INSTANTANEOUS INTRUSION DETECTION SYSTEM

By

ROHIT PILLAY

Bachelor of Engineering in Information Technology
Shri Guru Gobind Singhji Institute of Engineering and Technology
Nanded, MS, India
2007

Submitted to the Faculty of the
Graduate College of
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
December, 2010

INSTANTANEOUS INTRUSION DETECTION SYSTEM

Thesis Approved:

Dr. Johnson Thomas

Thesis Advisor

Dr. Subhash Kak

Dr. John Chandler

Dr. Mark E. Payton

Dean of the Graduate College

ACKNOWLEDGMENTS

I would like to take this opportunity to express my gratitude and appreciation to everyone involved in the successful completion of my thesis.

I would like to thank Dr. Johnson Thomas for his valuable guidance and support, it was indeed an honor to have him as my advisor and research supervisor. He helped me to grow both professionally and personally.

I would also like to thank the members of my committee Dr. Subhash Kak and Dr. John Chandler for their valuable time and inputs which helped improve my thesis. I would also like to thank Sumanth Gangasani, Abhishek Parekh and Amit Pillay for their valuable inputs to my thesis.

I cannot be grateful enough to my family; my father, my mother, my ever advising and loving elder brother and my beloved. I just can't thank them enough. Finally, I would to like to thank a ton to my friends Khushwant, Etesh, Kunal, Anushree, Sid, Vivek, Indu, Krishna, Sravanti and Vrishali for their endless support and friendship.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

CHAPTER 1

INTRODUCTION

The current era is an era of information where consumption, usage and manipulation of information is made easy by the use of computers and computer networks. Computer networks and digital informations have become a precious asset for various organizations, for example, e-commerce websites, government agencies, defense sites and growing industries. These networks are vulnerable to attacks, as today's attackers are well equipped, and with their advancing abilities in this field, there is an increased necessity to protect these networks and websites.

Network intrusions and network attacks have increased recently on major sites and networks; for example, major attacks have been launched on NATO, U.S. DOD, Pentagon and the white house. The safeguard and security of these networks and sites are becoming a very important and critical issue. As the technology and technical abilities of hackers are increasing and becoming more sophisticated, it is necessary to develop an effective intrusion detection system and intrusion prevention system which is more robust and more efficient in identifying and preventing these attacks. In the early 1980's and 1990's attacks like Denial of Service (DOS) and Sniffer were considered infrequent, but today a successful DOS attack can put any e-commerce or other retail sites out of business. Safeguarding these sites and networks is becoming more difficult and thus requires an advancement in the field of intrusion detection.

1.1 Intrusion and Intrusion Detection System

Intrusion is any set of action that attempts to compromise the integrity, confidentiality or availability of the resource [14]. An intrusion detection system is something that would identify these attacks and generate reports that would alarm the network administrator to take suitable action against the intrusion, which could be either strengthening the firewall or avoiding further occurrence of such attacks. If the intrusion detection system is properly deployed, it will play an important role in indicating whether the system is under attack or identifying any breach in the security [1], thus protecting the network against possible threats.

1.1.1 Types of Intrusion Detection System

There are various types of intrusion detection systems (IDS) [28] [1]. Based on the information source, IDS are classified as follows:

- Network based IDS: This uses network packets as the data source. The network adapter are used to monitor and analyze the traffic in real time. It uses various detection techniques to analyze and recognize the attack type. Once the attack is detected appropriate actions are taken by the network administrator like terminating the TCP connection, blocking particular IP packets by reconfiguring firewall and saving evidence to avoid further occurrences of such attacks [1]. Network based IDS includes wireless network monitoring and network analysis [28] [29].

- Host based IDS: The IDS mainly uses security logs and information gathered through a monitoring system, events and system logs. An intrusion is detected by analyzing these logs and regular checks for unexpected deviations from what is set as regular activity. A counteraction is taken, if an attack is detected, by either terminating the user login or disabling the account [1] [28] [29].

Based on the method of analysis IDS [11][12] are classified as follows:

- Anomaly based IDS: In anomaly based intrusion detection system the normal behavior of the user or group of users are monitored and recorded. A proper knowledge base is created for such normal activity. Anomaly based IDS identifies action that strays too far from normal behavior and alarms an intrusion to the administrator [12] [20] .

- Misuse based IDS: In misused based intrusion detection system the activities are recorded. These activities are compared with the known behavior of attackers and if there is a close match or the match is above the threshold then these activities are identified as intrusion [1] [10] [9] [11]. The IDS alarm the system for the attack and it's the responsibility of the network administrator or intrusion protection system to take the appropriate counter action.

1.2 Approaches to IDS

The major approaches to IDS are outlined below:

Data Mining: There are various techniques to create an effective intrusion detection system. In the data mining approach we collect a large quantity of network data and audit data to find normal usage patterns. In case of anomaly detection, any deviation from the normal usage pattern should raise an alarm, and, in the case of misuse, we use audit data for encoding and matching with known intrusion attacks [4].

Machine learning (Artificial neural network) : Neural network is a growing technique in the field of intrusion detection. Neural network based intrusion detection is mainly classified as memory based and memory free. Memory based neural network is when we train the network with absolutely error free data (anomaly detection). Once the training is complete if the input data set does not match the k-nearest neighbor or a pre-trained neuron the data is assumed to be anomalous and an alarm is raised. For the other case (misuse detection) the neural network is trained with all types of intrusion attack data. The neural

network is trained for these different attacks using their known patterns and signature. If any data matches with any of these signatures, the alarm is raised for intrusion [4]. There are more IDS approaches available like specification based, computer immunology, abstraction based intrusion detection; automatically build intrusion detection model, colored petri net model and information theoretic measures [4].

1.2.1 Advantages of using Neural Networks for IDS

The nature of attacks changes constantly. The linear mapping or rule based approach for intrusion detection is not enough to make a powerful network intrusion detection system. We require a flexible system which could analyze the huge network data more efficiently and accurately. Neural network could be a very effective approach to intrusion detection as it analyzes the data in a less structured way. It posses the ability to analyze the information from different network sources in nonlinear fashion which is a key factor in identifying such changing attack patterns [10]. Moreover, neural networks will provide the flexibility that any network would require. Network data or network information collected from different sources might be incomplete, unclear, inconsistent or distorted. The neural network is capable of learning such patterns and is also capable of analyzing data even though the data is unclear or distorted.

1.3 Deficiencies with Current methods and Proposed Hybrid Approach

The current neural network based intrusion detection is considered to be an efficient approach in classifying patterns but they require extensive computation and time consuming training. Training requires large amount of data which hinders most of its applications which stands true for its use in intrusion detection applications as well [21]. For example, an attack would be difficult to detect, if the attacker removes all its information from the system immediately after the attack, and since there will be no information to train the network for such kind of attack. Therefore, we would want to get the information about

the attack before its too late. Another issue with current neural network is assigning initial weights to neural networks which is an unresolved problem. Experiments conducted in 1998 showed different initial weights can give different results [18]. For these reasons we need a better IDS that could provide better support to existing firewalls and show better ability in identifying intrusions with more accuracy and fewer false positive and false negative rates [20]. The current misuse based IDS does not effectively identify a new attack in real time, which is another major issue.

In this thesis, we present a Hybrid network intrusion detection system which is not only faster but also accurate. The intrusion detection system is based on the *CC4* algorithm developed by Kak [2][6] that can detect attacks instantaneously and helps resolving the initial weight assigning issues of neural networks. The instantaneously trained *CC4* neural network [2] operates alongside a two-layer feed-forward neural network given by The Mathworks. The two-layer feed-forward backpropagation neural network identifies pattern with more precision once it is well trained and give a very low false negative and false positive rate and better accuracy. Thus the two IDS helps to create a hybrid IDS that is not only faster, but also provides higher accuracy in detecting known and unknown attacks in real time, and gives lesser false positives and false negatives rates. The hybrid system uses the instantaneous training and detection ability of the *CC4* neural network to identify new signatures, i.e. detecting unknown attacks in real time. Detection of new attacks in real time helps creating a more robust and better intrusion detection system. The hybrid system does not just detect whether incoming network packets belongs to an intrusion or a non intrusion type but it also detects the respective class of attack.

The rest of the document is divided into three sections: The next chapter gives a detail description of the underlying neural networks with a brief description of the intrusion dataset used for validating our approach. Chapter 3 describes the hybrid system and the simulation of its components. Chapter 3 also includes the results and findings. Chapter 4 includes the conclusion and related possible future work.

CHAPTER 2

BACKGROUND

2.1 Related Work

Intrusion detection system is an active area of research. Many approach and techniques have been used to design an effective intrusion detection system The most common technique is the one using data mining and machine learning technique. Our thesis is focused on machine learning based approach. In 1998, a neural network based intrusion detector was proposed by Ryan J., Lin M.J. & Miikkulainen R. [12] which identified intrusions based on the distribution of commands used by the user. This was an anomaly based intrusion detector. This system was divided into three phases; the first phase was to collect data from the audit log from different users for some period and construct a vector from the collected data to represent the command execution distribution from each user. The second phase is to train the network to identify these users based on these distribution vector and if the system identifies any significant deviation from normal behavior it was signaled as intrusion. In 1998, J. Cannady [9][11] proposed a misuse based IDS which was mostly to filter data from incoming packets for suspicious events. The data is send to expert system or to a stand alone neural network application for identifying intrusions. The final result was a two class identifier which detects normal and attack patterns for 89-90% of the cases. Further work has been on misuse based intrusion detection system using feed forward neural network and back propagation training algorithm by M. Moradi and Mohammad Zulkernine in 2004. They proposed an offline trained neural network approach which would detect known attacks with 91% of accuracy, here the novel attacks were not taken into consideration. There are lots of work done on unknown attack patterns recogni-

tion as well [9] [11] [12] [24] mostly based on anomaly based intrusion detection, i.e. by identifying activities which substantially defer from normal, as intrusion. A hybrid system was proposed by Stefanos Koutsoutos, Ioannis T. Christou and Sofoklis Efremidis in 2006 [23] using artificial neural network and rules based matching system to detect known and unknown network intrusions. They used outputs of net pattern classifier and system call classifier to make the decision whether to raise an alarm for intrusion or not. Both the neural network required good amount of training to detect the attacks. The experiment showed good results for known attack and they were able to detect novel attack with an accuracy of 80%. A similar work has been done by John Zhong Lei and Ali Ghorbani in 2004 [25] to simulate neural network based hybrid IDS which used Self Organizing Map (SOM), an unsupervised training algorithm to identify unknown attack by clustering the connection based on their similarities and a improved competitive learning method to identify known attacks.

Most of the work done is based on offline training of neural network and are mostly ineffective in real time detection of novel attacks. The simulation of neural network based IDS requires complex computation and time consuming training which hinders its real time unknown pattern recognition capability. In our thesis we have used *CC4* - instantaneously trained neural network as a replacement to the existing more common neural network to detect unknown attack patterns. The hybrid system we proposed handles the three possible type of input: normal, known attacks with classified individual classes and unknown attacks in more effective way.

2.2 CC4 Neural Network

The *CC4* neural network is an instantaneously trained neural network proposed by Kak in 1992 [6]. The instantaneously trained neural network is an attempt to model biological short term memory or the working memory. The amount of information the short term memory can hold is limited but can be extended by grouping such information [5]. The

corner classification network is based on the visio-spatial sketchpad and phonological loop [6]. The concept of radius of generalization was introduced in CC3. The hamming distance was used for classification between binary vectors, i.e. any test vector whose hamming distance is smaller than the radius of generalization of the network is classified in the same output class as that training vector [6]. A unique neuron is associated with each input vector and all hidden neurons are completely connected to the neurons at input layer. Each node in the network acts as filter for training samples to get the desired output. The algorithm called Corner Classification technique, as a hyper plane (the filter) is used to separate the corner of n dimensional cube by the training vectors [3].

The *CC4* algorithm consist of three layered feed forward network, namely, Input Layer, Hidden Layer, and Output Layer, with binary neurons which uses threshold logic as the activation function. The input layer requires input to be in unary format which makes the algorithm faster but hinders the applicability of the algorithm. Number of element in input vector determines the number of neuron input layer. Each input neuron corresponds to unique input vector, hence the input neurons is the sum of the range of the input vector. In addition to this neuron there is a bias neuron which always takes input as 1. The hidden layer contain hidden neurons which is connected to all the neuron in input layer. Each hidden neuron correspond to a single neuron in the training set. Therefore the size of hidden neurons is equal to the size of the input vector. The output layer contains output neurons which is equal to the minimum bits required to represent the output in the dataset in binary. The output layer is also fully connected to the hidden neuron [8].

The weights on hidden neurons are assigned by using the training sample itself. There can only be two possible values an input neuron can receive. Hence if an input neuron receives a 1, the weight between the hidden and input neuron is set to 1 else it is set to -1. There is an extra input neuron called a bias neuron. The weight between bias neuron and hidden neuron is computed as r-s+1 where 's' is the number of 1's in the input vector of the

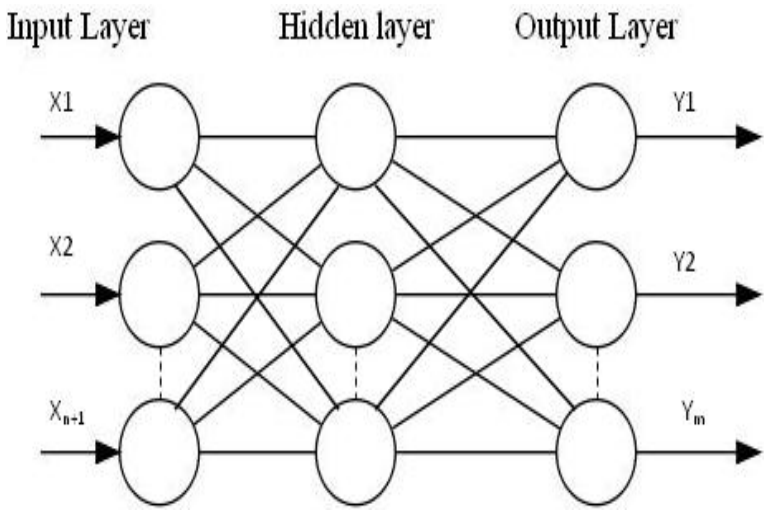


Figure 2.1: General Structure of *CC4* Neural Network [26] [5] [3] [8]

training sample and 'r' is the user specified radius of generalization. Similarly the weights between the hidden neuron and the output neuron are set to 1 or -1, but here we assume there is no extra bias neuron[2] [8]. The activation function for both hidden neuron and output layer for *CC4* network is the binary step function given as [2][8]

$$y = \begin{cases} 1 \text{ if x[n]} > 0 \\ 0 \text{ otherwise} \end{cases} \tag{2.1}$$

The *CC4* algorithm can be summarized as below [2][8]

$$Y = f(f((x_1, x_2, x_3, x_4, x_5, ..., x_N, 1) \times W) \times U) \tag{2.2}$$

The *CC4* algorithm is as follows

for each training vector x_k [n] do

s_k =no. of 1s in x_k[1:n-1]; // x = number of input vector

for j =1 to n-1 do

if x_i[j] = 1 then

w_i[j] = 1;

else // Input weights

w_i[j] = -1;

```
        end
    end
    w_i[n] = r-s_i+1; // r = radius of generalization
                      // s = number of 1s in input vector
    for k=1 to m do
        if y_i[k] =1 then // Y = number of bit in desired output.
                      u_i[k] = 1;
        else
                      u_i[k] = -1; // output weights
        end
    end
end
```

Basic *CC4* Algorithm [5]

The working of *CC4* algorithm can be explained using following example.

