 took 2 minutes. ID3 classification rate was 92.2%, which was the best result among the proposed methods. The main downstream of this method is the training time for ID3, because 2 minutes training is considered large in comparing to other systems, especially when using neural network properly the training time is measured in seconds even for a large dataset.

2.1.2 Intrusion Detection System based on Artificial Neural Network

Li, Zhang & Gu (2004) proposed an anomaly based network intrusion detection system based on multilayer perceptron with single hidden layer trained by backpropagation learning algorithm. The system operation was divided into three stages: Input Data Collection and Preprocessing, Training, and Detection stage. The result for the proposed module was 95% accuracy rate (2 Classes). The main advantage of the proposed system is that the researchers used only one hidden layer but they didn't mention how many hidden neurons they used. On the other hand the main disadvantage of the proposed system is that the output layer has only one neuron; therefore the system classifies records as either normal or attack with no specific attack type.

Depren et al. (2005) proposed an intelligent intrusion detection system for anomaly detection system using Self Organizing Map (SOM) to model the normal behavior. They used KDD Cup99 data set for implementation. Their results showed that their module achieved an accuracy rate of 98.96% (2 Classes) a false positive rate of 1.01%. The main advantage of this method is using the powerful unsupervised SOM which results a low false positive rate, but on the other hand the system didn't classify the records into 5 classes.

Sammany et al. (2007) developed an intrusion detection system and classification attacks using artificial neural networks. They were able to design a multilayer perceptron capable to distinguish only 2 types of attacks (Neptune, Satan) from normal. The proposed MLP architecture was trained using backpropagation algorithm with two hidden layers and three class output neurons. The results showed that the system was able to classify records with 93.43% detection rate. The proposed system was able to classify records into only 3 classes not 5 classes and the authors used 2 hidden layers which requires more processing time and storage space to store the equations.

Ahmad, Swati & Mohsin (2007) designed an intrusion detection mechanism using resilient backpropagation. The ANN architecture was input and output layers and two hidden layer, with 41, 14, 9, and 2 neurons respectively. KDD Cup 99 was used as the dataset that contains both training and testing sets. They achieved an accuracy rate of 95.93% (2 Classes). The proposed system had a very good accuracy rate but on the other hand they have used 2 hidden layers which are not necessary especially if the neural network parameters were selected optimally.

Kukielka & Kotulski (2008) analyzed different neural networks architecture for intrusion detection system. They designed three different neural network architectures: Backpropagation, Radial Basis Function, and Self Organizing Map. They used KDD99 as the input vector to the neural network. Multilayer perceptron was trained with different variances of the ordinary Backpropagation learning algorithm such as: Batch Gradient Descent, Batch Gradient Descent with momentum and Resilient Backpropagation. The results showed that multilayer perceptron was the most efficient and it requires less CPU performance and memory (RAM). That is why they tested without dividing the dataset into smaller subsets.

Jawhar & Mehrotra (2010a) designed an anomaly intrusion detection system using hamming network. They used KDD99 dataset for training and testing the proposed model. The results were inserting, they achieved a false negative of 4.94%, and the model was able to detect intrusions with a detection rate of 95% (5 Classes). The authors had a good advantage of using the supervised hamming network which is not usually used to detect intrusions. The system had a high detection rate but on the other hand the system had a reasonable high false negative rate in comparing to other intrusion detection systems.

2.1.3 Intrusion Detection System based on Hybrid Methods

Jawhar and Mehrotra (2010b) designed a hybrid intrusion detection system of fuzzy logic and neural network. Fuzzy C-Means (FCM) clustering algorithm was used to detect normal records while Multi-Layer Perceptron (MLP) was used to detect attacks (4 attack types). The MLP was trained using the resilient backpropagation. Their results were extremely high; the detection rate was 99.9% with false positive rate of about 0.01%. The main drawbacks of the proposed paper firstly the distribution of the records in the training dataset is not close to be equal between classes, specifically using 23084 of DoS attack (Denial of Service), 7 U2R (User to Root), 608 R2L (Root to Local) and 1301 Prob. will automatically lead the classifier to detect DoS and it will be very difficult to detect U2R because it's only 7 records. When designing a neural network it is very important to select how many records for representing each class so the neural network will not have converge to the larger class. Secondly the distribution of the testing dataset is not equal between classes, the normal class records dominate the other classes therefore the performance of the system will automatically increase especially that all proposed systems that have been designed by many researchers have a high detection rate for the normal class, while the most difficult classes to be classified which are U2R (User to Root) and R2L (Root to Local) are ignored in this paper. The system they proposed used only 2 records of U2R and 5 records from R2L, while 20463 records from normal.

Salama (2010) proposed a hybrid system of Support Vector Machine (SVM) and Deep Belief Network (DBN). DBN was used firstly for feature reduction process where the original feature columns were reduced from 41 to 5 columns. SVM is a classification technique based on Statistical Learning Theory (SLT). It is based on the idea of a hyper plane classifier, where it first maps the input vector into a higher dimensional feature space and then obtains the optimal separating hyper-plane. The goal of SVM is to find a decision boundary (i.e. the separating hyper-plane) so that the margin of separation between the classes is maximized. Then SVM was used to classify the reduced patterns to 5 classes: Normal, DoS, U2R, R2L and Prob. They have used NSL KDD dataset for evaluation and they achieved a classification rate of 92.84%. The authors have an excellent advantage of using the Deep Belief Network which consists of the Restricted Boltzmann Machine where Hinton proposed as an advanced version of the Boltzmann Machine. DBN seems to have a great interest in the past few years especially in the field of classification; it was used in Content Based Image Retrieval (CBIR) applications where it has promising results in terms of accuracy and speed.

Al-Rashdan (2011) has proposed an intelligent model using Hybrid Artificial Neural Networks, supervised and unsupervised learning capabilities to detect network intrusions from the KDDCup'99 dataset. She designed three cooperative phases by using an enhanced k-means clustering algorithm in Phase-1, a Hybrid Artificial Neural Network (Hopfield and Kohonen-SOM with Conscience Function) in Phase-2 and a Multi-Class Support Vector Machines in Phase-3. The Hybrid Neural Network Machine Learning Model achieved a detection rate of 92.5% and false positive rate of 3.5%. The main advantage of the proposed system is that the author used both supervised and unsupervised methods therefore minimizing the false positive rate, on the other hand using both supervised and unsupervised should expect to have a high detection rate than 92.5%.

Islim (2012) designed an intrusion detection system based on human immune system. The system model is divided into two subsystems: attack response system and learning system. The author mentioned that the attack response system contains both signature and profile databases, which is responsible for testing the received information and detecting if it contains a misuse attack and triggers an alarm or/and call the prevention system. The learning system is responsible for storing the new attack into the signature database. It contains 4 components but the 2 mainly components are: packet and system behavior analyzer component which clusters the normal behavior using the k-means clustering algorithm, anomaly detector classifier component which is responsible for classifying patterns using the naïve bayes classifier into attack or normal behavior and each database associated with each class is updated. The system was evaluated using the KDDCup99 dataset and the proposed system has a classification rate average for 4 attacks of 97.9% and false positive rate about 0.3%. The author used both supervised and unsupervised techniques in detecting attacks. The proposed system had an expectable detection rate and a low false positive rate but the false negative which is more critical than false positive hasn't been mentioned in the thesis. Our experiments demonstrate that the detection for the unseen attacks (novel attacks) is more effective and appropriate than using a machine learning algorithm such as naïve bayes classifier.

CHAPTER THREE

3 INTRUSION DETECTION SYSTEM

3.1 Introduction

Information security is one of the cornerstones of Information Society. Integrity and privacy of financial transactions, personal information and critical infrastructure data, all depend on the availability of strong and trustworthy security mechanisms. Network and Internet connectivity has provided great benefits to the modern society in terms of sharing and accessing information, however as more computers are connected to each other the higher the risk of attacks and sabotage occurs. One mechanism of information security that has been the subject of much attention in recent years is the intrusion detection systems, or IDSs. Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of intrusions. Intrusion is defined as attempts to compromise the confidentiality, integrity, availability or to bypass security mechanism of a computer or network (Kholfi, Habib & Aljahdali, 2006).

Kozushko (2003) defined that intrusion detection is considered by many to complement network firewalls, extending the security management capabilities of system administrators to include security audit, monitoring, attack recognition, and response. However, firewalls act as a barrier between the network, which is internal to the company, and the outside world. Filtering incoming traffic according to a security policy creates this barrier.

This would be a sufficient protection if it weren't for these facts (Kozushko, 2003):

1. Not all access to the Internet occurs through the firewall. Users, for various reasons ranging from ignorance to impatience, sometimes set up unauthorized modem connections between their systems that are connected to the network and outside Internet service providers. The firewall cannot mitigate risk associated with connections it never sees.

2. Not all threats originate outside of the firewall. The vast majority of loss due to security breaches is traced inside the company. Again, the firewall only sets barriers between the internal network and the Internet. If the traffic reflecting security breaches never passes the firewall, it cannot detect the problem.

3. Firewalls are subject to attacks themselves. Attack strategies for circumventing firewalls have been widely publicized since the first firewalls were fielded. A common strategy is to use tunnelling to bypass firewall protections. Tunnelling is the practice of encapsulating a message in one protocol that might be blocked by firewall filters, inside a second message.

Ali, Saleh & Badawy (2010) found that given the level and nature of modern network security threats, the question for security professionals should not be whether to use intrusion detection, but which intrusion detection features and capabilities to use. Intrusion Detection Systems have gained acceptance as a necessary addition to every organization's security infrastructure.

Intrusion Detection system can be performed manually or automatically. Manual detection is be performed by examining log files or other evidence for signs of intrusion. Automatic detection is the system that is called ***Intrusion Detection System***.

3.2 Computer Security

Stallings & Brown (2008) defined computer security that deals with computer related assets that are subjected to a variety of threats and for which various measures are taken to protect those assets.

Computer System Security is defined as the process of protecting the factors for any secure computer system. Those factors are: Confidentiality, Integrity, and Availability. These three concepts form what is often referred to as the CIA triad.

Confidentiality is the concealment of information or resources. The need for keeping information secret arises from the use of computers in sensitive fields such as government and industry. Confidentiality applies to the existence of data as well as hiding resources. Organizations may not wish others to know about specific equipment (Bishop, 2005).

Integrity refers to the trustworthiness of data or resources, and it is usually phrased in terms of preventing improper or unauthorized change. Integrity includes data integrity (the content of the information) and origin integrity (the source of the data, often called authentication). The source of the information may bear on its accuracy and credibility and on the trust that people place in the information. Working with integrity is very different from working with confidentiality. With confidentiality, the data is either compromised or it is not, but integrity includes both the correctness and the trustworthiness of the data. The origin of the data (how and from whom it was obtained), how well the data was protected before it arrived at the current machine, and how well the data is protected on the current machine all affect the integrity of data (Bishop, 2005).

Availability refers to the ability to use the information or resource desired. Availability is an important aspect of reliability as well as of system design because an unavailable system is as least bad as no system at all. The aspect of availability that is relevant to security is that someone may deliberately arrange to deny access to data or to a service by making it unavailable. Attempts to block availability, called Denial of Service attacks (DoS), can be the most difficult to detect, because the analyst must determine if the unusual access patterns are attributable to deliberate manipulation of resources or of environment (Bishop, 2005).

3.3 Firewalls

Stallings & Brown (2008) defined that Internet connectivity is no longer optional for organizations. The information and services available are essential to the organization. Moreover, individual users within the organization want and need internet access, and if this is not provided via their LAN, they will use dial-up capability from their PC to an Internet Service Provider (ISP). However, while internet access provides benefits to the organization, it enables outside world to reach and interact with local network. This creates a threat to the organization. While it is possible to equip each workstation and server on the premises network with strong security features, such as intrusion protection, this may not be sufficient and in some cases is not cost-effective. A widely accepted security service is the Firewall.

Kurose & Ross (2010) identified firewall as a combination of hardware and software that isolates an organization's internal network from the Internet at large, allowing some packets to pass and blocking others. A firewall allows a network administrator to control access between the outside world and resources within the administered network by managing the traffic flow to and from these resources.

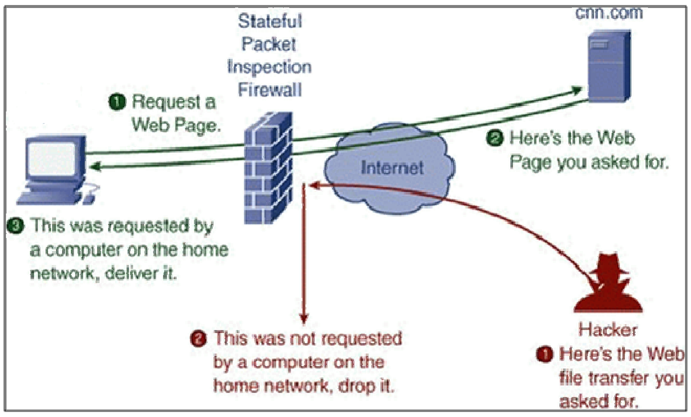

Stateful Packet Inspection Firewall
cnn.com
❶ Request a Web Page.
❷ Here's the Web Page you asked for.
Internet
❸ This was requested by a computer on the home network, deliver it.
❷ This was not requested by a computer on the home network, drop it.
Hacker
❶ Here's the Web file transfer you asked for.

Figure 3.1 Firewall

Stallings & Brown (2008) mentioned that the firewall figure 3.1 (Best Security Tips, 2007) is inserted between the network premises and the internet to establish a controlled link and to erect an outer security wall. The aim is to protect the premises network from the Internet-based attacks. Therefore, firewall provides an additional layer of defense, insulating the internal systems from external networks.

Kurose & Ross (2010) mentioned the following capabilities are within the scope or the goal of a firewall:

- **All traffic from outside to inside, and vice versa, passes through the firewall**. The above figure shows a firewall, sitting squarely at the boundary between the administered network and the rest of the Internet. While large organizations may use multiple levels of firewalls, firewall locating at single access point, makes it easier to manage.

- **Only authorized traffic, as defined by local security policy, will be allowed to pass**. With all traffic entering and leaving the institutional network passing through the firewall, the firewall can restrict access to authorized traffic.

- **The Firewall itself should be immune to penetration**. The firewall itself is a device connected to the network. If not designed or installed properly, it can be compromised, in which case it provides only a false sense of security which is worse than no firewall at all.

Stallings & Brown (2008) found that firewalls have their limitations in some situations where an Intrusion Detection System does not, this include the following:

- **Firewall can’t protect against attacks that bypass the firewall**. Internal systems may have dial-out capability to connect to an ISP.

- **Firewall may not fully protect against internal threats**. Such as a disgruntled employee or an employee who unwittingly cooperates with an external attacker.

- **Infected portable devices**. A laptop, PDA, or portable storage device may be used and infected outside the corporate network and then attached and used internally.

3.4 Intrusion Detection System

Firewall acts as a packet filter. A packet filter examines each datagram in isolation determining whether the datagram (inspects IP, TCP, UDP, & ICMP header fields) should be allowed to pass or should be dropped based on administrator specific rules. However to detect many attacks types, we need to perform deep packet inspection, that is, look beyond the header fields and into the actual application data that the packets carry (unlike the packet filter). When such a device observes a suspicious packet, or a suspicious series of packets, it could prevent those packets from entering the organizational network. Or, because the activity is only deemed as suspicious, the device could let the packets pass, but send alerts to a network administrator, who can then take a closer look at the traffic and take appropriate actions. A device that generates alerts when it observes potentially malicious traffic is called an intrusion detection system (IDS). A device that filters out suspicious traffic is called an intrusion prevention system (IPS). In this research we will focus on IDS systems, since the most interesting technical aspect of this system is how they detect suspicious traffic (Kurose & Ross, 2010).

Intrusion Detection is based on the assumption that the behavior of the intruder differs from that of a legitimate user in ways that can be quantified. Of course, we can't expect that will be exact distinction between an attack by an intruder and the normal use of resources by authorized users. Rather, we must expect that there will be some overlapping (Stallings & Brown, 2008). Therefore security violations can be detected by looking for abnormalities, so exploiting vulnerabilities requires an abnormal use of normal commands or instructions. Abnormalities such as deviation from usual actions (anomaly detection), execution of actions lead to break-ins (misuse detection).

Bishop (2005) mentioned that Intrusion Detection System goals are fourfold as follows:

1. **Detect a wide variety of intrusions**: Intrusions from within the site (Internal) as well as those from outside the site (Externals). Furthermore, both known and unknown attacks should be detected. This suggests a mechanism for learning or adapting new types of attacks.

2. **Detect intrusions in a timely fashion**: Timely here is to discover the intrusion within a short period of time not necessarily in real time.

3. **Present the analysis in a simple, easy to understand format**: Ideally this should alert the system red when an attack is detected and green for no detected intrusions. This will lead to the next requirement.

