"SCALABLE NETWORK INTRUSION DETECTION SYSTEM FOR DETECTION OF DOS AND DDOS ATTACKS"

A THESIS SUBMITTED TO

BHARATI VIDYAPEETH UNIVERSITY, PUNE

FOR AWARD OF DEGREE OF

DOCTOR OF PHILOSOPHY IN COMPUTER ENGINEERING

UNDER THE FACULTY OF ENGINEERING & TECHNOLOGY

SUBMITTED BY

MR. VIJAY DADASAHEB KATKAR

UNDER THE GUIDANCE OF

DR. S. G. BHIRUD

RESEARCH CENTRE

BHARATI VIDYAPEETH UNIVERSITY

COLLEGE OF ENGINEERING, PUNE. 411043

JULY 2016

CERTIFICATE

This is to certify that the work incorporated in the thesis entitled "Scalable Network Intrusion Detection System for detection of DoS and DDoS Attacks" for the degree of Doctor of Philosophy in the subject of Computer Engineering under the faculty of Engineering & Technology has been carried out by Mr. Vijay Dadasaheb Katkar in the Department of Computer Engineering at Bharati Vidyapeeth University, College of Engineering, Pune during the period from June 2009 to July 2016 under the guidance of Dr. S. G. Bhirud.

Place: Pune

Date:

(Prof. Dr. A. R. Bhalerao)

Principal & Dean

CERTIFICATION OF GUIDE

This is to certify that the work incorporated in the thesis entitled "Scalable Network Intrusion Detection System for detection of DoS and DDoS Attacks" submitted by Mr. Vijay Dadasaheb Katkar for the degree of Doctor of Philosophy in the subject of Computer Engineering under the faculty of Engineering & Technology has been carried out in the Department of Computer Engineering, Bharati Vidyapeeth University College of Engineering, Pune during the period from June 2009 to July 2016, under my direct supervision.

Place: Pune (Dr. S. G. Bhirud)

Date: Guide

DECLARATION BY CANDIDATE

I hereby declare that the thesis entitled "Scalable Network Intrusion Detection System for detection of DoS and DDoS Attacks" submitted by me to the Bharati Vidyapeeth University, Pune for the degree of Doctor of Philosophy (Ph.D.) in Computer Egineering under the faculty Engineering & Technology is original piece of work carried out by me under the supervision of Dr. S. G. Bhirud. I further declare that it has not been submitted to this or any other university or Institution for the award of any degree or Diploma.

I also confirm that all the material which I have borrowed from other sources and incorporated in this thesis is duly acknowledged. If any material is not duly acknowledged and found incorporated in this thesis, it is entirely my responsibility. I am fully aware of the implications of any such act which might have been committed by me advertently or inadvertently

Place: Pune (Mr. Vijay Dadasaheb Katkar)

Date: Research Student

ACKNOWLEDGEMENTS

I am grateful to my guru Shree Shivkrupanand Swami and Trilok Tripathi for their continuous blessings and unconditional love.

My first debt of gratitude goes to my guide Dr. S. G. Bhirud, who provided the vision, encouragement and advice necessary for me to progress not only through the doctoral program but also through various phases of life. He has always been a supportive teacher to me. I whole heartedly admire his excellence in every role that he plays and will always remain grateful to him for everything that is far beyond expressing it in words.

I am very much thankful to the Management of PCCOE, Pune for their valuable support and cooperation. I am also thankful to Dr. A. M. Fulambarkar, Principal PCCOE, Dr. Sudeep Thepade, Head, Dept of IT, PCCOE, Mr. Manish Narkhede, Dr. Surbhi Sengar, Ms. Minakshi Panchal, Mrs. Jaya Dewan, Mrs. Vaishali Kulloli, Mr. Sachin Jadhav, and Mr. Sarjerao Katkar for their valuable inputs, technical support and guidance.

I am thankful to Dr. A. R. Bhalerao, Dean, BVUCOE, Pune, Dr. Sunita Raut, Coordinator, R&D cell BVUCOE, faculty members of Department of IT, PCCOE for their important suggestions and support. My special thanks to the committee members Dr. Prasanna Joeg and Dr Manasi Patwardhan for their valuable suggestions and guidance.

I am grateful to my parents and wife Poornima for their unconditional support and co-operation they have extended throughout the research work. They were always a source of inspiration.

Finally I would like to thank all those who have directly or indirectly helped me in research work.

Vijay Dadasaheb Katkar

TABLE OF CONTENTS

ABSTRACT

With continuous growth in use of Internet for providing services to customers and amount of economy involved in these services; attacks on these services are also increased. Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks are most prevalent threats to the on-line services. These attacks consumes the resources required to provide the on-line services and makes these services unavailable for legitimate users. These attacks results in massive loss of data, resources, money and reputation of service providing organization.

Intrusion detection system is widely used tool by organizations to detect these attacks. Generally, intrusion detection system detects the attack by comparing the network traffic with signatures of known attacks. Data Mining approach is one of the widely used approach to automate the attack signature generation process. KDD 99 and CDMC 2012 datasets are widely used by researchers to evaluate the performance of their proposed intrusion detection systems.

Different categories of intrusion detection system, research done by various researchers over last two decades for designing intrusion detection systems belonging to these categories is reviewed and limitations of these systems are highlighted. KDD 99, CDMC 2012 and dataset created in the laboratory are analyzed. These dataset are used to evaluate the performance of proposed methodologies for detection of denial of service and distributed denial of service attacks.

Ensemble of classifiers approach is used by many researchers to improve the attack detection accuracy. This report presents the methodology for creation of adaptive ensemble of classifiers for detection of denial of service and distributed denial of service attacks. This approach adapts the continuous changes in normal user behavior and attack strategies for selecting the appropriate base classifiers. Adaptive ensemble of classifiers gives detection accuracy of 99.81%, 97.40%, 96.44% for KDD 99, CDMC 2012 and dataset generated in laboratory respectively.

Intrusion detection systems are categorized as anomaly detection and misuse detection based intrusion detection system. Misuse detection based system looks for patterns of known attacks in network traffic to detect these attacks. It can detect known attacks with high accuracy; however, it cannot detect novel attacks launched by attackers as their patterns are not known. Anomaly detection based system develops a model of

genuine network user access patterns and considers any deviation in this as attacks. It can detect novel attacks without prior knowledge; however, it observes very high false positive rate for novel attacks.

Many researchers have proposed a system using combination of misuse and anomaly detection approach to detect known as well as novel attacks with high accuracy. However, as behavior of genuine user changes continuously, sometimes normal user behavior is considered as novel attack.

Further methodology to detect known as well as novel attacks using combination of misuse and anomaly detection based system has been proposed. Frequent pattern mining approach has been used to detect known attacks and adaptive ensemble of classifiers have been used to detect novel attacks. Intelligent approach is presented to detect normal behavior, genuine variation in normal behavior and unknown attack. Proposed methodology achieves the 99.16%, 83.22% and 98.39% detection accuracy for KDD 99, CDMC 2012 and dataset generated in laboratory respectively.

With an exponential growth in computing power of machines and decreases in their cost; amount of network traffic generated is also increased. It is very difficult to process such a huge volume of network traffic for attack detection in real-time.

This report presents a distributed approach for processing huge amount of network traffic in real time using unused CPU cycles of the machines present within the organization. This approach filter outs the network connection records from attack detection process by analyzing the error indicating attributes of network connection records. This reduces the processing power and resource requirement of attack detection system and enables real time attack detections. The proposed approach detected; 95.39% attacks launched from inside the organization, 95.48% attacks launched from outside the organization and 100% spoofing activities.

When very large number of users starts using a particular service over network (e.g. shopping during online mega sale) or access a particular data present over the network (e.g. news about death of Michael Jackson) this results into a huge flood of network traffic towards servers in short period of time. Servers as well as intrusion detection system cannot handle such a huge volume of traffic generated in very short time span and considers it as distributed denial of service attack instead of normal user behavior.

Methodology is proposed to differentiate the genuine flood of user connections from intentional distributed denial of service attack in real time by analyzing the data present on social media, news RSS feeds and recent keywords searched on search engines. A methodology is suggested to gradually scale up and scale down the servers and intrusion detection nodes on cloud by analyzing the change in network traffic volume.

List of Figures

List of Tables

ABBREVIATIONS

AB-IDS	Agent based IDS
AEC	Adaptive Ensemble of Classifiers
A-IDS	Anomaly detection based IDS
AFRL	Air Force Research laboratory
B-IDS	Batch processing IDS
BDA-IDS	Big Data Analysis IDS
CDMC	Cyber Security and Data Mining Competition
DARPA	Defense Advance Research Project Agency
D-IDS	Distributed IDS
DoS	Denial of Service
DDoS	Distributed Denial of Service
EC	Ensemble of Classifiers
E-IDS	Ensemble of Intrusion Detection Engine based IDS
H-IDS	Host based IDS
Hi-IDS	Hierarchical IDS
HMM	Hidden Markov Models
IDE	Intrusion Detection Engine
IDS	Intrusion Detection System
KDD	Knowledge Discovery and Data Mining
KNN	K-Nearest Neighbor
M-IDS	Monolithic IDS
NPA-IDS	Network Packet Analysis based IDS
NPHA-IDS	Network Packet Header Analysis based IDS
NPPA-IDS	Network Packet Payload Analysis based IDS
N-IDS	Network based IDS
OS-ELM	Online Sequential Extreme Learning Machines
PTPC	Probability of True Positive Classification
R2L	Remote to Login
Rerror_rate	% of connections that have REJECT errors
R-IDS	Real time IDS
ROC	Receiver Operating Characteristic
Serror_rate	% of connections that have SYN errors
S-IDS	Signature Detection based IDS

SIGKDD	Special Interest Group on Knowledge Discovery and Data Mining
SPADE	Sequential PAttern Discovery using Equivalent classes
TF-ITF	Term Frequency-Inverse Term Frequency
U2R	User 2 Root attack

Chapter 1
Introduction

With the advancement of technology and decrease in cost of Personal Computers, access to the Internet has become easier and World Wide Web sites have become more sophisticated and inviting. In 1992 Paul Linder and Mark McCahill released Gopher tool which allowed researchers to retrieve required and specific data from numerous locations (Prezi, 2016a). In 1993 Marc Andreesen (founder of Netscape) developed a web browser at the University of Illinois and World Wide Web became a public domain (Mayo and Newcomb, 2009). In 1994 shopping malls arrived on the Internet, allowing to order pizza from Pizza Hut online or do online banking transactions (Prezi, 2016a). 16 million users were connected to Internet in 1995 and this number reached to 2,937 million in 2014 (Prezi, 2016b). With this rapid growth in number of users connected to internet, all organizations started provided their services over internet.

With rapid growth in the number of services provided over Internet; number of attacks on these services is also increased. Any act of compromising confidentiality, integrity or availability of online or offline computer system is called as Intrusion. 'Vladimir Levin' is the first publicly known Internet bank robber from Russia (Prezi, 2016a).

Albert Gonzalez stole the information of 45.7 million payment cards of US retailer TJX's customers, which resulted into loss of 256 million dollar (Palermo, 2015). In May 2014 Distributed Denial of Service (DDoS) attack was launched against Bank of China and the Bank of East Asia at a rate of 7.39 Gbps which denied the online services offered by these banks to legitimate customers. On 31st July 2015 customers of Royal Bank of Scotland, NatWest, and Ulster Bank were not able to access online services for fifty minutes due to DDoS attack (Honan, 2015). Attack on any online services results in massive loss in terms of money and reputation of organizations providing Internet based services (i.e. online services).

Figure 1.1 shows the average loss faced by top most companies of seven major countries due to Internet based attacks in 2015. This loss includes the cost of

investigation, detection, recovery and after effects (Pegram, 2016).

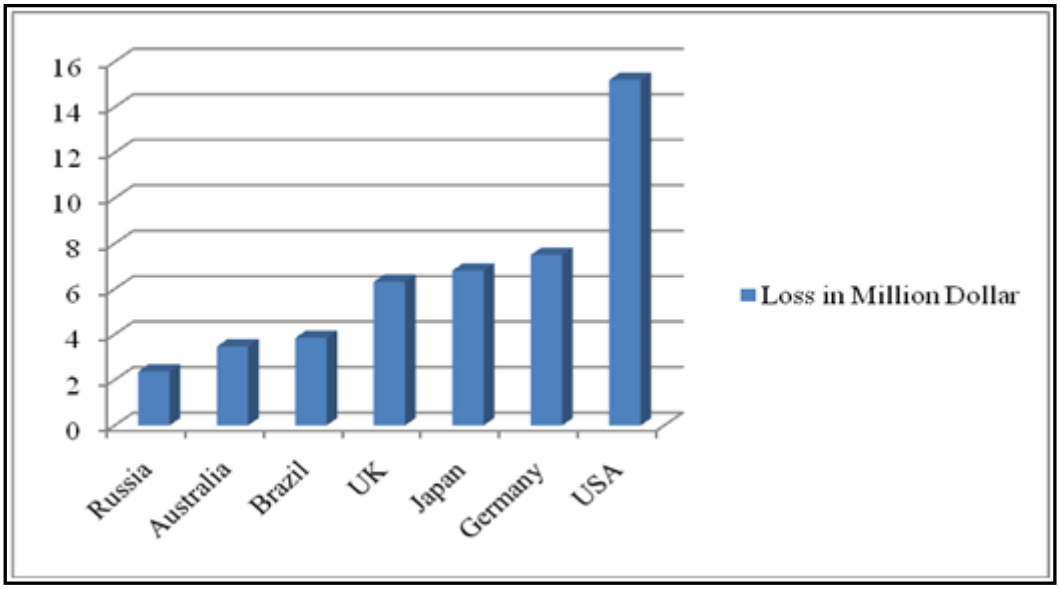


Figure 1.1: Average Loss Due to Cyber Crime in 2015

Firewall is widely used security enforcement tool by all the organizations however, it does not guarantee the complete security and sophisticated attacker may break this barrier. In this situation to know; how the attacker breached the security and what harm he did to the system Intrusion Detection System (IDS) is used by organizations as a second line of defense. Figure 1.2 shows typical placement of firewall and IDS in the organization (Veteranus, 2013).

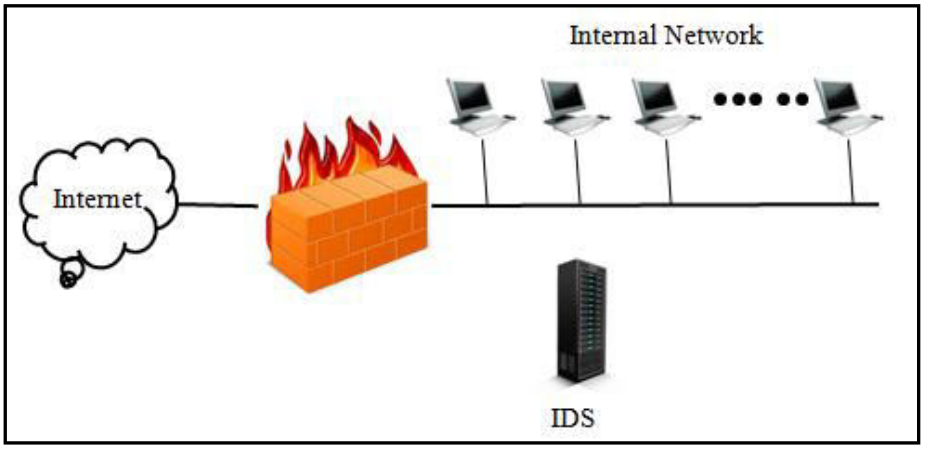


Figure 1.2: Placement of Firewall and IDS

1.1 Classification of Intrusions

Any sequence of related actions performed by a malicious adversary that results in the compromise of a target system is called as Intrusion. Intrusive activities are broadly

classified into four categories.

i. Probing: It is used for surveillance of network and machines in the network
ii. Remote to Local (R2L): It is an unauthorized access from a remote machine
iii. User to Root (U2R): It is an unauthorized access to local super-user (root) privileges
iv. Denial-of-Service (DOS) and Distributed Denial of Service (DDoS)

Out of these four intrusion activities DoS and DDoS attacks is the most prevalent threat which either exploits vulnerability in computing and communication resources or floods them in order to make the system unavailable for legitimate users. This results in massive loss of data, resources and money.

1.2 Categories of DoS and DDoS Attacks

DoS and DDoS attacks are broadly classified into four categories based on their working mechanism. These are Connection consumption based attacks, Resource consumption based attacks, Vulnerability exploitation attacks and Configuration modification based attacks.

1.2.1 Connection Consumption based Attacks

TCP is a connection oriented protocol. It establishes a connection between server and client before data exchange.

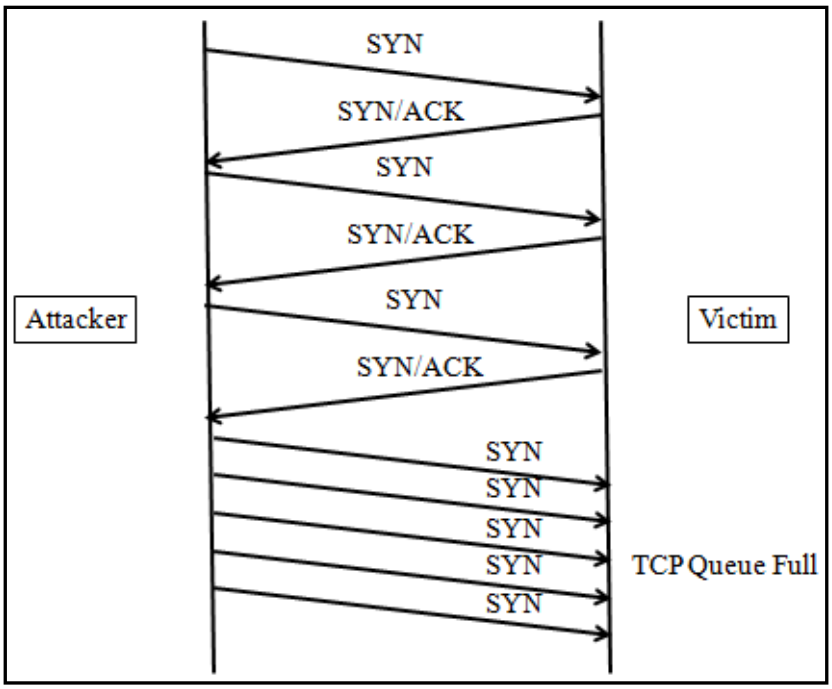


Figure 1.3: Working of SYN Flood Attack

Any server or machine can accept and serve limited number of connection requests. Attacker establishes a huge number of connections with server or target machine so that legitimate users cannot access the service provided by organization. This type of attacks consumes the Operating System's kernel resources required for connection establishment. SYN Flood attack is one of the widely used attacks which fall under this category. Figure 1.3 shows the working of SYN Flood attack in which the attacker establishes a huge number of half open TCP connections and exhausts the connection pool. Due to its working mechanism a cluster of servers can be slowed down using slow network connections (Geetha K. and Sreenath N., 2014).

1.2.2 Bandwidth Consumption based Attacks

Every network has a limited amount of bandwidth. If the volume of network traffic exceeds the bandwidth limit of the network; then it degrades the response time of servers and machines present on the network.

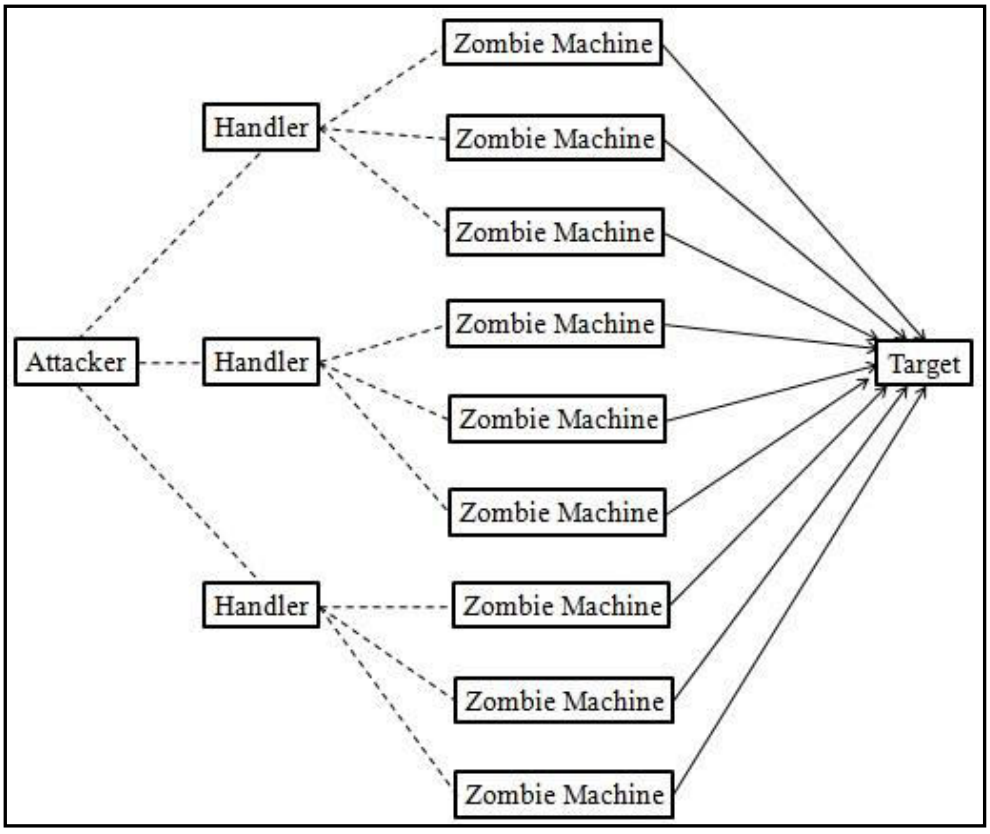


Figure 1.4: Bandwidth Consumption Attack using Zombie Machines

By exploiting this principal bandwidth computation based DoS and DDoS attacks are launched. Attacker uses preconfigured handler machines to control huge number of preconfigured zombie machine connected to Internet to create a huge flood as shown

in Figure 1.4. UDP flood is widely used attack of this category (Li et al., 2008).