XOR Function: The *CC4* neural network can be used to evaluate XOR function. The table 2.1 show the truth table of XOR function. As explained above input layer will have 3 input neuron which include 2 logic input and one neuron whose values is calculated as " r-s+1" as explained above. Since the truth table show 4 input vector therefore the number of hidden neuron will be same as the number of input vector and are connected to one output neuron in the output layer [3].

The three layers of *CC4* neural network are fully connected [3]. Each input vector is presented to network. The input interconnection weights are assigned as the sum of the product of each input element of the input vector with its corresponding interconnection weight elements. In this example input vector (1 0 1), where first two ones are the logic inputs and last 1 is the number of binary ones in the logic input, represented by 's'. Hence

Input		Output
X_1	X_2	Y
0	0	0
0	1	1
1	0	1
1	1	0

Table 2.1: XOR Truth Table

the weight vector of hidden neuron is given as (1 -1 0) and thus using both input vector and weight vector of hidden neuron we can determine the total input to the hidden neuron (1 * 1) + (-1 * 0) + (0 *1) = 1. The other hidden neuron will receive 0 or negative as input. Similarly the input for all hidden neuron based on other input vector and corresponding weight vector are determined. Similarly the weights of output layer and weights of hidden layer can be used to determine the final output. Radius of generalization, r, is set as zero as it is of no use in this example [3].

							Input to Hidden Neuron				Output of Hidden Neuron				Input	output
Input			s	Weights			H_1	H_2	H_3	H_4	H_1	H_2	H_3	H_4	Y	Y
0	0	1	0	-1	-1	1	1	0	0	-1	1	0	0	0	-1	0
0	1	1	1	-1	1	0	0	1	-1	0	0	1	0	0	1	1
1	0	1	1	1	-1	0	0	-1	1	0	0	0	1	0	1	1
1	1	1	2	1	1	-1	-1	0	0	1	0	0	0	1	-1	0

Table 2.2: *CC4* network parameters for XOR Function

The operation of *CC4* network i.e. its mapping of input to output is shown in table 2.2. There are other neural network techniques available but their learning rate and speed of pattern recognition is not real time i.e. other techniques are time consuming and requires substantial training. The *CC4* networks performance is better with respect to both general-

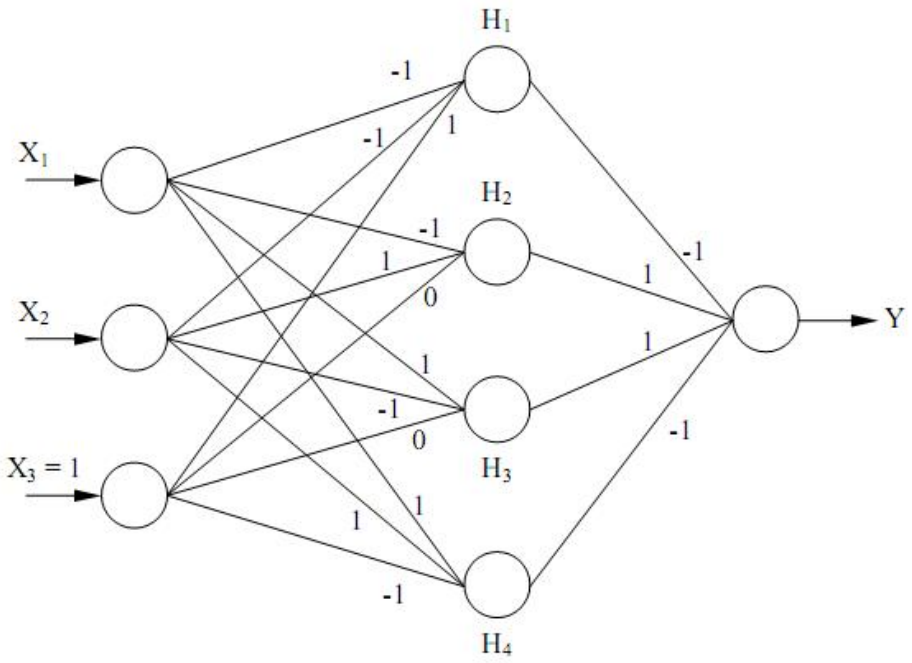


Figure 2.2: General structure of a *CC4* neural network for XOR Function [8] [3] [5]

ization of network and training speed. The speed is extremely fast, thus helping the neural network to identify various possible intrusions real time. The *CC4* neural network is fast but has certain limitations as it takes input in unary format only.

2.3 Two Layered Feed Forward Neural Network

The other neural network which we have used in our neural network based instantaneous intrusion detection system is a two-layer feed-forward network developed by The Mathworks popularly known as Matlab neural network toolbox. In our thesis, we are calling the two-layer feed-forward neural network based IDS as MLP based IDS just to shorten the name. This is a network with sigmoid neurons and linear output neurons. The network will be trained using the Levenberg-Marquardt algorithm [13] and the performance is measured using mean squared error algorithm.

The flow diagram for two-layer feed-forward network neural network is shown in figure

2.4 [27].

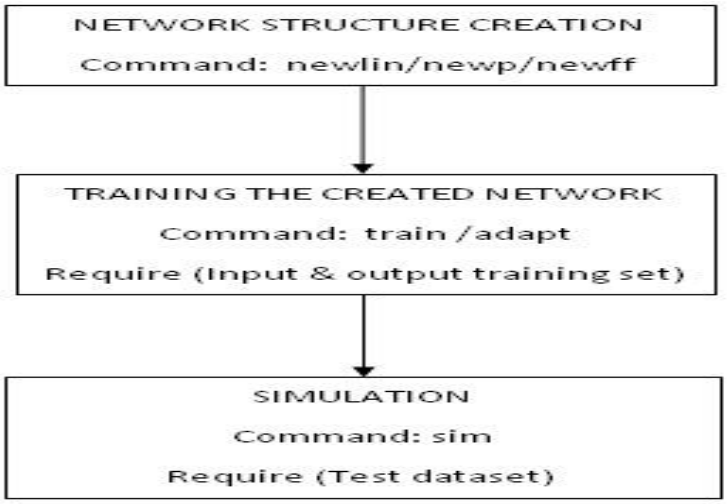


Figure 2.3: Flow Diagram of a two-layer feed-forward network Neural Network [27]

The working of the two-layer feed-forward network Neural network can be demonstrated by simulating a simple XOR function. The truth table of XOR is shown in figure 2.1. In this example we have 2 input neuron at the input level of two-layer feed-forward network neural network. Each neuron takes one binary value as input and has one output neuron at the output layer. We don't have an input layer. Input level is not considered as input layer because no processing takes place at this level, it just acts as a buffer, hence this neural network architecture is termed as two layered feed forward network [17]. For simulation of XOR function we are using newff() to create a new feed forward network and train() function to train the network. Train function is an inbuilt Matlab function which uses Levenberg-Marquardt algorithm for training. The network is simulated using the sim() function.

Input		Output	TestInput		TestOutput
0	0	0	0	0	4.640e-12
0	1	1	0	1	1.0000
1	0	1	1	0	1.0000
1	1	0	1	1	1.1883 e-12

Table 2.3: Two-layer feed-forward network Network: Values at Input and Output Neuron for XOR Function at Epoch=50

2.4 Intrusion Dataset

The data used for this experiment are the benchmark data collected by the Defense Advanced Research Projects Agency (DARPA) in 1998. It was collected by the Lincoln Laboratory of Massachusetts Institute of Technology in 1998. The Knowledge Discovery and Data Mining (KDD) dataset consist of different input cases of various attacks as well as normal network packets. We have used subset this dataset for training and testing the two respective intrusion detection system. There are 22 different intrusions signatures available which include signatures for Back, BufferOverflow, LoadModule, MultiHop, Guesspassword, R2L, U2R, Portsweep, warezClient, warezMaster, FTPWrite, Rootkit, Nmap, Smurf, Satan, Spy, Pod, Imap, Perl, PHF, TearDrop, IPSweep and Normal (legitimate TCP packet). There are 4900000 simulated records and each record has 41 different attributes. These 41 attribute network data features are labeled and shown in table 2.4.

Label	Network Data Feature	Label	Network Data Feature
A	duration	L	logged_in
B	protocol_type	M	num_compromised
C	service	N	root_shell
D	flag	O	su_attempted
E	src_bytes	P	num_root
F	dst_bytes	Q	num_file_creations
G	land	R	num_shells
H	wrong_fragment	S	num_access_files
I	urgent	T	num_outbound_cmds
J	hot	U	is_host_login
K	num_falied_logins	V	is_guest_login
W	count	AH	dst_host_same_srv_rate
X	srv_count	AI	dst_host_diff_srv_rate
Y	serror_rate	AJ	dst_host_same_src_port_rate
Z	srv_serror_rate	AK	dst_host_srv_diff_host_rate
AA	rerror_rate	AL	dst_host_serror_rate
AB	srv_rerror_rate	AM	dst_host_srv_serror_rate
AC	same_srv_rate	AN	dst_host_rerror_rate
AD	diff_srv_rate	AO	dst_host_srv_rerror_rate
AE	srv_diff_host_rate	AF	dst_host_count
AG	dst_host_srv_count		

Table 2.4: Network Data Feature Labels.

CHAPTER 3

HYBRID INSTANTANEOUS INTRUSION DETECTION SYSTEM

3.1 Problem Specification and goals

The major problems with current intrusion detection system is speed of detection and the accuracy of detection while detecting both anomalies and misuse attacks. False positives is when a detected intrusion is in fact not an intrusion, a false negative is when the attack is detected as a non attack. The current misuse based intrusion detection systems are effective against known attacks but are mostly ineffective against novel attacks (previously unknown attacks). Moreover, the current intrusion detection systems are less effective with real time detection of unknown attacks, as most neural network misuse based IDS are not trained for identifying unknown attack patterns. The current misuse based IDS gives an incorrect class type, when it encounters a new pattern. It is difficult to detect whether an abnormal behavior is a new attack or a known behavior with lots of variation. In this thesis we try to overcome the above mentioned issues with current misuse based intrusion detection systems through a more effective hybrid instantaneous intrusion detection system.

3.2 Intrusion Dataset Evaluation

The 41 attributes shown in table 2.4 fields signifies the signature of a particular attack. 22 attributes out of 41 related to the connection and remaining 19 attributes describe the connection to the same host in the last two seconds [21]. We have analyzed the signatures of different intrusion types, in search for fields that would least contribute in uniquely identifying a class of an attack. The purpose of analyzing these fields is to reduce the com-

putational efforts of the training algorithm as each field is involved in matrix multiplication and adds to the computational complexity. The method that we have used for analyzing is Principal Component Analysis (PCA) [19]. PCA is used when we have obtained the number of variables and variable have certain redundancies. PCA is used to develop a smaller number of artificial variable. We had an input vector the 41 different variable as listed in table 2.4 and we used PCA to see if we could reduce the redundant variables which are correlated and contribute least to determining the class of attack.

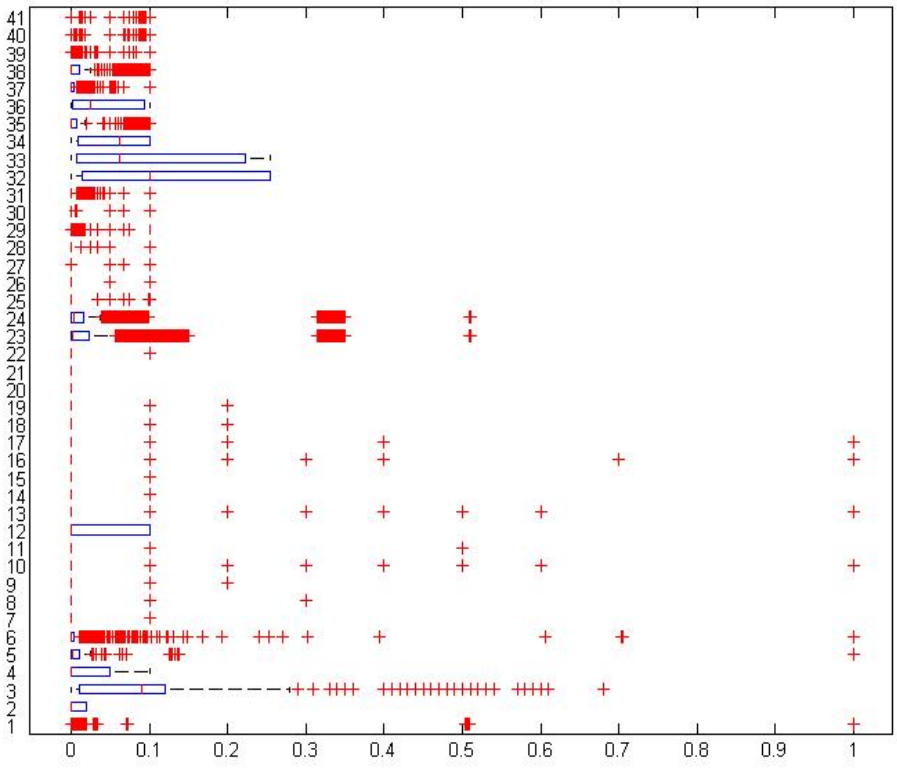


Figure 3.1: Column Value and Column- Distribution Plot for 17 different Intrusion Signature

Figure 3.1 shows the Attribute value vs the columns, this weight distribution plot is obtained by testing sample inputs of 17 different intrusion attacks on the Matlab PCA tool. It can be observed from the figure that column 7-22 has sparse significant value.