4. **Accurate**: A false positive occurs when an intrusion detection system reports an attack, but no attack is underway. False positives reduce confidence in the correctness of the results as well as increase the amount of work involved. However false negatives are worse because it occurs when an intrusion detection system fails to report an ongoing attack, where the purpose of these system is to report attacks. The goal of an intrusion detection system is to minimize both types of errors.

3.5 Intrusion Detection Classification

Intrusion Detection can be categorized into two classes, misuse intrusion detection and anomaly intrusion detection.

3.5.1 Misuse (Signature) Intrusion Detection System

Misuse (Signature) intrusion detection uses well-defined patterns of the attack that exploit weaknesses in system and application software to identify the intrusions. These patterns are encoded in advance and used to match against the user behavior to detect intrusion (Abraham, Grosan & Chen, 2006). The figure (Gadbois, n.d.) below represents the misuse intrusion detection system.

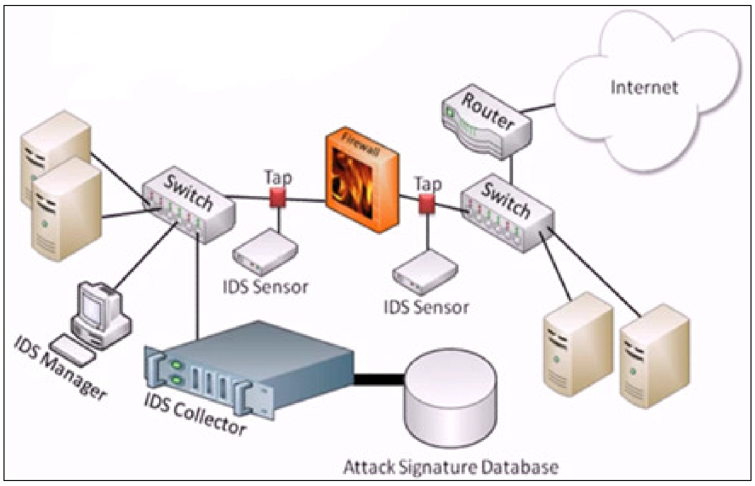


Figure 3.2 Signature (Misuse) Intrusion Detection System

Bishop (2005) defined the term “misuse” to an attack by an insider or authorized user. In the context of intrusion detection system, it means “rule-based detection”.

Misuse intrusion detection uses well-defined patterns of the attack (Attack Signature Database) that exploit weaknesses in system and application software to identify the intrusions. These patterns are encoded in advance and used to match against the user behavior to detect intrusion (Abraham, Grosan & Chen, 2006).

Bishop (2005) mentioned that misuse detection systems often use expert systems to analyze the data and apply the rule set. These systems can’t detect attacks that are unknown to the developers of the rule set. Later intrusion detection systems used adaptive methods involving neural networks to improve their detection abilities.

3.5.2 Anomaly Intrusion Detection System

Anomaly intrusion detection represented in figure 3.3 (Gadbois, n.d.) uses the normal usage behavior patterns to identify the intrusion. The normal usage patterns are constructed from the statistical measures of the system features. The behavior of the user is observed and any deviation from the constructed normal behavior is detected as intrusion.

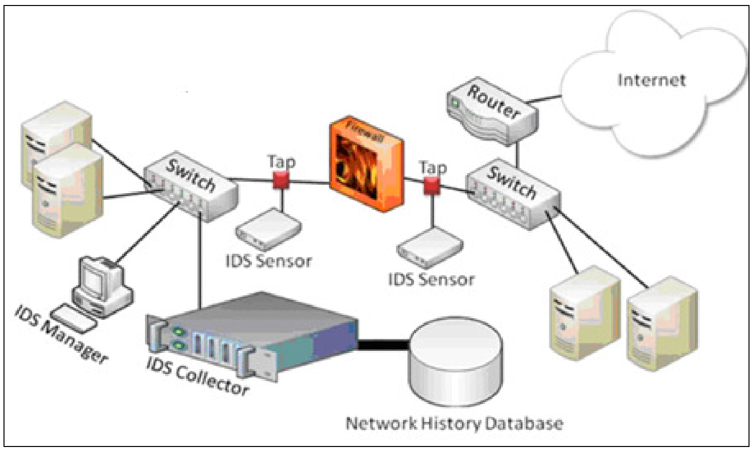


Figure 3.3 Anomaly Intrusion Detection System

Anomaly intrusion detection uses the normal usage behavior patterns to identify the intrusion, in which deviations from normal behavior indicate the presence of intentionally, or unintentionally excited attacks or faults. Anomaly detection approaches are based on building models of normal data and detect deviations from the normal model in observed data.

Scarfone & Mell (2007) found that the major benefit of anomaly-based detection methods is that they can be very effective in detecting previously unknown threats. For example, suppose that a computer becomes infected with a new type of malware. The malware could consume the computer's processing resources, send large numbers of e-mails, initiate large numbers of network connections, and perform other behavior that would be significantly different from the established profiles for the computer. Anomaly detection captures both known intrusions and unknown intrusions only if intrusions demonstrate a significant deviation from a normal profile.

3.5.3 Anomaly vs. Signature Intrusion Detection System

The advantages and disadvantages of both approaches are described as follows:

3.5.3.1 Anomaly Intrusion Detection System Advantages and Disadvantages

Kazienko & Dorosz (2004) mentioned that anomaly detection methods have the following advantages: possibility of detection of novel attacks as intrusions, less dependence of IDSs on operating environment (as compared with attack signature-based systems) and the ability to detect abuse of user privileges.

On the other hand this approach has the following disadvantages: a substantial false alarm rate and it requires a constant update of the normal behavior profile database (this may imply the need to close the system from time to time and may also be associated with greater false alarm rates) (Kazienko & Dorosz , 2004).

3.5.3.2 Misuse Intrusion Detection System Advantages and Disadvantages

Kazienko & Dorosz (2004) confirmed that signature detection methods have the following advantages: very low false alarm rate, simple algorithms, easy creation of attack signature databases, easy implementation and typically minimal system resource usage.

Kazienko & Dorosz (2004) described that this approach on the other hand has the following disadvantages: difficulties in updating information on new types of attacks (when maintaining the attack signature database updated as appropriate) therefore unable to detect unknown, novel attacks. A continuous update of the attack signature database for correlation is a must. The attack knowledge is operating environment–dependent, so misbehavior signature-based intrusion detection systems must be configured in strict compliance with the operating system (version, platform, applications used etc.) They seemed to have difficulty handling internal attacks. Typically, abuse of legitimate user privileges is not sensed by the system as a malicious activity (because of the lack of information on user privileges and attack signature structure).

3.6 Intrusion Detection Systems (IDSs) Architecture

There are many types of IDSs architecture. They are divided into the following two groups based on the type of events that they monitor and the way in which they are deployed: Host Based Intrusion Detection System (HIDS) & Network Based Intrusion Detection System (NIDS). NIDS responsibility is to protect the whole network in general, where any traffic across the network will be analyzed, but for critical end points, HIDS is used instead.

3.6.1 Host Based Intrusion Detection System (HIDS)

Scarfone & Mell (2007) defined that host based intrusion detection systems usually use systems and application logs to obtain records of events, and analyze them to determine if there is an intrusion. Host Based Intrusion Detection monitors the characteristics of a single host and the events occurring within that host for suspicious activity. Examples of the types of characteristics host-based IDS might monitor are system logs, running processes, application activity, file access and modification, and system and application configuration changes. Host-based IDSs are most commonly deployed on critical hosts such as publicly accessible servers and servers containing sensitive information.

3.6.2 Network Based Intrusion Detection System (NIDS)

Scarfone & Mell (2007) defined Network Based Intrusion Detection which monitors network traffic for particular network segments or devices and analyzes the network and application protocol activity to identify suspicious activity. It can identify many different types of events of interest. It is most commonly deployed at a boundary between networks, such as in proximity to border firewalls or routers.

3.6.3 Network Based vs. Host Based Intrusion Detection System

The difference between network based intrusion detection system and host based intrusion detection system is described in the table below (Kozushko, 2003):

Table 3.1 Network Based vs. Host Based Intrusion Detection System

Benefit	Network Based IDS	Host Based IDS
Deterrence	Strong deterrence for outsiders	Strong deterrence for insiders
Detection	Strong outsider detection weak insider detection	Strong insider detection weak outsider detection
Response	Strong response against outsider attacks	Weak real-time response

3.7 Passive and Reactive Intrusion Detection Systems

Intrusion Detection System can be categorized as Passive IDS and Reactive IDS depends on the type of response. In Passive IDS when there is an attack, the sensors will detect that there is an interesting information to be sent to the IDS collector, and then the IDS collector will compare this information to the Database and when it realizes it is an attack, it will send this information to the alarm server to alarm the user. Therefore the nature of detection does not involve any form of response to the intrusion.

To the contrary, in reactive IDS a response to the suspicious activity is performed by, for example, logging off a user or by reprogramming a firewall to block network traffic from the suspected malicious source. So Reactive IDS will perform the same as the Passive IDS except that when the alarm server alarms the user, the IDS collector will send information to the router or the firewall and notify these devices to block that activity to getting to the network (Gadbois, n.d.).

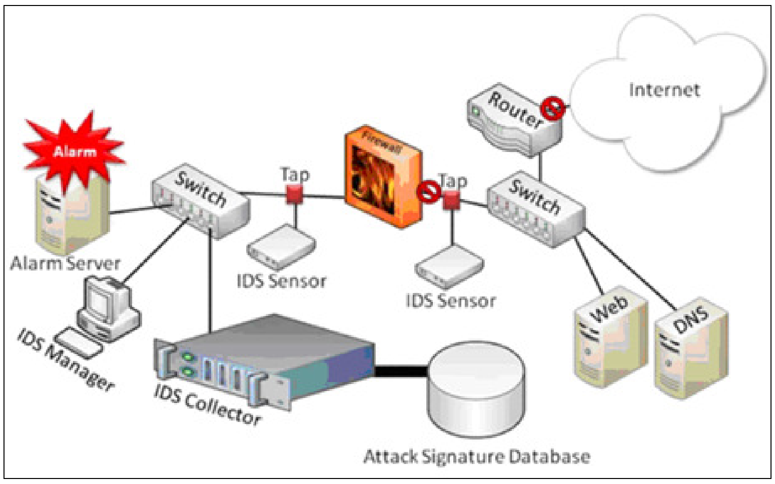


Figure 3.4 Reactive Intrusion Detection System

3.8 Network Intrusion Detection System (NIDS)

Stalling & Brown (2008) mentioned that network-based IDS (NIDS) monitors traffic at selected points on a network or interconnected set of networks. The NIDS examines the traffic packet by packet in real time, or close to real time, to attempt to detect intrusion patterns. The NIDS may examine network-, transport- and/or application- level protocol activity. Note the contrast with a host-based IDS; a NIDS examines packet traffic directed towards potentially vulnerable computer systems on a network. A host-based system examines user and software activity on a host.

3.8.1 Network Intrusion Detection System (NIDS) Strength

This research will focus on Network Intrusion Detection System. The primary responsibility of Network Intrusion Detection System (NIDS) is to protect the whole network in general (not a particular Host), so any traffic across the network will be analyzed by NIDS.

Al-Rashdan (2011) found that network-based IDS have many strengthens that cannot easily be offered by host-based intrusion detection alone. Many customers, in fact, deploy network-based intrusion detection when using IDS for the first time due to its low cost of ownership and rapid response times. Below are major reasons that make network-based intrusion detection a critical component of sound security policy implementation:

1. **Detects attacks that host-based systems miss**

Network-based IDS examine all packet headers for signs of malicious and suspicious activity. Host-based IDS do not see packet headers, so they cannot detect these types of attacks. For example, many IP-based denial-of-service (DoS) and fragmented packet (TearDrop) attacks can only be identified by looking at the packet headers as they travel across a network. This type of attack can be quickly identified by a network-based system looking at the packet stream in real-time. Network-based IDS can investigate the content of the payload, looking for commands or syntax used in specific attacks.

2. **More difficult for an attacker to remove evidence**

Network-based IDS use live network traffic for real-time attack detection. Therefore, an attacker cannot remove the evidence. Captured data includes not only the method of attack, but information that may help lead to identification and prosecution.

3. **Real-time detection and response**

Network-based IDS detect malicious and suspicious attacks as they occur, and so provide faster notification and response, where host-based systems usually do not recognize an attack or take action until after a suspicious log entry has been written. Real-time notification allows rapid reaction according to predefined parameters.

4. **Operating system independence**

Network-based IDS are not dependent on host operating systems as detection sources. By way of comparison, host-based systems require specific operating systems to function properly.

3.9 Network Attacks

The main purpose of IDS is to detect any potential network attacks. Different researches classify attacks into four categories (Mukkamala & Sung, 2003):

Denial of Service (DOS) Attacks

A denial of service attack is a class of attacks in which an attacker makes some computing or memory resource too busy or too full to handle legitimate requests, or denies legitimate users access to a machine. In other words an attacker tries to prevent legitimate users from using a service. Examples are Apache2, Back, Land, SYN Flood, Process table and Smurf.

User to Root Attacks (U2R)

User to root attacks are a class of attacks in which an attacker starts out with access to a normal user account on the system and is able to exploit vulnerability to gain root access to the system. Examples are Eject and Loadmodule.

Remote to Local Attacks (R2L)

A remote to local attack is a class of attacks in which an attacker sends packets to a machine over a network but who does not have an account on that machine. This means that an attacker does not have local account on the victim host and try to obtain it then exploits some vulnerability to gain illegally local access as a user of that machine. Examples are Xlock and Xsnoop.

Probing (Probe)

Probing is a class of attacks in which an attacker scans a network of computers to gather information or find known vulnerabilities. An attacker tries to find information about the target host, such as scanning victims in order to get knowledge about available services, using Operating System. Examples are Mscan, Nmap, Saint, and Satan.

Table below demonstrates attacks main and sub types:

Table 3.2 Attacks vs. Sub attacks

Main Attack	DoS	U2R	R2L	Prob
Sub Attack	apache2 back land mailbomb neptune pod processtable smurf teardrop udpstorm	buffer_overflow loadmodule Perl ps rootkit xterm	ftp_write guess_passwd imap mscan warezclient warezmaster xlock xsnoop	ipsweep nmap portsweep satan

3.10 Network Intrusion Detection System Dataset

In this research NSL-KDD dataset was used to evaluate the designed system. NSL-KDD is a modified version of the publically known KDD Cup99 dataset that was widely used for more than one decade.

3.10.1 KDD Cup99 Dataset

Tavallaee et al. (2009) mentioned that since 1999, KDD’99 has been the most widely used dataset for the evaluation of anomaly detection methods. This dataset is prepared by Stolfo and his colleagues and is built based on the data captured in DARPA’98 IDS evaluation program. DARPA’98 is about 4 gigabytes of compressed raw (binary) tcpdump data of 7 weeks of network traffic, which can be processed into about 5 million connection records, each with about 100 bytes. The two weeks of test data have around 2 million connection records.

In 1999, the original TCP dump files are pre-processed for the utilization of the IDS benchmark of the International Knowledge Discovery and Data Mining Tools Competition. To do so, information packets in the TCP dump file are summarized into connections. KDD training dataset consists of approximately 4,900,000 single connection vectors each of which contains 41 features and is labeled as either normal or an attack, with exactly one specific attack type (Tavallaee et al. 2009).

It is important to note that the test data is not from the same probability distribution as the training data, and it includes specific attack types not in the training data. This makes the task more realistic. Some intrusion experts believe that most novel attacks are variants of known attacks and the "signature" of known attacks can be sufficient to catch novel variants.

3.10.2 KDD99 Problems

Tavallaee et al. (2009) found an important deficiency in the KDD dataset is the huge number of redundant records. Analyzing KDD train and test sets, it has been found that about 78% and 75% of the records are duplicated in the train and test set, respectively. This large amount of redundant records in the train set will cause learning algorithms to be biased towards the more frequent records, and thus prevent it from learning infrequent records which are usually more harmful to networks such as U2R attacks. The existence of these repeated records in the test set, on the other hand, will cause the evaluation results to be biased by the methods which have better detection rates on the frequent records.

Tavallaee et al. (2009) provided a solution to solve the mentioned issues, resulting in new train and test sets which consist of selected records of the complete KDD dataset which is called NSL KDD99 Dataset.

3.10.3NSL KDD99 Dataset

Tavallaee et al. (2009) clarified that NSL-KDD is a dataset suggested to solve some of the inherent problems of the KDD'99 dataset. Although, this new version of the KDD dataset still suffers from some of the problems discussed by McHugh and may not be a perfect representative of existing real networks, because of the lack of public datasets for network-based IDSs, we believe it still can be applied as an effective benchmark data set to help researchers compare different intrusion detection methods. Furthermore, the number of records in the NSL-KDD train and test sets are reasonable. This advantage makes it affordable to run the experiments on the complete set without the need to randomly select a small portion. Consequently, evaluation results of different research work will be consistent and comparable.