1.2.3 Vulnerability Exploitation Attacks

Attacker identifies and exploits the vulnerabilities present in the target system to either crash or slow down it. These types of attacks are very difficult to identify as attacker completely mimics the behavior of legitimate user. In Slow HTTP Request attacks; parts of a HTTP header are sent to HTTP server at very low rate so that time required serving a single request is increased as shown in Figure 1.5. This in turn consumes the available resources for longer time duration.

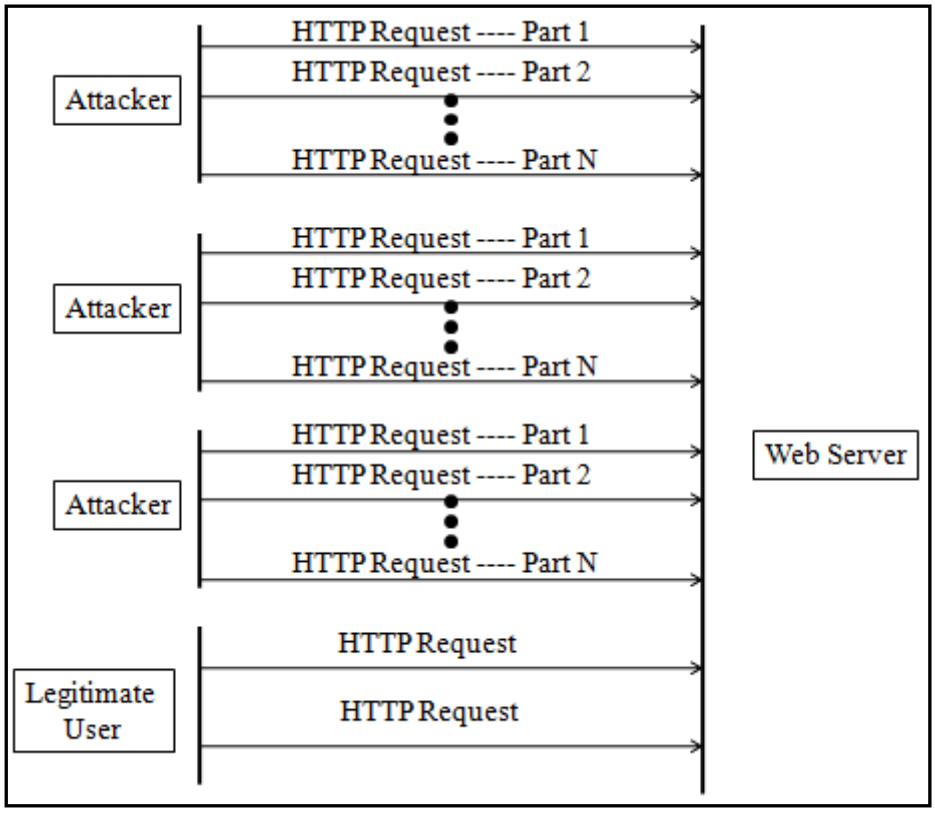


Figure 1.5: Working of Slow HTTP Request Attack

In Slow Read attack, attacker sends a request for large file to the server and then announces a small TCP window size. In response to this server send data at low rate to the client and resources are reserved for long duration. Such attacks can slow down a large cluster of servers with machines having slow or dial-up connection (Park et al., 2014).

1.3 Categorization of IDS

Figure 1.6 shows different criterion for categorization of IDS. It can be categorized based on eight criterions as detection method, deployment location, data source used

for attack detection, response time, architecture, number of detection engines used, volume of data processed and hardware/software implementation.

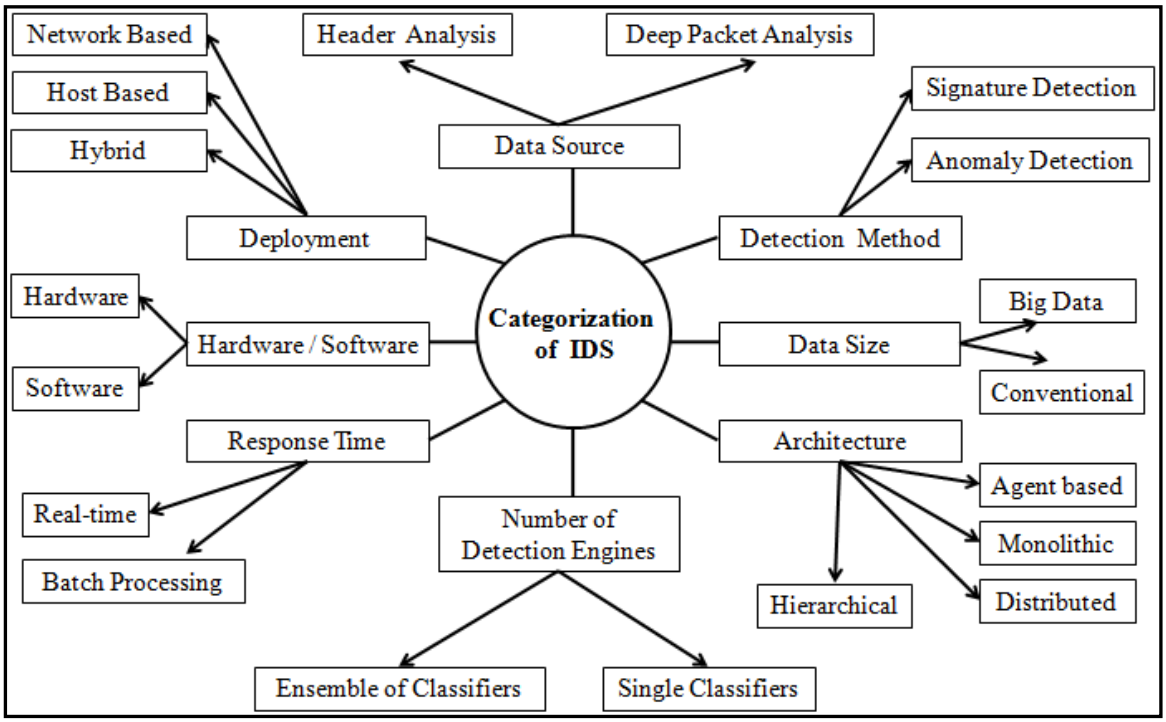


Figure 1.6: Categorization of IDS

1.3.1 Detection Method Categorization

Anomaly detection based IDS (A-IDS) and Signature (Misuse) detection based IDS (S-IDS) are two approaches of detecting intrusive activities. S-IDS look for patters or signatures of known attacks. If match is found then it generates the alarm. Signature database of known attacks is specified a priori (Shieh and Gligor, 1997). S-IDS can detect known attacks with high accuracy however; it has few drawbacks as below:

i. S-IDS cannot detect novel attacks.
ii. Signature database needs to be updated periodically

On the other hand, A-IDS attempts to estimate the normal behavior of the system to be protected and generate an alarm when the deviation between a current behavior of system and normal behavior exceeds a predefined threshold. A-IDS can detect novel attacks, however, it generates high false alarm rate as it is difficult to perfectly generate normal user behavior profile. Also A-ID needs to be updated periodically as normal user behavior changes over a period of time.

A-IDS can be implemented using supervised as well as unsupervised learning

approach. In unsupervised learning IDS builds normal user behavior profile by observing attack free traffic for certain period of time; whereas in supervised learning IDS builds normal user behavior profile by using labeled training dataset. Practically it is difficult to get a live network traffic which is attack free for sufficient duration to train IDS and reflects all possible normal user behaviors

Combination of A-IDS and S-IDS is used to detect known as well as novel attacks with high precision. S-IDS is used to detect known attacks whereas A-IDS is used to detect novel attack. Traffic or Audit Log which does not match with attacks signature as well as normal user behavior is considered as probable intrusive activity.

1.3.2 Deployment Categorization

Based on deployment location of IDS it is categorized as Host based IDS (H-IDS) and Network based IDS (N-IDS). Host based IDS monitors Operating System's sequence calls, audit logs and event logs of the machine on which it is installed to detect intrusive activities. H-IDS analyze more detailed information as compared to N-IDS and produces less false alarm rate as compared to it. It can also detect novel attacks with high precision as compared to N-IDS however; it is completely dependent on the system on which it is installed. H-IDS need to be installed and maintained on every machine of the organization which needs to be monitored. In large organization where thousands of machines need to be monitored, this becomes very difficult task.

N-IDS monitor the traffic of entire network with the help of sensors placed at various locations of network to detect intrusive activities. It can analyze less detailed information as compared to H-IDS however; it detects the intrusion before it reaches the victim machine as opposed to H-IDS. H-IDS normally detects successful intrusive attempts whereas N-IDS can detect both successful as well as failed intrusive attempts which make it more reliable solution for monitoring critical systems (Magalhaes, 2003).

N-IDS have following limitations:

i. N-IDS are vulnerable to packet spoofing attacks
ii. N-IDS cannot monitor the encrypted traffic in depth
iii. N-IDS cannot detect intrusions in real time under heavy network traffic
iv. N-IDS cannot detect insider attacks with high precision if sensors are not placed at appropriate locations

Therefore, combination of H-IDS and N-IDS is used to improve the detection precision wherein H-IDS is used to detect intrusions which targets specific host in the network like Back, Slow Read attacks, and N-IDS is used to detect resource and bandwidth consumption attacks like ICMP, UDP flood.

1.3.3 Data Source Categorization

Intrusion detection process uses captured network packets or audit logs present in the system(s) being monitored. Audit logs generated by Operating System and other programs running on top of it are used for intrusion detection process. Network Packet Analysis based IDS (NPA-IDS) analyzes either Header or Payload of captured network packer. Network Packet Header analysis based IDS (NPHA-IDS) are light weight as compared to Network Packet Payload analysis based IDS (NPPA-IDS) and can perform better under heavy network traffic. NPPA-IDS can detect intrusions with higher precision, however, NPPA-IDS cannot effectively analyze network traffic in which payload is encrypted (Al-Jarrah et al., 2016).

Combination of NPHA-IDS and NPPA-IDS is used to detect intrusions with higher precision and less computing power. First NPHA-IDS is used to detect possible intrusive packet and then NPPA-IDS is used for detailed analysis of possible intrusive packets.

1.3.4 Response-time Categorization

Based on response time of intrusion detection process IDS is categorized as Real time IDS (R-IDS) and Batch processing (i.e. Off-line) IDS (B-IDS). R-IDS analyzes the data present in data source within a finite and specified time period, B-IDS process the data in batches. R-IDS can detect intrusions as they occurs however, it cannot perform detailed analysis for intrusion detection process under heavy load which degrades it detection performance. B-IDS gives stable performance under heavy load however, due to its high response time under heavy load it is not used for intrusion detection in critical systems like Banking (Pfleeger and Pfleeger, 2003).

1.3.5 Architecture Categorization

Based on architecture IDS are categorized into following four categories:

i. Monolithic IDS (M-IDS)

ii. Hierarchical IDS (Hi-IDS)
iii. Agent based IDS (AB-IDS)
iv. Distributed IDS (D-IDS)

Monolithic IDS (M-IDS) are conventional IDS which perform intrusion detection activity as a single unit and decision making is done at only one level whereas in Hierarchical IDS (Hi-IDS) intrusion alert or aggregated information generated at lower level of IDS is used by higher level for further decision making. This hierarchical nature of information or alert processing improves the detection precision for complex attacks and reduces the false alarm rate. Agent based IDS (AB-IDS)divides the IDS system into small agents programs which are placed at different locations in the network. These agents capture, processes and passes the information for intrusion detection through a predefined chain. These specialized agents places a very little computing overload on a machine on which they are installed which may provide a light weight IDS solution. Distributed IDS (D-IDS) consist of multiple IDS present over a large network communicating with each other through central IDS or distributed agents. D-IDS gives the global view of intrusive activities in the large network and can effectively identify the intruders and intrusions in real time (Sen et al., 2006).

1.3.6 Number of Detection Engines Categorization

Conventional IDS uses single Intrusion Detection Engine (IDE) to detect intrusive activities. If training set for IDE is balanced then it give high detection precision for all the attacks, however, it is not practically possible to generate balanced training set for intrusion detection process. Conventional single IDE based IDS cannot detect all the attacks with high precision. To solve this problem Ensemble of IDE based IDS (E-IDS) is used. E-IDS uses multiple IDEs simultaneously and then combines their output using majority vote or weighted majority vote approach to detect the intrusion.

1.3.7 Data Size Categorization

Number of computing machines in an organization increases with a growth of origination, which increases the size of data needs to be analyzed by IDS for intrusion detection. This huge amount of data cannot be processed by conventional IDS servers in real time. To solve this problem Big Data Analysis based IDS (BDA-IDS) are used. If size of data to be analyzed is not big then conventional IDS servers are used.

1.3.8 Hardware and Software IDS

IDS can be implemented as Software application as well as Hardware device. Online hardware IDS devices can detect intrusions in real time; however, due to limited memory and computing power they have following lacunas:

i. Under heavy network traffic Online Hardware IDS itself becomes a bottleneck
ii. Hardware IDS need to be upgraded at regular intervals, which is very expensive

On other hand Software IDS requires more time for intrusion detection as compared to Hardware IDS; however, this problem can be solved using Distributed and parallel processing.

1.4 Motivation

First Intrusion Detection System was proposed by Denning (1987). She analyzed the System's audit records using statistical techniques like threshold, standard deviation and multivariate model for Host based Intrusion Detection. Lunt and Jagannathan (1988) have proposed real-time Intrusion-Detection Expert System which learns the normal behavior of user and predicts the attacks. It is based on the assumption that attacks deviates from the normal user behavior.

Meng et al. (2014) have presented a design of signature detection based N-IDS using packet filtering mechanism. In this, payload of incoming packets is analyzed to detect the presence of attack. This process requires huge amount of recourses in terms of memory and processing power to detect attacks in the presence of very heavy network traffic and DOS attacks. Signature based Intrusion Detection Systems can detect known attacks with high accuracy however, lacks in detection of novel attacks.

Meng et al. (2013) have described adaptive character frequency-based exclusive signature matching scheme for design of Signature based Network Intrusion Detection System. This scheme analyzes the network packets payload for signature matching; thus requires huge resources under heavy network traffic.

Paoet al. (2013) have implemented hardware signature detection based N-IDS using memory-based Non-deterministic Finite Automaton (NFA) regular expression match engine. Hardware Intrusion Detection Systems helps to speed up the detection process; and, becomes a bottleneck when, amount of network traffic is increased beyond a limit making the attack more critical.

Zhang et al. (2015) have proposed an Adaptive Stream Projected Outlier deTector (A-SPOT) technique for design of A-IDS. This technique also works effectively for high dimensional dataset. In general, A-IDS can detect novel attacks with high accuracy; however, it generates high false alarm rate. Anomaly detection based IDS perform poor against the attacks which mimics the behavior of normal user (e.g. Back, Slow read, Slow write attacks).

Dangeloa et al. (2015) have presented an A-IDS using uncertainty-managing batch relevance-based approach. Li and feng Xiao (2015) have proposed A-IDS using dual-ant clustering algorithm. Garcia-Teodoro et al. (2015) have proposed application layer A-IDS which automatically generates signature of attacks against HTTP servers. It analyzes the payload of traffic and if detected as anomalous; then payload is used for generating the signatures of attack. Application layer Intrusion Detection Systems requires more processing resources as it has to analyze the complete payload of traffic, also requires other IDS for detecting attacks on lower layers.

Hansen and Salamon (1990) have presented first ensemble of classifiers approach to improve the classification accuracy using majority vote approach. Filippi et al. (1994) argued that, ensemble of classifier approach gives good classification performance even in the presence of imbalanced training dataset.

Zare Moodi et al. (2015) have described design of ensemble of ensemble of one-class classifiers (i.e. ensemble of ensemble of classifiers) to detect known as well as novel attacks. Output of ensemble of classifiers is merged using majority vote approach. In this approach if more than half of the classifiers predicts wrong class then predicted output is also wrong.

Yin et al. (2015) have proposed dynamic creation of ensemble of ensemble of one-class classifiers for designing Intrusion Detection System using weighted majority vote approach. This approach works well for detection of known attacks but performs poor to detect novel attacks which mimics the behavior of normal user or slightly deviates from old attacks. Li et al. (2015) have proposed random ensemble of decision tree approach for deign of IDS. It uses majority vote approach for merging the output of classifiers in ensemble.

Elbasiony et al. (2013) and Kim et al. (2014) have presented design of hybrid Intrusion Detection System using combination of anomaly and signature detection

based IDS. In this approach, if network connection is not classified by S-IDS and detected as outlier by A-IDS, then it is considered as possible attack. This approach performs poor against attacks which mimic the behavior of normal user and detects deviation of normal user from regular profile as possible intrusion.

1.4.1 Limitations of Existing Approaches

Lot of research has been done over two decades for detection of various attacks. However, there is no generalized solution available for detection of attacks as diversity and volume of attacks are increasing day by day. The following section summarizes limitations of the systems suggested by various researchers.

- Signature based IDS is good at detection of known attacks as compared to novel attacks.
- Anomaly based Intrusion Detection System detects novel attacks; however, it generates high false positive rate.
- Hybrid (Anomaly-Signature based) Intrusion Detection System detects known as well as novel attacks; however, deviation from normal user profile is considered as possible intrusion; which is not always true.
- Payload analysis based IDS can detect attacks with high accuracy, however, analysis of entire payload of every packet requires huge memory and processing power. This requirement becomes critical for DoS attacks.
- Application Layer IDS detects attacks against particular service(s) with high accuracy. It requires analysis of entire payload and additional IDS for detecting attacks on lower layers.
- Hardware Intrusion Detection System can provide real time intrusion detection, however, it needs to be upgraded at regular interval, which makes it very expensive solution.
- Majority vote approach for Ensemble of classifiers gives better results for balanced dataset and performs poor against imbalance dataset.
- Weighted vote perform poor if the novel attack behaves like normal user or slightly deviates from old attacks.

1.5 Objectives and Problem Statement

Based on the literature survey and the research carried out by various researchers and the limitations there on, there is still need of IDS system which can address the issues

related to IDS. The main objectives of this research work are summarized as below:

i. To study and implement existing IDS systems
ii. To create real time traffic and develop a new dataset representing attacks which are not present in standard datasets
iii. To design a header analysis based IDS system for detection of Known DoS and DDoS attacks using Adaptive Ensemble of Nave Bayesian Classifiers
iv. To design a header analysis based Hybrid (Anomaly-Signature based) IDS for detection of Known as well as Novel DoS and DDoS attacks using Ensemble of Classifiers
v. To propose a lightweight system to detect Known as well as Novel DoS and DDoS attacks in Real-time using distributed processing.
vi. To propose a system to scale up/scale down the required infrastructure using virtual servers for real time Intrusion Detection based on the increase/decrease in the volume of network traffic.

Problem Statement

The nature of the network traffic is unpredictable. The network traffic datasets are voluminous, complex, heterogeneous, and of varying quality. There is no specific method to predict the intrusion. Different researchers have suggested different techniques to address the problem of intrusion detection.

After thorough understanding of the limitations of the existing systems, following problem has been identified.

"Design an Adaptive Ensemble of Classifiers based Hybrid (Anomaly-Signature detection based) Scalable Network Intrusion Detection System to detect Known and Novel Denial of Service and Distributed Denial of Service attacks in real time".