The value of these six features (land, urgent, num_failed_logins,num_shells, is_host_login num_outbound_cmds) are mostly zero [21], but when we carefully analyze individual signatures we came to the conclusion that all fields makes significant contribution. For example in case of Śpyáttack's signature, column 7-22 makes a significant contribution, thus we cannot filter out these columns. Moreover, there are various other attacks whose signatures are still unknown and these columns may have a significant effect on their detection. Srinivas Mukkamala & Andrew H. Sung, in 2002 [20] did a performance metric evaluation on the data vector which ranked these features into important, secondary and unimportant features based on the some tested rules [20]. The algorithm evaluates every input node based on following 5 class. These classes contains three categories {Important},< Secondary>, (Unimportant) and columns numbers are distributed accordingly.

- class 1: {1,3,5,6,8-10,14,15,17,20-23,25-29,33,35,36,38,39,41}, <2,4,7,11,12,16,18,19,24,30,31,34,37,40>, (13,32)
- class 2: {3,5,6,23,24,32,33}, <1,4,7-9,12-19,21,22,25-28,34-41>, (2,10,11,20,29,30,31,36,37)
- class 3: {1,3,5,6,8,19,23-28,32,33,35,36,38-41}, <2,7,9-11,14,17,20,22,29,30,34,37>, (4,12,13,15,16,18,19,21,3)
- class 4: {5,6,15,16,18,32,33}, <7,8,11,13,17,19-24,26,30,36-39>, (9,10,12,14,27,29,31,34,35,40,41)
- class 5: {3,5,6,24,32,33}, <2,4,7-23,26-31,34-41>, (1,20,25,38)

Every network incoming packet is compared with these classes and based on the results, the best suited class and its features are used for training, validation and testing. Evaluation of every input vector against these classes increases the computational overhead and hinders its real time applicability. Therefore all these reasons for our simulation we are not filtering any of the attributes out.

3.3 Proposed Approach

The IDS we propose in our thesis is a hybrid intrusion detection system which uses neural network machine learning approach for intrusion detection because of its advantages of pattern matching, pattern recognition and speed of detection. The hybrid intrusion detection system uses anomaly based and misuse based analysis for detecting unknown and known intrusion attacks respectively. The hybrid system which we propose is based on two different neural networks which are different in their learning speed and pattern recognition characteristics. The two neural networks are the *CC4* instantaneously trained neural network proposed by Kak in 1992 [6] and a simple two layered feed forward neural network, available in MLP . The data used for this experiment are the benchmark data used by Defense Advanced Research Projects Agency (DARPA)[16]. Figure 3.2 shows the basic architecture of the proposed hybrid intrusion detection system.

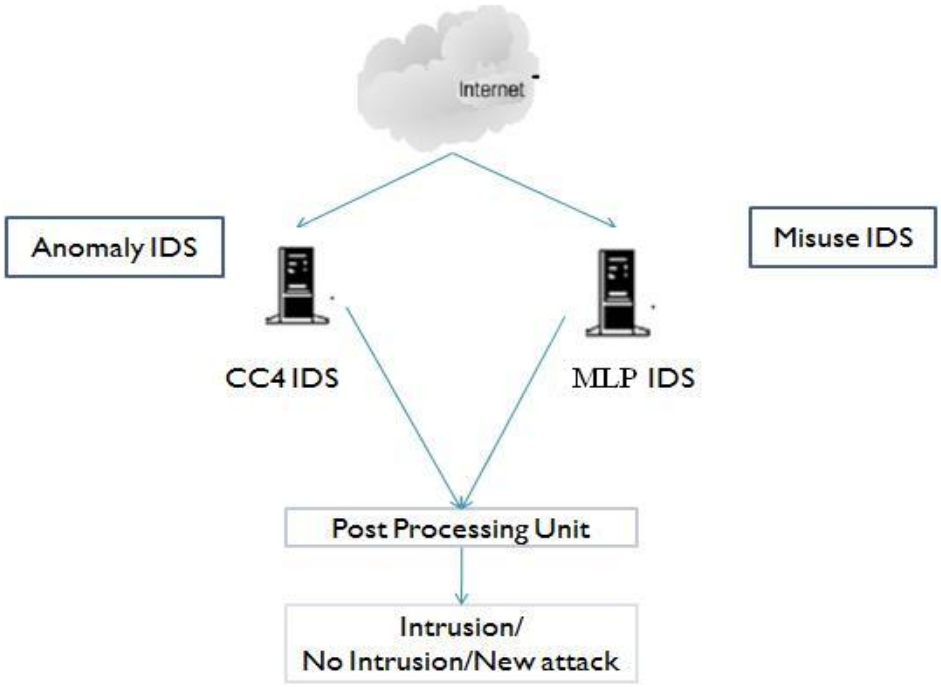


Figure 3.2: Basic Architecture of Hybrid Intrusion Detection System

The Hybrid intrusion detection system consists of three major component *CC4* IDS,

MLP IDSs, and the Post Processing Unit. Each of these components have a specific role. The idea behind using two different IDS is to use the advantages of their underlying neural networks. The advantage of using *CC4* based IDS is its property of instantaneous training and generalization which does not require a lot of training data and the computation is simple. The instantaneous training helps faster learning of new patterns, which help identifying known and unknown attacks in real time. We are using the *CC4* IDS as an anomaly based IDS which considers all attacks whose signatures are pre-recorded in a signature database as known behaviors of attack. If *CC4* based IDS encounters any new behavior which doesn't match to any of the known behaviors of attacks in the signature database, it considers that as an unknown attack and raises an alarm for it. The second component is a two-layer feed-forward neural network (by MLP) based IDS which shows better accuracy than *CC4* IDS while detecting known patterns or known attacks when it is well trained. The false positive and false negative rate is very low for MLP based IDS. We are using the *CC4* IDS for only identifying unknown attacks and not for known intrusion detection. This is because even though its speed of training is faster than the MLP based IDS, its accuracy for detecting known attacks is comparatively low and gives a higher false negative rate. On the other hand we are not using the MLP based IDS for unknown attack recognition because when tested with unknown attack samples MLP based IDS doesn't recognize patterns as a new attack. Instead it identifies new attacks as one of the known attacks or in some cases as normal data, which is not acceptable. Therefore, for these reasons we use *CC4* IDS for unknown attack pattern recognitions and MLP IDS for known attack pattern recognitions.

3.4 Component1: *CC4* IDS and its Simulation

We are using the basic *CC4* training algorithm for training and evaluating the test data sample for intrusion. While implementing *CC4* based IDS, every attribute in the input vector has to be separately converted to unary code[8] which is an issue in our case, as the data which has been collected is a 41 floating point network data (TCPDUMP). Since *CC4*

training algorithm accepts only the unary data, the intrusion data is converted into unary format using quantization mapping. A quantization mapping is a way of lossy compression; a process of representing a large possibly infinite set of values with a much smaller set [15]. The quantization mapping we used is shown in the table 3.1 below. The table shows the mapping of the different floating point values into their respective unary value. When observed the dataset in detail, we found the most of the network data lies more or less in these particular range. This range covers all the data values possibly present in the intrusion dataset. We decided to use these range based on the density range of 41 inputs values. The maximum value in the dataset for attributes is 0.6 and minimum is 0. Based on the 41 attributes values in entire dataset we came with following data range and mapping which actually helps distinguish between attacks signature in a better way. The range is not regular but its based on the concentration on attribute values which lies with the given range. We can have a regular range of data for better accuracy with known attacks detection but this irregular range give us better results while distinguishing between known and unknown signatures of attacks. This supports our sole purpose of identifying new attacks signatures more prominently. The conversion of real floating point network data to 410 unary values input vector and comparison of 410 bit output with all the available signatures adds approx. 0.9 seconds to computation and output generation of every input packet. Adding for the neural network to generate its output the total time is 3.4 seconds. This is for a 4.0 GB of memory and i5 core processor with a speed of 3.33 GHz system with Linux operating system (Fedora).

Each input for training and testing the CC4 network will be (41 attributes $\times$ 10 quantized bits) unary data and the output for training is either 0 or 1 based on the inputs, network weight and learning algorithm. These converted unary data will now be in the same format as that of the signatures of different intrusion attacks. The *CC4* network size increases as the different class of training sample increases and so the performance of a single network

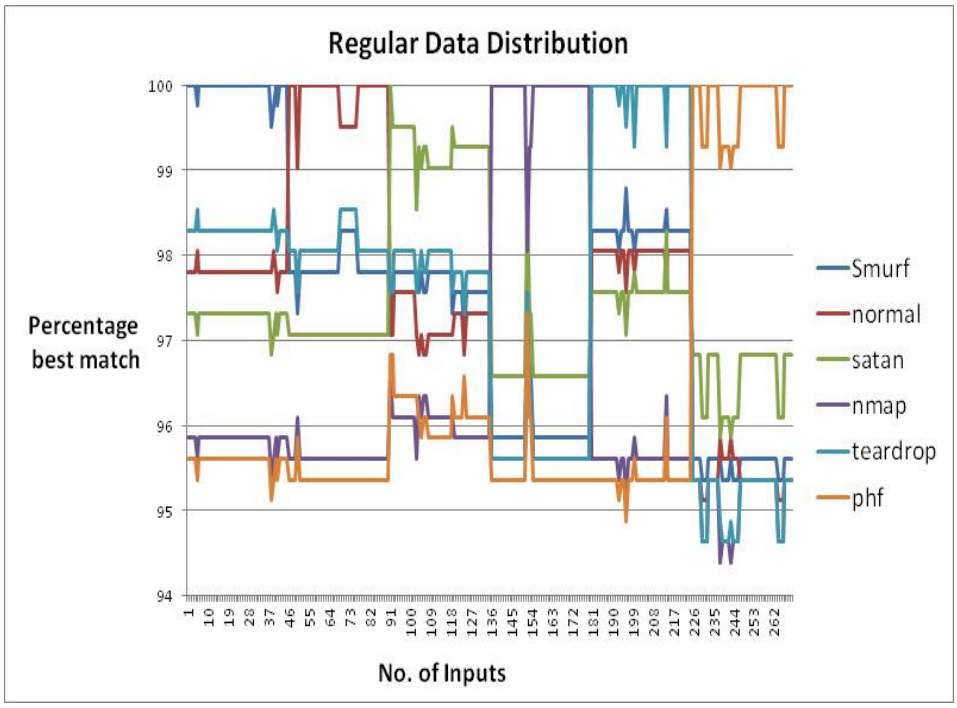


Figure 3.3: Regular Range Mapping

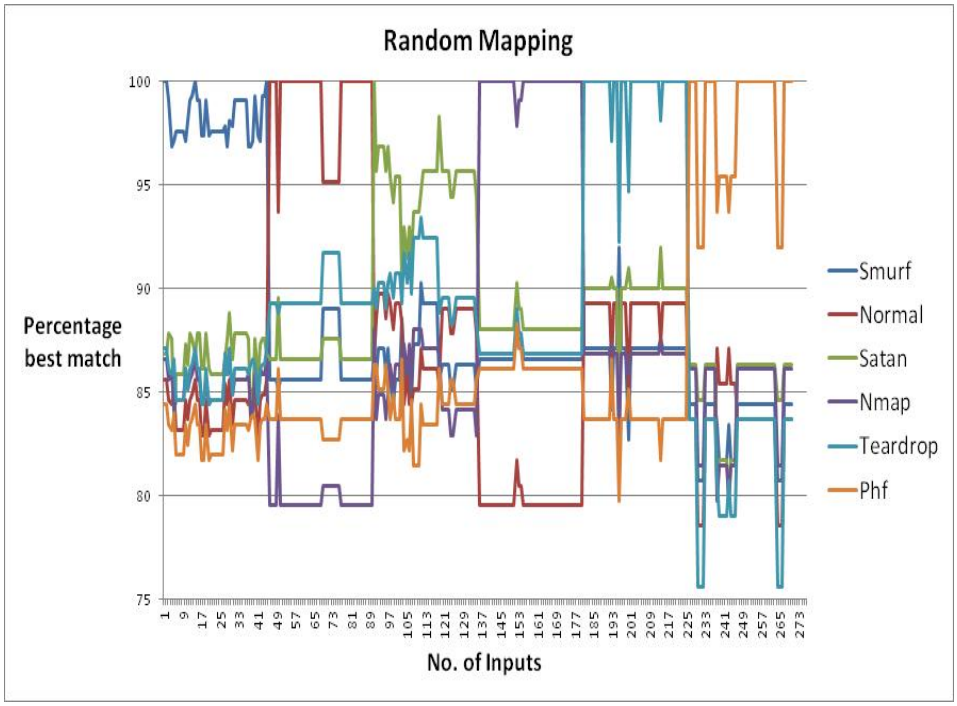


Figure 3.4: Irregular Range Mapping

Range	Mapping
0.0	0000000000
0.00000001 - 0.0000001	0000000001
0.0000001 - 0.000001	0000000011
0.000001 - 0.00001	0000000111
0.00001 - 0.0001	0000001111
0.0001 - 0.001	0000011111
0.001 - 0.01	0000111111
0.01 - 0.1	0001111111
0.1 - 0.3	0011111111
0.3 - 0.5	0111111111
0.5 - 0.7	1111111111

Table 3.1: Quantization mapping of floating point data to unary data

handling large network data mitigates. We are using a divide and conquer strategy in a parallel *CC4* network i.e. a chain of smaller *CC4* network executing together to reduce the computation complexity of a single network, which would allow the network to have better generalization thus resulting in output with lesser error. We are using a parallel chain of *CC4* neural network which has 410 neural network running in parallel, each generating one bit (0/1) as output per network as shown in figure 3.5.