The NSL-KDD data set has the following advantages over the original KDD dataset (Information Security Center of Excellence, 2009):

- It does not include redundant records in the train set, so the classifiers will not be biased towards more frequent records.
- There are no duplicate records in the proposed test sets; therefore, the performance of the learners is not biased by the methods which have better detection rates on the frequent records.
- The number of selected records from each difficulty level group is inversely proportional to the percentage of records in the original KDD dataset. As a result, the classification rates of distinct machine learning methods vary in a wider range, which makes it more efficient to have an accurate evaluation of different learning techniques.
- The numbers of records in the train and test sets are reasonable, which makes it affordable to run the experiments on the complete set without the need to randomly select a small portion.

3.11 Network Intrusion Detection System Evaluation

One of main issues involved in solving problems or trying to find optimal solution for them, is how we can test these proposed systems. As for NIDS, testing proposed method can provide a good indicator on whether the proposed method can give high performance compared with others or not. But the overall classification accuracy is not often an appropriate measure of performance especially in case of learning extremely imbalanced data. Metrics such as False Positive Rate, Recall, and Precision etc. are used to evaluate the performance of learning algorithms. These metrics have been widely used for comparisons. All metrics are functions of the confusion matrix as shown in the table below (Gau, Cai & Zhu, 2009):

Table 3.3 Predictive vs. Actual Classes Confusion Matrix

	Predictive Positive Class	**Predictive Negative Class**
Actually Positive Class	True Positive (TP)	False Negative (FN)
Actually Negative Class	False Positive (FP)	True Negative (TN)

A false-positive occurs when the system classifies an action as anomalous (a possible intrusion) when it is a legitimate action. Although this type of error may not be completely eliminated, a good system should minimize its occurrence to provide useful information to the users. A false-negative occurs when an actual intrusive action has occurred but the system allows it to pass as non-intrusive behavior, while true-positives (TP) and true-negatives (TN) are correct classifications. Recall Rate measures the proportion of actual positives which are correctly identified. Precision Rate is the ratio of true positives to combined true and false positives (Al-Rashdan et al., 2010).

To summarize the evaluation of concepts definitions (Sun & Wong, 2009, Gu, Cai & Zhu, 2009, Al-Rashdan, 2011, Revathi & Ramesh, 2011), the following definitions were given:

True Positive (TP): attack occurs and alarm rise

True Negative (TN): no attack and no alarm

False Negative (FN): attack occurs and no alarm

False Positive (FP): no attack but alarm rise

Detection Rate (DR): or classification rate for all classes (5 classes) where the system is evaluated by calculating the corrected classified records for each sub class (5 classes) of the total number of records.

Accuracy Rate (AR): the performance of the system is evaluated by calculating the ratio of correctly classified records as attacks (either normal or attack) to the total number of records.

Recall or Sensitivity Rate (NPV): measures the proportion of actual positives which are correctly identified.

$$Recall(Sensitivity\ NPV) = TP/_{(TP + FN)}$$

Precision Rate (PPV): is the ratio of true positives to combined true and false positives.

$$Precision\ (PPV) = TP/_{(TP + FP)}$$

False Positive Rate (FPR): is the ratio of incorrectly classified normal records (false alarms) to the total number of true negative and false positive.

False Negative Rate (FNR): is the ratio of incorrectly classified attacks (when system classify attacks as normal) records to the total number of true positive and false negative.

True Negative Rate (Specificity): is the ratio of correctly classified attacks records to the total number of true negative and false positive.

G-Mean: Geometry Mean measures the balanced performance of a learning algorithm between classes. G-mean will have high value when the performance of all classes is concerned, where both true positive rate (recall) and true negative rate are expected to be high simultaneously.

$$G - Mean = \sqrt{ACC^{-}.Recall}$$

F-Measure: is the harmonic mean of recall and precision tends to be closer to the smaller of the two. Hence a high f-measure value ensures that both recall and precision are reasonably high.

$$F - Measure = \left(2 * Precision * Recall \middle/ (Recall + Precision)\right) * 100\%$$

CHAPTER FOUR

4 ARTIFICIAL NEURAL NETWORK

4.1 Introduction to Artificial Neural Networks (ANN)

Jones (1997) research into potential systems of artificial intelligence looks to the brain for models rather than looking to technology for ideas from which to model the brain. A number of scientists are looking at the development of artificial intelligence from the basis of a developing understanding of the architecture of the human brain. This work is now represented in two interlocking disciplines: Computational neurobiology which involves understanding human/animal brains using computational models; and Neural Computing or simulating and building a machine to emulate the real brain. The analysis is made on two levels: coarse grained, examining and elucidating networks of interacting subsystems which is largely a neurophysiological activity; and fine grained, building theories and models of actual artificial neural networks as subsystems.

Lippmann (1987) defined that Artificial Neural Networks or simply "Neural Nets" go by many names such as connectionist models, parallel distributed processing models, and neuromorphic systems. Whatever the name, all these models attempt to achieve good performance via dense interconnection of simple computational elements. In this respect, artificial neural net structure is based on our understanding of biological nervous system. Neural net models have greatest potential in areas such as speech and image recognition. Instead of programming with sequential instructions, neural net models explore many competing hypotheses simultaneously using massively parallel nets composed of many computational elements connected by links with variable weights.

Lynn (n.d.) confirmed that Artificial Neural Networks (ANN) offer a different approach for analyzing data, and for recognizing patterns within that data, than traditional computing methods, therefore Artificial Neural Networks have been used in many applications such as: classification (Medical diagnosis, target recognition, character recognition, fraud detection, intruder detection and speech recognition), Function Approximation (machine diagnostics), and Data Mining (Clustering, data visualization and data extraction).

4.2 Biological Neural Network Model

The fundamental processing element of a biological neural network is a neuron. In the human brain, a typical neuron collects signals from others through a host of fine structures called dendrites. The neuron sends out spikes of electrical activity through a long, thin stand known as an axon, which splits into thousands of branches. At the end of each branch, a structure called a synapse converts the activity from the axon into electrical effects that inhibit or excite activity from the axon into electrical effects that inhibit or excite activity in the connected neurons. When a neuron receives excitatory input that is sufficiently large compared with its inhibitory input, it sends a spike of electrical activity down its axon. Learning occurs by changing the effectiveness of the synapses so that the influence of one neuron on another changes (Singh & Verma, 2011).

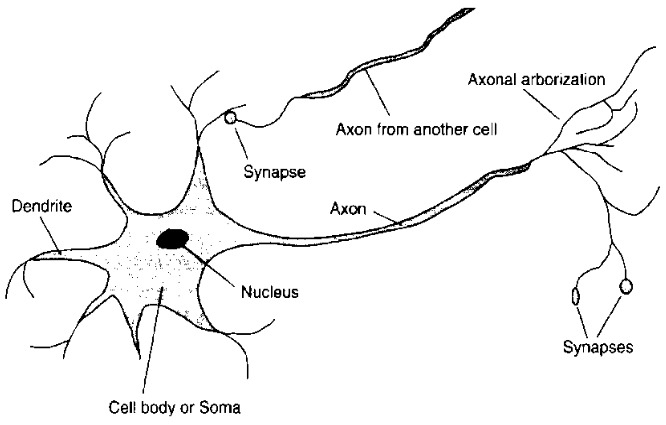


Figure 4.1 Biological Neural Net

The artificial neural networks try to replicate only the most basic elements of this complicated, versatile, and powerful organism. They do it in a primitive way. But for the software engineer who is trying to solve problems, neural computing was never about replicating human brains. It is about machines and a new way to solve problems (Reingold & Nightingale, 1999).

4.3 Artificial Neural Network Model

According to Haykin (1998) Artificial Neural Network (ANN) is a massively parallel distributed processor that has a natural propensity for storing knowledge and making it available for use. So it resembles the brain in two aspects: knowledge is acquired by the network through a learning process, and the interconnection strengths known as synaptic weights are used to store this knowledge.

The basic unit of neural networks, the artificial neurons, simulates the basic functions of natural neurons. Figure 4.2 (PrismNet, n.d.) below shows a fundamental representation of an artificial neuron.

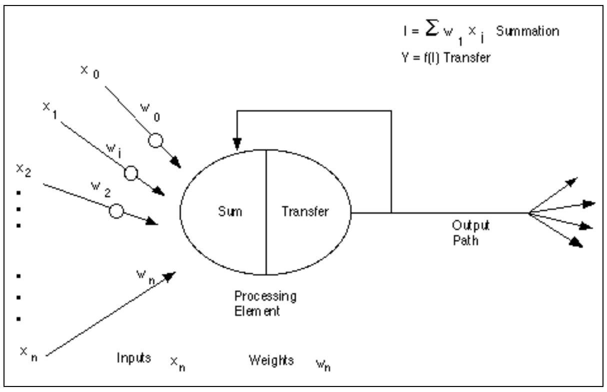


Figure 4.2 Basic Artificial Neural Network

In Figure 4.2, various inputs to the network are represented by the mathematical symbol, x_n. Each of these inputs is multiplied by a connection weight. These weights are represented by w_n. In the simplest case, these products are simply summed, fed through a transfer function to generate a result, and then neural output is provided (Naoum, 2011).

4.4 Artificial Neural Network Advantages

Stergiou (1996) stated that neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A trained neural network can be thought of as an "expert" in the category of information it has been given to analyze. This expert can then be used to provide projections given new situations of interest and answer "what if" questions. Other advantages include:

1. **Adaptive learning**: An ability to learn how to do tasks based on the data given for training or initial experience.
2. **Self-Organisation**: An ANN can create its own organisation or representation of the information it receives during learning time.
3. **Real Time Operation**: ANN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which take advantage of this capability.
4. **Fault Tolerance**: Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage.

4.5 Artificial Neural Network Architecture

A neural network has an input layer, output layer, and zero or more number of hidden layers. All these layers contain a number of neurons, the basic element of neural networks (Li, Zhang & Gu, 2004).

Bernacki & Włodarczyk (2004) presented the diagram of a neuron's operation Figure 4.3. The figure consists of some inputs emulating dendrites of the biological neuron, a summation module, an activation function and one output emulating an axon of the biological neuron. The architecture of this ANN consists of one input layer, one output layer and two hidden layers. Input layer consists of two neurons, first and second hidden layers have three and two neurons respectively, and finally the output layer contains a single neuron. The importance of a particular input can be intensified by the weights that simulate biological neuron's synapses. Then, the input signals are multiplied by the values of weights and next the results are added in the summation block. The sum is sent to the activation block where it is processed by the activation function (Transfer Function). Thus, we obtained neuron's answer y for the input signals "x_1" and "x_2".

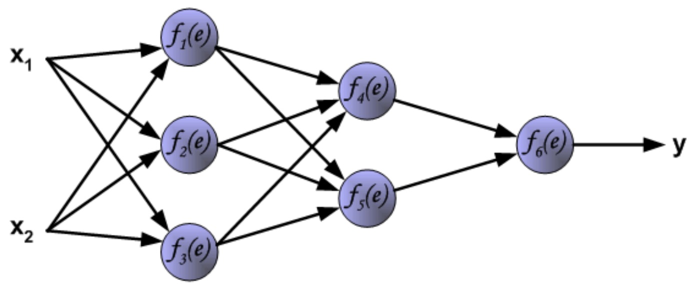


Figure 4.3 Artificial Neural Network

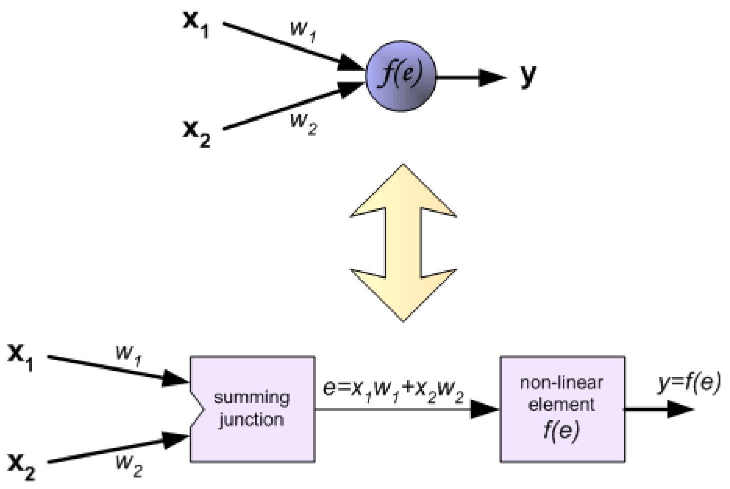


Figure 4.4 Summation and Transfer Functions

4.5.1 Layers

Reingold & Nightingale (1999) clarified that the most significant difference between artificial and biological neural nets is their organization. While many types of artificial neural nets exist, most are organized according to the same basic structure. There are three components to this organization: a set of input nodes, one or more layers of 'hidden' nodes, and a set of output nodes. The input nodes take in information, whether the information is in the form of a digitized picture, or a series of stock values, or just about any other form that can be numerically expressed, this is where the net gets its initial data (from the environment). The information is supplied as activation values, that is, each node is given a number, higher numbers representing greater activation.

The MIT Press declared that in order for the artificial neural net to carry out a useful task, one must connect the neurons in a particular configuration, set the weights, and choose the input-output functions (transfer function). The simplest artificial neural net would consist of a layer of input units connected to a single middle or "hidden" layer, which is linked to a layer of output units. To initialize the artificial neural net, whatever raw data is needed to perform the task is first fed into the input units. The resulting signal received by a neuron in the hidden layer depends on how the incoming raw data is weighted, and how it is modified by the transfer function. This procedure is repeated for the signal flowing out of the hidden layer before going on to the subsequent level.

For instance, gender recognition net might be presented with a picture of a man or woman at its input nodes and must set an output node to 0 if the picture depicts a man or 1 for a woman. In this way, the network communicates its knowledge to the outside world.

4.5.2 Transfer (Activation) Functions

Naoum (2011) clarified that the transfer function describes how a neuron's firing rate varies with the input it receives. Every neuron has an activation level. The summation function will compute this level, and according to this level we will have an exit value from the neuron or not. The relation between the activation level and the output maybe linear or non-linear and this relation can be represented by so called transfer function.

A neuron may sum its inputs, or average them, or something entirely more complicated. Each of these behaviors can be represented mathematically, and that representation is called the transfer (activation) function. Transformation is essential in order to improve the levels of outputs to a reasonable value between 0 and 1, due to in some cases outputs may be very large when we have more than one hidden layer (Naoum, 2011).

Remember information in ANN depends on the mathematical activation function. This function typically falls into different numbers of categories, such as (Naoum, 2011):

1. Sigmoid Transfer Function $f(x) = \frac{1}{1+e^{-\alpha x}}$
2. Hyperbolic Tangent Sigmoid Transfer Function $f(x) = \frac{2}{1+e^{-2x}} - 1$ (MathWorks, 2012)

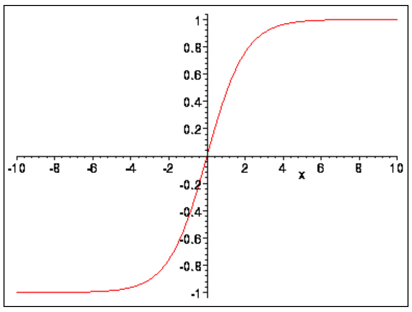


Figure 4.5 Hyperbolic Tangent Sigmoid (Willamette University, n.d.)

Naoum (2011) explained that for sigmoid (logical) activation function, the output varies continuously non-linearly as the input changes. The sigmoid function is bounded and differentiable real function and has positive derivative, and it has lower limit bound (0 or -1) and upper limit bound (+1).

4.6 Training an Artificial Neural Networks

Once a network has been structured for a particular application, then the network is ready to be trained. To start this process the initial weights are chosen randomly. Then, the training, or learning, begins. There are many approaches to training: supervised, unsupervised, and reinforcement training.

4.6.1 Supervised Training

Reingold & Nightingale (1999) described that in supervised training, both the inputs and the outputs are provided. The network then processes the inputs and compares its resulting outputs against the desired outputs. Errors are then propagated back through the system, causing the system to adjust the weights which control the network. This process occurs over and over as the weights are continually tweaked. The set of data which enables the training is called the "training set." During the training of a network the same set of data is processed many times as the connection weights are ever refined. Figure below (Aboshosha, n.d.) represents the supervised learning schema.