1.6 Major Contributions

The thesis has described methods and framework for detection of Known and Novel DoS and DDoS attacks. The major contributions of this thesis are:

i. A new dataset is prepared to represent HHTP Flood, Slow Read and Slow Write attacks which are not covered in standard datasets.
ii. Procedure to create an Adaptive Ensemble of Classifiers using Naive Bayesian classifier to detect Known DoS and DDoS attacks is described.

iii. Combination of Signature detection based IDS and Adaptive Ensemble of Classifiers is proposed to detect Known and Novel DOS and DDoS attacks.

iv. A framework to detection Known and Novel DOS and DDoS attacks in real-time using unutilized CPU cycles of computing machines within the organization is presented.

v. A methodology is proposed to process genuine flood of web user request and detect attacks in real time using virtual servers.

1.7 Organization of Thesis

Chapter 1 outlines the overview of IDS, need of IDS, types of attacks and IDS system, motivation for research work, problem statement, objectives and major contribution of the research work.

Chapter 2 presents the review of contributions made by researchers for design of A-IDS, S-IDS, Anomaly-Signature detection based IDS, H-IDS, N-IDS, Collaborative IDS, D-IDS, Hardware IDS, R-IDS, Big data analysis IDS and Ensemble of Classifiers based IDS.

Chapter 3 describes the analysis and limitations of KDD 99 and CDMC 2012 datasets used by researchers to evaluate the performance of IDS. It also describes the dataset created in laboratory to represent attacks not covered in standard datasets like KDD 99 and CDMC 2012.

Chapter 4 describes the process of creating Adaptive Ensemble of Classifiers for detection of Known DoS and DDoS attacks. It also presents the performance analysis of Adaptive Ensemble of Classifiers based N-IDS using KDD 999, CDMC 2012 and dataset created in laboratory.

Chapter 5 gives the process for combining S-IDS with Adaptive Ensemble of Classifiers based N-IDS to detect Known as well as Novel attacks. The performance of proposed combination is analyzed using KDD 999, CDMC 2012 and dataset created in laboratory.

Chapter 6 presents the framework for detecting DoS and DDoS attacks in real-time using unutilized CPU cycles of computing machines present within the organization. It also describes the experimental environment created in laboratory to evaluate the proposed framework and experimental results.

Chapter 7 describes the process to differentiate genuine flood of web requests from DDoS attacks and handle genuine flood of web requests in real time-using virtual servers. It also presents the experimental results and describes the experimental environment created in laboratory using virtual machines to evaluate the proposed system.

Chapter 8 summarizes the major contributions and also presents the overall conclusion of the thesis with suggestions for future work.

Chapter 2

Literature Survey

This chapter presents the brief review of contributions made by researchers over last two decades for design of A-IDS, S-IDS, Anomaly-Signature detection based IDS, H-IDS, N-IDS, Collaborative IDS, D-IDS, Hardware IDS, R-IDS, Big data analysis IDS and Ensemble of Classifiers based IDS.

2.1 Anomaly Detection based IDS

First A-IDS was presented by Denning (1987) to detect intrusions in Real time. Normal user behavior profile representing the user behavior with respect to other resources present in the system or over the network is used to detect the intrusions. It is created by applying various statistical analysis techniques like Mean and Standard Deviation, Multivariate Model, Markov Process Model, Time Series Model on six tuples namely user, action performed by user, target of action, exception condition, resource usage and time stamp present in Audit Records.

Kosoresow and Hofmeyr (1997) argued that sophisticated intruder may delete or alter the security audit trails after intrusive activity; so computationally efficient on-line intrusion detection mechanism is needed. They have demonstrated that real-time A-IDS can be created by analyzing the sequence of systems calls (generated by programs running on the machine) with the help of Deterministic Finite Automata.

Dickerson and Dickerson (2000) have proposed partially automated anomaly detection based N-IDS using fuzzy logic. Network traffic is aggregated to extract required features like number of connections from source to destination, connection status, number of packets exchanged, etc. Then these numeric features representing network connection state are fuzzified and passed as input to the Fuzzy Intrusion Recognition Engine to detect possible intrusions.

Tsai et al. (2003) have described protocol analysis and anomaly detection based N-IDS using finite automata. Network traffic converted into Connection Signature Pattern by protocol analysis is used for intrusion detection. Normal user behavior profile is created by integrating knowledge of domain expert and frequent Connection

Signature Patterns of normal user behavior discovered using SPIRIT sequential pattern mining algorithm. Finite automata matching are used to detect the deviation between normal user behavior and current user behavior.

Patcha and Park (2007) have proposed a design of anomaly detection based N-IDS for high speed network. Huge volume of network traffic is filtered using adaptive weighted packet sampling. This filtering process maintains the characteristics of network traffic. To extract the network traffic features; filtered traffic is aggregated based on destination IP and port address. These features are used to detect the anomalous user behavior.

Chen et al. (2010) have described a framework of anomaly detection based light weight N-IDS. It is based on a assumption that, initial packets of network connection carries sufficient information to detect attacks. Attributes are extracted from IP header, TCP header and payload to build a normal user behavior model.

Fiore et al. (2013) have proposed a design of anomaly detection based semi-supervised N-IDS using Discriminative Restricted Boltzmann Machine (DRBM). DRBM is trained using normal user behavior records present in the training file to generate normal user behavior model. This model is updated at regular interval to accommodate the change in user behavior over a period of time

Wanget al. (2014) have presented an anomaly detection based N-IDS using affinity propagation clustering algorithm. It analyzes the audit data stream and classifies the activities over network as normal activity, suspicious activity or attack. It continuously updates itself using network activities detected as normal to accommodate change in normal user behavior. Suspicious activities are further investigated for possible attacks using updated normal user behavior.

Erfani et al. (2016) have presented an approach for anomaly detection using Deep Belief Network and plane based one class Support Vector Machine. Deep belief network is used to select the most relevant features from training dataset and plane based one class Support Vector Machine is used to build the anomaly detection model using these features. They have argued that, this method is suitable for high dimensional large datasets processing applications.

2.2 Signature Detection based IDS

Christoph et al. (1995) have proposed a design of Distributed S-IDS using client-server architecture. Client program installed on each machine of the network periodically scans the audit trails for possible attacks and sends the result along with audit data to central server for further analysis and attack detection. Rules for S-IDS are developed using knowledge of domain experts and statistical analysis of audit logs which significantly deviates from normal user behavior. Rules for attack detection are categorized into three levels; Hour rules, Day rules and Week rules. These rules are applied at the end of predefined duration to detect the attacks.

Shieh and Gligor (1997) have described a pattern oriented model for implementation of S-IDS using State Transition Tables. S-IDS is trained using known sequences of systems calls used by attackers to intrude into the system. These sequences of system calls are represented as a State Transition Tables in S-IDS. User activities over the Unix system are matched with these sequences using State Transition Tables.

Xiang-Rong et al. (2001) have described the procedure to use 'Sequential Pattern Discovery using Equivalent classes' (i.e. SPADE) algorithm for design of S-IDS. SPADE algorithm is applied on training file containing patterns of normal user behavior and various attacks to identify the frequent sequential patterns. Then these frequent patterns are used as rules or signatures to detect known intrusions; however, this method fails against intrusions whose behavior is similar to normal user behavior.

Li et al. (2003) have proposed a design of S-IDS using Direct Hashing and Pruning algorithm which is a extension of apriori algorithm. In order to reduce the number of signatures; the signature generation process is divided into two phases. In first phase frequent patterns are identified using important attributes present in training file. Then frequent serial patterns are discovered from these patterns to use as attack and normal user pattern signatures.

Chavan et al. (2004) have presented a S-IDS using combination of Snort, Artificial Neural Network and fuzzy inference system. Attack signatures are discovered by applying Artificial Neural Network and fuzzy inference system on captured network traffic. These signatures are used by Snort to detect attacks in real time.

Wuu et al. (2007) have proposed a R-IDS using snort and pattern mining. Pattern discovery module identifies the single and sequential patterns from captured network

traffic and converts them into snort rules. These are then used for on-line intrusion detection by snort.

Mabu et al. (2010) have described the design of N-IDS using fuzzy set theory and genetic network programming. Combination of fuzzy set theory and genetic network programming is used to extract the attack and normal user behavior signatures from database containing continuous and discrete values.

Modia et al. (2012) have proposed a design of signature detection based N-IDS using combination of Snort and apriori algorithm. Apriori algorithm is used to generate the new rules from captured network traffic. These rules are added in the snort rule database to detect known attacks and variations of known attacks.

Meng et al. (2014) have presented a three stage approach for design of NPPA-IDS. In first stage list of blacklisted IP addresses is used for filtering the incoming packets, if their payload is matching with signatures of known attacks. If match is not found or packet does not belong to blacklisted IP addresses; then it is forwarded to N-IDS for further analysis. N-IDS is implemented using single character frequency based signature matching scheme. Intrusion detected by blacklisted IP address based filter and N-IDS are passed to false alarm filtering module implemented using KNN classifier to improve the false positive rate.

2.3 Hybrid (Anomaly-Signature) Detection based IDS

Endler (1998) proposed a IDS using combination of A-IDS and S-IDS to reduce false negative rate in detection of buffer overrun attacks. A-IDS is implemented using statistical analysis of user activities over the host and remote system as well as resources. S-IDS is implemented using Multi-Layer Perceptron. Audit records which are marked as anomalous by any of the IDS are considered as intrusive.

Depren et al. (2005) have proposed a design of hybrid N-IDS using combination of Self Organizing Map based A-IDS and C4.5 based S-IDS. To build a more precise normal user behavior model, A-IDS is implemented using different anomaly analyzer for TCP, UDP and ICMP protocols. Incoming network traffic is forwarded to corresponding anomaly analyzer for attack detection. Rule based decision support system is used to merge the output of S-IDS and S-IDS.

Zhang et al. (2008) have presented an Hybrid IDS using Random Forest based S-IDS and A-IDS. The design is based on a assumption that, attacks which uses less number

of connections are more harmful than attacks which uses more number of connections. S-IDS is build using network connection records representing various attacks and service based patterns discovered using Random Forest are used to build A-IDS. Network connection record which does not match with normal service patterns and known attack patterns is considered as a novel attack. They have used the concept of over sampling and under sampling to remove the class imbalance problem present in training dataset of IDS.

Catania et al. (2012) have proposed a design of N-IDS using combination of SNORT and A-IDS to detect novel attacks. SNORT is used to assign the class labels to records present in the unlabeled training dataset. Records labeled as 'Normal' are used to build the A-IDS using Support Vector Machine.

Kim et al. (2014) have presented a model of hybrid N-IDS using C4.5 based S-IDS and 1-Class Support Vector Machine based A-IDS. Initially rule based S-IDS is build using C4.5 algorithm; then these rules are used to divide the normal user connection records present in training dataset into subsets. These subsets are used to train different 1-Class Support Vector Machines. If attack is not detected by S-IDS; then network connection record is forwarded to the appropriate 1-Class Support Vector machine for further analysis.

2.4 IDS using Ensemble of Classifiers

Chebrolu et al. (2005) have presented a design of N-IDS using ensemble of Bayesian Network and Classification and Regression Trees to detect known attacks. Weights are assigned to both the classifiers based on their detection performance for individual classes. If output of both the classifiers is conflicting; these weights are used to take the final decision. They have also argued that different set of features should be used to detect different attacks.

Cabrera et al. (2008) have proposed a design of hierarchical distributed N-IDS using ensemble of C4.5 classifiers. Each machine present in the network detects the anomaly using C4.5 and sends the result to subnet head for further processing. Subnet heads aggregates the result of all the anomaly detectors and forwards to network manager for further processing. Network manager aggregates the results of all subnet heads to detect presence of attacks in the network. Anomaly detectors present on each machine are updated continuously to accommodate change in normal user behavior.

Govindarajan and Chandrasekaran (2011) have proposed an ensemble of Multilayer Perceptron and Radial Basis Function for design of IDS. Outputs of both the classifiers are merged using bagging algorithm.

Khreich et al. (2012) have described a design of Receiver Operating Characteristic (ROC) based ensemble of Hidden Markov Models (HMMs) for implementing A-IDS. Initially a set of HMMs are selected using ROC to form an ensemble of classifiers. When a new set of training data is available, set of HHMs are selected and merged into existing set of HHMs to update the ensemble.

Hu et al. (2014) have proposed a Dynamic DN-IDS using ensemble of ensemble of classifiers. Local Detection Model is present in each node of the system; these models are combined using Particle Swarm Optimization and Support Vector Machines to form a Global Detection Model. In order to adopt the change in normal user behavior and attack strategies; Global Detection Model is shared among all the nodes in the system and prediction results of Global Detection Model are used to update the Local Detection Model. Adaboost classifier is used to implement Local Detection Model and it is trained using a partial training set.

Aburomman and Reaz (2016) have described an ensemble of ensemble approach using Support Vector Machine, K-NN classifier and Particle Swarm Optimization. Support Vector Machine and K-NN classifier are used to create the binary base classifiers. Particle Swarm Optimization is used to generate the weight of each base classifier for each attack type. Outputs of base classifiers are merged using weighted majority voting approach.

2.5 Host based IDS

Lunt (1989) proposed a combination of A-IDS and rule based expert system to detect intrusions. Normal user behavior profile of users with respect to intrusive activity detection measures for system users, remote hosts and overall system is created by analyzing the audit records. These profiles are updated periodically to accommodate change in user behavior. Rule based expert system is created by defining the general rules for detecting possible intrusive activities. If user behavior is marked as intrusive by A-IDS or expert system it is considered as intrusion.

Liao and Vemuri (2002) have proposed H-IDS using K-NN classifier. Every executed process is represented in the form of vector and occurrence count of system

calls used by process represents the elements of vector. Occurrence count is calculated using Term Frequency-Inverse Term Frequency (TF-ITF) algorithm. These vectors are used to detect presence of attacks using K-NN classifier.

Peisert et al. (2007) have described a process of analyzing function calls to implement anomaly detection based H-IDS. They have argued that, sequence of function calls and function return points gives more information than sequence of system calls. Sequence of function calls and return points are analyzed to detect presence of unexpected events as well as absence of expected events.

Hu et al. (2009) have proposed a design of H-IDS using Hidden Markov Model. Sequence of system calls in training files is divided into sun sequences and used to train sub-Hidden Markov Models. These sub models are merged incrementally using weighted average algorithm to obtain the final model for attack detection.

Maggi et al. (2010) have presented a design of anomaly detection based H-IDS by analyzing sequence of system calls and their arguments. Hierarchical agglomerative clustering algorithm is used to create the clusters of system calls and their arguments used by various processes. These clusters and Hidden Markov Model is used to build the anomaly detection model

Garca et al. (2012) have presented a framework of log file analysis based H-IDS using K-Hidden Markov Model. Repetition of related sequence calls is identified from the log files and used to discover the signatures of normal user behavior. This reduces the amount of data used to build the detection model and increases the speed of log file analysis for attack detection.

Creech and Hu (2014) have proposed host dependent H-IDS using continuous and discontinuous kernel level system call patterns. Each system call is represented as a character and sequence of system calls are used to form the words. Context Free Grammar is used to generate the phrases using these words. These phrases are used to represent the normal user behavior. Any significant deviation between current sequence of system calls and known phrases is considered as an attack.

2.6 Network based IDS

Jiang et al. (2006) have described a design of N-IDS for detecting known as well as novel attacks. Clustering is used to partition the labeled training data. Label is

assigned to each partition by analyzing the labels of records present in it. Improved Nearest Neighbor algorithm and labeled partitions are used to detect the attacks.

Bankovic et al. (2007) have proposed a signature detection based N-IDS using Genetic Algorithm. Most relevant features for attack detection are selected from training file using Principal Component Analysis. These features are concatenated using 'AND' operator to generate initial set of attack signatures. Genetic Algorithm is applied on these signatures to obtain the final set of attack signatures.

Su et al. (2009) have presented a incremental fuzzy rule mining process for designing real time N-IDS. During training phase, attack free traffic is converted into network connection records using two second window. These records are used to generate the fuzzy rule set for anomaly detection. Real time network traffic is analyzed using this rule set. Change in normal user behavior is accommodated by updating this rule set using detected attack free records.

Wang et al. (2010) have proposed a framework of N-IDS using fuzzy clustering and Artificial Neural Network. Training data is divided into subsets using fuzzy clustering approach. Each subset is used to train one instance of Artificial Neural Network. Prediction of all Artificial Neural Networks is aggregated by Fuzzy aggregation module for attack detection.

Xu-sheng et al. (2013) have presented design of Core Vector Machine based N-IDS. It is based on an assumption that; attributes used to represent network traffic are correlated and it introduces a noise in training data. Partial Least Square method is applied on training data to remove this noise and select the most relevant features. Anomaly detection model is built using Core Vector Machine.

La Hoz et al. (2015) have proposed an anomaly detection based N-IDS using Self Organizing Map and Gaussian Mixture Model. Principal Component Analysis and Fisher Discriminant Ratio is used to select the relevant features from the training file. Self-organizing process is applied on these features to generate attack detection prototypes. These prototypes are modeled using Gaussian Mixture Models for incremental training of IDS.

2.7 Collaborative IDS

Miller and Inoue (2003) have proposed a C-IDS using distributed agents working in autonomous and asynchronous manner. These agents are implemented using self-

organizing maps. Each agent analyzes the network traffic and sends the analysis result to the central decision making unit. Central decision unit uses weighted voting approach to merge these results. Weights of each agent are updated continuously based on its detection precision.

Zhang et al. (2003) have presented a design of C-IDS using hierarchical co-ordination agents. Detection agents analyze the specified object or system for possible attacks. If it detects any attack or unable to interpret the system's or object's state, it sends the response to the collaboration agent. Collaboration agent uses this information for decision making. If it is not able to take the decision, it forwards this information to the collaboration agent present at higher level of hierarchy.

Kemmerer and Vigna (2005) have proposed scalable collaborative N-IDS. For in-depth and stateful analysis, the network traffic in divided into smaller partitions and these partitions are analyzed by different sensors. Every sensor is configured to detect a subset of known attacks. Sensors can be added into and removed from the system depending on the network traffic volume and required throughput.

Safaa et al. (2008) have described a working of collaborative N-IDS for SYN flood attack detection and prevention. Networks collaborate with each other using collaboration protocol. Traffic originating from and destined to the network is monitored for attack detection and this information is shared with other networks in the collaboration for attack detection and better bandwidth utilization.

Franois et al. (2012) have proposed a design of collaborative intrusion detection process using overlay network of virtual rings. Host can become a part of this collaborative network by using subscription protocol. Network traffic of all participating hosts is monitored using window based statistical analysis techniques for anomaly detection. Multiple levels of collaborative filtering is used to improve the detection rate and reduce the processing power requirement.

Wang et al. (2014) have described a process to accelerate string matching for design of S-IDS using collaborative finite state machines. 'K' number of finite state machines stored in separate memory are used in parallel to match 'K' number of characters at the same time. Value of 'K' is decided based on application and network traffic volume.

2.8 Distributed IDS

Ouyang et al. (2002) have described a design of DN-IDS using hierarchical Fuzzy Detection Engines. Each fuzzy detection engine monitors one subnet and converts the network traffic into network connection records. Different weights are assigned to each attributes of the connection record according to its contribution in attack detection. Fuzzy detection engine uses these weights and network connection records to classify activities over network as attack, normal or suspicious. The result of detection process id forwarded to the central fuzzy detection engine for further analysis.

Gaspary et al. (2005) have presented a framework of hierarchical distributed N-IDS for detection of known attacks. Monitoring agents present in the subnet analyzes the network traffic and sends the occurrence count of attack signatures to the middle level manager. Middle level manager analyzes the signature counts received from all monitoring agents for attack detection.

Peng et al. (2007) have proposed a mechanism for information sharing in scalable distributed N-IDS. Network is divided into subnets and each subnet is monitored by one anomaly detection based N-IDS. Every N-IDS uses Cumulative Sum (CUMSUM) approach to detect anomalies and shares this information with N-IDS present in other subnets for further processing. N-IDS aggregates the information received from other N-IDSs to make final prediction.

Zhang et al. (2011) have described a design of hierarchical distributed N-IDS. The network is organized into three levels namely home area network, neighboring area network and wide area network. Data segmentation module segments the information collected by information acquisition module by monitoring the home network. This information is used for attack detection using machine learning classifier. Information related to the attacks which cannot be detected at home area network is forwarded to intrusion detection module present at neighboring area network and so on.

Fung et al. (2012) have proposed a framework for implementation of D-IDS using collaborative H-IDSs. When H-IDS observe suspicious activities which it cannot classify; it sends information about these activities to the neighboring H-IDSs present in its collaborators list for their opinion. It calculates the false positive and false

negative rates of their opinions using Bayesian learning and uses Bayesian decision model to aggregates these opinions.

Mehmood et al. (2015) have presented a framework of D-IDS using H-IDS and mobile agents for detecting distributed attacks. Information of attacks detected by H-IDS is transferred to the central server by using mobile agents. Central server correlated the attacks information received from different H-IDSs to detect the presence of distributed attacks in the network.