The result of all these 410 neural networks together will be a match to one of the known intrusion signature or a legitimate data signature (Normal) and that will be the class of the input packet. The 410 output one from each of 410 neural network will be compared with each of the 410 bit signature of a known attack that is pre-saved, for example as shown in table 3.2, we gave normal network data as input to the *CC4* chain and its output is compared with all known signatures available. Later we calculate the best percentage of match with all respective signatures of known types which includes signatures of normal

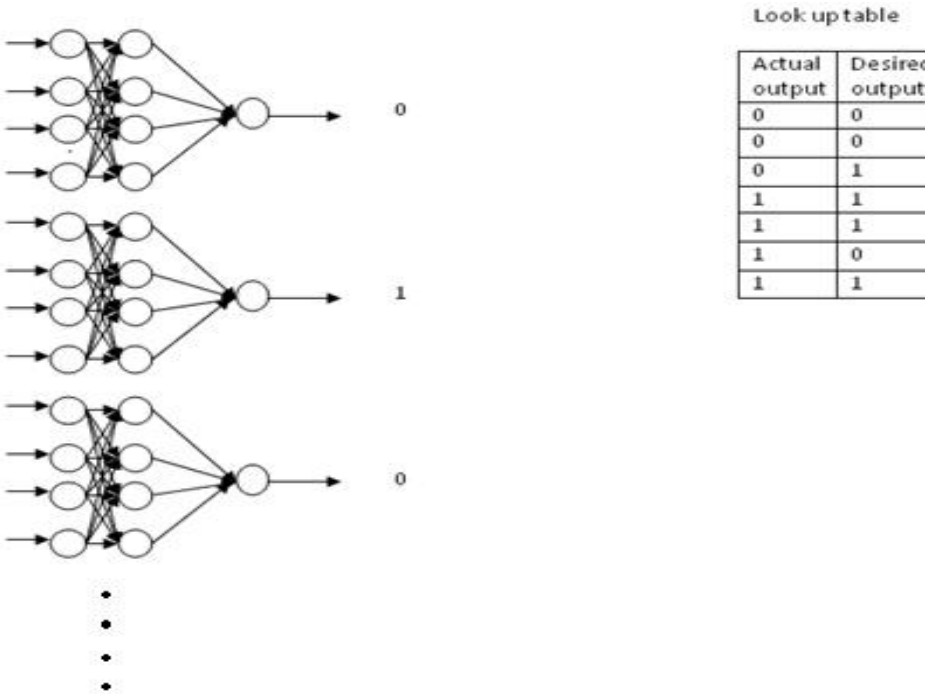


Figure 3.5: 410 Parallel CC4 Neural Network

data as well. The best match between them will decide whether the input belongs to normal or a particular class of known intrusion. For example, table 3.3 shows 10 different network data given as input to *CC4* IDS and the output from *CC4* IDS is compared against signature of 7 different known signatures (Smurf, Normal, Satan, Nmap, Neptune, Spy, Waraz) and based of their best match the class of network packet is decided. In our thesis, the graph obtained by analyzing the outputs from *CC4* IDS is the percentage best match graph.

Figure 3.6 shows the percentage best match graph generated when *CC4* based IDS is tested with 260 different network input data which includes randomly distributed 17 different intrusion types. The *CC4* IDS with radius of generalization r=0 is tested for Land, Bufferoverflow, Pod, Teardrop, Warazmaster, Neptune, Loadmodule, Guesspaswwrod, Smurf, Rootkit, Nmap, Ftpwrite, Spy, Phf, Perl, Multihop and Warazclient gave a 81.583% accuracy. The training and detection time for 410 parallel running *CC4* to detect any class of attack (normal /unknown/ known) is approx. 3.4 sec per signature when simulated on a system with Linux operating system (Fedora) with 4.0 GB of memory and i5 core proces-

S No.	Network Output	Smurf Signature	Normal Signature	Satan Signature	Neptune Signature
1	0	0	0	0	0
2	1	0	1	0	0
3	0	1	0	1	1
4	1	1	1	0	1
5	0	0	0	0	0
6	1	0	1	1	0
..	..	..	..	..	..
410	1	1	1	0	0

Table 3.2: Comparising the output from *CC4* IDS with all available signatures

Smurf	Normal	Satan	Nmap	Neptune	Waraz	Spy		Best Match		Attack Type
100	89.97	84.40	93.09	79.28	86.41	85.30		100		Smurf
89.75	99.77	82.18	90.86	85.53	91.75	88.41		99.77		Normal
79.28	85.30	80.62	80.84	100	84.40	81.51		100		Neptune
85.30	88.19	79.06	91.31	81.51	90.86	100		100		Spy
86.41	91.53	79.28	92.42	84.40	100	90.86		100		Waraz
88.41	93.54	81.29	90.42	86.41	97.99	88.86		97.99		Waraz
93.09	90.64	78.84	100	80.84	92.42	91.31		100		Nmap
100	89.97	84.40	93.09	79.28	86.41	85.30		100		Smurf
100	89.97	84.40	93.09	79.28	86.41	85.30		100		Smurf
100	89.97	84.40	93.09	79.28	86.41	85.30		100		Smurf

Table 3.3: Best Match Comparison to Determine Normal/Intrusion type Input

sor with a speed of 3.33 GHz. The *CC4* IDS used an one pass algorithm for training i.e. the *CC4* algorithm, requires one iteration to detect any class of pattern. This instantaneous training property of *CC4* IDS helps in real time identification of attacks.

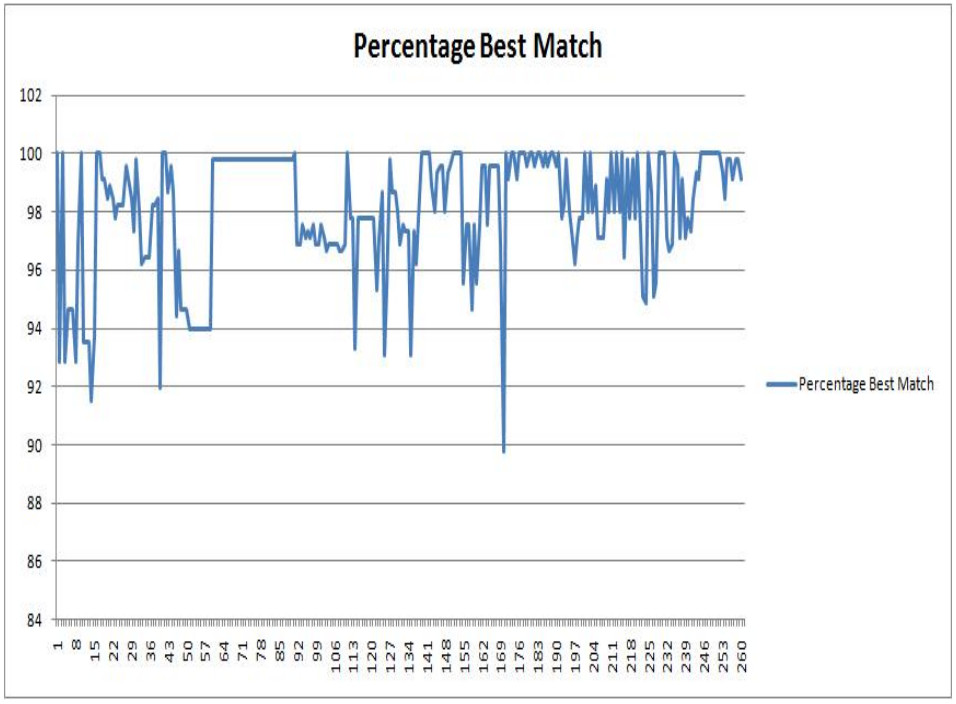


Figure 3.6: Testing *CC4* for 17 different known intrusion type data with r=0

3.4.1 Use of *CC4* for Detecting New Attack Signatures

As explained above *CC4* generates outputs matching close to one of the known signature only if the input belongs to one of the known attack classes for which we have a proper signature. We have set a threshold based on the observation of various simulations that we have run with various known and unknown type attack as input. Based on the this threshold we decide whether network has encountered a known or an unknown attack. If the percentage of best match amongst all known signature falls below this threshold there is possibility of input belonging to a known type class with lot of variation in it or its a new attack altogether. We have set 91% and above as threshold and any best match below it will alarm a signal for the network administrator. We have set 91% as our threshold because

after multiple simulation runs we found the minimum percentage of the best matches any input of a known class to a known signature is above 92 % on average. In a simulation the *CC4* neural network was intentionally fed with ”Back” intrusion type attack without any prior training for it i.e. we are assuming we have a new kind of attack coming through the network and we don’t have any signature available to identify the exact intrusion type. So in this case the best match goes below the set threshold and we can indicate the administrator for abnormal activity of unknown attack type.

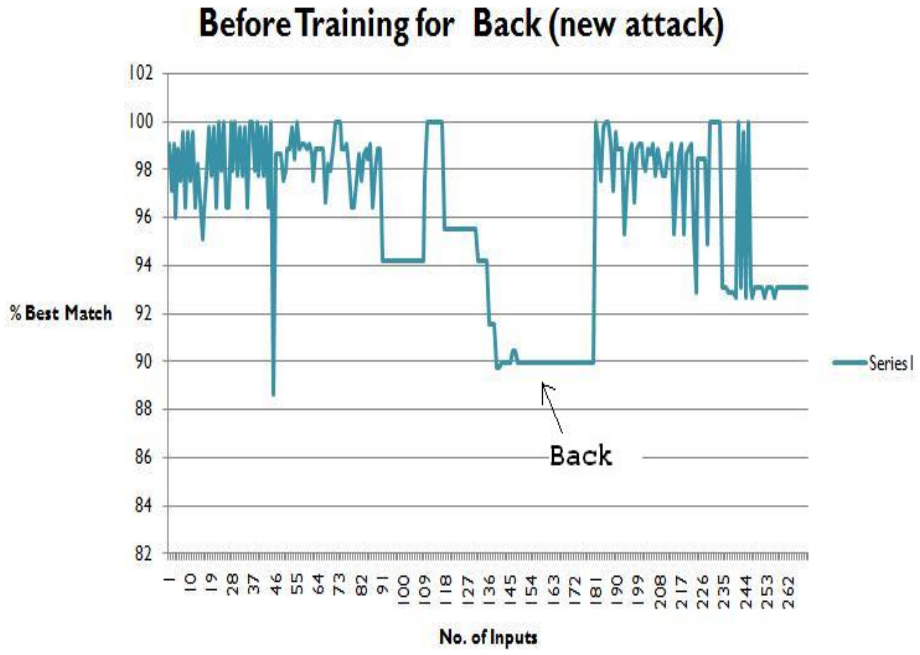


Figure 3.7: Best Match Graph of *CC4* Based IDS Before Training for Back

Figure 3.7 shows the best match graph that falls below the threshold when the IDS encountered a new attack ”Back” thus indicating abnormal malicious activity. The abnormal input packets are extracted and tested and, if it is a unique signature not seen before, the *CC4* based IDS will update itself with the new signature. The signature database for MLP IDS which contains the training dataset for the IDS, will also be updated. Figure 3.8 shows the best match graph of *CC4* IDS with proper identification when trained with the new signatures and tested against the same dataset which we used before the training. The IDS

now identifies the class of new attack with graph showing all the matches to be above the threshold percentage.

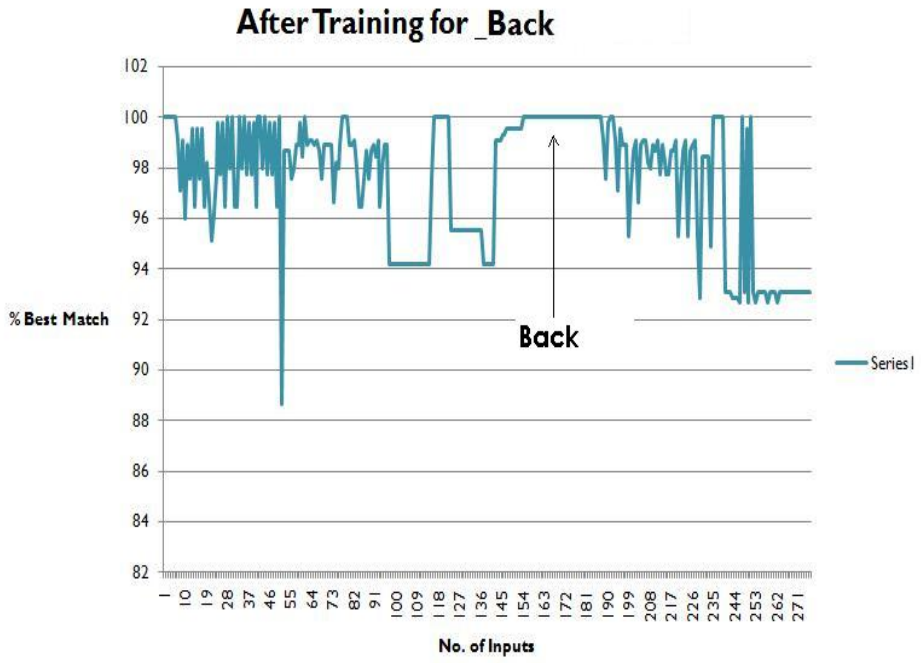


Figure 3.8: Best Match Graph of *CC4* Based IDS After Training for Back

3.5 Component 2: MLP IDS and its Simulation

In addition to the *CC4* network we created an intrusion detection system using the two-layer feed-forward neural network tool that uses the Levenberg-Marquardt algorithm to train the network. The reason behind developing another IDS is due to fact that even though the speed of training the network using *CC4* algorithm is faster than the two-layer feed-forward neural network, the percentage of accuracy in case of detection is considerably lower and the false negative alarms are comparatively higher. The MLP neural network tool uses a two layered feed forward network and utilizes the Levenberg-Marquardt algorithm to train the network. The tool uses the mean square error method to calculate the performance.

The two-layer feed-forward neural network contains 41 neuron at the input level and

15 hidden neuron at the hidden layer. The neurons at the output layer vary, depending on the number of known attacks. We have varied the number of neurons at the output layer depending on our simulation run. Every new signature detected adds 1 to the number of output neurons. Table show 3.4 shows the output layer with 5 neurons used to train the two-layer feed-forward neural network for 5 unique signatures.

Record Type	Output Vector
Normal	0 0 0 0 1
Satan	0 0 0 1 0
Smurf	0 0 1 0 0
Neptune	0 1 0 0 0
Nmap	1 0 0 0 0

Table 3.4: Output Vector Used for Training and Evaluating MLP IDS

The initial training of the two-layer feed-forward neural network is slow as it takes lots of iterations to adjust the internal weights. Figure 3.9 shows the accuracy of two-layer feed-forward after '1' epoch or iteration and figure 3.10 shows the accuracy after '50' epoch or iterations. After a substantial amount of training, the accuracy is good and shows less false positives and false negative rates.

Table 3.5 shows the distribution of data vectors in different subsets for training validation and testing set.