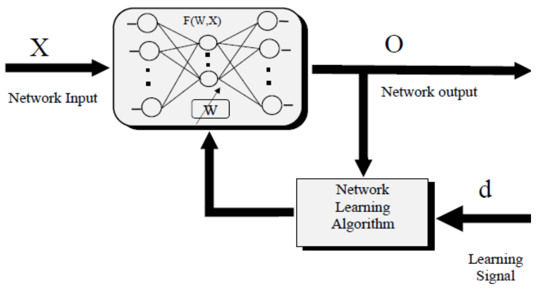


Figure 4.6 Supervised Learning Diagram

Haykin (2001) identified that this form of learning assumes the availability of a labeled (i.e., ground-trusted) set of training data made up of *N* input—desired examples:

$$\mathbf{T} = \{(\mathbf{x}_i, \mathbf{d}_i)\}_{i=1}^{N}$$

Where $\mathbf{x}_i$ = input vector of ith example

d_i = desired (target) response of ith example

N = Training set size

Given the training sample T, the requirement is to compute the free parameters of the neural network so that the actual output y_i of the neural network due to x_i is close enough to d_i for all i in a statistical sense. For example, we may use the Mean-Square Error (MSE) as the index of performance to be minimized.

$$E(n) = \frac{1}{N}\sum_{i=1}^{N}(d_i - y_i)^2$$

Stergiou & Siganos (1996) stated that a three-layer neural network can be trained to perform a particular task using the following procedure:

1. Present the network with training examples, which consist of a pattern of activities for the input units together with the desired pattern of activities for the output units.
2. Determine how closely the actual output of the network matches the desired output.
3. Change the weight of each connection so that the network produces a better approximation of the desired output.

4.6.2 Unsupervised Training

Naoum (2011) described that in unsupervised or self organized Learning, the network is not given any external indications as to what the correct responses should be nor whether the generated responses are right or wrong; it is based upon only local information.

It is simply exposed to the various input-output pairs and it learns by the environment, that is, by detecting regularizations in the structure of input patterns. This is often referred to as self-organization or adaption, figure 4.7 (Aboshosha, n.d.).

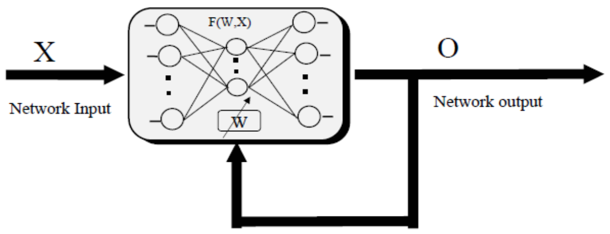


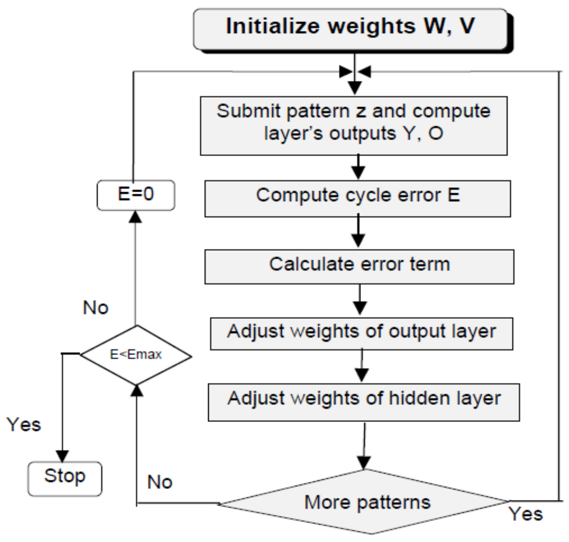
Figure 4.7 Unsupervised Learning Diagram

4.7 Backpropagation learning algorithm

One neuron cannot solve a complex problem that's why neural network composed of many neurons (Kukiełka & Kotulski, 2008). For the purpose of this research Multilayer Perceptron (MLP) will be used.

One of the architectures that are used most frequently is the MLP (Multilayer Perceptron). In such a network each neuron's output of the previous layer is connected with some neuron's input of the next layer. The MLP architecture consists of one or more hidden layers of neurons followed by an output layer. The signal is transmitted through the network in one direction from the input to the output, that's why this architecture is called feed forward. The MLP network is usually learned using the Backpropagation algorithm (BP).

Figure 4.8 (Aboshosha, n.d.) represents the backpropagation neural network algorithm which is one of the most powerful supervised neural networks. It has the same structure as multilayer perceptron and mainly used in complex logical operations, pattern classification and speech analysis. Like in multilayer perceptron, Backpropagation neural network has three layers: input, output and hidden layers (Li, Zhang & Gu, 2004)



Initialize weights W, V
Submit pattern z and compute layer's outputs Y, O
Compute cycle error E
Calculate error term
Adjust weights of output layer
Adjust weights of hidden layer
More patterns
Yes
No
E<Emax
No
E=0
Yes
Stop

Figure 4.8 Backpropagation Algorithm Diagram

The Backpropagation training algorithm (Lippmann, 1987):

The backpropagation training algorithm is an iterative gradient algorithm designed to minimize the mean square error between the actual output of a multilayer feed-forward perceptron and the desired output. The following assumes a sigmoid logistic non-linearity function which is used as the activation function,

$$f(x) = \frac{1}{1 + e^{-ax}}$$

Step 1: Initialize Weights and Offsets. Sets all weights and node offsets to small random values.

Step 2: Present Input and Desired Outputs

Present a continuous valued input vector x_0, x_1,… x_{n-1} and specify the desired outputs d_0,d_1,…,d_{m-1}. If the net is used as a classifier then all desired outputs are typically set to zero except for that corresponding to the class the input is from. The input could be new on each trial or samples from a training set could be presented cyclically until weights stabilize.

Step 3: Calculate Actual Outputs. Use the sigmoid nonlinearity from above formula to calculate the outputs y_0, y_1...y_{m-1}.

Step 4: Adapt weights. Use a recursive algorithm starting at the output nodes and working back to the first hidden layer. Adjust weights by:

$$w_{ij}(t+1) = w_{ij}(t) + \mu\delta x_i$$

In this equation $w_{ij}(t)$ is the weight from hidden node i or from an input to node j at time t, x_i is either the output of node i or is an input, μ is the gain term, and δ is an error term for node j. If node j is an output node, then

$$\delta_j = y_j(1 - y_j)(d_j - y_j)$$

Where d_j is the desired output of node j and y_j is the actual output.

If node j is an internal hidden node, then

$$\delta_j = x_j(1 - x_j)\sum_k \delta_k w_{jk}$$

Where k is over all nodes in the layers above node j.

Step 5: If the Mean Square Error is above some predefined value then repeat by going to step 2. Otherwise the Neural Network has been trained correctly.

The main advantages of Backpropagation neural nets are that they are great at prediction and classification. On the other hand, there is always a lack of explanation of what the net has been learned, suffers from local minima, slow training and temporary unstable (Naoum, 2011).

In order to solve the Backpropagation problem, many algorithms have been proposed so far to deal with the problem of appropriate weight-update by doing some sort of parameter adaptation during learning. These updated algorithms have a high performance, which they can converge from ten to one hundred times faster than the simple backpropagation algorithms. One of the fastest algorithms is the Resilient Backpropagation learning algorithm (Riedmiller & Braun, 1993).

4.8 Resilient Backpropagation learning algorithm

Riedmiller & Braun (1993) explained that the basic idea of the backpropagation learning algorithm is the repeated application of the chain rule to compute the influence of each weight in the network with respect to an arbitrary error-function E:

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial s_i}\frac{\partial s_i}{\partial net_j}\frac{\partial net_j}{\partial w_{ij}}$$

Where w_{ij} is the weight from neuron *i* to neuron *j*, s, is the output, and net_j is the weighted sum of the inputs of neuron *j*. Once the partial derivative for each weight is known, the aim of minimizing the error-function is achieved by performing a simple gradient descent (Riedmiller & Braun, 1993):

$$w_{ij}(t+1) = w_{ij}(t) - \epsilon \frac{\partial E}{(\partial w_{ij})(t)}$$

Obviously, the choice of the learning rate Є, which scales the derivative, has an important effect on the time needed until convergence is reached. If it is set too small, too many steps are needed to reach an acceptable solution; on the contrary a large learning rate will possibly lead to oscillation, preventing the error to fall below a certain value.

Riedmiller & Braun (1993) defined RPROP which stands for 'resilient propagation' as an efficient new learning scheme, which performs a direct adaptation of the weight step based on local gradient information. In crucial difference to previously mentioned adaptation technique, the effort of adaptation is not blurred by gradient behavior whatsoever. To achieve this, they introduce for each weight its individual update-value Δ_{ij}, which solely determines the size of the weight-update. This adaptive update-value evolves during the learning process based on its local sight on the error-function E, according to the following learning-rule:

$$\Delta_{ij}^{t} = \begin{Bmatrix} \eta^{+} * \Delta_{ij}^{t-1} & , if & \frac{\partial E}{\partial w_{ij}}^{t-1} * \frac{\partial E}{\partial w_{ij}}^{t} > 0 \\ \eta^{-} * \Delta_{ij}^{t-1} & , if & \frac{\partial E}{\partial w_{ij}}^{t-1} * \frac{\partial E}{\partial w_{ij}}^{t} < 0 \\ \Delta_{ij}^{t-1} & , & else \end{Bmatrix}, where\ 0 < \eta^{-} < 1 < \eta^{+}$$

The adaptation-rule works as follows: Every time the partial derivative of the corresponding weight w_{ij} changes its sign, which indicates that the last update was too big and the algorithm has jumped over a local minimum, the update-value Δ_{ij} is decreased by the factor η-. If the derivative retains its sign, the update-value is slightly increased in order to accelerate convergence in shallow regions. In all of their experiments, the choice of $\eta^{+} = 1.2$ gave very good results, independent of the examined problem. Slight variations of this value did neither improve nor deteriorate convergence time. So in order to get parameter choice more simple, they decided to constantly fix the increase/ decrease parameters to $\eta^+ = 1.2$ and $\eta^- = 0.5$

Once the update-value for each weight is adapted, the weight-update itself follows a very simple rule: if the derivative is positive (increasing error), the weight is decreased by its update-value, if the derivative is negative, the update-value is added:

$$\Delta w_{ij}^t = \begin{cases} -\Delta_{ij}^t & , if \quad \dfrac{\partial E}{\partial w_{ij}}^t > 0 \\ +\Delta_{ij}^t & , if \quad \dfrac{\partial E}{\partial w_{ij}}^t < 0 \\ 0 & , \quad else \end{cases}$$

4.9 Artificial Neural Network in Intrusion Detection System

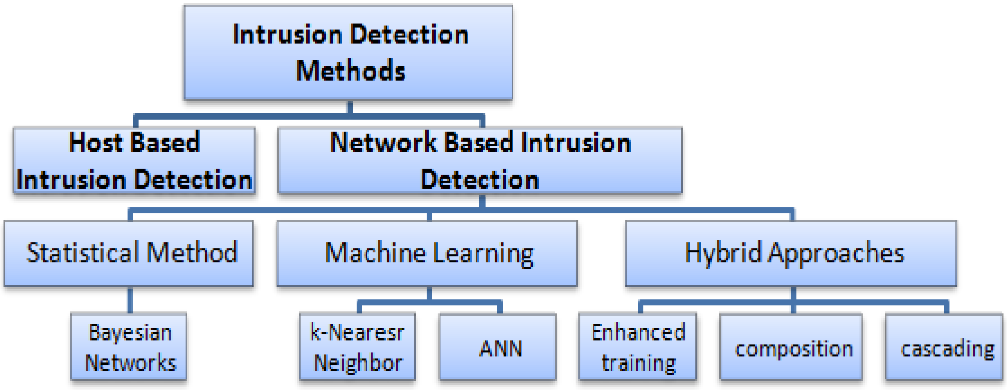


Figure 4.9 Network Intrusion Detection Methods

Figure above (Ippoliti, 2011) illustrates different number of classifiers for building intrusion detection systems. Neural Networks and k-Nearest Neighbor are classified under machine learning approach. Neural networks are a uniquely powerful tool in multiple class classification, especially when used in applications where formal analysis would be very difficult or even impossible, such as pattern recognition, nonlinear system identification, and control (Sammany et al., 2007). Because of their generalization feature, neural networks are able to work with imprecise and incomplete data. It means that they can recognize also patterns not presented during a learning phase. That is why the neural networks could be a good solution for detection of a well- known attack, which has been modified by an aggressor in order to pass through the firewall system. In that case, traditional Intrusion Detection Systems, based on the signatures of attacks or expert rules, may not be able to detect the new version of this attack (Kukielka and Kotulski, 2010).

CHAPTER FIVE

5 PROPOSED MODEL & METHODOLOGY

5.1 Proposed Model

The proposed system contains two approaches in detecting intrusions using machine learning algorithm and a neural network. The proposed system is divided into five phases: the environment phase, data preprocessing phase, feature classification phase, training phase and testing phase. Figure 5.1 represents the proposed model phases.

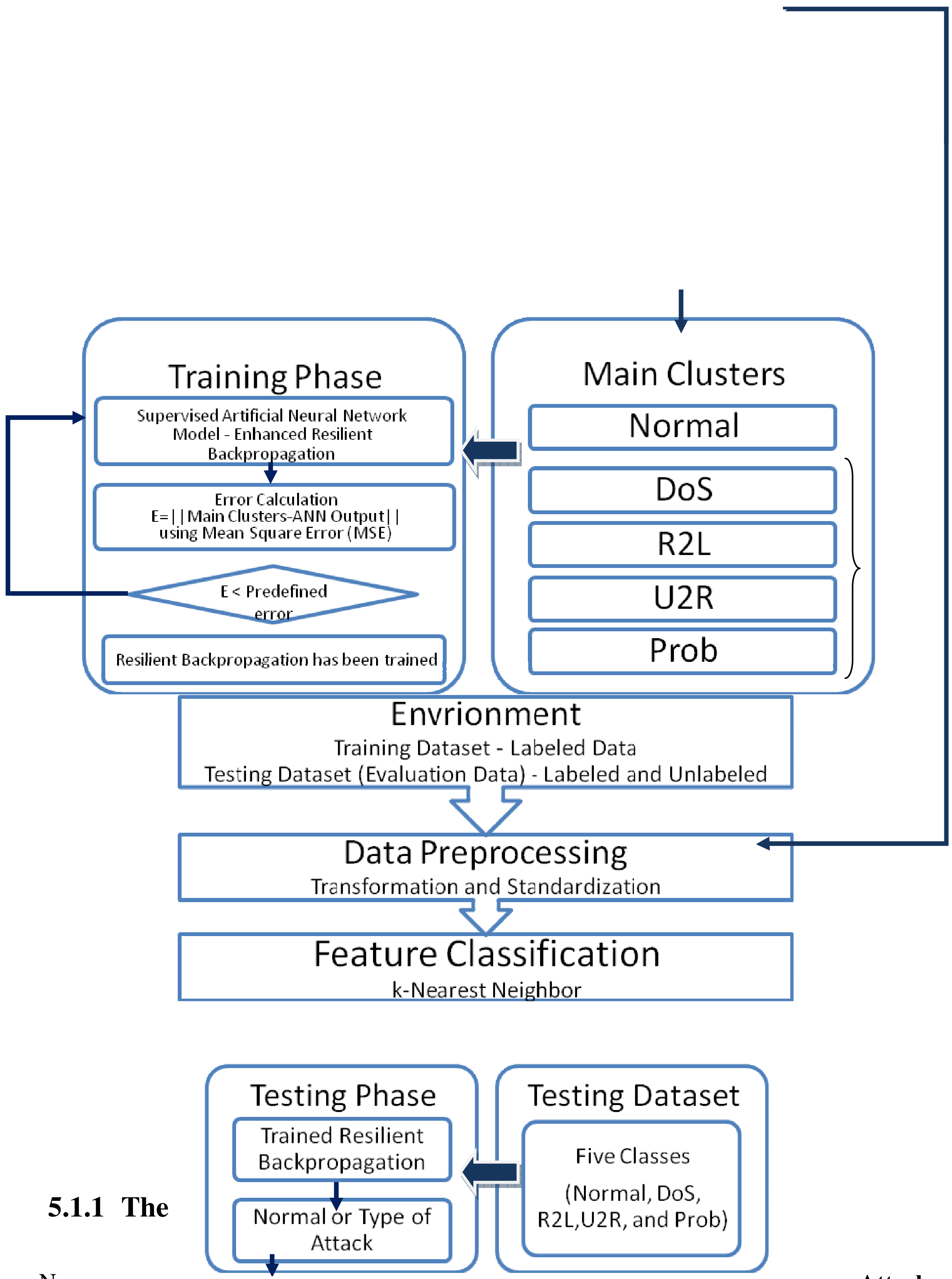

Training Phase
Supervised Artificial Neural Network Model - Enhanced Resilient Backpropagation
Error Calculation
E=||Main Clusters-ANN Output||
using Mean Square Error (MSE)
E < Predefined error
Resilient Backpropagation has been trained
Main Clusters
Normal
DoS
R2L
U2R
Prob
Attack Types
Envrionment
Training Dataset - Labeled Data
Testing Dataset (Evaluation Data) - Labeled and Unlabeled
Data Preprocessing
Transformation and Standardization
Feature Classification
k-Nearest Neighbor
Testing Phase
Trained Resilient Backpropagation
Normal or Type of Attack
Testing Dataset
Five Classes
(Normal, DoS, R2L,U2R, and Prob)
No
Yes
Yes

Figure 5.1 Proposed Hybrid Model (kNN_ERBP)

5.1.1 The Environment Phase

This unit presents records from NSL KDD99 dataset (Information Security Center of Excellence, 2009, Tavallaee et al., 2009). It's divided into two subsets, training subset and testing subset. The NSL KDD dataset includes a wide variety of intrusions together with normal activities simulated in a military network environment. NSL KDD records belong to one of the following five categories: Normal, DoS (denial of service), R2L (root to local), U2R (user to root) and Probing (surveillance). There are 41 features columns and they are either symbolic or continuous.