2.9 Hardware IDS

Hutchings et al. (2002) have proposed a design of reconfigurable N-IDS using Field Programmable Gate Array. Attack signatures are represented using non-deterministic finite automata and attacks are detected using string matching with the help of programmed hardware to speed the detection process.

Schuehler et al. (2004) have presented an design of Field-Programmable Port Extender based TCP/IP content processing system for building high speed N-IDS and NPPA-IDS. Payload of TCP/IP packets is analyzed using regular expression for detection of attacks and monitoring the current network state.

Baker and Prasanna (2005) have proposed a design of hardware N-IDS using Field Programmable Gate Array. Attacks signatures are represented as a sequence of characters and attack are detected by matching the current network traffic with attack signatures using KnuthMorrisPrat algorithm. In order to improve the detection speed and reduce the memory requirement, KnuthMorrisPrat algorithm is modified to match two characters at a time.

Maci-Prez et al. (2011) have proposed a DN-IDS using autonomous N-IDS embedded into smart sensor nodes. Autonomous N-IDS is implemented using self-organizing map. Every N-IDS analyzes the TCP/IP traffic in promiscuous mode and reports the detected anomalies to the central service in asynchronous mode.

Pontarelli et al. (2013) have described a design of high speed signature detection based N-IDS using Field-Programmable Gate Array. N-IDS is implemented using functionally equivalent hardware blocks programmed to detect complete set of attack signatures. To speed up the detection process, network traffic is grouped according to the destination service and forwarded to different blocks for attack detection.

Cronin and Wang (2013) have presented a design of signature detection based NPPA-IDS. To speed up the repetitive regular expression matching process of attack signature detection; IDS is implemented using Glushkov Non-Deterministic Finite Automata based on counting Bit-Parallel architecture.

Erdem (2016) has proposed a enhanced tree-based pattern matching mechanism consisting of many binary search trees using Field Programmable Gate Arrays for designing N-IDS. Each binary search tree is configured to represent a fixed width pattern and stored on a separate Static RAM based pipeline.

2.10 Real-time IDS

Jiang et al. (2005) have described a design of real time scalable N-IDS for high speed networks using flow-based dynamic load balancing scheme. N-IDS is consisting of multiple analyzers trained to detect known attacks. Load balancer continuously monitors the load of each analyzer and new network connection is forwarded to the least loaded analyzer for analysis.

Eric Y. K. Chan and Ju (2006) have proposed a design of R-IDS using Bloom filter, leaky bucket algorithm and adaptive threshold for detection of DoS and DDoS attacks. Bloom filter and leaky bucket algorithm is used to monitor the network traffic and heavy flows. Network traffic volume varies from time to time; thus to differentiate the legitimate traffic from DoS and DDoS attacks an adaptive threshold is used.

Das et al. (2008) have described a design of anomaly detection based on-line N-IDS using Principal Component Analysis. Features for attack detection are extracted from protocols header and payload section at data link layer. Principal Component Analysis is used for dimensionality reduction of feature set and outlier detection. It is implemented on Field Programmable Gate Array using RocketIO multi-gigabit transceivers.

Jamdagni et al. (2013) have proposed a real time NPPA-IDS using Principal Component Analysis and Mahalanobis Distance Map. N-gram text categorization is used to extract the features from network packet payload. These features are processed using Principal Component Analysis based iterative feature selection engine to extract most significant features. Mahalanobis distance map is used to build the normal user behavior profile using selected features.

Bulajoul et al. (2015) have proposed a design of R-IDS using Cisco catalyst switches and parallel Snort N-IDS. Four queues are created and network traffic is forwarded to one of the queue using Quality of Service parameters. These queues are processed using filtering criterions specified in access control list and packets in queue are forwarded to corresponding Snort for further processing.

2.11 Big Data IDS

Singh et al. (2014) have described a design of anomaly detection based scalable BDA-IDS using Hadoop, Hive, Mahout and Tshark to detect botnet based attacks in quasi real time. Tshark is used to extract the required features from captured network traffic. These features are loaded into the Hadoop file system using Hive. Anomaly detection based IDS is implemented using Random Forest of 100 trees.

Mylavarapu et al. (2015) have proposed a combination of S-IDS and A-IDS for detecting known and novel attacks in real time using apache storm framework. S-IDS is implemented using Multi-Layer Perceptron and A-IDS is implemented using CC4 instantaneous neural network.

Bandre and Nandimath (2015) have presented a framework of N-IDS using Hadoop and General Purpose Graphical Processing Unit (GPGPU). Huge volume of network traffic data is proposed using Flume, Pig, Hive and HBase installed on Hadoop. This processed data is analyzed for intrusion detection using Pattern Frequency-Inverse Cluster Frequency (PF-ICF) approach. PF-ICF is implemented on GPGPU to improve the speed of intrusion detection.

Huang et al. (2016) have described a design of ensemble of Online Sequential Extreme Learning Machines (OS-ELM) and demonstrated its effectiveness for intrusion detection process. They have argued that, training phase of OS-ELM is independent of other OS-ELMs in ensemble and proposed a method for parallel training and testing of OS-ELM using map-reduce framework.

2.12 Summary

This chapter briefly describes the research carried out in the field of Intrusion Detection. N-IDS can effectively detect the Intrusions by analyzing the traffic of entire network; however, it requires huge amount of resources to process the traffic of entire network in real-time. Hardware IDS can process huge volume of network traffic in real-time; however, it needs to be upgraded at regular time interval which is

very expensive process. Distributed IDS can process huge volume of traffic in small amount of time period; whereas Ensemble of Classifiers approach improves the detection performance of IDS. Distributed N-IDS designed using Ensemble of Classifiers can detect the Intrusions in real-time.

Chapter 3

Test Bed for Intrusion Detection System

Performance of self-learning and fully automated system developed using techniques like machine learning and data mining depends on their training dataset. If training dataset is biased or incomplete, these systems generate undesired response in some conditions. To avoid these incidence self-learning automated IDS systems must be tested in all respect before deploying them in real world environment. To solve this problem Defense Advance Research Project Agency (DARPA) and Air Force Research laboratory (AFRL) took the initiative in 1998 and developed training and testing datasets to evaluate the performance of IDS. These datasets were further processed and converted into network connection records by Special Interest Group on Knowledge Discovery in Data (SIGKDD) in 1999 to generate ready to use training and testing datasets for Data Mining and Data Analytics based IDS. Then Cyber Security and Data Mining Competition (CDMC) developed new training and testing datasets in 2010 to reflect recent intrusive patterns and network infrastructure.

This Chapter presents the analysis and limitations of KDD 99 and CDMC 2012 datasets used by researchers to evaluate the performance of IDS. It also describes the dataset created in laboratory to represent attacks not covered in standard datasets like KDD 99 and CDMC 2012.

3.1 DARPA 1998 Dataset

DARPA 1998 dataset was prepared by MIT Lincoln Labs under the sponsorship of DAARPA and AFRL with an objective to survey and evaluate research in Intrusion detection (Mchugh J., 2000). Figure 3.1 shows the network setup used for simulation of U.S. Air force LAN. Many attacks were simulated in the environment and nine weeks of TCP dump data is captured which can be used for evaluation of all types of IDS as payload as well as header information is present in TCP dump file. It can be observed from Figure 3.1 that flat network topology is used with only two physical subnets to simulate the U.S. Air force LAN.

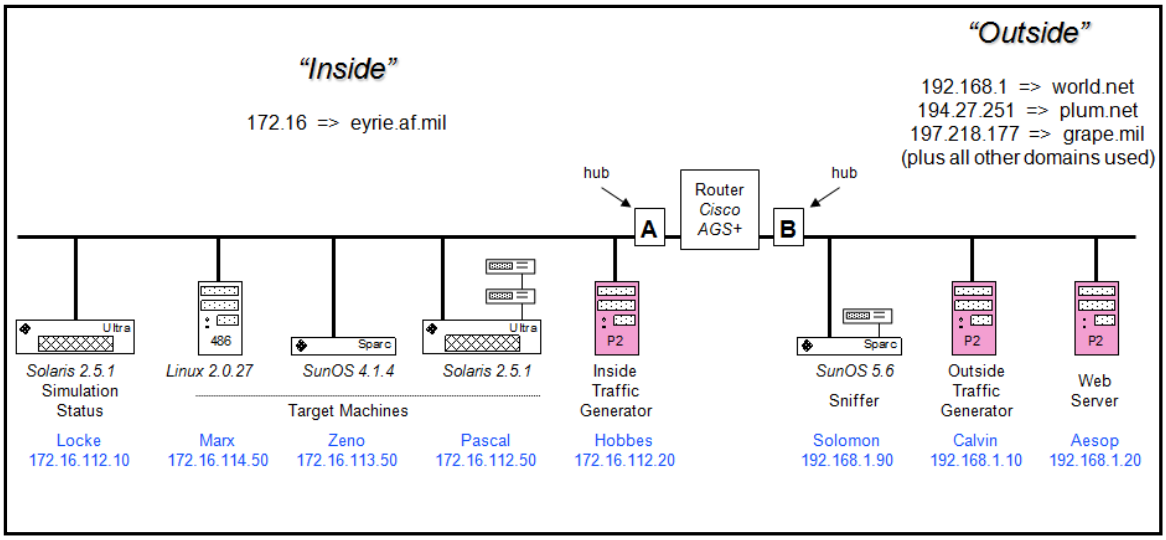


Figure 3.1: Flat Network Topology With Only Two Physical Subnets Used to Simulate the U.S. Air Force LAN

Network traffic representing following DoS attacks is present in DARPA 1998 dataset:

- Apache2: Intruder sends HTTP requests containing very high number of HTTP headers.
- Back: Intruder sends URL to the server by embedding many front slashes in it.
- Land: Intruder sends an IP packet with same Source IP address and Destination IPaddress. It is only effective against old systems.
- Mailbomb: Intruder transmits a huge flood of messages to the server in a short time period to overflow the servers queue.
- SYN Flood (Neptune): Intruder establishes many half-open TCP connections with target to exhaust its connection pool.
- Ping of Death: Intruder sends oversized IP packet to the target system.
- Processtable: Intruder opens many connections with target UNIX system in order to completely fill its process table.
- Smurf: Intruder sends a flood of ICMP echo reply packets to the target with the help of spoofed ICMP echo request packets.
- Teardrop: Intruder sends mal-fragmented packets to the target system to crash it.
- UDP storm: Intruder creates a huge flood of UDP packets between echo ports of two machines present in the network.

Lacunas of DARPA dataset are highlighted by many researchers. These are:

i. DARPA dataset is never validated and it differs from real network traffic (McHugh 2000).

ii. It is an irregular dataset as Time to Live (TTL) for normal packets and intrusive packets is different (Mahoney and Chan 2003).

3.2 Data Discretization using Fuzzy Logic

To improve the learning efficiency of algorithms; the continuous input values are mapped into discrete bins. The continuous values are mapped into discrete bins using fuzzy logic with the help of triangular membership functions shown in Figure 3.2. The data available is mapped into 2-bins to 20-bins. Accordingly, 2 to 20 triangular membership functions are created to discretize the data into 2-bins to 20-bins.

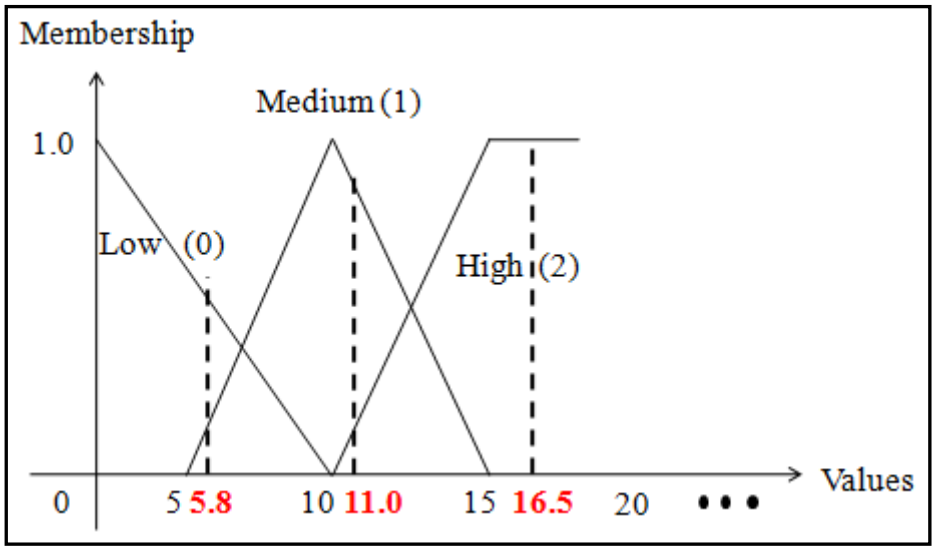


Figure 3.2: Example: Triangular Membership Functions for Fuzzy Discretization into 3-Bins

Figure 3.2 shows triangular membership function for fuzzy discretization of a continuous valued function into 3-bins. Example: Consider values 5.8, 11.0 and 16.5. Figure 3.2 shows that the Membership (μ) of value 5.8 is 0.72 for low; 0.16 for medium and 0.0 for high. Therefore, value 5.8 is discretized as 0 (low) and replaced by 0. Value 11.0 will be discretized as 1 (medium) and 16.5 will be discretized as 2 (high). For N-bins, N-triangular membership functions are created and continuous values are discretized into 'N' levels.

3.3 KDD 99 Dataset

KDD 99 is a version of DARPA 1998 dataset (Qu G. et al., 2005). Seven weeks of raw training data is processed into about five million connection records, which

contains 24 different attacks. Two weeks of test data is processed into around two million connection records, which contains 38 (additional 14) different attacks. Every connection record contains 41 features and is labeled as Normal or Intrusive. If a connection record represents an intrusive behavior; its label specifies the intrusion name. The Features of KDD 99 dataset are categorized as Basic features, Traffic features and Content features.

- Basic Features: These features are extracted from the individual TCP/IP connection established over the network. e.g. Number of bytes transferred from source to destination and vice versa, protocol used for communication, connection status, etc.
- Traffic Features: These features are extracted using a time and connection window to examine the status of simultaneous or subsequent connections to the same destination machine or service. e.g. connection to same service or destination, rejected connection, etc.
- Content Features: These features are extracted from the payload of the network packets. e.g. user login information, number of commands executed by users over network, etc..

Table 3.1 shows the list of selected 16 features out of 41 features to design the IDS systems proposed in this thesis.

Table 3.1: Selected 16 features for design of IDS systems proposed in the thesis

Sr. No.	Feature	Type	Sr. No.	Feature	Type
1	Protocol Type	Symbolic	9	Service Serror Rate	Continuous
2	Service	Symbolic	10	Rerror Rate	Continuous
3	Flag	Symbolic	11	Service Rerror Rate	Continuous
4	Source Bytes	Continuous	12	Destination Host Serror Rate	Continuous
5	Destination Bytes	Continuous	13	Destination Host Service Serror Rate	Continuous
6	Count	Continuous	14	Destination Host Rerror Rate	Continuous
7	Service Count	Continuous	15	Destination Host Service rerror Rate	Continuous
8	Serror Rate	Continuous	16	Class	Continuous

KDD 99 provides following three files containing connections records to train and evaluate the IDS:

- kddcup.data.gz: It contains complete dataset to train IDS (i.e. Training Dataset) kddcup.data 10 percent.gz: It contains 10% records of Training Dataset

- corrected.gz: It contains test dataset with labels (i.e. Testing Dataset)
- corrected.gz: It contains test dataset with labels (i.e. Testing Dataset)

Distribution of records in KDD 99 Training and Testing Dataset which representing normal user activity and DoS and DDoS attack is shown in Table 3.2. It can be observed from the table that, dataset is highly imbalanced.

Table 3.2: Distribution of Records in KDD 99 Training and Testing Dataset

Class	Training Dataset Records	Testing Dataset Records
Normal	9,72,781	60,593
Back	2,203	1,098
Pod	264	87
Smurf	28,07,886	1,64,091
Teardrop	979	12
Neptune	10,72,017	58,001
Land	21	9

Table 3.3 shows the Correlation between numeric attributes when KDD 99 training dataset is discretized into 3 bins using fuzzy triangular membership function. It is shown that, Attributes A4 and A5 are having maximum correlation of 0.79; whereas, correlation among other attributes is less than 0.6.

Table 3.3: Correlation Between Numeric Attributes of KDD 99 Training Dataset Discretized into 3-Bins

Attributes	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12
A1	1.00											
A2	-0.20	1.00										
A3	-0.09	0.26	1.00									
A4	-0.14	0.12	-0.04	1.00								
A5	-0.11	0.14	-0.04	0.79	1.00							
A6	0.24	-0.39	0.05	-0.07	-0.12	1.00						
A7	-0.12	0.20	0.05	0.02	0.03	-0.26	1.00					
A8	0.14	-0.11	0.05	0.00	-0.04	0.22	-0.06	1.00				
A9	0.13	-0.15	0.07	0.02	0.00	0.31	-0.35	0.49	1.00			
A10	-0.02	-0.08	0.08	-0.07	-0.09	0.15	-0.26	0.11	0.02	1.00		
A11	-0.08	0.16	-0.01	0.46	0.40	-0.08	0.02	0.03	0.04	0.02	1.00	
A12	-0.10	0.13	-0.04	0.55	0.57	-0.10	-0.04	-0.03	0.05	-0.06	0.52	1.00

Correlation between numeric attributes of KDD 99 training dataset discretized into 20-bins is shown in Table 3.4. Correlation between Attributes A4 and A5 is 0.81. Correlation among remaining attributes is less than 0.7. This clearly indicates that, numeric attributes of KDD 99 dataset are not correlated.

Table 3.4: Correlation Between Numeric Attributes of KDD 99 Training Dataset Discretized into 20-Bins

Attributes	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12
A1	1.00											
A2	0.38	1.00										
A3	-0.13	0.46	1.00									
A4	-0.34	0.10	-0.08	1.00								
A5	-0.32	0.12	-0.07	0.81	1.00							
A6	0.40	-0.66	0.05	-0.06	-0.07	1.00						
A7	-0.32	0.52	0.16	0.08	-0.05	-0.58	1.00					
A8	0.29	-0.31	0.10	0.13	0.10	0.50	-0.31	1.00				
A9	0.28	-0.39	0.08	0.16	0.14	0.60	-0.53	0.68	1.00			
A10	-0.07	-0.16	0.10	-0.06	-0.08	0.24	-0.29	0.02	0.16	1.00		
A11	-0.21	0.11	-0.05	0.50	0.65	-0.03	-0.11	0.16	0.17	-0.05	1.00	
A12	-0.17	0.10	-0.06	0.58	0.65	-0.03	-0.14	0.20	0.23	-0.09	0.64	1.00

Table 3.5: Average Values of Numeric Attributes of KDD 99 Training Records Discretized into 3-Bins

Attribute	Average						
	Back	Normal	Pod	Teardrop	Land	Neptune	Smurf
Attribute 1	1.00	0.57	0.00	0.00	0.00	0.01	0.00
Attribute 2	1.00	1.01	1.00	1.11	1.00	1.27	1.50
Attribute 3	1.00	1.01	1.00	1.00	1.00	1.00	1.50
Attribute 4	1.27	1.15	1.00	1.00	1.10	1.29	1.00
Attribute 5	1.45	1.14	1.00	1.00	1.00	1.29	1.00
Attribute 6	2.00	1.89	2.00	1.89	1.80	1.29	2.00
Attribute 7	1.48	1.43	1.12	1.89	1.00	1.87	1.88
Attribute 8	1.48	1.31	1.29	1.28	1.00	1.02	1.25
Attribute 9	2.00	1.51	1.71	1.33	1.70	1.06	1.63
Attribute 10	1.12	1.23	1.71	1.33	1.70	1.02	1.63
Attribute 11	1.15	1.15	1.00	1.61	1.00	1.27	1.38
Attribute 12	1.15	1.15	1.00	1.00	1.00	1.27	1.00

Average, standard deviation and median of numeric attributes when records of Training Dataset are discretized into 3-bins using triangular fuzzy membership function are shown in Tables 3.5, Table 3.6 and Table 3.7 respectively.

It can be observed from these tables that, average, standard deviation and median of attribute-2, attribute-3, attribute-7, attribute-8, attribute-11, attribute-12 of Back class, Normal class are similar and training file is dominated by records of Normal class

having 9,72,781 records and 2,203 records of Back class. Thus it is real challenge for IDS to differentiate records of Back class from Normal class.