MLP IDS give inaccurate results, if the MLP IDS is not sufficiently trained with respect to the number in the training sample. The MLP IDS gives better performance if it has better a understanding of the patterns it is looking for. In a simulation run, the training dataset contained 6010 packets, which included 1000 packets each for 6 other attack class and just 10 packets of teardrop i.e. MLP IDS was trained with 10 training packets of Teardrop attack. MLP IDS was later tested for its performance to detect the TearDrop attack. While testing, Due to improper training, instead of detecting the samples as a 'Teardrop' Attack,

Record Type	Training Set	Validation Set	Test Set
Normal	1000	1000	50
Satan	1000	1000	50
Smurf	1000	1000	50
Neptune	1000	1000	50
Nmap	1000	1000	50

Table 3.5: Distribution of Data Vectors in Different Subsets used for Training, Validation and Testing Set

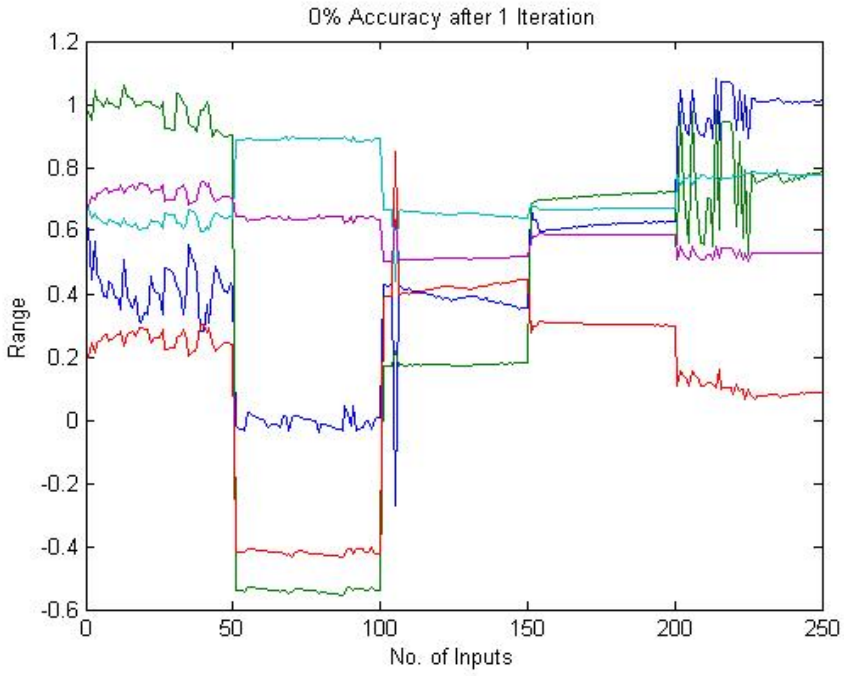


Figure 3.9: Accuracy of MLP IDS after 1 iteration(0% Accuracy)

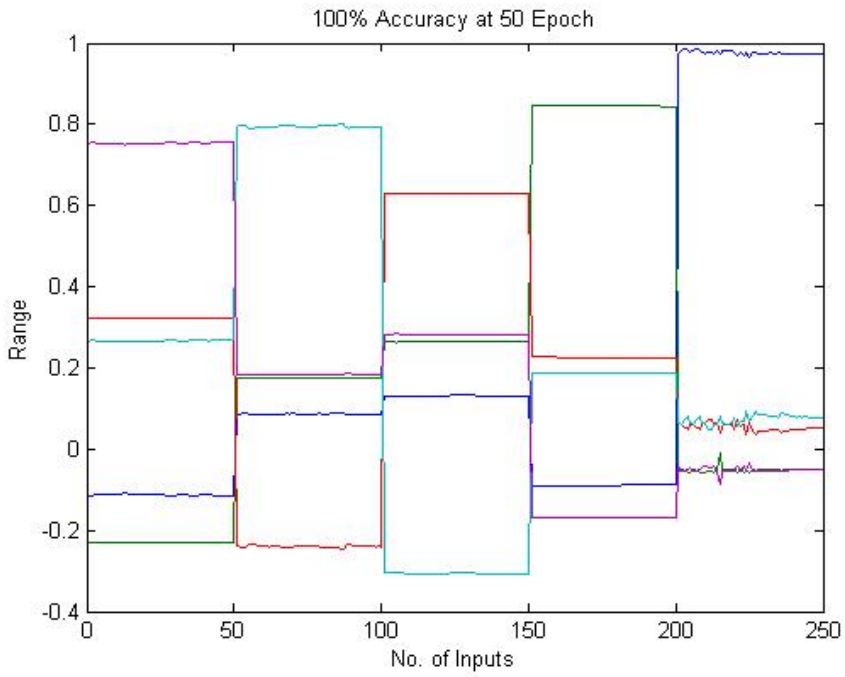
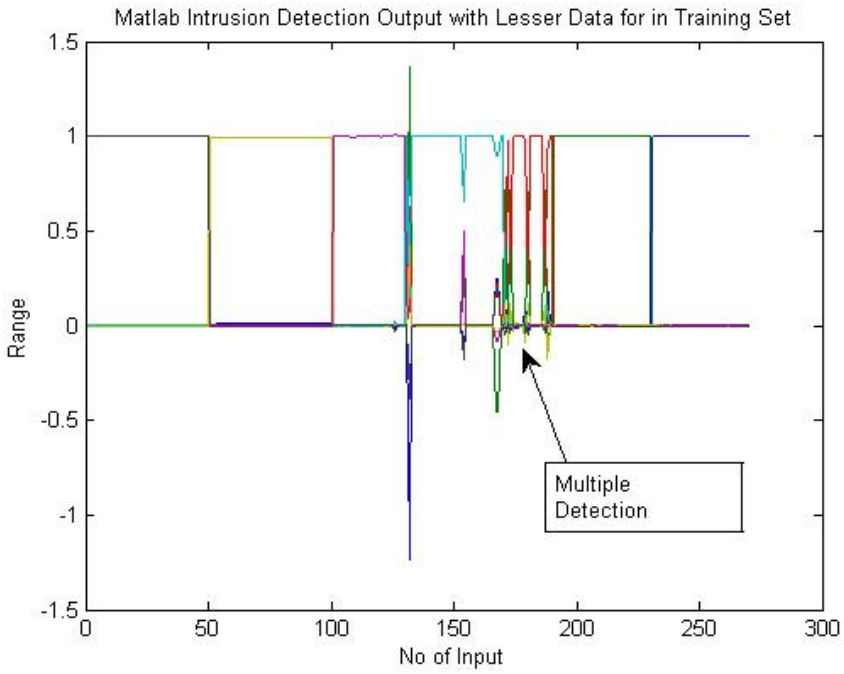


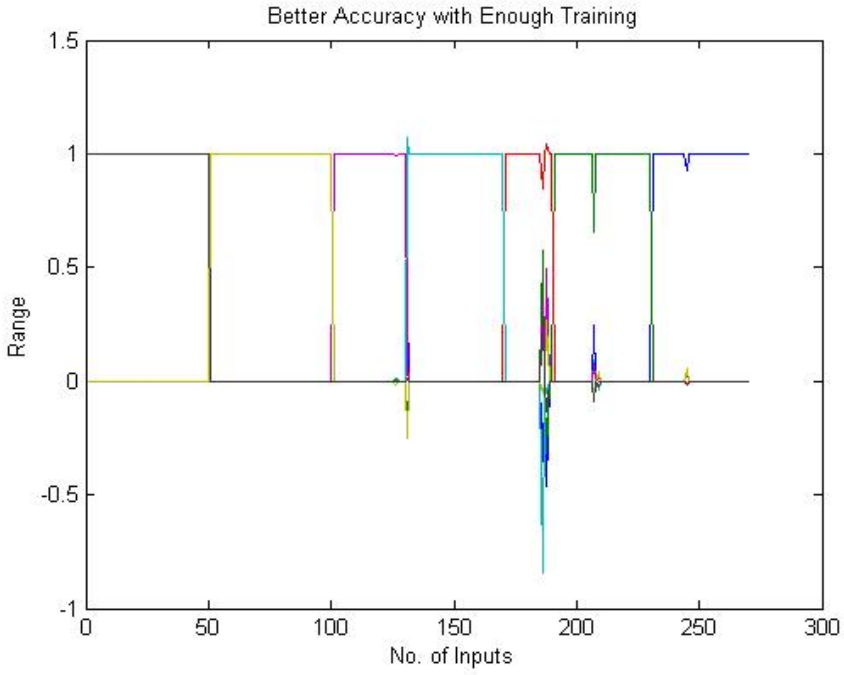
Figure 3.10: Accuracy of MLP IDS after 50 iteration(100% Accuracy)

multiple attacks were detected in the sample inputs. Figure 3.11 (a) shows the output of MLP IDS with multiple attacks detection. Figure 3.11 (b) shows a better performance for Teardrop attack detection when MLP IDS was fed 1000 samples of the Teardrop attack while training and tested against the same test dataset which was used before the training. The accuracy was approx. 100% for this dataset.

MLP based IDS uses the network data in their original form as it does not require any mapping unlike the case of *CC4* IDS. MLP IDS requires a good amount of training for better performance and it learns gradually with every iteration which requires a large number of training data and more iterations. Once the training is complete, the accuracy is very high, with a 92% of accuracy. The false positive and false negative rate is fairly low if the IDS have well understood patterns of the incoming network packets. The MLP IDS learns better with more number of samples. The first time training of the MLP IDS requires good number of iterations. On average MLP IDS took 15-50 epochs to validate the training dataset of 7000 training samples. The training and detection time of MLP based



(a) Before Proper Training



(b) After Proper Training

Figure 3.11: MLP IDS Performance Before and After Proper Training

IDS to detect any class of attack (normal /unknown/ known) is approx. 1.5 - 2.5 minutes when simulated on a system with windows vista operating system, 4.0 GB of memory and i5 core processor with a speed of 3.33 GHz. This training time depends on the number of epochs set for training and the size of training dataset.

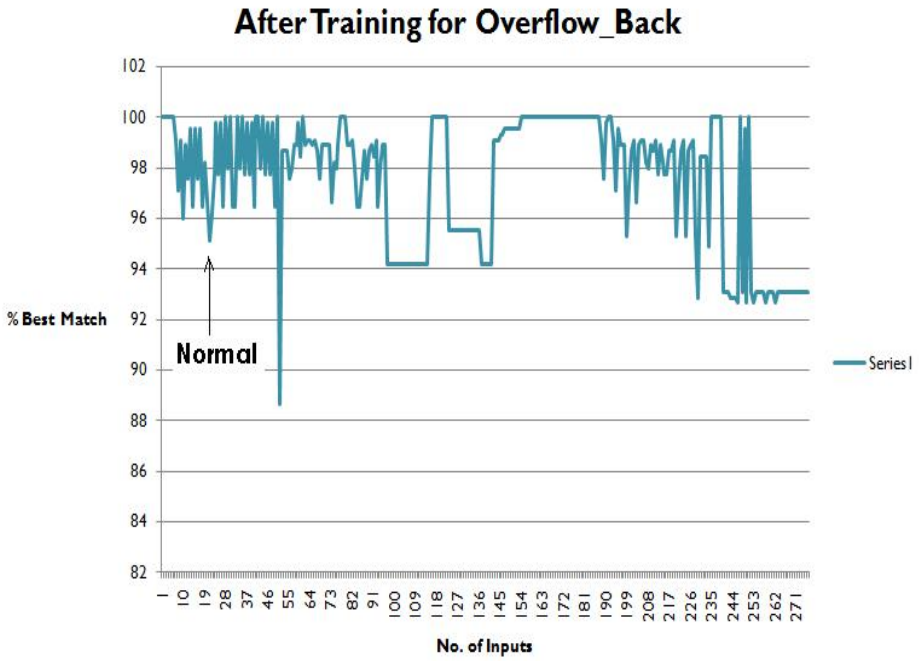
The Post Processing Unit will use the output of MLP based IDS to determine the class of network packet based on known signature for which MLP IDS is trained until and unless there is no new attack encountered. If a new attack is encountered the *CC4* output will be considered because of its instantaneous learning even with less amount of data and quick pattern recognition which helps identifying new attacks in real time.

3.6 Component 3: Post Processing Unit

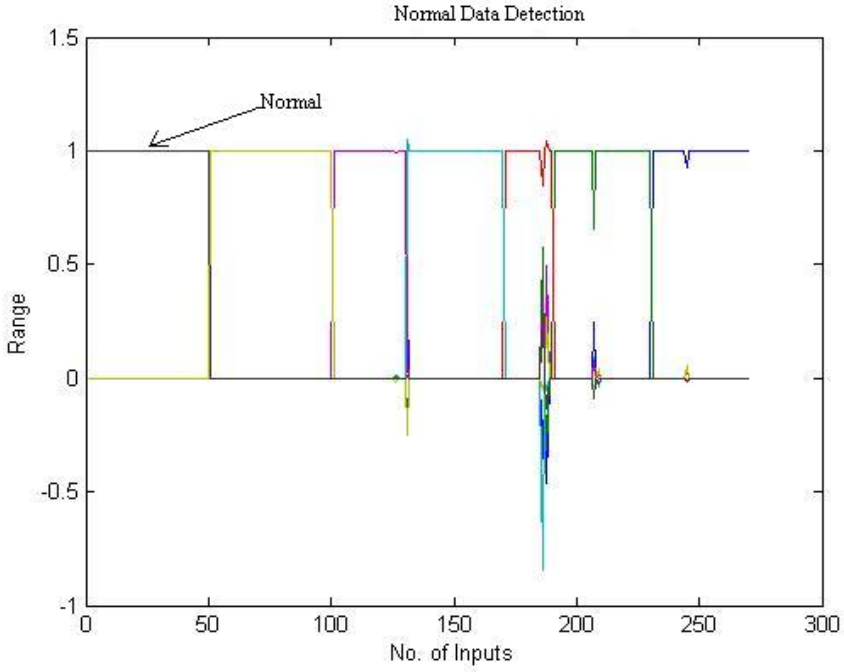
The Post Processing Unit is responsible to determine whether the incoming packet from the network belongs to normal legitimate class type, a known attack type or a unknown attack type i.e. a new attack. It use the basic architecture of the hybrid IDS as shown in figure 3.1 for determining whether packet belongs to normal or known attack class. This architecture is extended to a hybrid IDS as shown in figure 3.16 to determine the unknown attack class. The Post Processing Unit will utilize the output from both *CC4* IDS and MLP IDSs to determine the class of incoming packets. he There will be three cases depending on the class of incoming network packet.

1. Normal Packets: If all the values of the ”percentage best match graph” of *CC4* IDS is above the threshold (91%) and output from MLP IDS shows that the packet belongs to normal class, the Post Processing Unit declares it as normal packet. Figure 3.12 (a) shows the percentage best match graph of *CC4* for incoming packet of normal class and figure 3.12 (b) shows the output of MLP IDS for packets of normal class, when both IDSs were tested with the same dataset.