The following table represents the features name and types of NSL KDD dataset (University of California, Irvine, n.d.).

Table 5.1 NSL-KDD Feature Columns Name and Type

	Feature Name	Feature Type		Feature Name	Feature Type
1	duration	continuous.	22	is_guest_login	discrete.
2	protocol_type	symbolic.	23	count	continuous.
3	service	symbolic.	24	srv_count	continuous.
4	flag	symbolic.	25	serror_rate	continuous.
5	src_bytes	continuous.	26	srv_serror_rate	continuous.
6	dst_bytes	continuous.	27	rerror_rate	continuous.
7	land	discrete.	28	srv_rerror_rate	continuous.
8	wrong_fragment	continuous.	29	same_srv_rate	continuous.
9	urgent	continuous.	30	diff_srv_rate	continuous.
10	hot	continuous.	31	srv_diff_host_rate	continuous.
11	num_failed_logins	continuous.	32	dst_host_count	continuous.
12	logged_in	discrete.	33	dst_host_srv_count	continuous.
13	num_compromised	continuous.	34	dst_host_same_srv_rate	continuous.
14	root_shell	continuous.	35	dst_host_diff_srv_rate	continuous.
15	su_attempted	continuous.	36	dst_host_same_src_port_rate	continuous.
16	num_root	continuous.	37	dst_host_srv_diff_host_rate	continuous.
17	num_file_creations	continuous.	38	dst_host_serror_rate	continuous.
18	num_shells	continuous.	39	dst_host_srv_serror_rate	continuous.
19	num_access_files	continuous.	40	dst_host_rerror_rate	continuous.
20	num_outbound_cmds	continuous.	41	dst_host_srv_rerror_rate	continuous.
21	is_host_login	discrete.	42	Label	symbolic.

5.1.2 Data Pre-processing Phase

The data from the environment unit will be processed before entering the classification unit. Feature columns are processed at 2 steps as follows:

1. Transformation: Symbolic columns which are protocol, service, flag and label are transformed to numeric values using customize transformation table. Each column has its own customization table, depending on the column content. After analyzing the protocol column, it has been shown that it has three protocols values: TCP, UDP, & ICMP. Tables below demonstrate the customize transformation tables for protocol and flag feature columns.

Table 5.2 Protocol Column Feature Transformation Table

Protocol -2	No
TCP	1
UDP	2
ICMP	3

Table 5.3 Flag Column Feature Transformation Table

Flag-4	No
OTH	1
REJ	2
RSTO	3
RSTO0	4
RSTR	5
S0	6
S1	7
S2	8
S3	9
SF	10
SH	11

The same procedure is applied to the service column. Label column contains either normal or the sub-type attack label. Transforming this column was applied at two steps. First the sub attack type was represented with the main attack type, and then the main attack type was transformed to numeric using 5 columns, each class is represented with value one using one column. Table below represents the customization transformation for the main classes.

Table 5.4 Label Transformation Table

Label -42	Column1	Column2	Coulmn3	Column4	Column5
Normal	1	0	0	0	0
DoS	0	1	0	0	0
U2R	0	0	1	0	0
R2L	0	0	0	1	0
Prob.	0	0	0	0	1

The following figures represent the data feature columns before and after transformation:

Table 5.5 Feature Columns before transformation

0, tcp, ftp_data, SF, 491, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2, 0, 0, 0, 0, 1, 0, 0, 150, 25, 0.17, 0.03, 0.17, 0, 0, 0, 0.05, 0, normal

Table 5.6 Feature Columns after transformation

0, 1, 18, 10, 491, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2, 0, 0, 0, 0, 1, 0, 0, 150, 25, 0.17, 0.03, 0.17, 0, 0, 0, 0.05, 0, 1,0,0,0,0

2. Standardization: Means that subtract the mean and divide by the standard deviation, so the training subset matrix is processed by mapping each row's means to 0 and standard deviations to 1. It's important to mention that the main testing dataset also should be standardized using the mean and the variance of the training dataset before performing the simulation. The standardization can be done using the Matlab function mapstd.

5.1.3 Feature Classification Phase

The dataset will be divided into 5 groups (classes) according to the attack type using the k- Nearest Neighbor machine learning algorithm.

K-Nearest Neighbor algorithm is a supervised learning, which has corresponding target to each instance in the training dataset. A training dataset is an ordered pair (x, y) where x is an instance and y is a label. The goal is to predict labels for testing dataset. The k-nearest neighbor algorithm is amongst the simplest of all machine learning algorithms: an object is classified by a majority vote of its neighbors, with the instance being assigned to the most common class amongst its k-nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of its nearest neighbor (Elkan, 2011).

kNN classifier only requires an integer k, a set of labeled examples and a measure of "closeness". kNN has the following advantages: analytically tractable, simple implementation and if we assume that the points are d-dimensional, then the straight forward implementation of finding k Nearest Neighbor takes O(n) time (Thirumuruganathan, 2010). On the other hand kNN classifier has the following disadvantages: large storage requirements, computationally intensive recall and highly susceptible to the curse of dimensionality (TAMU Computer Science and Engineering, n.d.).

In this stage the kNN classifier is used to classify the NSL-KDD dataset into 5 classes (Normal, DoS, U2R, R2L and Probe). The neighbor distance was measured using first norm instead of Euclidean distance and first nearest neighbor was used instead of using large value of k. Using large values for k will lead destroy the locality of the estimation and in addition, it will increase the computational burden (TAMU Computer Science and Engineering, n.d.).

k- Nearest Neighbor classification algorithm procedure:

1. Store NSL-KDD99 training dataset with its corresponding label.

2. For each connection in the NSL-KDD99 testing dataset calculate the norm difference (distance) with every connection in the training set using first norm. The maximum absolute column sum norm $\|A\|_1$ is defined as (Naoum, 2011):

$$\|A\|_1 = max_j \sum_{i=1}^{n} |a_{ij}|$$

3. Sort the distances in ascending order. Then with k=1 select the first minimum nearest neighbor.

4. Select the label for the nearest neighbor example. This label is the prediction for this test example.

5. Repeat the above procedure for all the connections in the testing dataset.

After classifying the testing set into 5 classes using k-Nearest Neighbor, the dataset will also be classified using the enhanced resilient backpropagation neural network.

5.1.4 Training Phase

The enhanced resilient backpropagation neural network will be trained by adjusting the weights until the error between the desired output and the neural output is below some predefined value (e.g. e^{-10}). Mean Square Error (MSE) will be used to find the norm between the desired output and the neural output. The learning process is essentially an optimization process in which the parameters of the best set of connection coefficients (weights) for solving a problem are found (Moradi & Zulkernine, n.d.)

It is very difficult to know which training algorithm will be the fastest for a given problem. Our experiments have shown that the resilient backpropagation (trainrp) algorithm is the fast algorithm and may be the fastest on detecting intrusions, and the memory requirements for this algorithm are relatively small in comparison to the other algorithms considered.

In order to improve the convergence speed of the resilient backpropagation an optimal learning factor parameter was derived, and the algorithm was enhanced and it is called the enhanced resilient backpropagation.

5.1.4.1 Enhanced Resilient Backpropagation (ERBP)

The general learning rule formula is identified as:

$$w^{(m+1)} = w^m + \xi(t^m - d^m)z^m$$

Where

The optimal weight $\mathbf{w}^*$ which is the correct weight solution will be used to improve the convergence speed where the neural network will settle in the global minima instead the local. Using the above equation $\mathbf{w}^*$ is subtracted at both sides of the equation. The learning rule, where assuming $\mathbf{w}^*$ is the correct weight solution becomes:

Now if z^m is correctly classifed there is no need to update the weights, but if z^m is misclassified, then:

$$\|w^{(m+1)} - w^*\|^2 = \|w^m - w^*\|^2 + \xi^2\|z^m\|^2 + 2\xi(t^m - d^m)[(w]^m - w^*)z^{\square}$$

Where $\|\cdot\|$ can be any norm, however in this research 1- norm was used. The term $(t^m - d^m)^2$ equals 1, because when the target is one the neural output will be zero and vice versa. The target and the neural output will never be equal because we assumed from the beginning that $z^{□}$ is missclassified.

Now it can be shown that:

and

Then substitute the above two formulas in the main equation, we have:

$$\|w^{(m+1)} - w^*\|^2 = \|w^m - w^*\|^2 + \xi^2\|z^m\|^2 - 2\xi\left(\left|\llbracket(w\rrbracket^{*})^T z^m\right| + \left|\llbracket(w\rrbracket^{m})^T z^{□}\right|\right)$$

Then the optimal step size can be derived by minimizing the mean square error (MSE), where $\|w^{(m+1)} - w^m\| \to 0$ over ξ_{opt}:

$$\xi_{opt} = \frac{\left|\llbracket(w\rrbracket^{*})^T z^m\right| + \left|\llbracket(w\rrbracket^{m})^T z^m\right|}{\|z^m\|^2}$$

Substitute ξ_{opt} in the learning rule equation:

$$w^{(m+1)} = w^m + \frac{\llbracket(w\rrbracket^{*} - w^m)^T z^m}{\|z^m\|^2} z^{□}$$

We can't use ξ_{opt} since the optimal weight value can't be determined in advance. Therefore a relaxation method will be used by replacing the unknown term $\llbracket(\mathbf{w}\rrbracket^{*})^{\mathbf{T}}□^{\mathrm{m}}$ by $\delta,$ where $0 \leq □ \leq \delta^*$: $\delta^* = min_m|w^{*T}z^m|$

Thus the learning rule becomes:

$$w^{(m+1)} = w^{m} + \frac{(t^{m} - d^{m})(\delta + |w^{mT} z^{m}|)}{\|z^{m}\|^{2}} z^{m}$$

Using ξ_{opt}, the pseudo code for the enhanced resilient propagation becomes (Riedmiller & Braun 1993, Naoum 2011):

```
$\Delta(m) = minimum(\Delta(m-1) * \eta^{+}, \Delta_{\max}\,\square)$
$\Delta w(m) = -sign\left(\frac{\partial E}{\partial w}(m)\right) * \Delta(m)$
$w(m+1) = w(m) + \xi_{opt}\Delta w(m)$
}



$\Delta(m) = maximum(\Delta(m-1) * \eta^{-}, \Delta_{\min}\,\square)$
$w(m+1) = w(m) - \xi_{opt}\Delta w(m-1)$
$\frac{\partial E}{\partial w}(m) = 0$
}



 $\Delta w(m) = -sign\left(\frac{\partial E}{\partial w}(m)\right) * \Delta(m)$
$w(m+1) = w(m) + \xi_{opt}\Delta w(m)$
}
}
```

5.1.4.2 Training the Enhanced Resilient Backpropagation Neural Network (ERBP)

Two of the problems that occur during the training of the enhanced resilient backpropagation are over-fitting and false well. Over-fitting occurs when the enhanced resilient backpropagation neural network produced an error on the testing dataset to be larger than the error on the training dataset. The enhanced resilient backpropagation neural network has memorized the training examples, but it has not learned to generalize to new intrusions. False Well occurs when the enhanced resilient backpropagation settles in the local minima instead the global minima (solution).

One method to solve both problems and to improve the enhanced resilient backpropagation network generalization, is to use an enhanced resilient backpropagation neural network that is just large enough (optimum number of hidden layers and hidden neurons) to provide an adequate fit. The idea of using an optimum number of hidden layers and neurons that the enhanced resilient backpropagation neural network will not have enough power to over-fit the data. The experiments have shown that the best number of hidden layers and neurons was one hidden layers with 34 hidden neurons.

5.1.4.2.1 Early Stopping

The default method for improving generalization is called early stopping. This technique is automatically provided for all of the supervised network creation functions, including the pattern recognition network creation functions patternnet (MathWorks Matlab Help, 2011).

In this technique the available data is divided into three subsets. The first subset is the training set, which is used for computing the gradient and updating the network weights and biases. The second subset is the validation set. The error on the validation set is monitored during the training process. The validation error normally decreases during the initial phase of training, as does the training set error. However, when the network begins to over-fit the data, the error on the validation set typically begins to rise. When the validation error increases for a specified number of iterations (net.trainParam.max_fail), the training is stopped, and the weights and biases at the minimum of the validation error are returned. The test set error is not used during training, but it is used to compare different models (MathWorks Matlab Help, 2011).

In our experiments random division function was used to divide the main training subset using the default ration in Matlab which is: 70% training subset, 15% validation subset and 15% testing subset.

To make the proposed system more realistic in this research, a new inputs in the testing dataset were represented, records that the neural network have never been seen including new sub attacks that the network did not see in the training process.

To provide maximum generalization, we started with only one hidden layer using different number of hidden neurons iteratively. We used the iterative process because using high number of hidden neurons will lead to over-fitting problem, where the neural will not be able to classify new records. Generally if there are no good results then a second layer can be added to improve the neural performance. In this paper we started the training process using 10 hidden neurons as the minimum number up to 37 hidden neurons as the maximum number. The maximum number of hidden neurons was calculated using the following equations (Heath, 2011):

I (input dimensionality) =41, O (classes) =5 and Ntrn (training vectors) = 2471

Neq (Number of output equations) = Ntrn*O=2471*5= 12355

Hmax1=floor ((Neq-O)/(I+O+1))=floor((12355-5)/(41+5+1))=262

To select the best number of hidden neurons Neq>>Nw, where:

Nw (Number of unknown weights) = (I+1)*H+ (1+H)*O; where H is the number of hidden neurons.

Suppose that Neq > r * Nw, where r=7, then:

Hmax7=round (Hmax1/r) =37

Experiments have shown that when H=34 the enhanced resilient backpropagation neural performance gave best classification rate. Therefore if H=34, number of output equations (Neq) will be greater than number of unknown weights (Nw):

$$Nw = (41+1)*34+(1+34)*5=1603$$

and

$$(Neq)\quad 12355 >> (Nw)\quad 1603$$

According to the above equations the maximum number of hidden neurons is 37, therefore we started the training process with 10 incremented by 2 neurons ending with the maximum which is 37. The results have shown that the best number of hidden neurons was 34.

Training phase demonstrates that one hidden layer might be enough to solve a classification problem. This verifies the Kolmogarov theorem (Naoum, 2011) which states that a 3 layer perceptron could be used to create any continuous like hood function required in a classifier.

5.1.5 Testing Phase

In this phase, testing dataset will be classified by both k-Nearest Neighbor and the enhanced resilient backpropagation neural network which was trained during the training phase using the best number of hidden neurons and layers. The designed system will be evaluated by calculating the Detection Rate (DR), Accuracy Rate (AR), False

Positive Rate (FPR) etc. for each classifier and then finally the results of the combined models will be calculated. The results and experiments details are described in chapter six.

CHAPTER SIX

6 EXPERIMENTS RESULTS AND CONCLUSION

6.1 Matlab Programming Language

In this research, k-Nearest Neighbor and enhanced resilient backpropagation was trained to detect intrusions using Matlab 2011 software.

MATLAB (Matrix Laboratory) is a programming environment for algorithm development, data analysis, visualization, and numerical computation. MATLAB can solve technical computing problems faster than with traditional programming languages, such as C, C++, and FORTRAN. MATLAB can be used in a wide range of applications, including signal and image processing, communications, control design, test and measurement, financial modeling and analysis, and neural networks. For a million engineers and scientists in industry and academia, MATLAB is the language of technical computing (MathWorks Matlab Help, 2011).

6.2 NSL-KDD Testing Dataset

Training dataset was used to tune the weights and testing dataset was used for the network evaluation. Testing dataset contains some attacks that is not represented in the training dataset. The testing dataset (labeled and unlabeled) details are shown in the tables 6.1 and table 6.2 respectively:

Table 6.1 Testing Datasets (Labeled) Analysis Details

Testing (Labeled) Datasets	Class Size
Normal	1000
Denial of Service (DoS)	2200
User to Root (U2R)	37
Root to Local (R2L)	2200
Prob.	2200
Total	7637

Table 6.2 Testing Datasets (Unlabeled) Analysis Details

Testing Dataset (Unlabeled)	Class Size
Unknown	3750

6.3 k- Nearest Neighbour Results (kNN)

At the first stage in classification, k-Nearest Neighbor classifier will be used to classify the testing dataset into 5 classes, using the parameters which are given in table 6.3:

Table 6.3 kNN Parameters

K- Nearest Neighbour value	1
Distance Measure	Norm -1

Using the first norm demonstrates better results than using the second norm. Tables (6.4 & 6.5) below represent the classification rates for each class (labeled) and unlabeled using the second norm:

Table 6.4 kNN Classification Results – Second Norm (Labeled)

Testing Datasets (Labeled)	Class Size	Detected Size	Attack Detection Rate
Normal	1000	920	92%
DoS	2200	1928	87.6%
U2R	37	16	43.2%
R2L	2200	808	36.7%
Prob.	2200	2024	92%
Total	7637	5696	74.58%

Table 6.5 kNN Classification Results – Second Norm (Unlabeled)

Testing (Unlabeled) Datasets	Class Size	Detected Size	Detection Rate
Unknown attacks	3750	2041	54.4%

The results for using the first norm in the k-Nearest Neighbor for labeled testing dataset are given in the table 6.6:

Table 6.6 kNN Classification Results – First Norm (Labeled)

Testing Datasets (Labeled)	Class Size	Detected Size	Attack Detection Rate
Normal	1000	929	92.9%
DoS	2200	1950	88.6%
U2R	37	18	48.6%
R2L	2200	801	36.4%
Prob.	2200	2114	96.1%
Total	7637	5812	76.1%

Classification rates in table 6.6 shows that the results for the first norm were better than the second norm; therefore the first norm was used in the k-Nearest Neighbor implementation as the distance measure.