Table 3.6: Standard Deviation Values of Numeric Attributes of KDD 99 Training Records Discretized into 3-Bins

Attribute	Std Dev						
	Back	Normal	Pod	Teardrop	Land	Neptune	Smurf
Attribute 1	0.00	0.50	0.00	0.00	0.00	0.07	0.00
Attribute 2	0.00	0.09	0.00	0.32	0.00	0.44	0.52
Attribute 3	0.00	0.08	0.00	0.00	0.00	0.00	0.52
Attribute 4	0.45	0.36	0.00	0.00	0.32	0.45	0.00
Attribute 5	0.51	0.35	0.00	0.00	0.00	0.45	0.00
Attribute 6	0.00	0.32	0.00	0.32	0.42	0.46	0.00
Attribute 7	0.51	0.50	0.33	0.32	0.00	0.33	0.34
Attribute 8	0.51	0.46	0.47	0.46	0.00	0.15	0.45
Attribute 9	0.00	0.50	0.47	0.49	0.48	0.25	0.50
Attribute 10	0.33	0.42	0.47	0.49	0.48	0.12	0.50
Attribute 11	0.36	0.35	0.00	0.50	0.00	0.45	0.50
Attribute 12	0.36	0.36	0.00	0.00	0.00	0.45	0.00

Table 3.7: Median Values of Numeric Attributes of KDD 99 Training Records Discretized into 3-Bins

Attribute	Median						
	Back	Normal	Pod	Teardrop	Land	Neptune	Smurf
Attribute 1	1.00	1.00	0.00	0.00	0.00	0.00	0.00
Attribute 2	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Attribute 3	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Attribute 4	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Attribute 5	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Attribute 6	2.00	2.00	2.00	2.00	2.00	1.00	2.00
Attribute 7	1.00	1.00	1.00	2.00	1.00	2.00	2.00
Attribute 8	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Attribute 9	2.00	2.00	2.00	1.00	2.00	1.00	2.00
Attribute 10	1.00	1.00	2.00	1.00	2.00	1.00	2.00
Attribute 11	1.00	1.00	1.00	2.00	1.00	1.00	1.00
Attribute 12	1.00	1.00	1.00	1.00	1.00	1.00	1.00

KDD 99 dataset provides following advantages over DARPA 1998 dataset:

i. No need to convert network traffic present in TCP dump file into relational structure
ii. Direct as well as derived features are readily available in dataset
iii. Requires less processing power and memory

Due to aforementioned advantages of KDD 99 over DARPA 1998 dataset, it has been extensively used by the many researchers for evaluating the performance of their proposed IDS. However, it has few Lacunas, these are:

i. Training dataset is a highly imbalanced and includes redundant records, thus learning algorithms are biased towards more frequent class labels and fails to learn less patterns of less frequented class labels.
ii. It cannot be used to evaluate performance of payload analysis based IDS.

3.4 CDMC 2012 Dataset

CDMC 2012 dataset is a real traffic data collected from several types of honeypots and a mail server over 5 different networks inside and outside of Kyoto University (Chen et al., 2014). The dataset is composed of 14 features including label information which indicates whether each session is attack or not. Distribution of records in CDMC 2012 Training and Testing dataset are shown in Table 3.8. It can be observed that, dataset is highly imbalanced.

Table 3.8: Distribution of Records in CDMC 2012 Training and Testing Dataset

Class	Training Dataset Records	Testing Dataset Records
1	71,046	57,155
-1	57,310	70,053
-2	364	149

Table 3.9: Correlation Between Numeric Attributes of CDMC 2012 Training Dataset Discretized into 3-Bins

Attributes	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13
A1	1.00												
A2	-0.01	1.00											
A3	0.00	-0.01	1.00										
A4	0.00	-0.01	0.17	1.00									
A5	0.00	-0.15	0.00	0.01	1.00								
A6	0.01	-0.68	0.01	0.01	0.18	1.00							
A7	0.00	0.03	0.00	0.00	-0.01	0.06	1.00						
A8	-0.01	0.52	0.00	0.00	-0.09	-0.45	0.12	1.00					
A9	0.00	-0.03	0.00	0.00	-0.03	-0.08	0.12	-0.02	1.00				
A10	0.01	-0.87	0.01	0.01	0.16	0.63	-0.04	-0.53	0.14	1.00			
A11	0.00	0.03	0.00	0.01	-0.04	-0.05	-0.01	-0.02	0.14	0.02	1.00		
A12	0.00	-0.02	0.00	0.00	-0.01	-0.04	0.22	0.03	0.24	0.00	0.04	1.00	
A13	-0.01	0.40	0.00	0.00	-0.07	-0.34	0.13	0.23	0.01	-0.39	-0.06	0.22	1.00

Table 3.9 shows the correlation between numeric attributes when training dataset is discretized into 3-bins using fuzzy triangular membership function. It is shown that correlation between attribute-2 and attribute-10 is -0.87; whereas, correlation among remaining attributes is less than 0.7. This clearly indicates that, numeric attributes are not correlated.

Correlation between numeric attributes when training dataset is discritized into 20-bins using fuzzy triangular membership function is shown in Table 3.10. It can be observed that, attribute-2 and attribute-10 are having negative correlation of -0.89 while correlation among remaining attributes is less than 0.7. This clearly indicates that, numeric attributes are not correlated.

Table 3.10: Correlation Between Numeric Attributes of CDMC 2012 Training Dataset Discretized Into 20-Bins

Attributes	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13
A1	1.00												
A2	-0.08	1.00											
A3	0.03	-0.01	1.00										
A4	0.00	-0.01	0.31	1.00									
A5	0.04	-0.40	0.01	0.01	1.00								
A6	0.08	-0.69	0.01	0.01	0.47	1.00							
A7	-0.01	0.05	0.00	0.00	-0.03	0.06	1.00						
A8	-0.06	0.61	-0.01	0.00	-0.28	-0.51	0.13	1.00					
A9	-0.01	-0.02	0.00	0.00	-0.03	0.07	0.10	-0.07	1.00				
A10	0.09	-0.89	0.01	0.01	0.41	0.67	-0.04	-0.63	0.21	1.00			
A11	-0.03	0.00	0.00	0.01	-0.08	-0.03	-0.01	-0.03	0.14	0.06	1.00		
A12	-0.01	0.02	0.00	0.00	-0.05	-0.06	0.22	0.01	0.37	0.02	0.03	1.00	
A13	-0.05	0.41	0.00	0.00	-0.18	-0.35	0.12	0.27	-0.06	-0.42	-0.07	0.19	1.00

Average, standard deviation and median of numeric attributes when records of CDMC 2012 training dataset are discretized into 3-bins using triangular fuzzy membership function are shown in Table 3.11, Table 3.12, and Table 3.13.

It can be observed from these tables that; average, standard deviation and median of attribute-1, attribute-3, attribute-4, attribute-5, attribute-6, attribute-7 of class -1 and class 2 are similar and training file is dominated by records of class -1 and class 1. Thus it is real challenge for IDS to differentiate records of class 2 from class -1.

Table 3.11: Average Values of Numeric Attributes of CDMC 2012 Training Records Discretized into 3-Bins

Attribute	Average		
	1	-1	-2
Attribute 1	1.0	1.0	1.0
Attribute 2	1.02	1.86	2.0
Attribute 3	1.0	1.0	1.0
Attribute 4	1.0	1.0	1.0
Attribute 5	1.06	1.0	1.0
Attribute 6	1.86	1.11	1.89
Attribute 7	1.0	1.01	1.0
Attribute 8	1.01	1.41	1.08
Attribute 9	1.01	1.05	1.0
Attribute 10	1.98	1.10	1.39
Attribute 11	1.04	1.04	1.0
Attribute 12	1.0	1.01	1.0
Attribute 13	1.0	1.24	1.02

Table 3.12: Standard Deviation Values of Numeric Attributes of CDMC 2012 Training Records Discretized into 3-Bins

Attribute	Std Dev		
	1	-1	-2
Attribute 1	0.02	0.01	0.00
Attribute 2	0.14	0.35	0.00
Attribute 3	0.01	0.00	0.00
Attribute 4	0.01	0.00	0.00
Attribute 5	0.24	0.00	0.00
Attribute 6	0.35	0.31	0.32
Attribute 7	0.01	0.10	0.00
Attribute 8	0.09	0.49	0.27
Attribute 9	0.08	0.22	0.00
Attribute 10	0.15	0.29	0.49
Attribute 11	0.20	0.19	0.00
Attribute 12	0.00	0.11	0.00
Attribute 13	0.00	0.43	0.13

KDD 99 dataset is generated in a simulated environment. It does not reflect the diversity of networking components which are present in today's networks. These limitations are addressed by CDMC 2012 dataset. As a result researchers have started using this dataset to evaluate performance of their IDS. However, this dataset has some drawbacks:

i. Information about attributes in dataset is not available.

ii. Name and Type of attack is not mentioned in dataset.

iii. Dataset does not contain attacks against variety of services.

iv. It cannot be used to evaluate performance of payload analysis based IDS.

Table 3.13: Median Values of Numeric Attributes of CDMC 2012 Training Records Discretized into 3-Bins

Attribute	Median		
	1	-1	-2
Attribute 1	1.0	1.0	1.0
Attribute 2	1.0	2.00	2.0
Attribute 3	1.0	1.0	1.0
Attribute 4	1.0	1.0	1.0
Attribute 5	1.0	1.0	1.0
Attribute 6	2.00	1.0	2.0
Attribute 7	1.0	1.0	1.0
Attribute 8	1.0	1.0	1.0
Attribute 9	1.0	1.0	1.0
Attribute 10	2.0	1.0	1.0
Attribute 11	1.0	1.0	1.0
Attribute 12	1.0	1.0	1.0
Attribute 13	1.0	1.0	1.0

3.5 Dataset Created in Laboratory

The KDD 99 dataset contains information about six DoS attacks. Many new attacks are created by attackers after preparation of KDD 99 dataset. Some of the commonly observed attacks are HTTP flood, Slow Read, Slow Write.

A setup was established as shown in Figure 3.3 to create training and testing dataset containing these attacks. These attacks were launched using java code and tools available in Kali Linux. Distribution of records in training and testing dataset created in laboratory is shown in Table 3.14. It can be observed that, dataset is highly imbalanced.

Table 3.14: Distribution of Records in Training and Testing Dataset Created in Laboratory

Class	Training Dataset Records	Testing Dataset Records
HTTP Flood	3,45,487	17,27,438
Slow Read	57,426	2,87,134
Slow Write	49,194	2,45,971
Normal	12,68,287	11,73,148

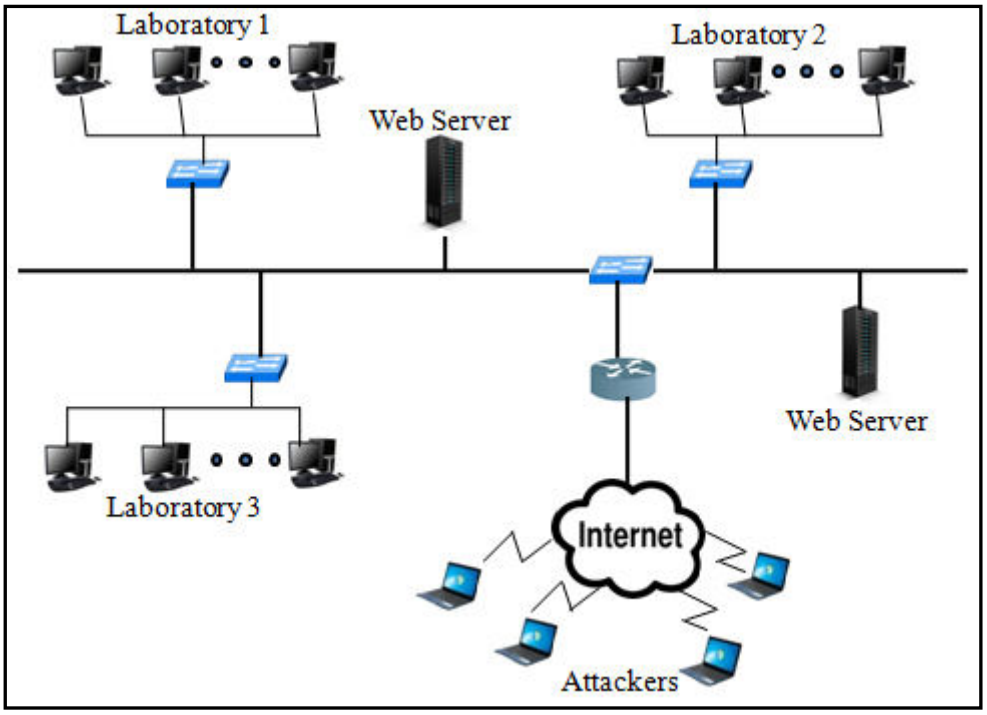


Figure 3.3: Experimental Setup to Create New Dataset Representing Attacks not Covered in Standard Dataset

The captured network traffic was converted into network traffic records containing sixteen features as shown in Table 3.1. Table 3.15 shows the correlation between numeric attributes when training dataset is discretized into 3-bins using fuzzy triangular membership function. It can be seen that correlation between attribute-4, attribute-5 is 0.77; whereas, correlation among remaining attributes is less than 0.7.

Table 3.15: Correlation Between Numeric Attributes of Training Dataset Created in Laboratory Discretized into 3-Bins

Attributes	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12
A1	1.00											
A2	-0.24	1.00										
A3	-0.13	0.62	1.00									
A4	-0.24	0.04	-0.10	1.00								
A5	-0.18	0.07	-0.09	0.77	1.00							
A6	0.21	-0.35	0.11	0.08	0.03	1.00						
A7	-0.20	0.32	0.12	-0.12	-0.07	-0.41	1.00					
A8	0.14	-0.14	0.08	0.16	0.09	0.38	-0.17	1.00				
A9	0.15	-0.20	0.06	0.20	0.14	0.49	-0.48	0.48	1.00			
A10	-0.10	-0.02	0.16	-0.06	-0.11	0.17	-0.25	0.07	0.28	1.00		
A11	-0.13	0.12	-0.04	0.61	0.53	0.06	-0.15	0.15	0.19	-0.01	1.00	
A12	-0.11	0.07	-0.10	0.62	0.61	0.04	-0.18	0.09	0.20	-0.06	0.59	1.00

Correlation between numeric attributes when training dataset is discretized into 20-bins using fuzzy triangular membership function is shown in Table 3.16. It can be observed that, Attribute-4 and attribute-5 are having positive correlation of 0.84 while correlation among remaining attributes is less than 0.7. This clearly indicates that, numeric attributes are not correlated.

Table 3.16: Correlation Between Numeric Attributes of Training Dataset Created in Laboratory Discretized into 20-Bins

Attributes	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12
A1	1.00											
A2	-0.51	1.00										
A3	-0.12	0.47	1.00									
A4	-0.38	0.47	-0.10	1.00								
A5	-0.36	0.45	-0.10	0.84	1.00							
A6	0.56	-0.68	0.16	-0.62	-0.40	1.00						
A7	-0.49	0.57	0.08	0.45	0.43	-0.66	1.00					
A8	0.53	-0.35	0.25	-0.39	-0.37	0.52	-0.27	1.00				
A9	0.59	-0.45	0.22	-0.48	-0.46	0.63	-0.57	0.61	1.00			
A10	-0.18	-0.04	0.32	-0.23	-0.23	0.27	-0.28	0.13	0.28	1.00		
A11	-0.39	0.49	-0.07	0.68	0.66	-0.43	0.37	-0.33	-0.42	-0.22	1.00	
A12	-0.37	0.46	-0.10	0.69	0.68	-0.42	0.33	-0.31	-0.48	-0.23	0.58	1.0

Average, standard deviation and median of numeric attributes when records of training dataset created in laboratory are discretized into 3-bins using triangular fuzzy membership function are shown in Table 3.17, Table 3.18, Table 3.19.

Table 3.17: Average Values of Numeric Attributes of Training Dataset Created in Laboratory Discretized into 3-Bins

Attribute	Average			
	HTTP Flood	Slow Read	Slow Write	Normal
Attribute 1	1.14	0.36	0.38	1.09
Attribute 2	1.28	0.41	0.49	1.13
Attribute 3	1.82	1.17	1.19	1.36
Attribute 4	1.82	1.17	1.18	1.36
Attribute 5	1.68	1.32	1.35	0.37
Attribute 6	1.69	1.31	1.35	0.39
Attribute 7	1.78	1.57	1.61	1.19
Attribute 8	1.77	1.58	1.62	1.20
Attribute 9	1.65	1.31	1.36	0.34
Attribute 10	1.65	1.31	1.35	0.36
Attribute 11	1.77	1.58	1.62	1.21
Attribute 12	1.78	1.58	1.62	1.21

Table 3.18: Standard Deviation Values of Numeric Attributes of Training Dataset Created in Laboratory Discretized into 3-Bins

Attribute	Std Dev			
	HTTP Flood	Slow Read	Slow Write	Normal
Attribute 1	0.35	0.61	0.65	0.67
Attribute 2	0.53	0.64	0.64	0.68
Attribute 3	0.44	0.67	0.71	0.67
Attribute 4	0.48	0.65	0.70	0.63
Attribute 5	0.60	0.66	0.66	0.65
Attribute 6	0.60	0.68	0.64	0.68
Attribute 7	0.48	0.73	0.69	0.69
Attribute 8	0.49	0.73	0.71	0.70
Attribute 9	0.57	0.68	0.70	0.67
Attribute 10	0.61	0.71	0.70	0.67
Attribute 11	0.49	0.71	0.69	0.69
Attribute 12	0.50	0.68	0.65	0.69

Table 3.19: Median Values of Numeric Attributes of Training Dataset Created in Laboratory Discretized into 3-Bins

Attribute	Median			
	HTTP Flood	Slow Read	Slow Write	Normal
Attribute 1	1.0	0.0	0.0	1.0
Attribute 2	1.0	0.0	0.0	1.0
Attribute 3	2.0	1.0	1.0	1.0
Attribute 4	2.0	1.0	1.0	1.0
Attribute 5	2.0	1.0	1.0	0.0
Attribute 6	2.0	1.0	1.0	0.0
Attribute 7	2.0	2.0	2.0	1.0
Attribute 8	2.0	2.0	2.0	1.0
Attribute 9	2.0	1.0	1.0	0.0
Attribute 10	2.0	1.0	1.0	0.0
Attribute 11	2.0	2.0	2.0	1.0
Attribute 12	2.0	2.0	2.0	1.0

It can be observed from Table 3.17, Table 3.18 and Table 3.19 that; average, standard deviation and median of attribute-1, attribute-2, attribute-3, attribute-4, attribute-5, attribute-6, attribute-7, attribute-8, attribute-9, attribute-10, attribute-11, attribute-12 of Slow Read class and Slow Write class are similar. Thus it is real challenge for IDS to differentiate records of Slow Read class from Slow Write class.

3.6 Summary

This chapter provides the analysis of dataset created in laboratory and standard datasets (KDD 99 and CDMC 2012) used to evaluate the performance of proposed IDS. KDD 99 dataset is used by many researchers and is a benchmark dataset for evaluation of IDS; however it does not reflect the recent network behavior and traffic characteristics. Some of the researchers have also used CDMC 2012 dataset to evaluate performance of their IDS. Dataset has also been created in library to represent the attacks which are not present in both the standard datasets.

Chapter 4

Known DoS and DDoS Attack Detection using Adaptive Ensemble of Classifiers

Attack which exploits known vulnerability of a system or network using known approach is called as known attack. Ensemble of Classifiers (EC) approach is widely used by researchers to improve the prediction accuracy of IDS. This chapter describes the process for creation of Adaptive Ensemble of Classifiers (AEC) to accommodate continuous changes in normal user behavior, DoS and DDoS attack strategies.

4.1 Introduction

DoS and DDoS attacks either consumes the resources available over the network or exploits the vulnerability present in the target system in order to make the network services or resources unavailable to legitimate users (Park et al., 2014).

Figure 4.1: 200-400 Gbps DDoS Attacks are Becoming a Normal Scenario

Infected or misconfigured machines present over the network, which can be used for launching attacks are called as zombie machines. With the growth in number of machines connected to the Internet, number of zombie machines is also increased. Attackers use these zombie machines to generate more than 100 Gbps traffic while launching DoS and DDoS attacks. Figure 4.1 shows report from krebsonsecurity which alerts that attackers can generate a 200-400 Gbps DDoS attacks using vulnerability present in old implementations of Network Time Protocol.

N-IDS can detect DoS and DDoS attacks by analyzing either header or entire payload of captured packets. Network Packet Header Analysis based IDS (NPHA-IDS) examine only header of captured packets to detect attacks, whereas, Network Packet Payload Analysis based IDS (NPPA-IDS) inspects the complete payload section of captured packets. NPPA-IDS require huge processing power and memory to detect attacks with a huge volume of traffic. It becomes very difficult for NPPA-IDS to examine entire payload of packets in real time when network traffic volume is increased beyond 20 Gbps and becomes almost impossible when it is increased beyond 100 Gbps. We proposed NPHA-IDS to detect DoS and DDoS attacks.