2. Known Attack: If all the values of the percentage best match graph of *CC4* IDS is above the threshold and output graph for MLP IDS shows the packet belongs to one of the



(a) Best Match Graph Output of *CC4* IDS for Normal Packets



(b) Output of MLP IDS Showing 100 % Accuracy for Normal Packets

Figure 3.12: Post Processing Unit Outputs ”Normal” for Normal Incoming Packets

Record Type	Test Set
Normal	50
Smurf	50
Satan	30
Back	50
Teardrop	20
Netpune	30
Nmap	40

Table 3.6: Distribution of Data Vector for *CC4* IDS and MLP IDS Test Set

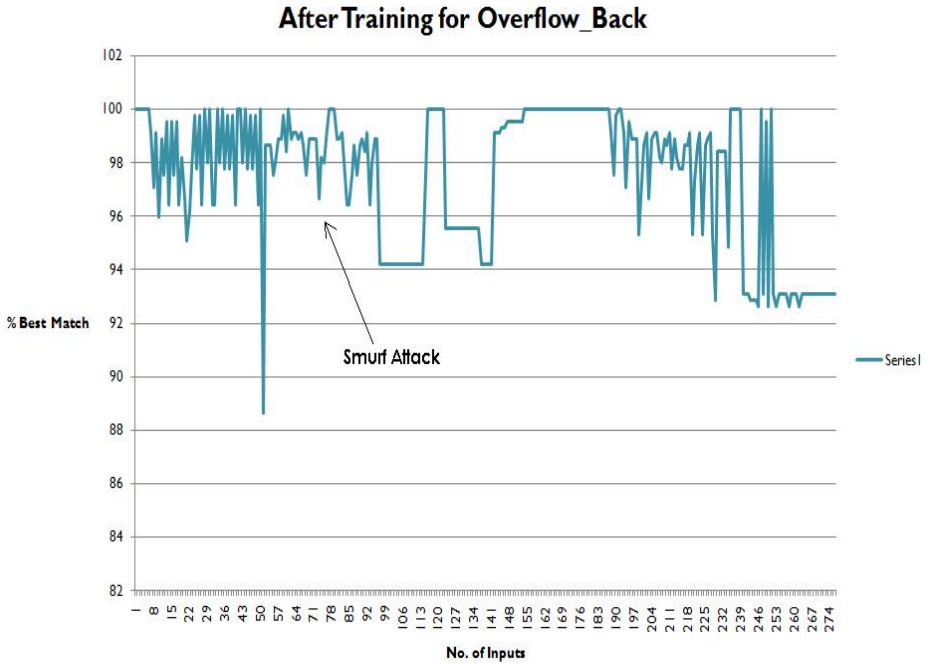
Record Type	Training Set	Validation set	Test Set
Normal	1000	1000	50
Smurf	1000	1000	50
Satan	1000	1000	30
Back	1000	1000	50
Teardrop	1000	1000	20
Neptune	1000	1000	30
Nmap	1000	1000	40

Table 3.7: Distribution of Data Vector in Different Subset for Training, Validation and Testing Set for MLP IDS

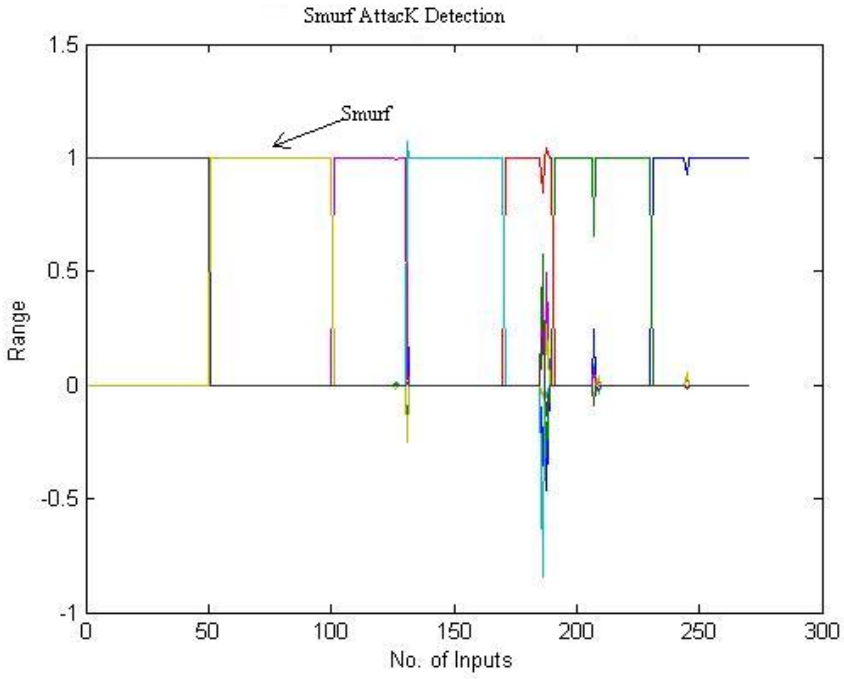
known intrusion attack classes, the Post Processing Unit declares that this packet belong to the respective class of attack. Figure 3.13 (a) show the best match graph of *CC4* for incoming packet of Smurf attack class and figure 3.13 (b) show the output of MLP IDS for packets of Smurf attack, when tested with same test dataset.

3. Unknown Attack: Unknown attacks are identified when output graph of MLP IDS and best match output graph of *CC4* IDS contradicts each other. The hybrid system monitors the *CC4* graph carefully for any drop in the best match percentage graph below the threshold. The *CC4* IDS doesn't give a 100% accurate detection for all attacks if there is a huge variation amongst two input packets belonging to a same class graph may fall below the threshold for that particular input case as shown in figure 3.14. After various test run for *CC4* IDS it is observed that the accuracy of correct detection of attack class for *CC4* IDS lies between 80-85% if Radius of generalization is set to zero i.e. r=0 *CC4* IDS gives best performance. Radius of Generalization is the term given to *classification of input vector within the Hamming distance from the stored vector as belonging to the same class of stored vector* [3]. Figure 3.15 shows *CC4* IDS's accuracy vs number of intrusions in test sample at different radius of generalization values. We have kept a waiting time for hybrid system. The hybrid system waits for 15 inputs packets to make sure whether the fall in the percentage best match graph is just a huge variation or a new attack altogether. If even after the wait time the packets show similar behavior i.e. percentage best match graph value for these packets is still below the threshold percentage (91%), the Post Processing Unit saves these input packets into the signature database. The signature database is then used as the training set to train the MLP IDS. We chose to scan 15 input before deciding whether to save the input packet or not because any attack normally take more than 15 packets to start an attack and moreover, MLP based IDS gives very low accuracy when trained with less than 15 data samples for a given attack as explained above in figure 3.11.

The Hybrid system has a backup MLP IDS, MLP IDS-2 as shown in figure 3.16, which is trained offline whenever the system encounters a new signature or a new attack type for



(a) Best Match Graph Output of *CC4* IDS for Smurf Attack Packets



(b) Output of MLP IDS Showing 100 % Accuracy for Smurf Attack Packets

Figure 3.13: Post Processing Unit Outputs ”Smurf Attack” for Smurf Incoming Packets

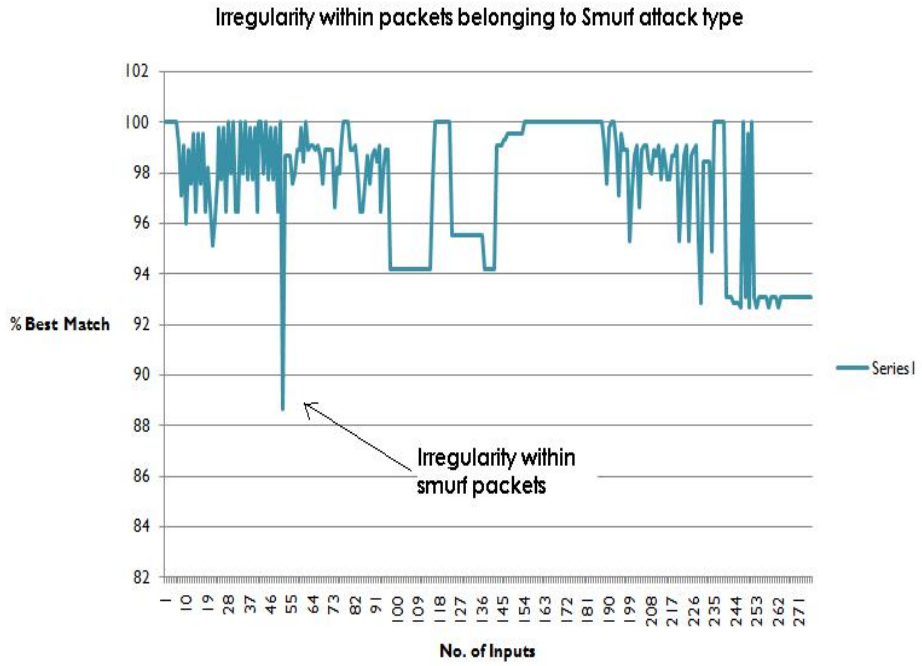


Figure 3.14: Irregularity within packets belonging to Smurf Attack type

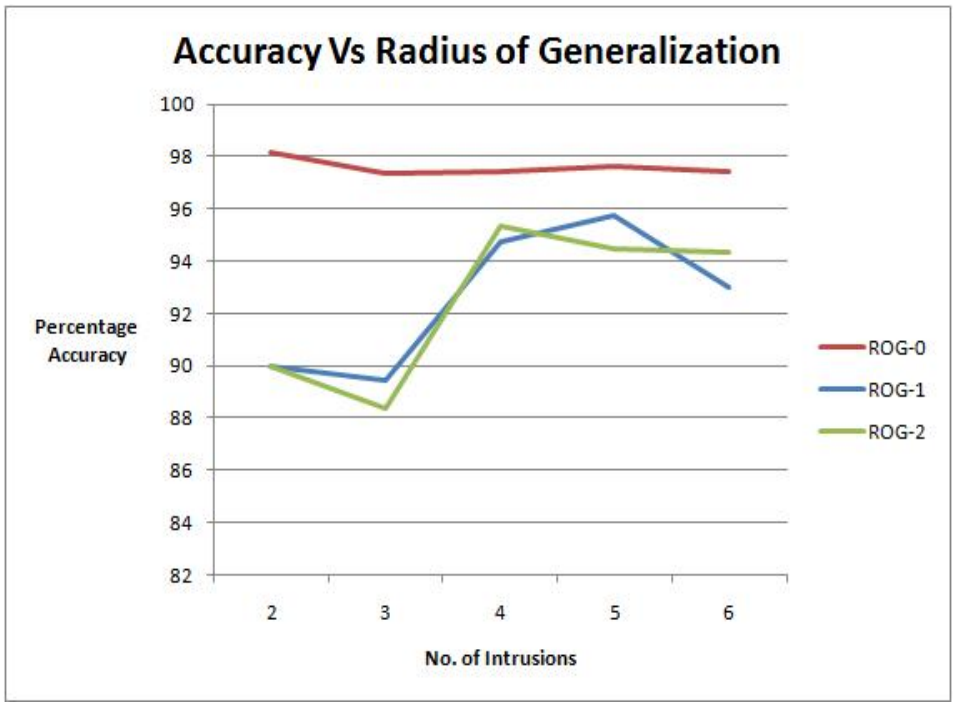


Figure 3.15: No. of intrusion Vs Accuracy of *CC4* IDS at different ROG

which the live MLP IDS, i.e. the MLP IDS-1, is not trained for. As mentioned above there are two cases for which the graph falls below the threshold value. Firstly, when hybrid system encounters new attack and secondly, when there is a huge variation in packets belonging to known attack class. The Hybrid system behaves differently for both cases.

Case 1: If the best match percentage graph is still below the threshold value even after the wait time, the system assumes it's a new attack and uses the saved signatures from the signature database to train MLP IDS-2 immediately with the new training dataset which now includes the signature of new attacks. As soon as MLP IDS-2 updates itself with the new signatures and start correctly identifying the new attack, MLP IDS-1 will be swapped with MLP IDS-2 instantaneously without any delay. The network therfore gets protected immediately, which is our primary concern. Now MLP IDS-2 will go live with its updated signature and MLP IDS-1 will wait offline until the system encounters an unknown attack via *CC4* IDS. This process is repeated for every new attack found.

Case 2: If the best match percentage graph comes back above threshold value before the wait time, the hybrid system assumes that the input was just a variation not a new attack and the system takes no action.

3.7 Comparison of MLP IDS and *CC4* IDS

3.7.1 Accuracy of MLP IDS Vs *CC4* IDS based on number of Iterations

The behavior of percentage best match graph of *CC4* IDS remains unchanged unless its radius of generalization is changed, on the other, hand MLP IDS's accuracy increases with number of iterations. Figure 3.17 show the accuracy of MLP IDS Vs *CC4* IDS for increasing number of iterations. The MLP IDS and *CC4* IDS where tested with a fixed number of test inputs with included 4 different attacks types (Smurf , Neptune, Nmap and Satan) and normal data. The accuracy of *CC4* IDS remained the same irrespective of the number of iterations because *CC4* has a one pass training algorithm and is based on prescriptive learn-

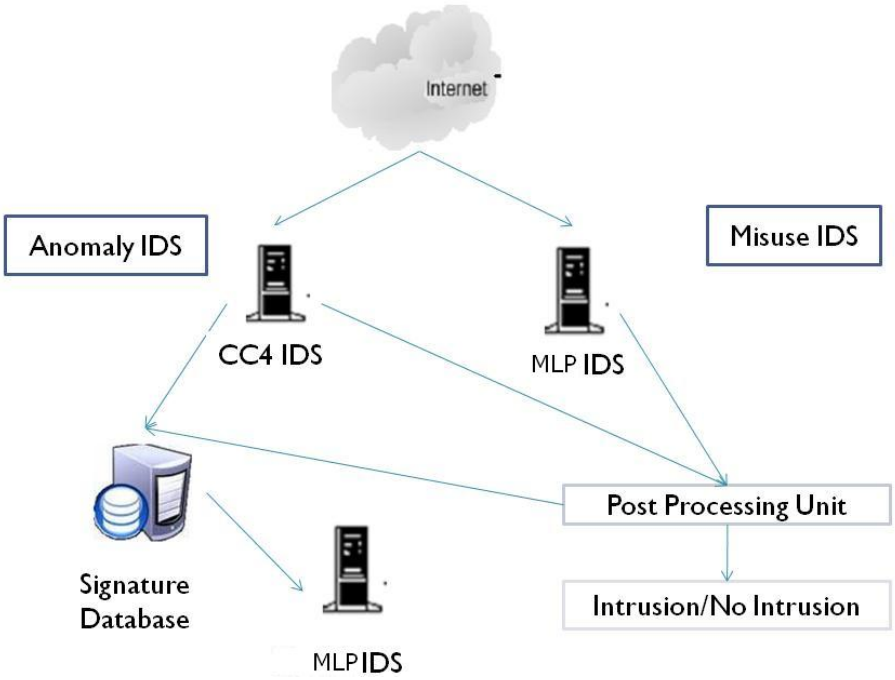


Figure 3.16: Modified Architecture of Hybrid IDS for Detecting and Training IDS for Unknown Attacks

ing,whereas MLP IDS's learns better with every iteration and hence its accuracy increase with the number of iterations.