Classifiers are best judged by the classification rate distribution in the confusion matrix. Table 6.7 below demonstrates the confusion matrix of the kNN classifier:

Table 6.7 kNN Classifier Confusion Matrix

Confusion Matrix	kNN Result					
Target	**1**	**2**	**3**	**4**	**5**	**Grand Total**
1	929	19	3	13	36	1000
2	17	1950	0	2	231	2200
3	12	0	18	6	1	37
4	1077	52	77	801	193	2200
5	0	79	7	0	2114	2200
Grand Total	**2035**	**2100**	**105**	**822**	**2575**	**7637**

k-Nearest Neighbor classifier was unable to detect the unlabeled attacks in the testing dataset, the results reveal that kNN was able to detect only about 55 % of the unlabeled records, as shown in the table 6.8:

Table 6.8 kNN Classification Results (Unlabeled)

Testing (Unlabeled) Datasets	Class Size	Detected Size	Detection Rate
Unknown attacks	3750	2063	55%

As noticed in table 6.6, kNN was able to classify normal class with good detection rate, but U2R attack is considered as one of the hardest class to classify; has an attack detection rate about 48.6%. DoS and Probe attacks have a reasonably good detection rate but kNN had a very poor result in R2L detection.

Table 6.9 demonstrates the TP, FP, FN, Precision, Recall, FPR and FNR for each attack:

Table 6.9 Evaluation Results for Each Class (kNN Classifier)

Attack	TP	FP	FN	Recall	Precision	FPR	FNR
DoS	1950	19	17	99%	99%	0.9%	0.9%
U2R	18	3	12	60%	85.7%	8.1%	40%
R2L	801	13	1077	42.7%	98.4%	0.6%	57.3%
Prob.	2114	36	0	100%	98.33%	1.64%	0%

Tables (6.10 & 6.11) below demonstrate the confusion matrix of the predictive classes versus the actual classes and the evaluation formulas respectively:

Table 6.10 kNN Predictive vs. Actual Classes

	Predictive Positive	Predictive Negative
Actual Positive	5531 (TP)	1106 (FN)
Actual Negative	71 (FP)	929 (TN)

Table 6.11 kNN Classifier Evaluation Formulas

Detection Rate = 76.1%	Accuracy Rate = 84.6%
Recall = 83%	Precision = 99%
False Positive Rate = 7%	False Negative Rate = 17%
Specificity = 93%	G-Mean = 88%
F-Measure = 90%	

As noticed in table 6.11, kNN classifier was able to detect records with a classification rate of approximately 76%, but kNN had a low false positive, which means only 71 normal records were detected as intrusion. Therefore the precision rate was 99% which is considered a high rate. kNN had a very bad performance in terms of false negative rate, where 1106 attacks were detected and seen as normal. Therefore using kNN classifier independently is not a good choice because the system is not very accurate in detecting intrusions.

6.4 Enhanced Resilient Backpropagation Artificial Neural Network Results (ERBP)

At the second stage in classification, the enhanced resilient backpropagation neural network will be used also to classify the testing dataset into 5 classes. The neural network was trained using the following parameters:

Table 6.12 Enhanced Resilient Artificial Neural Network Parameters

Parameters	Details
Learning	Supervised
Input Layer	One input layer with 41 neurons (input dimensionality)
Hidden Layer	One hidden layer with 34 neurons
Output Layer	One output layer with 5 neurons (Classes)
Number of epochs	194
Performance	Mean Square Error
Transfer Function	Hyperbolic tangent sigmoid (tansig)

As mentioned in the training phase section, the number of hidden neurons was selected carefully and precisely according to the confusion matrix results. The neural network was trained using the ordinary resilient backpropagation with one hidden layer containing 34 hidden neurons that were selected as the best result in terms of classification rate. The classification rate for different number of hidden neurons is described in table 6.13:

Table 6.13 Numbers of Hidden Neurons vs. Detection Rate

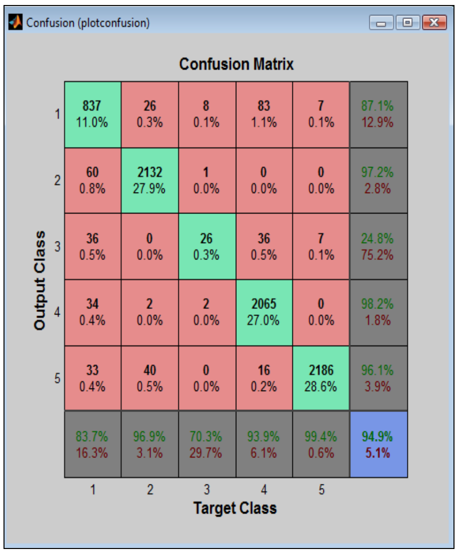
Hidden Neurons	Detection Rate	Iterations (Epochs)
10	93.4%	230
12	93.4%	414
14	94.5%	245
16	92.4%	159
18	93.3%	169
20	91.5%	265
22	93.2%	193
24	94.6%	214
26	93.6%	207
28	93.4%	217
30	92.8%	223
32	94.2%	154
34	94.9%	325
36	94.6%	174
38	94.5%	178
40	91.2%	109

Using the enhanced resilient backpropagation for 34 hidden neurons has improved the performance of the system in terms of classification rate and number of epochs in comparison with the ordinary resilient backpropagation. Table 6.14 demonstrates the difference between them:

Table 6.14 Enhanced resilient backpropagation vs. Ordinary resilient backpropagation

Algorithm	Detection Rate	Epochs
Ordinary resilient backpropagation	94.9%	325
Enhanced resilient backpropagation	95.7%	194

Figures (6.1 & 6.2) below represent the confusion matrix of the enhanced and the ordinary resilient backpropagation using 34 hidden neurons:

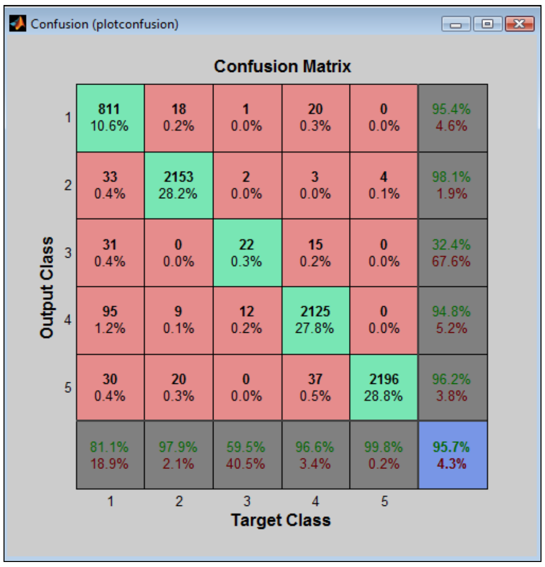


Confusion (plotconfusion)

Confusion Matrix

Output Class \ Target Class	1	2	3	4	5	
1	837 11.0%	26 0.3%	8 0.1%	83 1.1%	7 0.1%	87.1% 12.9%
2	60 0.8%	2132 27.9%	1 0.0%	0 0.0%	0 0.0%	97.2% 2.8%
3	36 0.5%	0 0.0%	26 0.3%	36 0.5%	7 0.1%	24.8% 75.2%
4	34 0.4%	2 0.0%	2 0.0%	2065 27.0%	0 0.0%	98.2% 1.8%
5	33 0.4%	40 0.5%	0 0.0%	16 0.2%	2186 28.6%	96.1% 3.9%
	83.7% 16.3%	96.9% 3.1%	70.3% 29.7%	93.9% 6.1%	99.4% 0.6%	94.9% 5.1%

Figure 6.1 Ordinary Resilient Backpropagation Neural Network (Labeled)



Confusion (plotconfusion)

Confusion Matrix

Output Class \ Target Class	1	2	3	4	5	
1	811 10.6%	18 0.2%	1 0.0%	20 0.3%	0 0.0%	95.4% 4.6%
2	33 0.4%	2153 28.2%	2 0.0%	3 0.0%	4 0.1%	98.1% 1.9%
3	31 0.4%	0 0.0%	22 0.3%	15 0.2%	0 0.0%	32.4% 67.6%
4	95 1.2%	9 0.1%	12 0.2%	2125 27.8%	0 0.0%	94.8% 5.2%
5	30 0.4%	20 0.3%	0 0.0%	37 0.5%	2196 28.8%	96.2% 3.8%
	81.1% 18.9%	97.9% 2.1%	59.5% 40.5%	96.6% 3.4%	99.8% 0.2%	95.7% 4.3%

Figure 6.2 Enhanced Resilient Backpropagation Neural Network (Labeled)

More specifically the enhanced resilient backpropagation neural network was able to classify the labeled testing dataset as follows:

Table 6.15 Enhanced Resilient Artificial Neural Network Results (Labeled)

Testing Datasets (Labeled)	Class Size	Detected Size	Attack Detection Rate
Normal	1000	811	81.1%
DoS	2200	2153	97.9%
U2R	37	22	59.5%
R2L	2200	2125	96.6%
Prob.	2200	2196	99.8%
Total	7637	7307	95.7%

Enhanced resilient backpropagation had a reasonable good detection rate when detecting novel attacks:

Table 6.16 Enhanced Resilient Artificial Neural Network Results (Unlabeled)

Testing (Unlabeled) Datasets	Class Size	Detected Size	Detection Rate
Unknown attacks	3750	3210	86%

The enhanced resilient backpropagation was trained using the early stopping technique, where the validation subset was used in order to stop the training process when the number of maximum validation check point was reached. In this research the number of maximum validation fail was 6. Figures (6.3 & 6.4) represent the neural network architecture and the training parameters including the training time required (seconds):

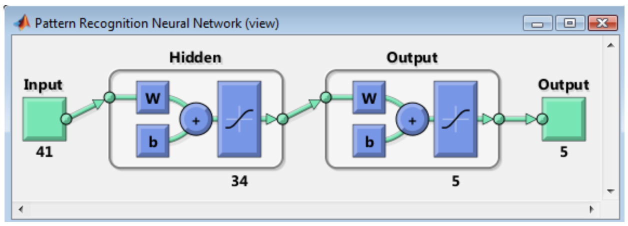
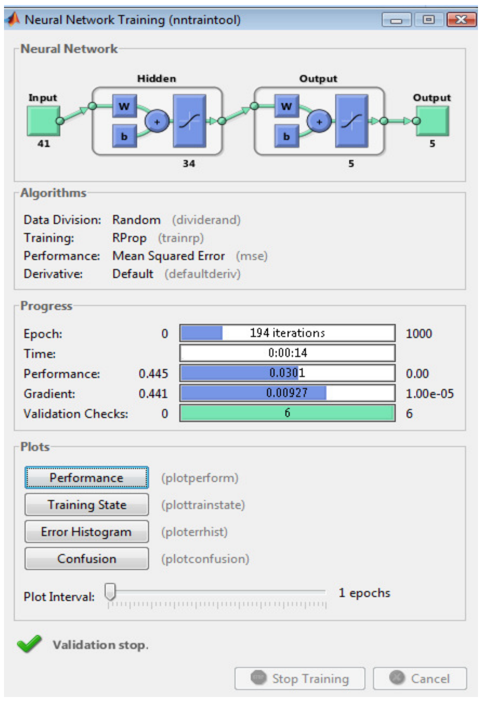


Figure 6.3 Enhanced Resilient Backpropgation Neural Network Architecture



Algorithms

Data Division:	Random (dividerand)
Training:	RProp (trainrp)
Performance:	Mean Squared Error (mse)
Derivative:	Default (defaultderiv)

Progress

Epoch:	0	194 iterations	1000
Time:		0:00:14	
Performance:	0.445	0.0301	0.00
Gradient:	0.441	0.00927	1.00e-05
Validation Checks:	0	6	6

Plots

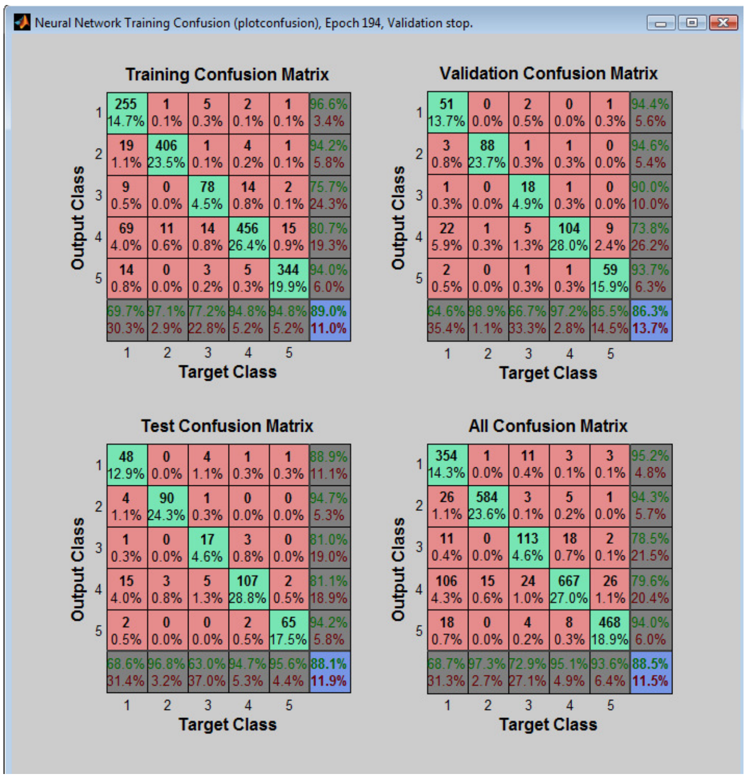
Performance	(plotperform)
Training State	(plottrainstate)
Error Histogram	(ploterrhist)
Confusion	(plotconfusion)

Plot Interval: 1 epochs

✓ Validation stop.

Figure 6.4 Enhanced Resilient backpropagation training process

Figure 6.5 represents the confusion matrix of the training, validation and testing subsets:


Neural Network Training Confusion (plotconfusion), Epoch 194, Validation stop.