4.2 Ensemble of Classifiers and Adaptive Ensemble of Classifiers

Ensemble Classifier (EC) is a set of classifiers which predicts the final decision using predictions (i.e. votes) of classifiers present in the set. Majority voting and weighted voting are widely used approaches by EC for predicting the final decision (Yin et al., 2015). Each classifier present within the EC is called as Base Classifier (BC). Two approaches to build EC are:

i. Single BC Approach: EC is created using multiple instances of single BC trained using
    a. Multiple non-overlapping subsets of training dataset
    b. Different training datasets or inputs
ii. Multiple BC approach: EC is created using different BCs trained using same training dataset

Different versions of same training dataset can be created using various pre-processing methods. Discretization is widely used pre-processing method in which continuous numeric values are mapped into 'N' discrete bins. Discretization can be used to create different versions of training dataset by changing the number of bins.

EC is classified into categories as static or adaptive.

i. **Static EC:** BCs present within Static EC are fixed and does not change after training dataset is updated

ii. **AEC:** BCs present within AEC are changed to adapt the change in training pattern; after every update in training dataset.

Working of AEC is demonstrated using following examples. Figure 4.2 and Figure 4.3 shows how AEC works in two different situations.

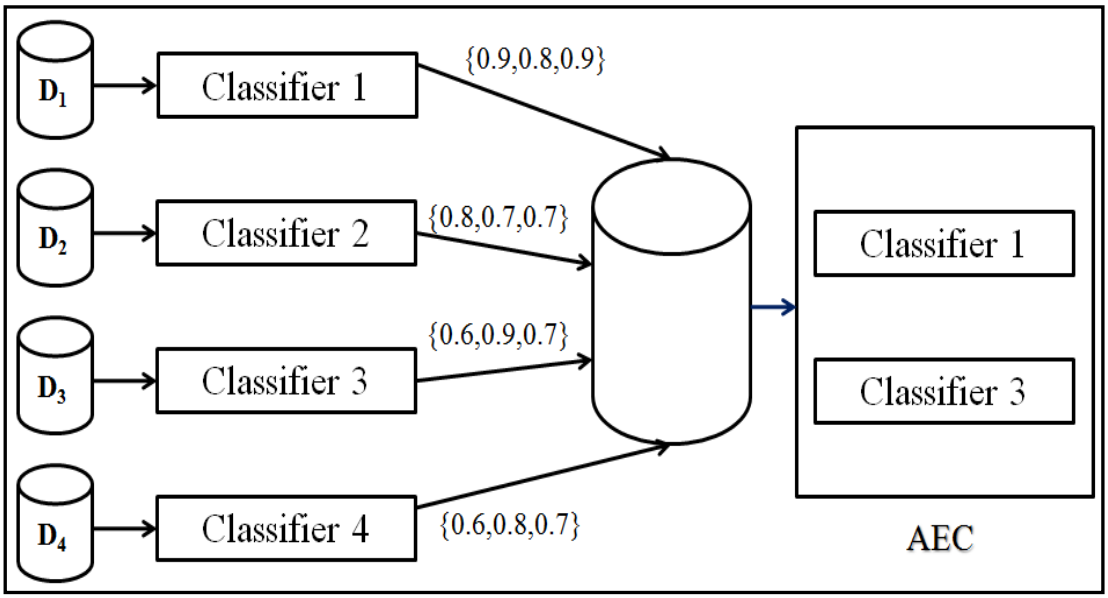


Figure 4.2: Adaptive Ensemble of classifiers: Example 1

Example 1: Let, D be a training dataset containing records belonging to three different classes; class-1, class-2 and class-3. Datasets D_1, D_2, D_3 and D_4 are derived from D by discretizing it into 2-bins, 3-bins, 4-bins and 5-bins respectively.Classifier-1, Classifier-2, Classifier-3 and Classifier-4 shown in Figure 4.2 are trained and evaluated using D_1, D_2, $D_3$and D_4 respectively.

Assume that each classifier classifies the records with certain true positive probabilities as shown in Figure 4.2. As shown in Figure 4.2 Classifier-1 predicts records of class-1, class-2 and class-3 with probability of true positive prediction of 0.9, 0.8 and 0.9 respectively. Similarly, Classifier-2 predicts records of class-1, class-2 and class-3 with probability of true positive prediction of 0.8, 0.7 and 0.7 respectively and so on. Classifier-1 has highest probability of true positive prediction for class-1 and class-3 compared to other classifiers and Classifier-3 has highest probability of true positive prediction for class-2 compared to other classifiers. Thus,

Classifier-1 and Classifier-3 are considered for creating AEC.

Example 2: Let the original dataset D updated and new datasets D_1', D_2', D_3' and D_4'shown in Figure 4.3 are derived from updated dataset D by discretizing it into 2-bins, 3-bins, 4-bins and 5-bins respectively. Classifier-1, Classifier-2, Classifier-3 and Classifier -4 are re-trained and evaluated using updated datasets D_1', D_2', D_3' and D_4' respectively.

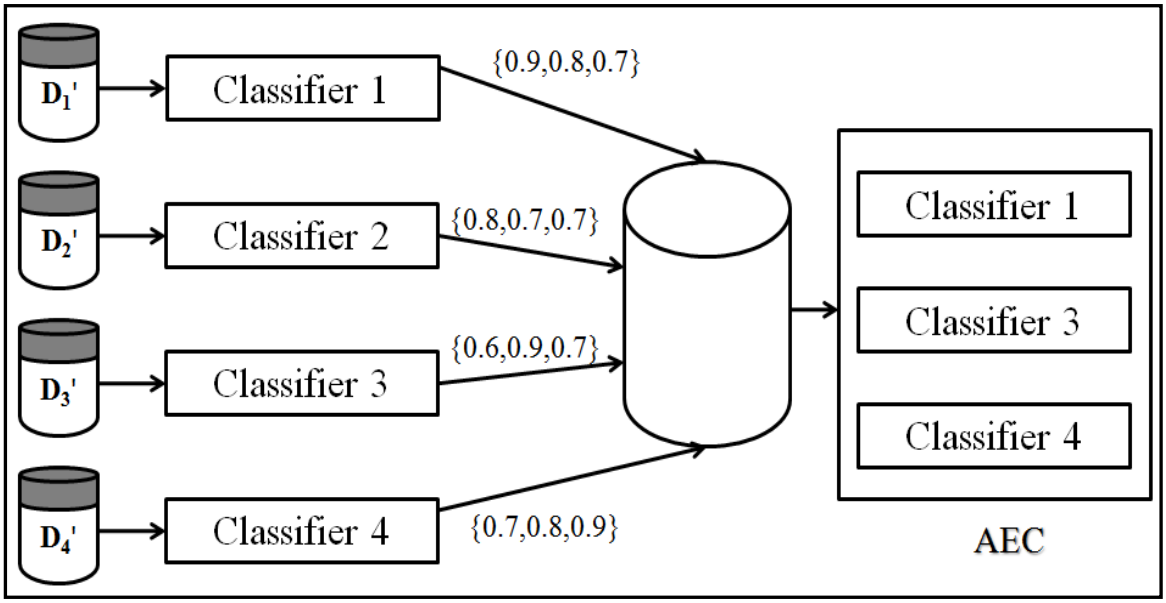


Figure 4.3: Adaptive Ensemble of classifiers: Example 2

Assume that each classifier classifies the new records with certain true positive probabilities as shown in Figure 4.3. After re-training these classifiers on new dataset say Classifier-1 predicts records of class-1, class-2 and class-3 with probability of true positive prediction of 0.9, 0.8 and 0.7 respectively.

Similarly Classifier-2 predicts records of class-1, class-2 and class-3 with probability of true positive prediction of 0.8, 0.7 and 0.7 respectively and so on. It is observed that Classifier-1, Classifier-3 and Classifier-4 have highest probability of true positive prediction for class-1, class-2 and class-3 respectively compared to other classifiers. Thus, Classifier-1, Classifier-3 and Classifier-4 are selected to create a new AEC after updation of training dataset D.

It is observed from above examples shown in Figure 4.2 and Figure 4.3 that, BCs within AEC are adaptively selected to accommodate changes in training dataset D.

4.3 Proposed Methodology for Detection of Known DoS and DDoS Attacks using AEC

Figure 4.4 shows the proposed methodology for detection of known DoS and DDoS attacks using AEC. The BCs within AEC are adaptively selected depending on the changes in normal user behavior, DoS and DDoS attack strategies.

Feature selection is the first step in machine learning and data mining based IDS to select only relevant and important features from training dataset. Information gain of attributes is used to select the important features from dataset.

Benefits of feature selection process are:

i. Reduced dimensionality of the dataset
ii. Reduced processing power and memory requirements
iii. It improves the quality of training dataset

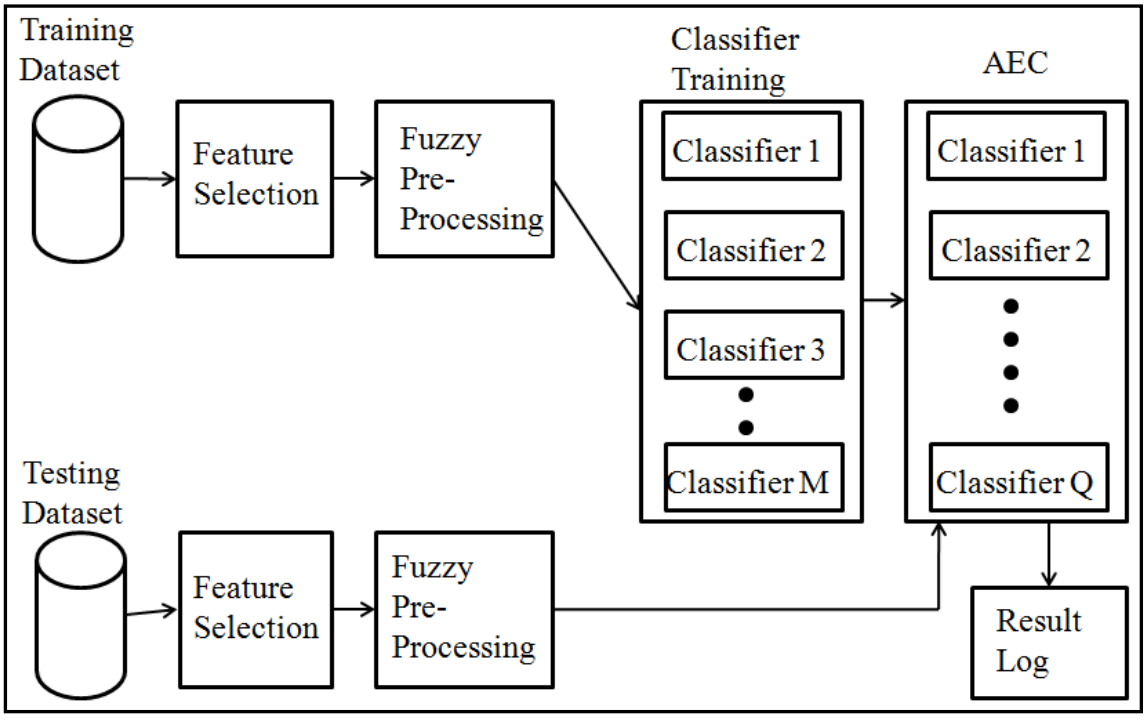


Figure 4.4: Methodology for Detection of Known DoS Attacks using Ensemble of Classifiers

Continues valued data is then mapped into discrete bins using Fuzzy Pre-Processing. Figure 4.5 shows the fuzzy preprocessing algorithm for conversion of continuous values of dataset into 'N' discrete bins using triangular membership function of fuzzy

logic. Number of discretization bins 'N' is equal to number of linguistic values used by fuzzy preprocessing algorithm. Set of linguistic values :

L = {Low, High} for N = 2,

L = {Low, Mid, High} for N = 3,

L = {Low, Mid, High, Very High} for N = 4 and so on.

Step-2 converts the continuous values of records present in dataset into linguistic value (i.e. discrete values) and adds the converted record into fuzzy dataset (FD_N).

Algorithm: Fuzzy Preprocessing
Input: KddTrD – KDD99 Training dataset
A – Set of attributes $\{A_1, A_2, \ldots, A_n\}$
L – Set of Linguistic values
N – Number of Linguistic values
Output: $FTrD_N$ – Fuzzy Training Dataset, FD_N – Fuzzy Dataset
Steps:
Step 1: Initialize FD_N = NULL
Step 2: For each record R_j in dataset KddTrD
For each attribute A_i in Record R_j do
If (A_i is a numeric attribute) Then
Convert numeric value of A_i into Linguistic value
Add Linguistic value of A_i to Fuzzy Record (i.e. FR_j)
Else
Add Value of A_i to FR_j
End If
End For
Add Fuzzy record FR_j to Fuzzy Dataset (i.e. FD); $FD_N \leftarrow FD_N \cup R_j$
Step 3: Initialize $FTrD_N$ = NULL
Step 4: For each row R_j in FD do
If (R_j is not present in $FTrD_N$) Then
$FTrD_N \leftarrow FTrD_N \cup R_j$
End If
End For
Step 5: Stop

Figure 4.5: Dataset Discretization using Fuzzy Logic

Step-4 copies only unique records from FD_N into fuzzy training dataset ($FTrD_N$) to remove redundancy. Fuzzy preprocessing algorithm generates two datasets as an output; one with redundant records (FD_N) and another with unique records ($FTrD_N$).

$FTrD_N$ and FD_N are used as training and testing datasets respectively for creating AEC as shown in Figure 4.6. Nineteen different versions of dataset; {FD_2, $FTrD_2$} to {FD_{20}, $FTrD_{20}$} are created using nineteen different linguistic values (i.e. discretization bins). Naive Bayesian classifier NB_2 is trained using $FTrD_2$ and its performance is evaluated using FD_2. Probability of true positive classification (PTPC) for every class-c_i by Naive Bayesian classifier NB_2 (i.e. PTP_{i2}) is stored in the set PTP (i.e. Probability of True Positive Classification). Then, Naive Bayesian classifier NB_3 is trained using $FTrD_3$ dataset and its performance is evaluated using FD_3. PTPC for every class-c_i by NB_3 (i.e. PTP_{i3}) is stored in the set PTP and so on.

Algorithm: Classifier Performance Evaluation

Input: KddTrD – KDD99 Training dataset

A – Set of attributes $\{A_1, A_2, \ldots, A_n\}$

C – Set of Classes in KddTrD dataset $\{c_1, c_2, \ldots, c_m\}$

Output: PTP - Probability of True Positive classification

Steps:

Step 1: Number of Linguistic values i.e. N = 2

Step 2: Call Fuzzy Preprocessing (KddTrD, A, L, N)

Step 3: Train Naïve Bayesian classifier NB_N using $FTrD_N$

Step 4: Test the detection accuracy of trained classifier using FD_N as testing dataset

Step 5: PTP_{ij} = Probability of True Positive Classification of class c_i by classifier NB_j;

Where i = 1 to m and j=N

Step 6: If (N < 20) Then

N = N + 1

Go to step 2

End If

Step 7: Stop

Figure 4.6: Base Classifiers Performance Evaluation

The process of creating AEC using appropriate BCs is explained with following example. Example 1: Let, D be a training dataset consisting of five classes namely {c_1, c_2, c_3, c_4, c_5} and it is converted into 19-bins (D2, D3,….D20). Consider nineteen BCs namely NB_2, NB_3, ...$NB2_0$ which are trained and evaluated on {$FTrD_2$, FD_2}, {$FTrD_3$, FD_3}…,{$FTrD_{20}$, FD_{20}} respectively.

Assume that the PTPC of every class c_i by available BCs (NB_2 to NB_{20}) as shown in Table 4.1

Class c_1 is classified by NB_3 with highest PTPC, class c_2 and class c_4 are classified by NB_{16} with highest PTPC and class c_3 and class c_5 are classified by NB_2 with highest PTPC. Therefore, AEC is created using combination of classifier NB_2, classifier NB_3 and classifier NB_{16}.

Table 4.1: Example: Creating Adaptive Ensemble of Classifiers

Classifier	c_1	c_2	c_3	c_4	c_5
NB_2	0.6	0.6	**0.9**	0.8	**0.9**
NB_3	**0.9**	0.8	0.6	0.6	0.4
NB_4	0.6	0.7	0.8	0.8	0.6
NB_5	0.8	0.6	0.7	0.7	0.8
NB_6	0.7	0.5	0.6	0.6	0.7
NB_7	0.6	0.4	0.5	0.5	0.6
NB_8	0.5	0.8	0.4	0.4	0.5
NB_9	0.4	0.4	0.8	0.8	0.4
NB_{10}	0.8	0.6	0.4	0.4	0.8
NB_{11}	0.4	0.8	0.6	0.6	0.4
NB_{12}	0.6	0.6	0.8	0.8	0.6
NB_{13}	0.8	0.5	0.6	0.6	0.8
NB_{14}	0.6	0.7	0.5	0.5	0.6
NB_{15}	0.5	0.7	0.7	0.7	0.5
NB_{16}	0.7	**0.9**	0.7	**0.9**	0.7
NB_{17}	0.7	0.7	0.4	0.4	0.7
NB_{18}	0.4	0.8	0.6	0.6	0.4
NB_{19}	0.6	0.7	0.5	0.5	0.6
NB_{20}	0.5	0.7	0.7	0.7	0.5

Network connection record is given as a input to all BCs within the AEC and their outputs are aggregated as shown in Figure 4.7. If all BCs have highest PTPC for their predicted class; then, class predicted with highest posterior probability is selected as a output of AEC.

If more than one BCs have highest PTPC for their predicted class; then, first select the classes predicted by BCs having highest PTPC for their predicted class. Among these selected classes; the class predicted with highest posterior probability is considered as a output of AEC.

If only one BC among the AEC has highest PTPC for its predicted class; then predicted class of that BC is selected as output of AEC. If none of the BCs has highest

PTPC for their predicted class; then, the class predicted with highest posterior probability is selected as a output of AEC.

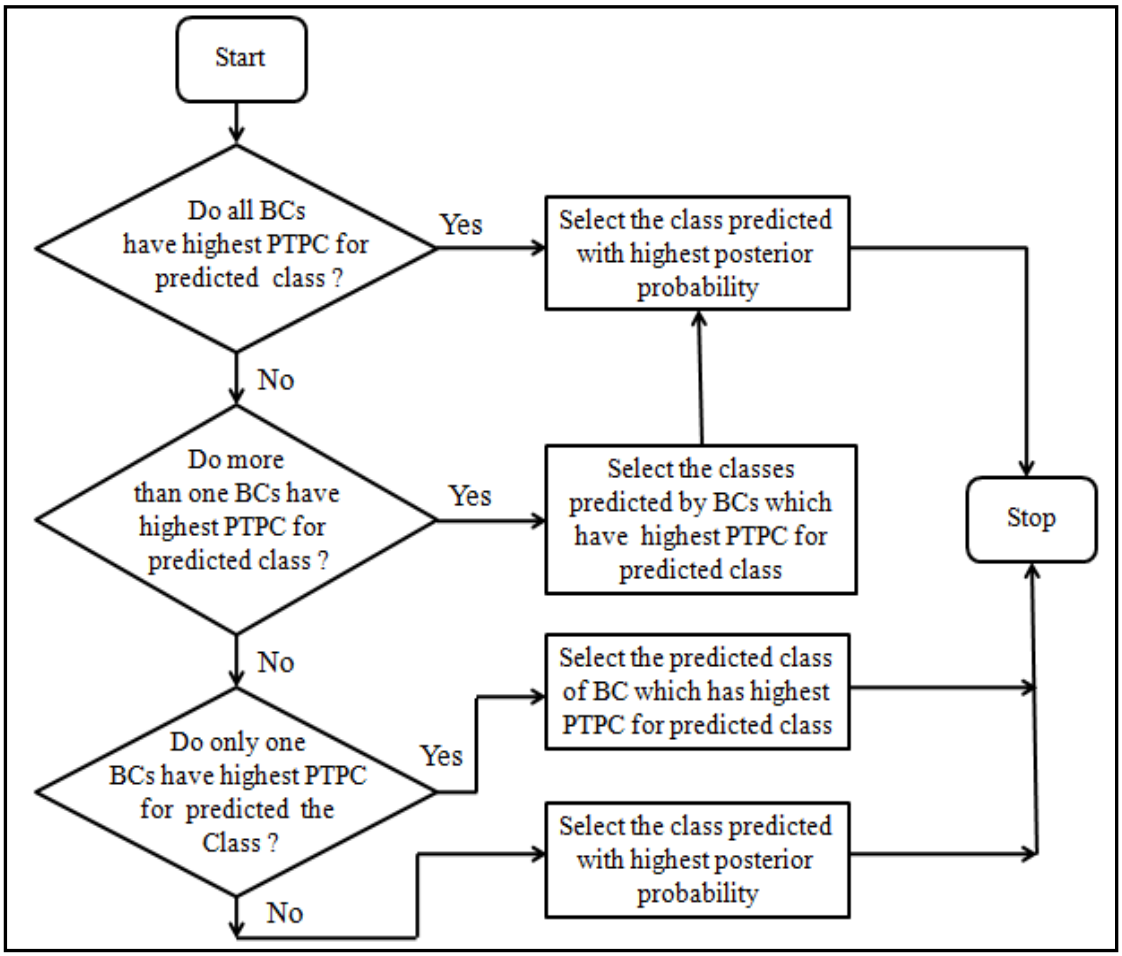


Figure 4.7: Output Aggregation of BCs

4.4 Experimental Results

Proposed Methodology is evaluated using three different datasets namely KDD 99, CDMC 2012 and a dataset prepared in Laboratory

4.4.1 Experiments on KDD 99 Dataset

Table 4.2 shows the detection accuracy of BCs and AEC. It is observed that, the highest overall detection accuracy (99.34%) is obtained for NB_{17}, however, it fails to detect records of Teardrop and Land class with high precision. NB_2 predicts records of Teardrop, Neptune and Land class with highest detection accuracy, NB_3 predicts records of Pod and Smurf class with highest detection accuracy and NB18 predicts records of Back and Normal class with highest detection accuracy as compared to any other BCs. ACE is build using three BCs i.e. NB_2, NB_3 and NB_{18}. The overall

detection accuracy obtained with this AEC is 99.81%. Table 4.3 shows the detection accuracy of DoS, DDoS attacks and normal user behavior.