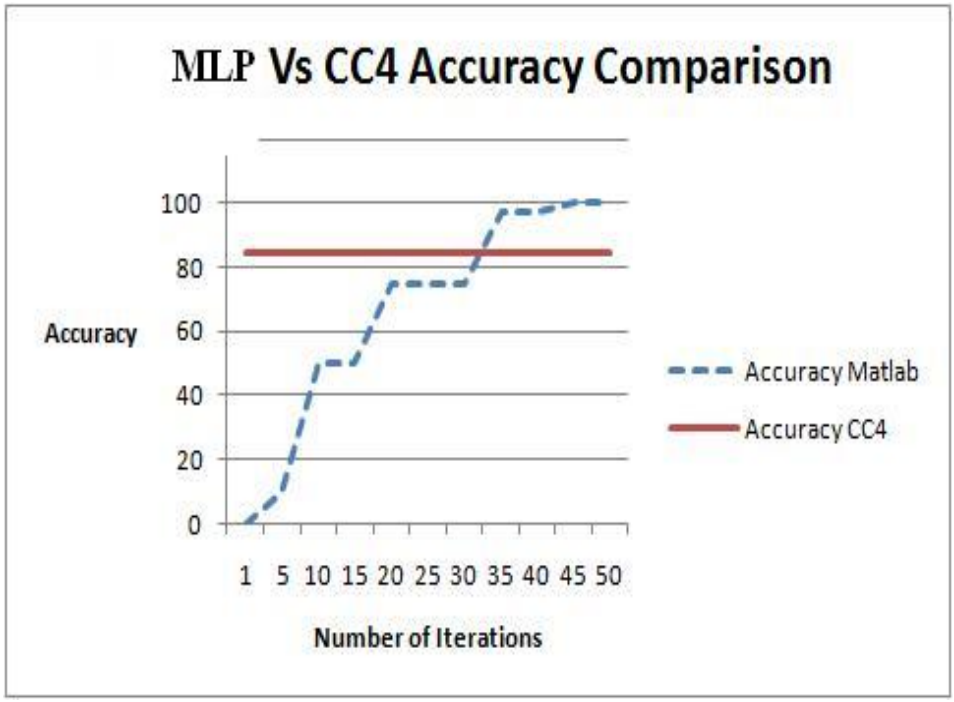


Figure 3.17: *CC4* IDS vs MLP IDS on Variable Number of Iterations

3.7.2 False positive and false negative comparison of MLP IDS Vs *CC4* IDS

MLP IDS shows a very low, close to 0% false positive and false negative rate when it is fully trained with signatures of known intrusion attack types and with signature of normal type. The *CC4* IDS shows some false negative behavior. Figure 3.18 and figure 3.19 show the graph of *CC4* based IDS and MLP IDS when both where tested with the same dataset. For better understanding and better comparison we just used one output neuron at the output layer which will output '1' if the input vector belongs to any of the known attack classes and '0' if input the vector belongs to a normal data class of input. MLP IDS gave 0% false negative and false positive rate whereas *CC4* IDS gave 1.515% false negative rate when tested with the first 183 attack type data and next 80 normal type data.

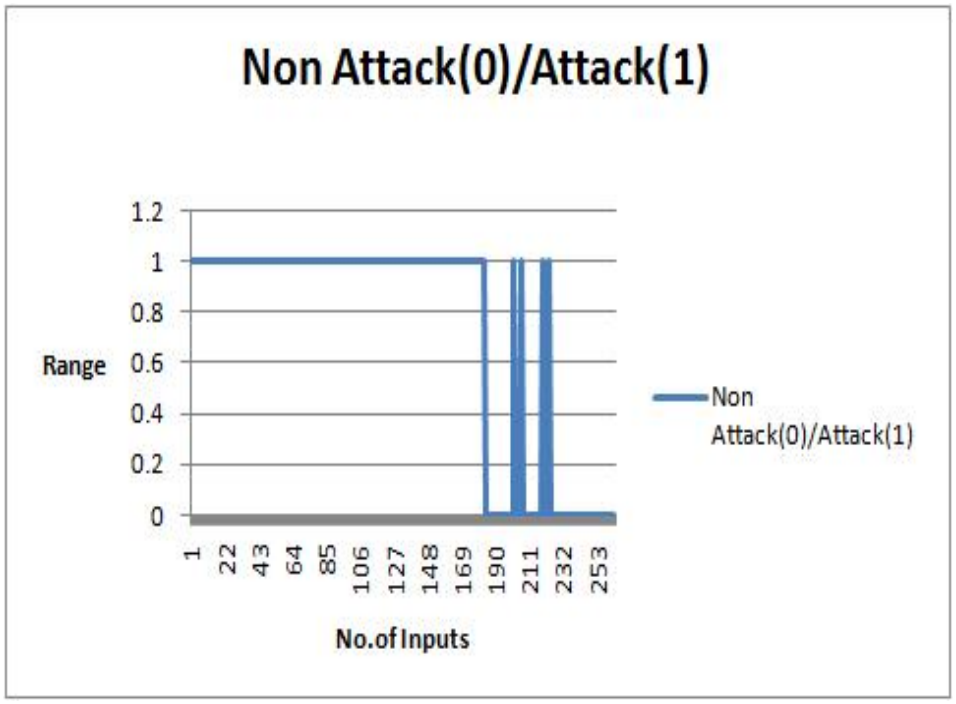


Figure 3.18: *CC4* IDS False Positive and False Negative rate

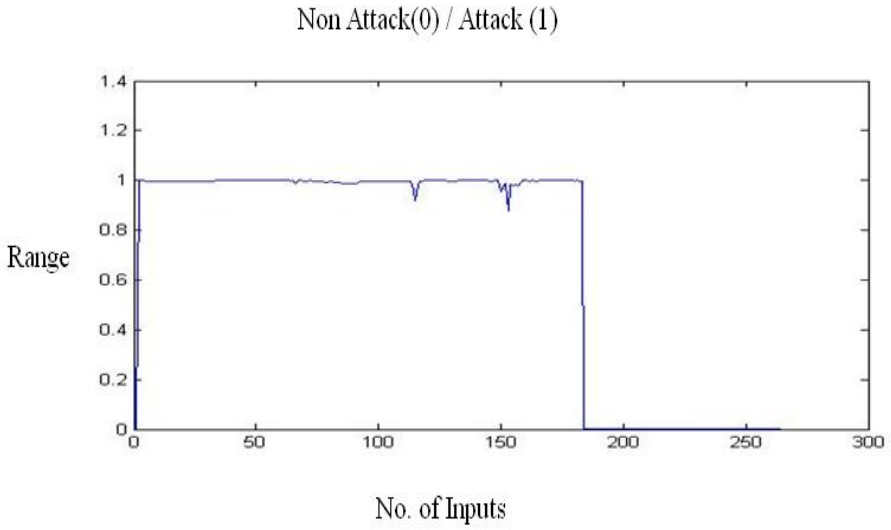


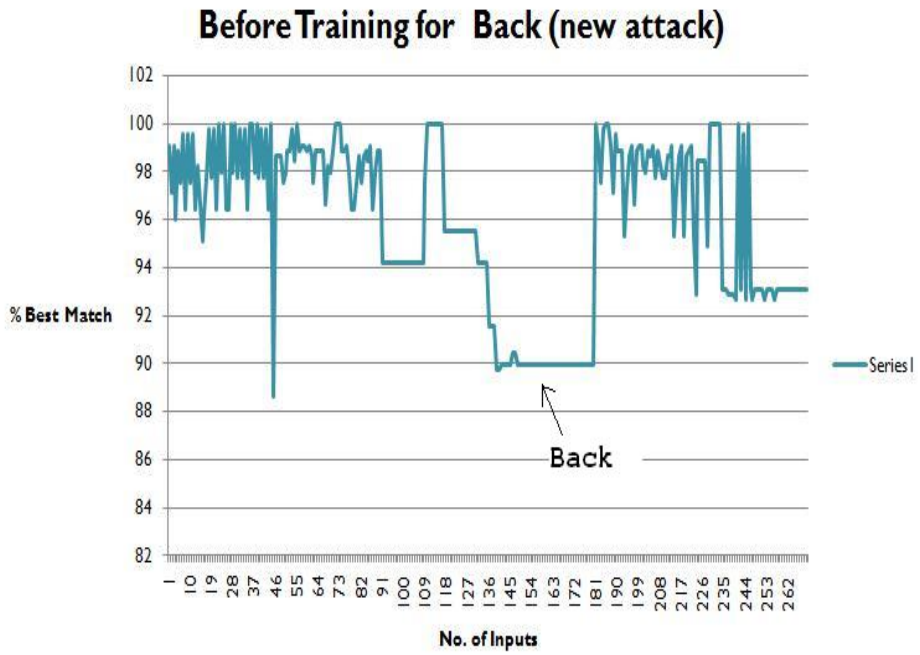
Figure 3.19: MLP IDS False Positive and False Negative Rate

3.7.3 Performance of MLP IDS Vs *CC4* IDS for Unknown Attack

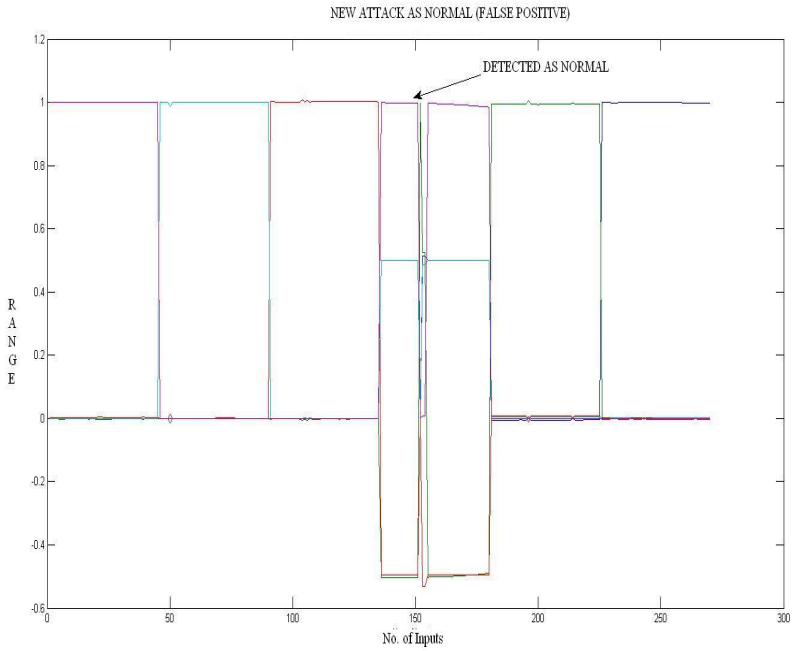
Multiple simulation run on MLP IDS showed more false positive rate and even identified a wrong class of attack for an unknown attack input, for which it had no prior training. The main purpose of the hybrid system is to avoid this behavior of MLP IDS. The *CC4* IDS on the other hand shows a downfall in its percentage best match graph below the threshold for any unknown input data, which helps to identify the abnormal behavior of the input against all known patterns. Both the IDSs can be trained for new attacks by collecting their signatures and retraining the neural networks. Thus further occurrences to that particular unknown attack can be detected properly. Figure 3.20 (a) shows percentage best match graph of *CC4* IDS before training for "Back" attack and figure 3.20 (b) shows graph with the test dataset shown in table 3.7 after training. Similarly figure 3.21 (a) shows the output graph of MLP IDS before training for "Back" attack and figure 3.21 (b) show output graph with same dataset after training for "Back" attack. Figure 3.22 shows the mean square error of the Levenberg-Marquardt training for MLP IDS procedure vs the training epochs for a two layer feed forward neural network with configuration- $\{41\ 10\ 10\ 6\}$. The decrease in the error was completely satisfactory for intrusion detection.

Record Type	Training Set	Validation set	Test Set
Normal	1000	1000	45
Satan	1000	1000	45
Smurf	1000	1000	45
Back	1000	1000	45
Teardrop	1000	1000	45
Neptune	1000	1000	45

Table 3.8: Distribution of Data Vectors in Different Subsets used for Training, Validation and Testing Set for MLP IDS and *CC4* IDS to Analyze their behavior for an Unknown Attack