Training Confusion Matrix

Output Class \ Target Class	1	2	3	4	5	
1	255 14.7%	1 0.1%	5 0.3%	2 0.1%	1 0.1%	96.6% 3.4%
2	19 1.1%	406 23.5%	1 0.1%	4 0.2%	1 0.1%	94.2% 5.8%
3	9 0.5%	0 0.0%	78 4.5%	14 0.8%	2 0.1%	75.7% 24.3%
4	69 4.0%	11 0.6%	14 0.8%	456 26.4%	15 0.9%	80.7% 19.3%
5	14 0.8%	0 0.0%	3 0.2%	5 0.3%	344 19.9%	94.0% 6.0%
	69.7% 30.3%	97.1% 2.9%	77.2% 22.8%	94.8% 5.2%	94.8% 5.2%	89.0% 11.0%

Validation Confusion Matrix

Output Class \ Target Class	1	2	3	4	5	
1	51 13.7%	0 0.0%	2 0.5%	0 0.0%	1 0.3%	94.4% 5.6%
2	3 0.8%	88 23.7%	1 0.3%	1 0.3%	0 0.0%	94.6% 5.4%
3	1 0.3%	0 0.0%	18 4.9%	1 0.3%	0 0.0%	90.0% 10.0%
4	22 5.9%	1 0.3%	5 1.3%	104 28.0%	9 2.4%	73.8% 26.2%
5	2 0.5%	0 0.0%	1 0.3%	1 0.3%	59 15.9%	93.7% 6.3%
	64.6% 35.4%	98.9% 1.1%	66.7% 33.3%	97.2% 2.8%	85.5% 14.5%	86.3% 13.7%

Test Confusion Matrix

Output Class \ Target Class	1	2	3	4	5	
1	48 12.9%	0 0.0%	4 1.1%	1 0.3%	1 0.3%	88.9% 11.1%
2	4 1.1%	90 24.3%	1 0.3%	0 0.0%	0 0.0%	94.7% 5.3%
3	1 0.3%	0 0.0%	17 4.6%	3 0.8%	0 0.0%	81.0% 19.0%
4	15 4.0%	3 0.8%	5 1.3%	107 28.8%	2 0.5%	81.1% 18.9%
5	2 0.5%	0 0.0%	0 0.0%	2 0.5%	65 17.5%	94.2% 5.8%
	68.6% 31.4%	96.8% 3.2%	63.0% 37.0%	94.7% 5.3%	95.6% 4.4%	88.1% 11.9%

All Confusion Matrix

Output Class \ Target Class	1	2	3	4	5	
1	354 14.3%	1 0.0%	11 0.4%	3 0.1%	3 0.1%	95.2% 4.8%
2	26 1.1%	584 23.6%	3 0.1%	5 0.2%	1 0.0%	94.3% 5.7%
3	11 0.4%	0 0.0%	113 4.6%	18 0.7%	2 0.1%	78.5% 21.5%
4	106 4.3%	15 0.6%	24 1.0%	667 27.0%	26 1.1%	79.6% 20.4%
5	18 0.7%	0 0.0%	4 0.2%	8 0.3%	468 18.9%	94.0% 6.0%
	68.7% 31.3%	97.3% 2.7%	72.9% 27.1%	95.1% 4.9%	93.6% 6.4%	88.5% 11.5%

Figure 6.5 Training, Validation and Testing subsets Confusion Matrix

The training process was terminated as the validation point was reached. The MSE for the 3 subsets and the main testing subsets was as follows:

Table 6.17 Training, Validation, Testing and Main Testing MSE

H	Nepochs	MSE train	MSE Val	MSE Test	MSE Main Test
34	194	0.0305	0.0394	0.0325	0.0202

The above table 6.17 represents the mean squsre error for the three subsets in the training process. As noticed from the table the best performance was at 0.0394 MSE for the validation subset, after this point the validation MSE starts to rise up, that means that the enhanced resilient backpropagation is strarting to over-fit the data, that when the enhanced resilient backpropagation neural network stops training and the previous best perfromance for the validation subset is recorded.

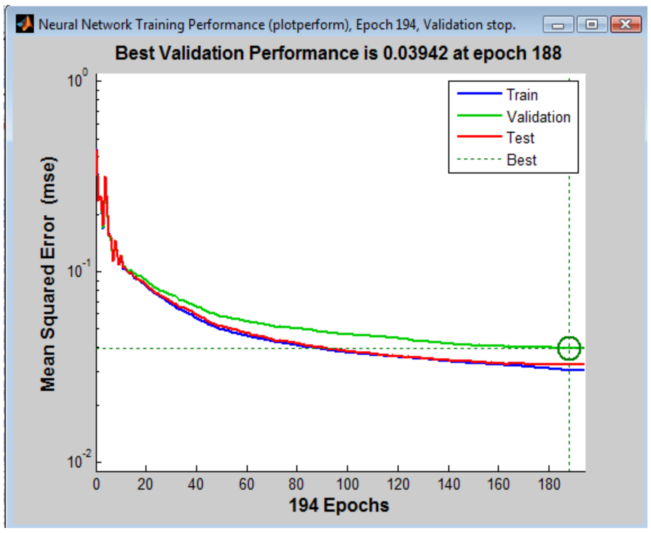


Figure 6.6 Neural Network Training Performance

The evaluation formulas for each intrusion attacks are:

Table 6.18 Evaluation Results for Each Class (ERBP Classifier)

Attack	TP	FP	FN	Recall	Precision	FPR	FNR
DoS	2153	33	18	99.2%	98.5%	1.5%	0.8%
U2R	22	31	1	95.7%	41.5%	83.8%	4.3%
R2L	2125	95	20	99.1%	95.7%	4.3%	0.9%
Prob.	2196	30	0	100%	98.65%	1.36%	0.0%

Tables (6.19 & 6.20) demonstrate the confusion matrix for the predictive classes versus the actual classes and the evaluation formulas respectively for ERBP classifier:

Table 6.19 ERBP Predictive vs. Actual Classes

	Predictive Positive	Predictive Negative
Actual Positive	6598 (TP)	39 (FN)
Actual Negative	189 (FP)	811 (TN)

Table 6.20 ERBP Classifier Evaluation Formulas

Detection Rate = 95.7%	Accuracy Rate = 97%
Recall = 99%	Precision = 97%
False Positive Rate = 19%	False Negative Rate = 1%
Specificity = 81%	G-Mean = 90%
F-Measure = 98%	

From table 6.20 it can be concluded that ERBP classifier was able to detect records with a classification rate of approximately 96%, which is considered very good performance and with false negative rate of 1%. ERBP Recall and Precisions have very good results were the system trustworthy is guaranteed to detect intrusions especially when the false negative records were only 39 (intrusions detected as normal). But the drawback of the ERBP was the false positive rate, where the system will have many false alarms and this is considered as a disadvantage of the ERBP classifier because it requires extra work for the administrator to check whether the alarm is correct or incorrect for that connection.

6.5 Hybrid System (kNN_ERBP) Results

Each classifier phase has its own advantages and disadvantages, therefore combining the results for both classifiers the performance of the hybrid system is improved. The results of the hybrid kNN_ERBP are shown in tables (6.20 & 6.21) which represent the confusion matrix of the 5 classes (labeled) and the detection rate for the unlabeled testing dataset:

Table 6.21 Hybrid System (kNN_ERBP) Confusion Matrix (Labeled)

Confusion		**Target**				
		Normal	DoS	U2R	R2L	Prob.
Output	Normal	929	18	1	20	0
	DoS	19	2153	2	3	4
	U2R	3	0	22	15	0
	R2L	13	9	12	2125	0
	Prob.	36	20	0	37	2196
	DR	92.9%	97.9%	59.5%	96.6%	99.8%

The main advantage of the hybrid system is that it was able to detect normal class with a reasonable good detection rate using the k-Nearest Neighbor classifier, and using the enhanced resilient backpropagation neural network it was able to detect denial of service (DoS), root to local (R2L) and prob. with a very high attack detection rate, but unfortunately the hybrid system is still unable to correctly classify the user to root (U2R) attack.

The hybrid system was able to detect novel (unseen) attacks with a good detection rate:

Table 6.22 Hybrid System (kNN_ERBP) (Unlabeled)

Testing (Unlabeled) Datasets	Class Size	Detected Size	Detection Rate
Unknown attacks	3750	3451	92%

Evaluation results for each class using the hybrid system was as follows:

Table 6.23 Evaluation Results for Each Class (Hybrid)

Attack	TP	FP	FN	Recall	Precision	FPR	FNR
DoS	2153	19	18	99.2%	99.1%	0.9%	0.8%
U2R	22	3	1	95.7%	88%	8.1%	4.3%
R2L	2125	13	20	99.1%	99.4%	0.6%	0.9%
Prob.	2196	36	0	100%	98.39%	1.64%	0.0%

The evaluation rates for the hybrid system was improved using both classifiers (k-Nearest Neighbor and Enhanced Resilient Backpropagation Neural Network), the following tables (6.24 & 6.25) demonstrate the confusion matrix of the predictive classes versus the actual classes and the evaluation formulas respectively:

Table 6.24 Hybrid Predictive vs. Actual Classes

	Predictive Positive	Predictive Negative
Actual Positive	6598 (TP)	39 (FN)
Actual Negative	71 (FP)	929 (TN)

Table 6.25 Hybrid Classifier Evaluation Formulas

Detection Rate = 97.2%	Accuracy Rate = 99%
Recall = 99%	Precision = 99%

False Positive Rate = 7%	False Negative Rate = 1%
Specificity = 93%	G-Mean = 96%
F-Measure = 99%	

As noticed above, the hybrid (kNN_ERBP) system has improved the performance of the previous models, where the classification rate for 5 classes rises up to 97%. The recall and precision were improved especially when the false positive minimized to 71 instead of 189 using ERBP. The main advantage of the hybrid system is the false negative rate which it's considered the most critical evaluation is about 1%. On the other hand the system has a false positive rate (false alarm) of 7%.

Finally the hybrid system performance is compared to other intrusion detection systems that use either neural network (supervised, unsupervised), k-means machine learning algorithm and naïve bayes classifier.

The hybrid system is compared to k-means intrusion detection system proposed by Nieves. The results demonstrate that the proposed system has better result in terms of detection rate but the k-means system has better results in terms of false positive rate. Unsupervised learning (ex. K-means) usually has better results in terms of false positive rate.

Table 6.26 Intrusion Hybrid (kNN_ERBP) vs. K-means (Nieves, 2009)

Evaluation	Hybrid (kNN_ERBP)	K-Means
DR	97.2%	89%
FPR	7%	4.8%

The proposed hybrid system is compared to Kohonen Self Organizing Maps (SOM) with Conscience function. Table 6.27 shows that the proposed system has better results in terms of detection rate, recall rate, precision, false negative rate and f-measure. Kohonen SOM (unsupervised learning) has better results in terms of false positive rate.

Table 6.28 demonstrates that the proposed hybrid system has better results in detecting R2L and Prob attacks than the Conscience SOM.

Table 6.27 Hybrid (kNN_ERBP) vs. SOM (Al-Rashdan, 2011)

Evaluation	Hybrid (kNN_ERBP)	SOM+ Conscience
DR	97.2%	92.5%
AR	99%	-
Recall	99%	94.7%
Precision	99%	96.4%
FPR	7%	3.5%
FNR	1%	5.2%
F-Measure	99%	95.5%

Table 6.28 Hybrid (kNN_ERBP) vs. SOM (Al-Rashdan, 2011) (4 attacks)

Evaluation	Hybrid (kNN_ERBP)			SOM + Conscience Function		
	DR	FPR	FNR	DR	FPR	FNR
DoS	97.9%	0.9%	0.8%	100%	0%	0%
U2R	59.5%	8.1%	4.3%	80%	0%	20%
R2L	96.6%	0.6%	0.9%	88.2%	5.9%	0%
Prob.	99.8%	1.64%	0%	94.1%	0%	5.9%

Also the proposed hybrid system is compared to ordinary backpropagation proposed by Brifcani and Issa. Table 6.29 illustrates that the detection rate of proposed system has better results than the backpropagation neural network. Table 6.30 shows that the hybrid system has better detection rate for all attack types than the backpropagation neural network.

Table 6.29 Hybrid (kNN_ERBP) vs. BP (Brifcani and Issa, 2011)

Evaluation	Hybrid (kNN_ERBP)	BP
DR	97.2%	91.8%

Table 6.30 Hybrid (kNN_ERBP) vs. BP (Brifcani and Issa, 2011) (4 attacks)

Evaluation	Hybrid (kNN_ERBP)	BP
	DR	DR
DoS	97.9%	97.25%
U2R	59.5%	0%
R2L	96.6%	32.5%
Prob.	99.8%	81.23%

Finally the proposed hybrid system is compared to k-Means & Naïve Bayes proposed by Islim. Table 6.31 illustrates that the recall, precision and f-measure of proposed system has better results. Table 6.32 shows that the hybrid system has better detection rate for R2L and Prob attacks.

Table 6.31 Hybrid (kNN_ERBP) vs. k-Means and Naïve Bayes (Islim, 2012)

Evaluation	Hybrid (kNN_ERBP)	k-Means & Naive Bayes
DR	97.2%	97.9%
AR	99%	-
Recall	99%	97.9%
Precision	99%	98.6%
FPR	7%	0.3%
FNR	1%	-
F-Measure	99%	98.2%

Table 6.32 Hybrid (kNN_ERBP) vs. k-Means and Naïve Bayes (Islim, 2012) (4 attacks)

Evaluation	Hybrid (kNN_ERBP)	k-Means & Naive Bayes
	DR	DR
DoS	97.9%	98.8%
U2R	59.5%	82.1%
R2L	96.6%	61.3%
Prob.	99.8%	96.8%

Figures (6.7 and 6.8) below demonstrate the comparison between the proposed systems against the above mentioned system:

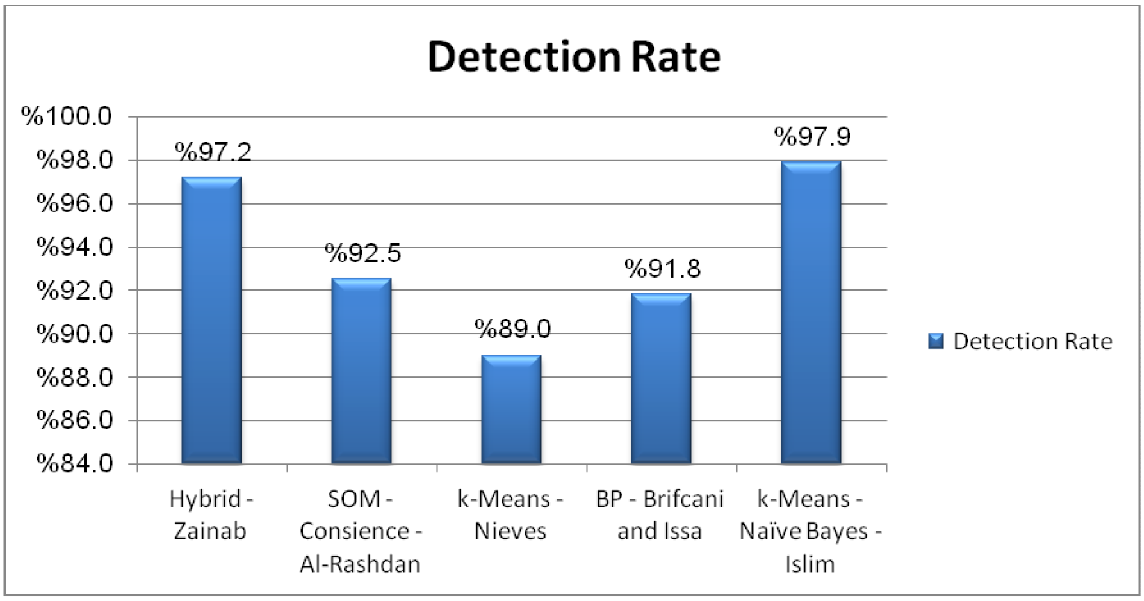


Figure 6.7 Detection Rates of the proposed system vs. other systems

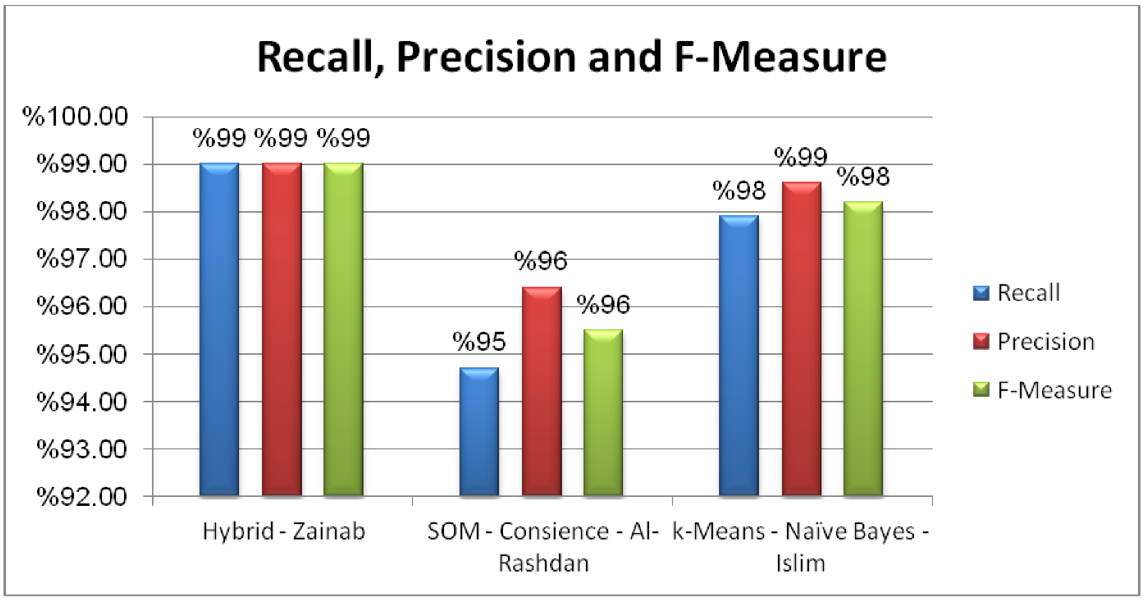


Figure 6.8 Recall, Precision and F-Measure Comparisons

6.6 Conclusion

An intrusion detection system was designed. The detection of intrusions is preformed at two stages using the k-nearest neighbor and the enhanced resilient backpropagation neural network. Each stage was able to classify the records into 5 classes with reasonable good detection rates, but when combining the results of the two stages the system was able to classify intrusions with a high detection rate (DR) about 97.2% and with false negative rate (FNR) of 1% and false positive rate (FPR) of 7%.