Table 4.2: Detection Accuracy of BCs and AEC for KDD 99 Dataset

Classifier	Pod	Smurf	Back	Teardrop	Land	Neptune	Normal	Overall
NB_2	96.55%	99.98%	7.1%	**100%**	**77.78%**	**99.93%**	80.02%	95.35%
NB_3	**97.70%**	**99.993%**	17.76%	66.67%	77.78%	98.77%	97.31%	98.85%
NB_4	97.70%	96.990	11.66%	50%	66.67%	98.36%	99.52%	99.21%
NB_5	97.70%	99.98%	11.66%	58.33%	66.67%	98.23%	99.52%	99.18%
NB_6	97.70%	99.98%	9.47%	66.67%	55.56%	98.13%	99.54%	99.16%
NB_7	97.70%	99.98%	9.47%	66.67%	55.56%	98.14%	99.55%	99.16%
NB_8	97.70%	99.98%	9.38%	66.67%	55.56%	98.12%	99.55%	99.15%
NB_9	97.70%	99.98%	21.86%	41.67%	55.56%	98.08%	99.53%	99.19%
NB_{10}	97.70%	99.98%	36.97%	41.67%	55.56%	98.04%	99.61%	99.26%
NB_{11}	97.70%	99.98%	34.15%	41.67%	55.56%	97.92%	99.61%	99.22%
NB_{12}	97.70%	99.98%	52.55%	41.67%	55.56%	97.89%	99.59%	99.28%
NB_{13}	97.70%	99.98%	52.55%	41.67%	55.56%	97.88%	99.59%	99.28%
NB_{14}	97.70%	99.98%	69.85%	33.33%	55.56%	97.76%	99.59%	99.32%
NB_{15}	97.70%	99.98%	70.31%	41.67%	55.56%	97.75%	99.60%	99.33%
NB_{16}	97.70%	99.98%	70.67%	41.67%	55.56%	97.75%	99.60%	99.33%
NB_{17}	97.70%	99.98%	70.67%	33.33%	55.56%	97.76%	99.63%	99.34%
NB_{18}	97.70%	99.98%	**72.77%**	33.33%	55.56%	97.68%	**99.68%**	99.33%
NB_{19}	97.70%	99.98%	72.77%	41.67%	55.56%	97.68%	99.63%	99.32%
NB_{20}	97.70%	99.98%	71.68%	41.67%	55.56%	97.67%	99.63%	99.32%
AEC	97.70%	99.993%	72.77%	100%	77.78	99.93%	99.68%	**99.81%**

Table 4.3: Detection Accuracy of DoS, DDoS Attacks and Normal User Behavior for KDD 99 Dataset

Class	Testing Records	Correctly Classified Records	Mis-Classified Records	Detection Accuracy (%)
Normal	60593	60397	196	99.68
Back	1098	799	299	72.77
Pod	87	85	2	97.7
Smurf	164091	164080	11	99.99
Teardrop	12	12	0	100
Neptune	58001	57961	40	99.93
Land	9	7	2	77.77

Average, standard deviation and median of numeric attributes when records of training dataset are discretized into 3-bins using triangular fuzzy membership function are shown in Table 3.5., Table 3.6 and Table 3.7 respectively.

It is observed from Table 3.5, Table 3.6 and Table 3.7 that, average, standard deviation and median of attribute-2, attribute-3, attribute-7, attribute-8, attribute-11 and attribute-12 of Back class and Normal class are similar and training file of KDD 99 dataset is dominated by records of Normal class having 9,72,781 records and 2,203 records of Back class. Records of Back class are classified as Normal class which in turn reduces the detection accuracy for Back class.

Table 4.4: Average values of Numeric Attributes of KDD 99 Training Records Discretized into 18-Bins

Attribute	Average						
	Back	**Normal**	**Pod**	**Teardrop**	**Land**	**Neptune**	**Smurf**
Attribute 1	1.00	0.57	0.00	0.00	0.00	0.00	0.00
Attribute 2	1.00	1.17	1.09	3.05	1.29	5.49	11.55
Attribute 3	1.01	1.27	1.09	2.36	1.00	1.29	11.55
Attribute 4	3.94	3.32	1.00	1.02	2.57	4.76	1.00
Attribute 5	5.48	3.15	1.00	1.00	1.00	4.75	1.00
Attribute 6	16.89	15.45	17.00	14.48	13.14	3.43	17.00
Attribute 7	9.35	6.80	4.03	15.52	1.43	15.73	14.14
Attribute 8	9.35	8.43	4.24	4.52	1.29	2.01	8.45
Attribute 9	17.00	10.88	9.12	4.90	10.71	2.16	9.90
Attribute 10	1.69	3.81	9.12	4.90	10.71	1.07	9.90
Attribute 11	2.70	3.55	1.00	6.59	1.29	4.73	3.13
Attribute 12	2.70	3.64	1.00	1.00	1.00	4.72	1.00

Table 4.5: Standard Deviation Values of Numeric Attributes of KDD 99 Training Records Discretized into 18-Bins

Attribute	Std Dev						
	Back	**Normal**	**Pod**	**Teardrop**	**Land**	**Neptune**	**Smurf**
Attribute 1	0.00	0.49	0.00	0.00	0.00	0.01	0.00
Attribute 2	0.06	0.79	0.29	1.63	0.76	2.79	5.62
Attribute 3	0.09	0.95	0.29	0.97	0.00	0.45	5.62
Attribute 4	3.87	5.49	0.00	0.14	4.16	6.75	0.00
Attribute 5	3.85	5.15	0.00	0.00	0.00	6.75	0.00
Attribute 6	0.67	3.47	0.00	4.91	6.74	3.19	0.00
Attribute 7	6.07	5.06	5.42	3.10	0.53	3.60	3.90
Attribute 8	6.07	5.70	3.93	2.75	0.49	1.08	5.04
Attribute 9	0.00	5.82	5.62	2.96	7.85	1.94	5.17
Attribute 10	2.15	4.15	5.62	2.96	7.85	0.59	5.17
Attribute 11	2.96	4.97	0.00	5.14	0.49	6.76	3.00
Attribute 12	2.96	5.11	0.00	0.00	0.00	6.76	0.00

Average, standard deviation and median of numeric attributes when records of training dataset are discretized into 18-bins using triangular fuzzy membership function are shown in Table 4.4, Table 4.5 and Table 4.6 respectively.

Table 4.6: Median Values of Numeric Attributes of KDD 99 Training Records Discretized into 18-Bins

Attribute	Median						
	Back	**Normal**	**Pod**	**Teardrop**	**Land**	**Neptune**	**Smurf**
Attribute 1	1.00	1.00	0.00	0.00	0.00	0.00	0.00
Attribute 2	1.00	1.00	1.00	3.00	1.00	5.00	13.00
Attribute 3	1.00	1.00	1.00	2.00	1.00	1.00	13.00
Attribute 4	2.00	1.00	1.00	1.00	1.00	1.00	1.00
Attribute 5	5.00	1.00	1.00	1.00	1.00	1.00	1.00
Attribute 6	17.00	17.00	17.00	17.00	17.00	2.00	17.00
Attribute 7	9.00	5.00	1.00	17.00	1.00	17.00	17.00
Attribute 8	9.00	7.00	2.00	4.00	1.00	2.00	8.00
Attribute 9	17.00	12.00	9.00	5.00	17.00	2.00	10.00
Attribute 10	1.00	2.00	9.00	5.00	17.00	1.00	10.00
Attribute 11	2.00	1.00	1.00	8.00	1.00	1.00	1.00
Attribute 12	2.00	1.00	1.00	1.00	1.00	1.00	1.00

It is observed from Table 4.4, Table 4.5 and Table 4.6 that, when records of KDD 99 training dataset are discretized into 18-bins; average, standard deviation and median values of attribute-2, attribute-3, attribute-7, attribute-8, attribute-11 and attribute-12 of Back class, Normal class deviates much more as compared to average, standard deviation and median of these numeric attributes when records were discretized into 3-bins. Thus, less number of records of Back class are classified as Normal class which in turn increases the detection accuracy of Back class.

4.4.2 Experiments on CDMC 2012 Dataset

Experimental results on CDMC 2012 dataset given in Table 4.7 show the detection accuracy of BCs (NB_2 to NB_{20}) and AEC. Highest overall detection accuracy 95.99% is obtained for classifier NB19, however, it fails to detect records of class 1 and class -2 with highest detection accuracy. Classifier NB5, classifier NB9, and classifier NB19 predicts records of class 1, class -2 and class -1 with highest detection accuracy respectively as compared to any other BCs. ACE is build using three BCs namely NB5, NB9 and NB19. The overall detection accuracy of AEC is 97.40%. Table 4.8 shows the detection accuracy of ACE for both the attacks and normal user behavior.

Table 4.7: Detection Accuracy of BCs and AEC for CDMC 2012 Dataset

Classifier	1	-1	-2	Overall
NB_2	93.30	84.98	0.00	88.61
NB_3	93.13	88.17	12.75	90.31
NB_4	93.24	88.86	39.60	90.77
NB_5	**96.01**	85.85	36.91	90.35
NB_6	93.16	94.01	46.31	93.57
NB_7	93.19	96.44	49.66	94.92
NB_8	93.17	92.31	44.97	92.64
NB_9	92.81	97.96	**50.34**	95.59
NB_{10}	92.90	98.32	48.32	95.82
NB_{11}	92.89	98.04	47.65	95.67
NB_{12}	92.90	98.22	45.64	95.77
NB_{13}	92.90	98.40	40.94	95.86
NB_{14}	92.90	98.16	40.94	95.73
NB_{15}	92.90	98.21	46.98	95.76
NB_{16}	92.91	97.92	47.65	95.60
NB_{17}	92.90	98.15	48.32	95.73
NB_{18}	92.89	98.19	45.64	95.75
NB_{19}	92.90	**98.64**	38.93	**95.99**
NB_{20}	92.90	97.34	40.27	95.27
AEC	96.01	98.64	50.34	**97.40**

Table 4.8: Detection Accuracy of Attacks and Normal User Behavior for CDMC 2012 Dataset

Class	Testing Records	Correctly Classified	Mis-Classified	Detection Accuracy (%)
1	57,155	54,877	2,278	96.01
-1	70,053	69,099	954	98.64
-2	149	75	74	50.34

Average, standard deviation and median of numeric attributes when records of CDMC 2012 training dataset are discretized into 3-bins using triangular fuzzy membership function are shown in Table 3.11, Table 3.12, and Table 3.13 respectively.

It is observed from Table 3.11, Table 3.12, Table 3.13 that, average, standard deviation and median of attribute-1, attribute-3, attribute-4, attribute-5, attribute-6 and attribute-7 of class -1 and class -2 are similar when records of CDMC 2012 training dataset are discretized into 3-bins and training file is dominated by records of class -1 and class 1. Thus, records of '-2' class are classified as either '-2' or '-1' which in turn reduces the detection accuracy of '-2' class.

Average, standard deviation and median of numeric attributes when records of CDMC 2012 training dataset are discretized into 10-bins using triangular fuzzy membership function are shown in Table 4.9, Table 4.10, and Table 4.11.

Table 4.9: Average Values of Numeric Attributes of CDMC 2012 Training Records Discretized into 10-Bins

Attribute	Average		
	1	-1	-2
Attribute 1	1.00	1.00	1.00
Attribute 2	2.21	6.74	8.97
Attribute 3	1.00	1.00	1.00
Attribute 4	1.00	1.00	1.00
Attribute 5	1.66	1.03	1.00
Attribute 6	7.22	2.04	7.30
Attribute 7	1.00	1.08	1.00
Attribute 8	1.13	4.16	3.09
Attribute 9	1.43	2.37	2.35
Attribute 10	8.64	2.74	4.28
Attribute 11	1.64	1.27	1.00
Attribute 12	1.00	1.12	1.00
Attribute 13	1.00	2.65	1.11

Table 4.10: Standard Deviation Values of Numeric Attributes of CDMC 2012 Training Records Discretized into 10-Bins

Attribute	Std Dev		
	1	-1	-2
Attribute 1	0.00	0.04	0.00
Attribute 2	1.31	3.46	0.41
Attribute 3	0.03	0.00	0.00
Attribute 4	0.06	0.02	0.00
Attribute 5	1.30	0.17	0.00
Attribute 6	3.27	2.65	3.07
Attribute 7	0.08	0.77	0.00
Attribute 8	0.94	3.54	2.43
Attribute 9	1.01	2.51	1.35
Attribute 10	1.37	2.70	1.53
Attribute 11	2.08	1.45	0.00
Attribute 12	0.03	0.78	0.00
Attribute 13	0.03	3.18	0.92

It is observed from Table 4.9, Table 4.10, Table 4.11 that, average, standard deviation and median of attribute-1, attribute-3, attribute-4, attribute-5, attribute-7, attribute-10,

attribute-12 of class 1, class -1 and class -2 are similar when records of CDMC 2012 training dataset are discretized into 10-bins and training file is dominated by records of class -1 and class 1 . Thus, records of '-2' class are classified as '1' or '-1' or '-2' which in turn reduces the detection accuracy of '-2' class.

Table 4.11: Median Values of Numeric Attributes of CDMC 2012 Training Records Discretizd into 10-Bins

Attribute	Median		
	1	-1	-2
Attribute 1	1.00	1.00	1.00
Attribute 2	2.00	9.00	9.00
Attribute 3	1.00	1.00	1.00
Attribute 4	1.00	1.00	1.00
Attribute 5	1.00	1.00	1.00
Attribute 6	9.00	1.00	9.00
Attribute 7	1.00	1.00	1.00
Attribute 8	1.00	1.00	3.00
Attribute 9	1.00	1.00	2.00
Attribute 10	9.00	1.00	4.00
Attribute 11	1.00	1.00	1.00
Attribute 12	1.00	1.00	1.00
Attribute 13	1.00	1.00	1.00

Average, standard deviation and median of numeric attributes when records of CDMC 2012 training dataset are discretized into 20-bins using triangular fuzzy membership function are shown in Table 4.12, Table 4.13, and Table 4.14.

Table 4.12: Average Values of Numeric Attributes of CDMC 2012 Training Records Discretized into 20-Bins

Attribute	Average		
	1	-1	-2
Attribute 1	1.00	1.00	1.00
Attribute 2	2.56	13.97	18.92
Attribute 3	1.00	1.00	1.00
Attribute 4	1.00	1.00	1.00
Attribute 5	2.64	1.08	1.23
Attribute 6	15.00	3.34	15.21
Attribute 7	1.00	1.17	1.00
Attribute 8	1.28	8.09	5.66
Attribute 9	2.05	4.06	3.98
Attribute 10	18.16	4.91	8.38
Attribute 11	2.44	1.62	1.00
Attribute 12	1.00	1.27	1.00
Attribute 13	1.00	4.70	1.24

Table 4.13: Standard Deviation Values of Numeric Attributes of CDMC 2012 Training Records Discretized into 20-Bins

Attribute	Std Dev		
	1	-1	-2
Attribute 1	0.00	0.09	0.00
Attribute 2	3.14	7.73	0.98
Attribute 3	0.08	0.00	0.00
Attribute 4	0.14	0.05	0.00
Attribute 5	2.85	0.32	0.42
Attribute 6	7.37	5.97	6.90
Attribute 7	0.18	1.74	0.00
Attribute 8	2.12	7.96	5.51
Attribute 9	2.17	5.62	2.97
Attribute 10	3.17	6.02	3.40
Attribute 11	4.68	3.27	0.00
Attribute 12	0.07	1.76	0.00
Attribute 13	0.08	7.16	2.08

Table 4.14: Median Values of Numeric Attributes of CDMC 2012 Training Records Discretized into 20-Bins

Average	Median		
	1	-1	-2
Attribute 1	1.00	1.00	1.00
Attribute 2	2.00	19.00	19.00
Attribute 3	1.00	1.00	1.00
Attribute 4	1.00	1.00	1.00
Attribute 5	2.00	1.00	1.00
Attribute 6	19.00	1.00	19.00
Attribute 7	1.00	1.00	1.00
Attribute 8	1.00	1.00	5.00
Attribute 9	1.00	1.00	4.00
Attribute 10	19.00	1.00	9.00
Attribute 11	1.00	1.00	1.00
Attribute 12	1.00	1.00	1.00
Attribute 13	1.00	1.00	1.00

It is observed from Table 4.12, Table 4.13, Table 4.14 that; average, standard deviation and median of attribute-1, attribute-3, attribute-4, attribute-7, attribute- 12 and attribute-13 of class 1 and class -2 are similar when records of CDMC 2012 training dataset are discretized into 20-bins and training file is dominated by records of class -1 and class 1. Thus, records of '-2' class are classified as either '-2' or '1' which in turn reduces the detection accuracy of '-2' class.

4.4.3 Experiment on Dataset created in Laboratory

Table 4.15 shows the detection accuracy of BCs and AEC on the dataset created in the laboratory. It is observed that, highest overall detection accuracy (95.73%) is obtained for classifier NB_{12}, however, it fails to detect records of HTTP Flood class, Slow Read class, Slow Write class and Normal class with highest detection accuracy. Classifier NB_7 predicts records of HTTP Flood class and Normal class with highest detection accuracy as compared to other BCs. Classifier NB_{14} and classifier NB_{15} predicts records of Slow Write class and Slow Read class with highest detection accuracy respectively. ACE consisting of NB_7, NB_{14} and NB_{15} predicts the records with 96.44% detection accuracy. Table 4.16 shows the detection accuracy of attacks and Normal User Behavior.

Table 4.15: Detection Accuracy of BCs and AEC for Dataset Created in Laboratory

Classifier	HTTP Flood	Slow Read	Slow Write	Normal	Overall
NB_2	89.56	71.53	74.31	88.21	86.50
NB_3	90.62	71.53	74.31	90.03	87.65
NB_4	91.94	71.98	75.25	91.85	89.04
NB_5	93.27	71.98	75.25	93.41	90.25
NB_6	95.28	80.18	81.48	95.62	93.14
NB_7	**96.14**	82.25	83.55	**96.38**	94.16
NB_8	96.14	85.16	85.92	96.38	94.57
NB_9	96.14	88.38	89.21	96.38	95.08
NB_{10}	96.14	90.45	91.25	96.38	95.40
NB_{11}	96.14	91.72	93.58	96.31	95.65
NB_{12}	96.04	93.86	95.07	95.87	95.73
NB_{13}	95.54	95.12	96.28	94.18	95.09
NB_{14}	94.91	96.58	**97.82**	93.27	94.70
NB_{15}	93.58	**97.31**	97.82	91.47	93.47
NB_{16}	92.23	97.31	97.82	89.53	92.13
NB_{17}	91.03	97.31	97.82	87.72	90.91
NB_{18}	89.52	97.31	97.82	86.29	89.66
NB_{19}	87.21	97.22	97.82	84.26	**87.80**
NB_{20}	85.84	97.22	97.52	82.75	86.57
AEC	96.14	97.31	97.82	96.38	**96.44**

All the classes of dataset created in laboratory represent the HTTP traffic. HTTP Flood class, Slow Read class and Slow Write class represents the DoS and DDoS attacks which establishes connection with target HTTP server and consumes the resources required for HTTP connection in order to make the server unavailable for legitimate users. During these attacks, initial connections are established and

completed as normal user connections. As a result average, standard deviation and median values of network connection records representing these initial intrusive connections are similar to average, standard deviation and median values of network connection records representing normal user behavior. Thus, these initial connections of HTTP Flood class, Slow Read class and Slow Write class are classified as Normal class. This reduces the detection accuracy of these classes.