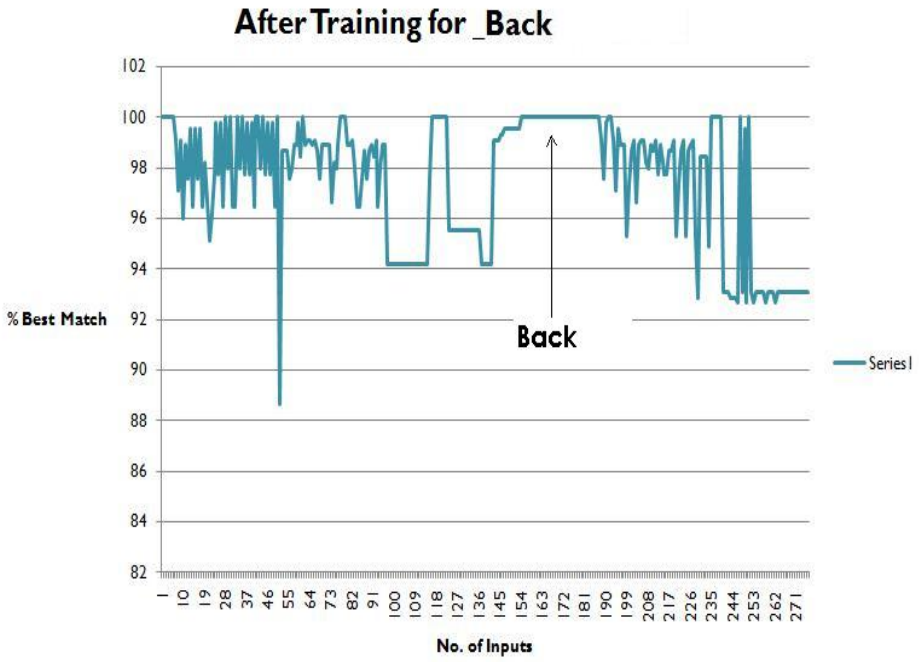


(a) *CC4* IDS Percentage Best Match graph for Unknown Attack before training

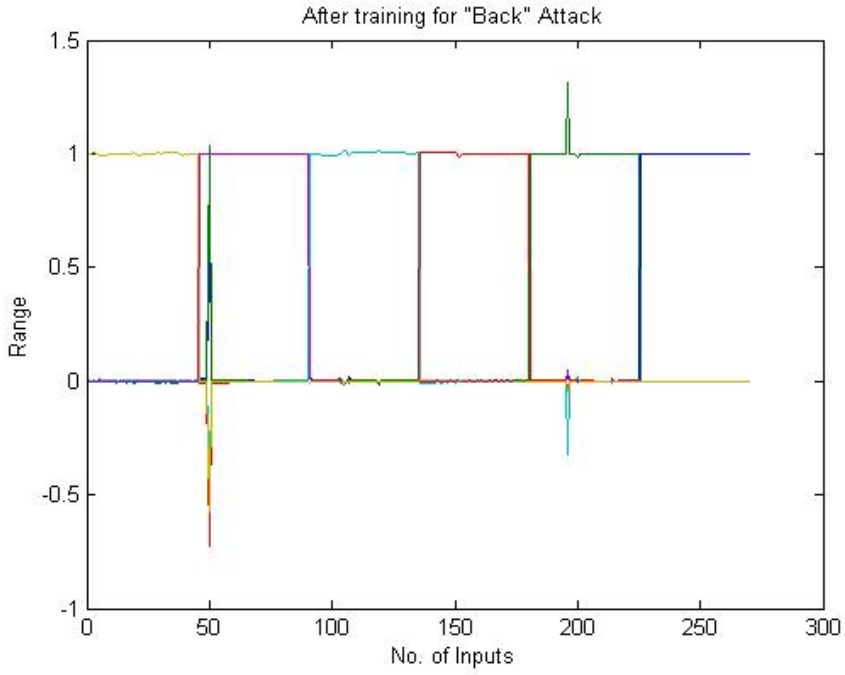


(b) *CC4* IDS Percentage Best Match graph for Unknown Attack after training

Figure 3.20: Behavior of MLP IDS vs *CC4* IDS for Unknown Attack Before Training



(a) MLP IDS graph for Unknown Attack Before Training



(b) MLP IDS graph for Unknown Attack After Training

Figure 3.21: Behavior of MLP IDS vs *CC4* IDS for Unknown Attack After Training

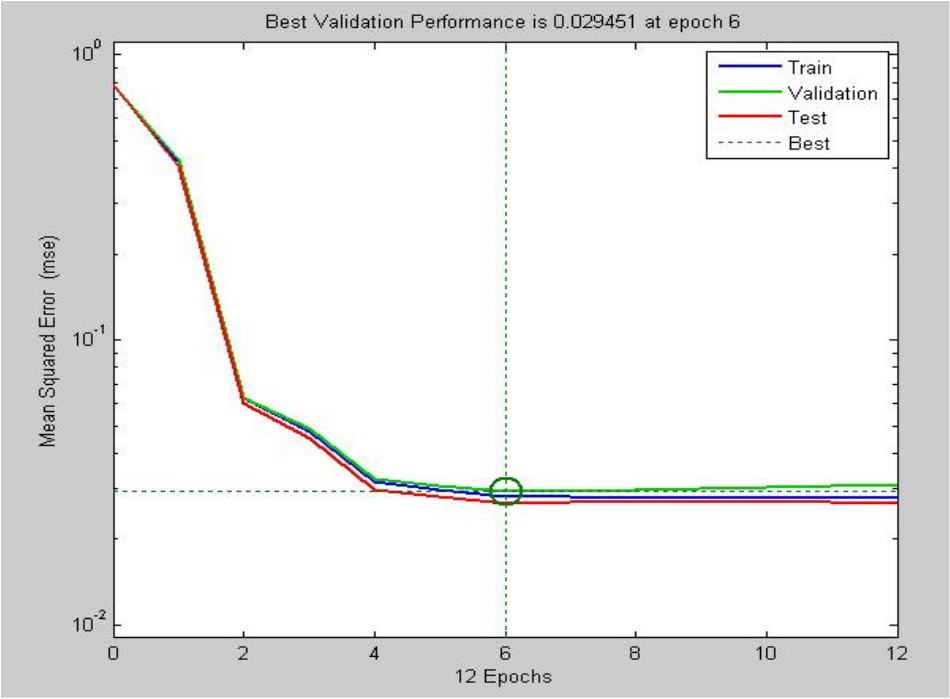


Figure 3.22: The Mean Square Error of the Levenberg-Marquardt Training Procedure vs Training Epochs

CHAPTER 4

CONCLUSIONS

4.1 Summary

We have successfully implemented an instantaneous and accurate intrusion detection system using a hybrid network architecture and proved that instantaneous intrusion detection system is very efficient with both known and unknown attack detection. The hybrid system uses an anomaly based analysis to identify new attacks which is implemented using the *CC4* algorithm- an instantaneously trained neural network and uses a misuse based analysis for known attack detection which is implemented using a basic two layered feed forward network, the MLP neural network tool.

It is observed that the hybrid IDS with three components, the *CC4* IDS, two-layer feed-forward based IDS and the Post Processing Unit work together to increase the accuracy of intrusion detection. The hybrid intrusion detection system detects not just whether the input is an attack or a non attack, but its also tells to which class the attack belongs. The Hybrid system reduces false positive and false negative rates through a well trained two layered feed forward network IDS. The Hybrid system is capable of instantaneous detection of unknown attacks, hence the name Instantaneous Intrusion Detection System. The simulation on the 1998 KDD dataset shows that the instantaneous intrusion detection system is capable of real time detection of known and unknown attacks with high accuracy and low false positive and false negative rates.

4.2 Summary of Results

In our thesis we present an effective intrusion detection system which can detect normal, specific attack class as well as unknown attacks, thus improving on current misuse based intrusion detection system. The results shows the hybrid system is capable of detecting attack classes with 90-92% accuracy and with less than 3% of false positive and false negative rate, when two-layer feed-forward neural network based IDS is well trained. The hybrid system detects new or unknown attack with an accuracy of 80-85%.

4.3 Critique

- The *CC4* algorithm requires data to be converted into unary form which requires extra computation efforts
- The results of both the IDS should be synchronized to analyze the output of both the IDS in real time scenario.

4.4 Future Work

- The *CC4* based IDS requires quantization mapping to convert floating point values to unary value. A fast classification network was proposed by Kun Won Tang and Subhash Kak [22] in 2002, which possesses the same generalization characteristics and fast training speed as that of *CC4* neural network and does not require any data conversions. This could be a replacement to *CC4* IDS and may give better results in terms of detection speed as pre mapping and post output comparisons can be avoided.
- A better data mining approach can be used to get an ideal input vector which contains fewer attributes than the original input vector, to qualify for all possible attacks class.
- The hybrid system can be implemented and tested with real time network data and see its performance. A evaluated delay can be added to synchronize the output from

the respective intrusion detection systems.

BIBLIOGRAPHY

[1] John McHugh, Alan Christie, Julia Allen, "Defending Yourself: The Role of Intrusion Detection Systems," IEEE Software, vol 17, no. 5, pp. 42-51, Sep./Oct. 2000,

[2] S. Kak, "A class of instantaneously trained neural networks", Information Sciences, vol 148, 97-102, 2002.

[3] P. Rajagopal, S. C. Kak, "Instantaneously trained neural networks with complex-valued neurons ", In Complex-Valued Neural Networks: Theories and Applications, Akira Hirose (ed.), World Scientific Publishing Co., 2004.

[4] Peng Ning, Sushil Jajodia, "Intrusion Detection Techniques", The Internet Encyclopedia. John Wiley & Sons." H. Bidgoli (Ed.), ISBN: 0-471-22201-1. December 2003.

[5] Abhilash Ponnath, "Instantaneously Trained Neural Networks", Computer Research Repository, vol abs/cs/0601129, 2006.

[6] S. Kak, "New algorithms for training feedforward neural networks", Pattern Recognition Letters, vol 15, pp.295-298; 1994.

[7] S. Kak, "Better web searches and faster prediction using instantaneously trained neural networks", IEEE Intelligent Systems, vol. 14(6), pp. 78-81, 1999.

[8] G. S. Reddy "Generalization and efficient implementation of *CC4* Neural Network ", Oklahoma State University, Masters Thesis; 2008.

[9] J. Cannady "Artificial neural networks for misuse detection", Proceedings of the National Information Systems Security Conference, pp. 368-81, 1998.

[10] Idea for multilevel Percepton, "$http://www.zurzuvi.com/neuralnetwork.html$" [Last Accessed May 23rd 2010]

[11] J. Cannady "Artificial neural networks for misuse detection", Proceedings of the National Information Systems Security Conference, p. 368-81, 1998.

[12] J. Ryan , M.J. Lin , R. Miikkulainen "Intrusion detection with neural networks", Advances in Neural Information Processing Systems, Cambridge, MA: MIT Press"; vol. 10, 1998.

[13] Donald Marquardt, "An Algorithm for Least-Squares Estimation of Nonlinear Parameters", SIAM Journal on Applied Mathematics,vol 11, pp- 43 441, 1963.

[14] D. Russell and G. T. Sr. Gangemi , Computer Security Basics, Reilly & Associates, Inc., California, 1991.

[15] Ernesto Kofman and Sergio Junco. "Quantized-state systems: A DEVS approach for continuous system simulation", Transactions of Society for Modeling & Simulation International,vol 18(3):123 132, 2001.

[16] MIT Lincoln Laboratory, DARPA Intrusion Detection Evaluation Data Sets, "http:$//www.ll.mit.edu/IST/ideval/data/data_index.html$" [Last Accessed March-15-2009]

[17] Matwork Neural Network Tool Guide, "$hhtp://www.mathworks.com/help/pdf\_doc/nnet/nnet.pdf$" [Last Accessed March-10-2009]

[18] A.Ghosh, A.Schwartzbard, and M. Shatz, 1999, ”Learning Program Behavior Profiles for Intrusion Detection”, Proceedings of the First USENIX Workshop on Intrusion Detection and Network Monitoring, Santa Clara, California, April 1999.

[19] I. T. Jolliffe, Principal Component Analysis, 2nd edition, Springer, 2002.

[20] S Mukkamala, G Janowski, A H. Sung Intrusion Detection Using Neural Networks and Support Vector Machines Proceedings of IEEE International Joint Conference on Neural Networks Hawaii, May 2002), pp.1702-1707, 2002.

[21] M. Moradi, Mohammad Zulkernine. ”A Neural Network Based System for Intrusion Detection and Classification of Attacks”, Proceedings of the 2004 IEEE International Conference on Advances in Intelligent Systems-Theory and Applications, Luxembourg-Kirchberg, Luxembourg, 2004

[22] K. W. Tang and S. Kak, ”Fast classification networks for signal processing”, Circuits, Systems Signal Processing Vol 21, pp.207- 224, 2002.

[23] Stefanos Koutsoutos, Ioannis T. Christou, Sofoklis Efremidis,” An Intrusion Detection System for Network-Initiated Attacks Using a Hybrid Neural Network”, Artificial Intelligence Applications and Innovations, pp 228-235, 2006.

[24] H. Debar, M. Becker, D. Siboni: ”A Neural Network Component for an Intrusion Detection System”, Proceedings of the 1998 National Information Systems Security Conference (NISSC’98) Ariington, VA. October 5-8 1998.

[25] John Zhong Lei, Ali Ghorbani, ”Network Intrusion Detection Using an Improved Competitive Learning Neural Network”, Proceedings of the Second Annual Conference on Communication Networks and Services Research (CNSR’04), pp.190-197, 2004

[26] B. Shu and S. Kak, "Neural Network-based Intelligent Metasearch Engine", Information Science, vol 120, 1-4, pp. 1-11, 1999.

[27] NEURAL NETWORKS: Basics using MATLAB Neural Network Toolbox $http : //s2.e-monsite.com/2010/04/09/38379014nn2000-pdf.pdf$ [Last Accessed 12-March-2010]

[28] R. Trost, Practical Intrusion Analysis, Addison-Wesley, 2009.

[29] William Stallings, Cryptography and Network Security: Principles and Practice, Prentice Hall, 5/E , 2011

VITA

Rohit Pillay

Candidate for the Degree of

Master of Science

Thesis: INSTANTANEOUS INTRUSION DETECTION SYSTEM

Major Field: Computer Science

Biographical:

Personal Data: Born in Bilaspur, India on April 24th, 1986.

Education:

Received the B.E. degree from Shri Guru Gobind Singhji Institute of Engineering and Technology, Nanded, India, 2007, in Information Technology
Completed the requirements for the degree of Master of Science with a major in Computer Science from Oklahoma State University in Dec 2010.

Experience:

Graduate Technical Lab Assistant at Physical Science headed by Dr. RJ Hauenstein, Stillwater, OK.

Jan 2009- Dec 2010

Asst. Software Engineer at Tata Consultancy Services Limited, Mumbai, India.

Jan 2008- May 2008

Name: Rohit Pillay Date of Degree: December, 2010

Institution: Oklahoma State University Location: Stillwater, Oklahoma

Title of Study: INSTANTANEOUS INTRUSION DETECTION SYSTEM

Pages in Study: 53 Candidate for the Degree of Master of Science

Major Field: Computer Science

The security of computer networks is a critical issue. Deficiencies within these networks makes them venerable to malicious actions that compromise the integrity, confidentiality or availability of the resources. The major problems with current intrusion detection systems (IDS) is the speed and accuracy of detection. The current neural network based intrusion detection systems requires offline training and are unable to detect new or unknown attacks in real time. In our thesis we present a faster neural network based hybrid intrusion detection system which can detect known and unknown patterns in real time. The hybrid system utilizes the *CC4* instantaneously trained neural network as an anomaly based IDS to detect unknown class of attacks and a two layered feed forward neural network as a misuse based IDS to detect known attacks, Furthermore, the hybrid system classifies attacks into classes.

The hybrid IDS has three components, a *CC4* IDS which is used as an anomaly based IDS to detect unknown attacks, a two-layer feed-forward Levenberg-Marquardt training algorithm based IDS which is used as a misuse base IDS and the Post Processing Unit. The outputs of the respective IDSs are processed at the Post Processing Unit and based on the output the class of attack is determined. Results shows the hybrid system is capable of detecting known attacks class with 90-92% accuracy and with less than 3% of false positive and false negative rates. The hybrid system detects new or unknown attacks with an accuracy of 80-83%. The hybrid system is also able to detect new attacks in 1 iteration, thereby making it applicable for real time intrusion detection.

ADVISOR'S APPROVAL: _____