Number of records in the training dataset has a great effect on the system performance. Classes in training dataset don't have the same size number, for instance normal and denial of service classes dominate the dataset, therefore in designing both classifiers the number of records for each class was selected approximately equally. The k value for the k-nearest neighbor classifier was selected where the classification rate for the labeled and unlabeled testing datasets was the maximum among other suggested values. k-Nearest Neighbor in the hybrid system was powerful in detecting normal class and it has a very good performance in detecting DoS and Prob attacks, but U2R and R2L attacks were not classified correctly because these specific attacks are very difficult to detect because the dataset contains a few records for both classes, so in this case a learning algorithm such as k-nearest neighbor that uses norm distance as the classification criteria is not an appropriate classifier for detecting those attacks, therefore the enhanced resilient backpropagation neural network was used as the second classifier.

The resilient backpropagation neural network is an enhanced version of the ordinary backpropagation which only uses the sign of the error derivative to determine the amount of the weight modification while the ordinary backpropagation uses both the value and the sign of the derivative. This will lead to the result that the resilient backpropagation require ***less memory*** in storing the parameters where it only stores the derivative sign. To improve the convergence of our proposed system, an optimal learning factor in the weight update function was derived. The experiments results show that the enhanced resilient did improve the system classification rates and the neural network reaches the validation maximum fail number faster than the resilient backpropagation. The best number of hidden layers and neurons was searched among the best minimum mean square error for the validation subset.

The hybrid system of both supervised classifiers was compared against several systems, such as unsupervised learning (k-Means and Kohonen SOM with conscience function), supervised learning (backpropagation BP) and hybrid unsupervised and supervised (k-Means and Naïve Bayes). It can be concluded that if a hybrid system of a supervised and unsupervised neural network models is used, then the intrusion detection system can have high detection, accuracy, recall and precision rate, while maintaining low false negative and false positive rates.

6.7 Future Work

1. Integrating different supervised and unsupervised learning architectures that have not been used before in detecting intrusions.

2. Reducing the dataset features dimensionality.

3. Combining several models (more than 2) in detecting intrusions.

4. Employing Reinforcement/Anti-Reinforcement learning mechanism instead of supervised or unsupervised learning.

5. Using parallel computing.

6. Developing a new Artificial Neural Network to remove local minima.

7. Developing a new theory to enhance the optimal number of hidden layers and neurons; therefore eliminating the over-fitting problem.

8. Using this system as a base for intrusion prevention system.

REFERENCES

[1] Aboshosha, A & Nat, R. (n.d.). Artificial Intelligence, An Introductory Course Lecture 3. Retrieved August 3, 2011, from http://www.icgst.com/cc/AI/Lecture3.pdf

[2] Aboshosha, A & Nat, R. (n.d.). Artificial Intelligence, An Introductory Course Lecture 4. Retrieved August 3, 2011, from http://www.icgst.com/cc/AI/Lecture4.pdf

[3] Abraham, A, Grosan, C, & Chen, Y. (2006). Evolution of Intrusion Detection Systems. Retrieved October 18, 2011, from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.161.5620

[4] Ahmad, I, Swati, S & Mohsin, S. (2007). Intrusions Detection Mechanism by Resilient Back Propagation (RPROP). *European Journal of Scientific Research, EuroJournals Publishing, Inc*, *17*, 523-531. Retrieved November 2, 2011, from http://www.eurojournals.com/ejsr%2017%204.pdf

[5] Ali, A, Saleh, A & Badawy, T. (2010). Intelligent Adaptive Intrusion Detection Systems Using Neural Networks (Comparative study). *International Journal of Video & Image Processing and Network Security IJVIPNS-IJENS*, *10*. Retrieved October 25, 2011, from http://www.ijens.org/101701-6363%20IJVIPNS-IJENS.pdf

[6] AL-Rashdan, W, Naoum, R, Al_Sharafat, W & Al-Khazaaleh, M. (2010). Novel network intrusion detection system using hybrid neural network (Hopfield and Kohonen SOM with conscience function). *IJCSNS International Journal of Computer Science and Network Security*, *10*(11). Retrieved January 26, 2012, http://paper.ijcsns.org/07_book/201011/20101103.pdf

[7] Al-Rashdan, W. (2011). *A Hybrid Artificial Neural Network Model (Hopfield-SOM with Conscience) for Effective Network Intrusion Detection System*, (Doctoral dissertation), The Arab Academy for Banking and Financial Sciences, Jordan.

[8] Bernacki, M. & Włodarczyk, P. (2004). *Backpropagation*. Retrieved October 31,

2011, from http://home.agh.edu.pl/~vlsi/AI/backp_t_en/backprop.html

[9] Best Security Tips [Image] (2007). Retrieved March 14, 2012, from http://www.bestsecuritytips.com/modules/soapbox/images/firewall/2/firewall1.gif

[10] Bishop, M. (2005). *Introduction to Computer Security*. Boston: Pearson Education.

[11] Brifcani, A. & Issa, A. (2011). Intrusion detection and attack classifier based on three techniques: a comparative study. *Eng. & Tech. Journal*, *29*(2). Retrieved December 12, 2011, from http://www.uotechnology.edu.iq/tec_magaz/volum292011/No.2.2011/text/Text%20%2817%29.pdf.

[12] Cannady, J. (1998). Artificial Neural Networks for Misuse Detection. *National Information Systems Security Conference*. Retrieved October 18, 2011, from http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.39.5179

[13] Depren, O., Murat, T., Anarim, E. & Ciliz, M. (2005). An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks. *Expert Systems with Applications, Elsevier*,713-722. Retrieved October 20, 2011, from http://www.ft.unicamp.br/RedesComplexas/downloads/An_intelligent_intrusion_detection_system_for_anomaly_and_misuse_detection_in_computer_networks.pdf

[14] Elkan, C. (2011). *Nearest Neighbor Classification*, University of California, San Diego (UCSD CSE). Retrieved November 13, 2011, from http://cseweb.ucsd.edu/~elkan/250B/nearestn.pdf

[15] Gadbois, P. (n.d.). *Train Signal's CompTIA Security+ course*, YouTube. Retrieved October 18, 2011, from http://www.youtube.com/watch?v=O2Gz-v8WswQ

[16] Gu, Q., Cai, Z. & Zhu, Li. (2009). Classification of Imbalanced Datasets by using

the Hybrid Re-Sampling Algorithm Based on Isomap. Advance in Computation and Intelligence 4th International Symposium, ISICA. China. Retrived March 14, 2012, from http://books.google.jo/books?id=pYqIOup99sEC&pg=PA307&dq=nsl+kdd+anomaly+intrusion+detection&hl=ar&sa=X&ei=HWRgT6b7GJTY8QOE96SdBw&ved=0CEgQ6AEwBA#v=onepage&q=nsl%20kdd%20anomaly%20intrusion%20detection&f=false

[17] Haykin, S. (1998). *Neural Networks: A Comprehensive Foundation*. NY: Macmillan.

[18] Haykin, S. (2001). Feedforward Neural Networks: An Introduction. In I. Sandberg, J. Lo. C, Fancourt, J.Principe & S.Katagiri (Eds), *Nonlinear Dynamical Systems: Feedforward Neural Network Perspectives*. Wiley. Retrieved March 19, 2011, from http://media.wiley.com/product_data/excerpt/19/04713491/0471349119.pdf

[19] Heath, G. (2011, December 28). Validation set and Parameters in Backpropagation ANN, [Newsreader]. December 28, 2011, Answer posted to http://www.mathworks.com/matlabcentral/newsreader/view_thread/315487#865508

[20] Information Security Center of eXcellence (ISCX). (2009). *The NSL-KDD Data Set*. Retrieved October 26, 2011, from http://www.iscx.ca/NSL-KDD/

[21] Ippoliti, D. (2011). *Automated Network Anomaly Detection with Learning and QoS Mitigation*, (Doctoral proposal dissertation), University of Colorado, USA.

[22] Islim, E. (2012). *A Hybrid Intrusion Detection Model Based on Human Immune System*, (Master dissertation), Middle East University, Jordan.

[23] Jawhar, M. & Mehrotra, M. (2010a). Anomaly Intrusion Detection System using Hamming Network Approach. *International Journal of Computer Science & Communication*. *1*(1), 165-169. Retrieved October 25, 2011, from http://www.csjournals.com/IJCSC/PDF1-1/32.pdf

[24] Jawhar, M. & Mehrotra, M. (2010b). Design Network Intrusion Detection System using hybrid Fuzzy-Neural Network. *International Journal of Computer Science and*

Security, *4*(3). Retrieved March 3, 2012, from http://www.cscjournals.org/csc/manuscript/Journals/IJCSS/volume4/Issue3/IJCSS-288.pdf

[25] Jones, S. (1997). Consciousness: The Development of Neural Networks. Retrieved December 17, 2010, from http://www.psych.utoronto.ca/users/reingold/courses/ai/cache/neur_net.htm

[26] Kazienko, P. & Dorosz, P. (2004). *Intrusion Detection Systems (IDS) part 2- Classification; methods; techniques*. Retrieved March 4, 2012, from http://www.windowsecurity.com/articles/ids-part2-classification-methods-techniques.html

[27] Kholfi, S., Habib, M. & Aljahdali, H. (2006). Best hybrid classifiers for intrusion detection. *Journal of Computational Methods in Sciences and Engineering*, *6* (5, 6). Retrieved March 11, 2011, from http://cit.tu.edu.sa/~sultan/opm.pdf

[28] Kotulski, Z. & Kukiełka, P. (2008). Analysis of Different Architectures of Neural Networks for Application in Intrusion Detection Systems. *Proceedings of the International Multiconference on Computer Science and Information Technology*. Retrieved October 25, 2011, from http://www.proceedings2008.imcsit.org/pliks/197.pdf

[29] Kozushko, H. (2003). Intrusion Detection: Host-Based and Network-Based Intrusion Detection Systems, Independent Study. Retrieved October 18, 2011, from http://infohost.nmt.edu/~sfs/Students/HarleyKozushko/Papers/IntrusionDetectionPaper.pdf

[30] Kukiełka, P. & Kotulski, Z. (2010). Adaptation of the neural network-based IDS to new attacks detection. arXiv - Cornell University. Retrieved October 26, 2011, from http://arxiv.org/ftp/arxiv/papers/1009/1009.2406.pdf

[31] Kurose, J. & Ross, K. (2010). *Computer Networking A Top-Down Approach* (5th

ed.). Boston: Pearson Education.

[32] Li, J., Zhang, G. & Gu, U. (2004). The Research and Implementation of Intelligent Intrusion Detection System Based on Artificial Neural Network. *Proceedings of the Third International Conference on Machine Laming and Cybernetics*. Shanghai. Retrieved October 18, 2011, from http://svn.assembla.com/svn/odinIDS/.../01378582.pdf

[33] Lippmann, R. (1987). An Introduction to Computing with Neural Nets. *IEEE ASSP Magazine*.

[34] Lynn, G. (n.d.). *Introduction to Neural Networks Solutions and NeuroSolutions*. Retrieved 2010, from http://www.nddownloads1.com/videos/nns_and_neurosolutions/nns_and_neurosolutions.html

[35] MathWorks Matlab Help. (2011). Speed and memory comparison for training multilayer networks summary.

[36] MathWorks Matlab, *Hyperbolic tangent sigmoid transfer function – Matlab*. Retrieved March 8, 2012, from http://www.mathworks.com/help/toolbox/nnet/ref/tansig.html

[37] Moradi, M. & Zulkernine, M. (n.d.). A Neural Network Based System for Intrusion Detection and Classification of Attacks. University of British Columbia. Retrieved October 18, 2011, from http://research.cs.queensu.ca/~moradi/148-04-MM-MZ.pdf

[38] Mukkamala, S. & Sung, A. (2003). Feature Selection for Intrusion Detection using Neural Networks and Support Vector Machines. *To appear in Journal of the Transportation Research Board (of the National Academies)*. Retrieved October 18, 2011, from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.94.8518

[39] Naoum, R. (2011). *Artificial Neural Network* [Acrobat Reader], Middle East

University (MEU), Jordan.

[40] Nieves, J. (2009). Data clustering for anomaly detection in network intrusion detection. Oak Ridge National Laboratory. Retrieved November 20, 2011, from http://info.ornl.gov/sites/rams09/j_nieves_rodrigues/Documents/report.pdf

[41] PrismNet [Image] (n.d.). Retrieved 2012, from http://www.prismnet.com/~hcexres/power_tools/audience_task/neural2.gif

[42] Rashid, F. (2012). *Malware, Hacking Most Common Attacks in 2011 Data Breaches: Verizon DBIR-Security-News & Reviewers- eweek.com*. Retrieved March 30, 2012, from http://www.eweek.com/c/a/Security/Malware-Hacking-Most-Common-Attacks-in-2011-Data-Breaches-Verizon-DBIR-210278/

[43] Reingold, E. & Nightingale, J. (1999). *Artificial Intelligence tutorial review for psychology students, PSY371*. Retrieved March 3, 2012, from http://www.psych.utoronto.ca/users/reingold/courses/ai/ai.html

[44] Revathi, M. & Ramesh, T. (2011). Network intrusion detection system using reduced dimensionality. *Indian Journal of Computer Science and Engineering (IJCSE)*, *2*(1), 61-67. Retrieved January 28, 2012, from http://www.ijcse.com/docs/IJCSE11-02-01-056.pdf

[45] Riedmiller, M. & Braun, H. (1993). A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm. IEEE INTERNATIONAL CONFERENCE ON NEURAL NETWORKS. Retrieved October 31, 2011, from http://paginas.fe.up.pt/~ee02162/dissertacao/RPROP%20paper.pdf

[46] Salama, M., Eid, H., Ramadan, R., Darwish, A & Hassanien, A. (2010). Hybrid Intelligent Intrusion Detection Scheme. 15th Online World Conference on Soft Computing in Industrial Applications, 15th to 27th November 2010, Springer in "Advances In Intelligent and Soft Computing. Retrieved March 14, 2012, from http://rabieramadan.org/papers/103_Hybrid%20Intelligent%20Intrusion%20Detection%20Scheme.pdf

[47] Sammany, M., Sharawi, M., El-Beltagy, M. & Saroit, I. (2007). Artificial Neural Networks Architecture For Intrusion Detection Systems and Classification of

Attacks. Faculty of Computers and Information Cairo University. Retrieved October 18, 2011, from http://infos2007.fci.cu.edu.eg/Computational%20Intelligence/07177.pdf

[48] Scarfone, K. & Mell, P. (2007). Guide to Intrusion Detection and Prevention Systems (IDPS). Retrieved October 18, 2011, from http://csrc.nist.gov/publications/nistpubs/800-94/SP800-94.pdf

[49] Singh, M. & Verma, K. (2011). Speech Recognition Using Neural Networks. *International Journal of Technology And Engineering System(IJTES)*, *2*(1) Retrieved March 11, 2012, from http://www.ijcns.com/pdf/324.pdf

[50] Stallings, W. & Brown, L. (2008). *Computer Security Principals and Practice*. USA: Pearson Education.

[51] Stergiou, C. & Siganos, D. (1996). NEURAL NETWORKS. Retrieved March 6, 2012, from http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html

[52] Stergiou, C. (1996), *Neural Networks*. Retrieved March 6, 2012, from http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol1/cs11/article1.html

[53] Sun, Y. & Wong, A. (2009). Classification of imbalanced data: A Review. *International Journal of Pattern Recognition and Artificial Intelligence*, *23*(4), 687-719. Retrieved March 19, 2012, from http://www.worldscinet.com/ijprai/23/2304/free-access/S0218001409007326.pdf

[54] TAMU Computer Science and Engineering (Texas A&M). (n.d.). *PRISM Lectures*. Retrieved March 12, 2012, from http://research.cs.tamu.edu/prism/lectures/iss/iss_l12.pdf

[55] Tavallaee, M., Bagheri, E., Lu, W. & Ghorbai, A. (2009). A Detailed Analysis of

the KDD CUP 99 Data Set. *Proceedings of the 2009 IEEE Symposium on Computational Intelligence in Security and Defense Applications (CISDA 2009).* Retrieved October 25, 2011, from http://www.tavallaee.com/publications/CISDA.pdf

[56] The MIT Press (n.d.). *Artificial Neural Networks.* Retrieved March 6, 2012, from http://mitpress.mit.edu/books/chapters/0262062410chap1.pdf

[57] Thirumuruganathan, S. (2010). *A Detailed Introduction to K-Nearest Neighbor (KNN) Algorithm*. Retrieved March 20 , 2012, from http://saravananthirumuruganathan.wordpress.com/2010/05/17/a-detailed-introduction-to-k-nearest-neighbor-knn-algorithm/

[58] University of California, Irvine. (n.d.). *KDD-CUP-99 Task Description*. Retrieved December 24, 2011, from http://kdd.ics.uci.edu/databases/kddcup99/task.html

[59] Willamette University [Image] (n.d.). Retrieved March 8, 2012, from http://www.willamette.edu/~gorr/classes/cs449/Maple/ActivationFuncs/images/active16.gif