Table 4.16: Detection Accuracy of Attacks, Normal User Behavior for Dataset Created in Laboratory

Class	Testing Dataset Records	Correctly Classified	Mis-Classified	Detection Accuracy (%)
HTTP Flood	17,27,438	16,60,759	66,679	96.14
Slow Read	2,87,134	2,79,411	7723	97.31
Slow Write	2,45,971	2,40,609	5362	97.82
Normal	11,73,148	11,30,681	42467	96.38

4.5 Conclusion

This chapter presents a process for dynamic selection of Base Classifiers to create an AEC based on the updated input records. AEC adopt itself with the continuous evolution and change in normal user behavior, DoS and DDoS attack strategies as well as normal user behavior to improve the detection accuracy. The detection accuracy obtained is 99.81%, 97.40%, 96.44% for KDD 99, CDMC 2012 and dataset generated in laboratory respectively. It is observed that AEC improves the detection accuracy of IDS; however, with increase in number of Base Classifiers in AEC, processing power and memory requirement of IDS also increases. This makes it an unrealistic solution for design of real-time IDS. Thus there is a need of filtering mechanism which can effectively reduce the amount of data processed by IDS; which will allow the use of AEC for design of real-time IDS.

Chapter 5

Novel DoS and DDoS attack Detection

Novel attack behavior significantly differs from normal user behavior and known attacks. This chapter presents the proposed methodology for detection of novel DoS, DDoS attacks and known attacks using S-IDS and AEC proposed in previous chapter.

5.1 Introduction

Every attack launched over the network has a predefined pattern; S-IDS learns these patterns and use them to detect these attacks. On-line system use the known signature patterns to remove the vulnerabilities, lacunas and modify the working of the system. Attackers continuously modify existing attacks or develop new attacks in order to launch successful and stealth attacks (Barker, 2015).

Figure 5.1: Modification in working mechanism of known attack: DDoS Reflection Attacks using SSDP protocol

All the attacks whose signature or information is known are called known attacks, whereas, attacks which exploits either publically unknown vulnerability or publically known vulnerability using new approach are called as novel attack. Novel attacks are

classified into two categories:

i. Modification in working mechanism of known attack
ii. Zero-day attack

Modification in working mechanism of known attack:

Attacker modifies the working mechanism of known attack to exploit publically known vulnerability of system in novel manner and makes it unavailable for legitimate user. Figure 5.1 shows the report published by 'ddosattacks.net' informing how attacker launched the reflection DDoS attack by using Simple Service Discovery Protocol present on home computers instead of using DNS servers.

Zero day attacks:

Attacker exploits the publically unknown vulnerability of system to launch a successful stealth DDoS attack and make the system unavailable for legitimate user (Jiang, 2016). Figure 5.2 shows how Slowloris attack exploited the working of HTTP server in novel manner to launch a stealth attack. Instead of generating a huge flood of HTTP requests; Slowloris attack opens many connections with HTTP server and keeps them open as long as possible. The huge flood exhaust the connection pool of HTTP server restricting processing of legitimate client requests.

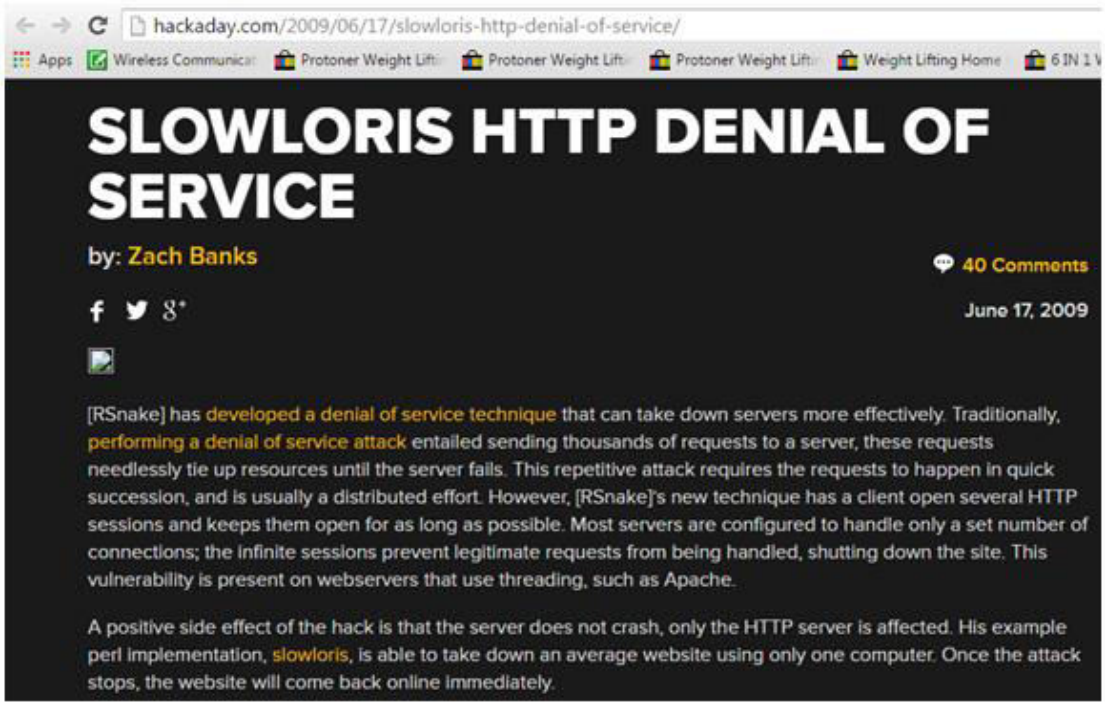


Figure 5.2: Zero Day Attack: Slowloris

Novel attacks remains stealth and continues to harm on-line systems with huge impact till their signatures are not generated by security experts and analyst. Once these signatures are discovered, the same are used to develop a defense mechanism and remove vulnerabilities in the system. The objective is to automate signature generation process of novel attacks.

5.2 Proposed Methodology for Detection of Known as well as Novel DoS and DDoS Attacks

The methodology proposed in section 4.3 for detection of known DoS and DDoS attacks using AEC is further extended for detection of known, novel DoS and DDoS attacks as shown in Figure 5.3. It is based on the assumption that; if network connection record does not matches with the signatures of known attacks as well as normal user behavior, then it is treated as novel attack.

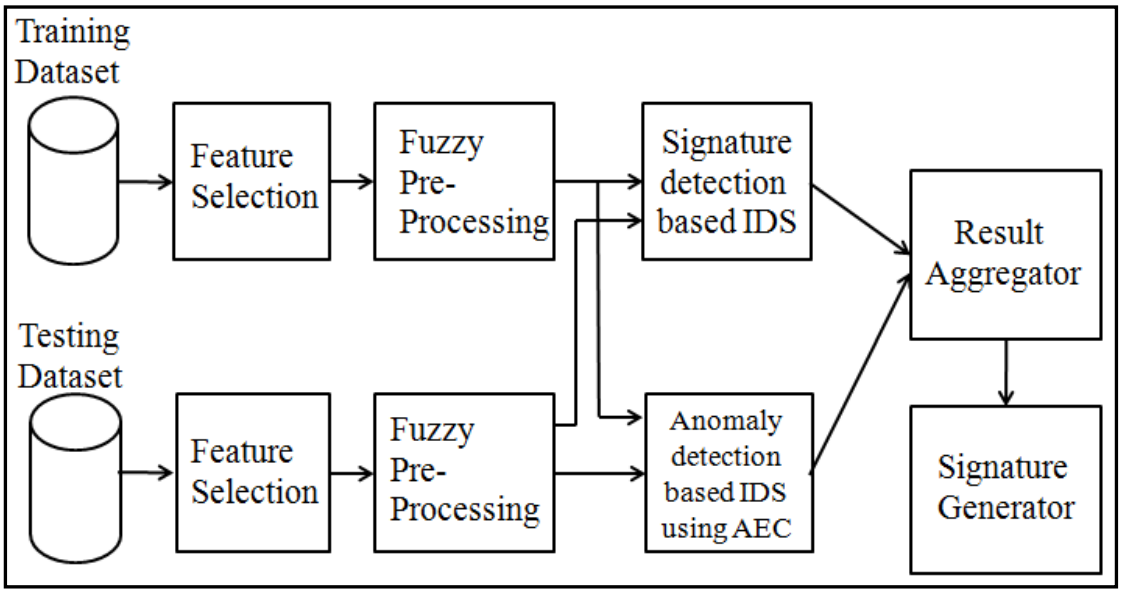


Figure 5.3: Methodology for Detection of Known, Novel DoS and DDoS Attacks using S-IDS and AEC

Combination of S-IDS and AEC is proposed to detect known, novel DoS and DDoS attacks. When features to be used for building S-IDS are known in advanced, we can directly discover frequent itemsets of size N.

Support and Confidence of patterns (i.e. itemset) present in training dataset are used to find the frequent patterns representing normal user behavior and known attacks. Support, Confidence of patterns are calculated using equation. 5.1 and equation 5.2.

$$Support\ of\ pattern\ X \rightarrow Y = Supp(X \rightarrow Y) = P(X \cup Y) \quad \text{---------- (5.1)}$$

$$Confidence\ of\ Pattern\ X \rightarrow Y = Conf(X \rightarrow Y) = \frac{\text{Supp (X} \cup \text{Y)}}{Supp(X)} \quad \text{---------- (5.2)}$$

Where, X is the pattern of network connection records present in training dataset and Y is the class of the network connection record (i.e. type of attack or normal user behavior). Support $(X \rightarrow Y)$ specifies the probability of occurrence of class Y and pattern X together in training dataset. Confidence $(X \rightarrow Y)$ specifies the probability of class Y if pattern of network connection record is X.

S-IDS developed using frequent itemsets generates high true positive rate and very low false positive rate; however, it fails to learn the normal user behavior and attacks characteristics which does not satisfy the minimum Support and Confidence criterion. As a result S-IDS cannot classify network connection records representing these characteristics. To solve this problem, AEC is trained using records of normal user behavior and known attacks. Output of S-IDS and AEC is given as input to result aggregator which aggregates these outputs and detects known as well as novel DoS and DDoS attacks as shown in Figure 5.4.

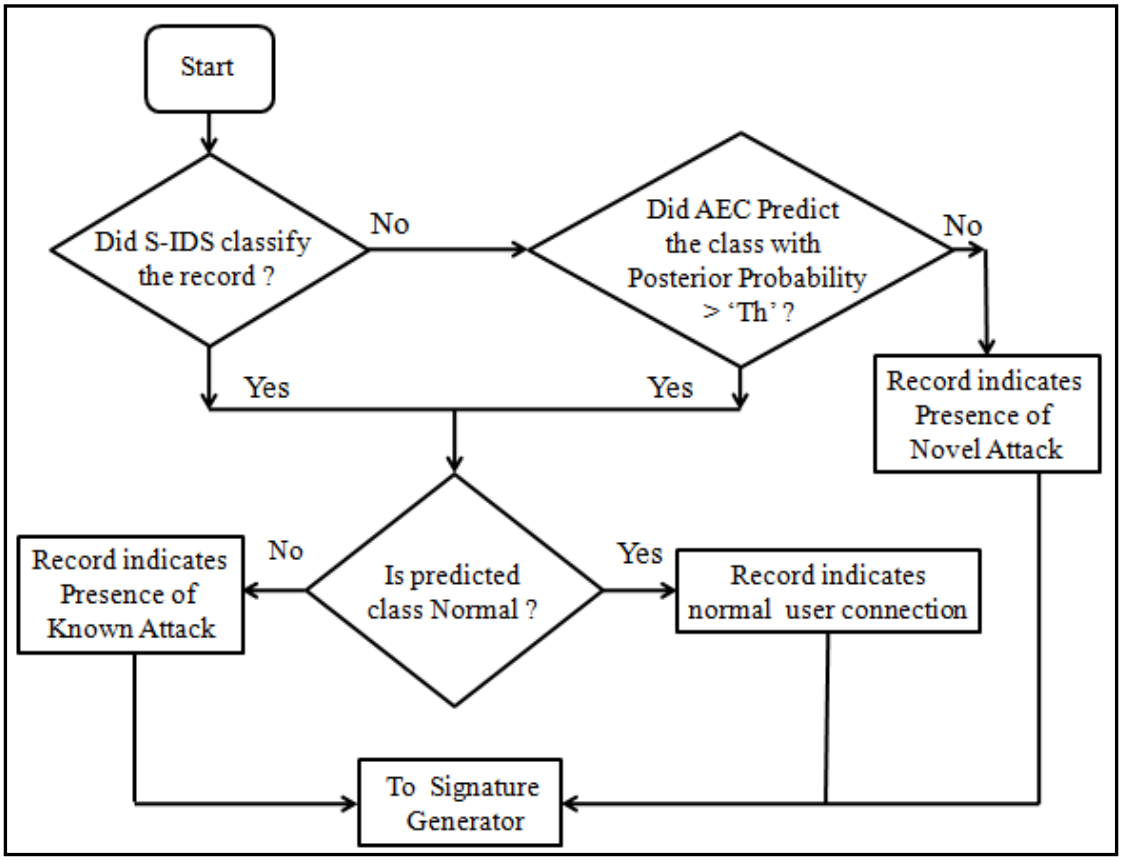


Figure 5.4: Working of Result Aggregator

If network connection record matches with the signature present in the S-IDS; it predicts whether record represents the attack or not and its result is considered as final prediction by Result Aggregator. If signatures of S-IDS are not matching with network connection record and it does not predict the class of network connection record then output of AEC is used for decision making. If AEC predicts the result with posterior probability greater then predefined threshold 'Th', then result of AEC is considered as final prediction, otherwise, network connection records is considered be a representative of novel attack. The process of calculating the threshold value 'Th' for detection of novel attacks is explained in Figure 5.5.

Threshold calculation:

- Let, 'D' denotes the Training set

  $D = \{r_1, r_2, \ldots\ldots r_n\}$

- Let, 'C' denotes the set of classes of records belonging to 'D'

  $C = \{ c_1, c_2, \ldots\ldots c_m \}$

- Let, 'S' denotes the classifier
- f is a function such that,

  $f(S, r_i) = (c_j, p_i)$ where, p_i is probability of class c_j for record r_i

- g is a function such that,

  $g(S, r_i) = 1$ if r_i is correctly classified by 'S'

  $= 0$ Otherwise

- 'Q' denotes a set of real numbers (i.e. probabilities) such that

  $Q = \{f(S, r_i).pi \mid g(S, r_i) = 1\}$

- $T = \min(q_k)$ $\forall\ q_k \in Q$

Thus, Threshold 'Th' = T - (0.25 * T)

Figure 5.5: Threshold Calculation

Classifier 'S' is trained and evaluated using training dataset 'D'. Then, Minimum probability with which classifier 'S' correctly predicts the records of every class c_i in dataset D is calculated as 'T'. In this process AEC is used as classifier 'S'. Then threshold 'Th' is calculated as:

$Th = T - (0.25 * T)$ 'Th' ---------- (5.3)

The output of result aggregator is given as an input to signature generation module. Signature Generator module uses frequent pattern discovery process to generate the signatures of novel attacks using network connection records considered as representative of novel attacks by result aggregator. This automated process generate the approximate signatures of novel attacks.

5.3 Experimental Results

KDD 99, CDMC 2012 and dataset developed in laboratory are used to test the proposed mechanism.

5.3.1 Experiments using KDD 99 Dataset

KDD 99 testing dataset representing four novel DDoS attacks namely 'Process table', 'UDP Storm', 'Mail bomb' and 'Apache2' whose signatures are not present in training dataset along with seven DOS and DDoS attacks is used to test the proposed methodology. Table 5.1 shows the detection accuracy on KDD 99 Dataset for known, novel DoS and DDoS attacks detection using proposed combination of S-IDS and AEC.

Table 5.1: Detection Accuracy on KDD 99 Dataset for Known and Novel Attacks using Proposed Combination of S-IDS and AEC

Class	Testing Records	Correctly Classified	Mis-Classified	Detection Accuracy (%)
Normal	60593	60397	196	99.68
Back	1098	799	299	72.77
Pod	87	85	2	97.7
Smurf	164091	164080	11	99.99
Teardrop	12	12	0	100
Neptune	58001	57961	40	99.93
Land	9	7	2	77.77
Novel Attack	6555	6500	55	99.16

Table 5.2, Table 5.3 and Table 5.4 shows the average, standard deviation and median values of numeric attributes when network connection records belonging to normal user behavior, known and novel attacks are discretized into-3 bins using triangular fuzzy membership functions.

Table 5.2: Average Values of Numeric Attributes of KDD 99 Records Belonging to Normal User Behavior, Known Attacks and Novel Attacks Discretized into 3-Bins

Attribute	Average							
	Back	**Normal**	**Pod**	**Teardrop**	**Land**	**Neptune**	**Smurf**	**Novel Attack**
Attribute 1	1.00	0.57	0.00	0.00	0.00	0.01	0.00	0.84
Attribute 2	1.00	1.01	1.00	1.11	1.00	1.27	1.50	1.00
Attribute 3	1.00	1.01	1.00	1.00	1.00	1.00	1.50	1.00
Attribute 4	1.27	1.15	1.00	1.00	1.10	1.29	1.00	1.08
Attribute 5	1.45	1.14	1.00	1.00	1.00	1.29	1.00	1.09
Attribute 6	2.00	1.89	2.00	1.89	1.80	1.29	2.00	2.00
Attribute 7	1.48	1.43	1.12	1.89	1.00	1.87	1.88	2.00
Attribute 8	1.48	1.31	1.29	1.28	1.00	1.02	1.25	1.93
Attribute 9	2.00	1.51	1.71	1.33	1.70	1.06	1.63	1.93
Attribute 10	1.12	1.23	1.71	1.33	1.70	1.02	1.63	1.0
Attribute 11	1.15	1.15	1.00	1.61	1.00	1.27	1.38	1.08
Attribute 12	1.15	1.15	1.00	1.00	1.00	1.27	1.00	1.05

Table 5.3: Standard Deviation Values of Numeric Attributes of KDD 99 Records Belonging to Normal User Behavior, Known Attacks and Novel Attacks Discretized into 3-Bins

Attribute	Std Dev							
	Back	**Normal**	**Pod**	**Teardrop**	**Land**	**Neptune**	**Smurf**	**Novel Attack**
Attribute 1	0.00	0.50	0.00	0.00	0.00	0.07	0.00	0.36
Attribute 2	0.00	0.09	0.00	0.32	0.00	0.44	0.52	0.00
Attribute 3	0.00	0.08	0.00	0.00	0.00	0.00	0.52	0.00
Attribute 4	0.45	0.36	0.00	0.00	0.32	0.45	0.00	0.28
Attribute 5	0.51	0.35	0.00	0.00	0.00	0.45	0.00	0.28
Attribute 6	0.00	0.32	0.00	0.32	0.42	0.46	0.00	0.04
Attribute 7	0.51	0.50	0.33	0.32	0.00	0.33	0.34	0.01
Attribute 8	0.51	0.46	0.47	0.46	0.00	0.15	0.45	0.25
Attribute 9	0.00	0.50	0.47	0.49	0.48	0.25	0.50	0.25
Attribute 10	0.33	0.42	0.47	0.49	0.48	0.12	0.50	0.00
Attribute 11	0.36	0.35	0.00	0.50	0.00	0.45	0.50	0.27
Attribute 12	0.36	0.36	0.00	0.00	0.00	0.45	0.00	0.21

It is observed from Table 5.2, Table 5.3, Table 5.4 that; average, standard deviation and median values of records representing novel attacks and records representing normal user behavior, known attacks are different. Thus, AEC classifies the records representing novel attacks with posterior probability less than predefined threshold

'Th' and result aggregator marks these records as novel attacks. In this case detection accuracy of 99.16% is achieved for novel attacks.

Table 5.4: Median Values of Numeric Attributes of KDD Records Belonging to Normal User Behavior, Known Attacks and Novel Attacks Discretized into-3 Bins

Attribute	Median							
	Back	**Normal**	**Pod**	**Teardrop**	**Land**	**Neptune**	**Smurf**	**Novel Attack**
Attribute 1	1.00	1.00	0.00	0.00	0.00	0.00	0.00	1.00
Attribute 2	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Attribute 3	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Attribute 4	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Attribute 5	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Attribute 6	2.00	2.00	2.00	2.00	2.00	1.00	2.00	2.00
Attribute 7	1.00	1.00	1.00	2.00	1.00	2.00	2.00	2.00
Attribute 8	1.00	1.00	1.00	1.00	1.00	1.00	1.00	2.00
Attribute 9	2.00	2.00	2.00	1.00	2.00	1.00	2.00	2.00
Attribute 10	1.00	1.00	2.00	1.00	2.00	1.00	2.00	1.00
Attribute 11	1.00	1.00	1.00	2.00	1.00	1.00	1.00	1.00
Attribute 12	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

5.3.2 Experiments using CDMC 2012 Dataset

Records belonging to class '1' and class-1' in training dataset are used to train the S-IDS, AEC and records belonging to class '-2' are eliminated from training process. Records belonging to class '-2' in testing file are used to represent novel attacks and test the proposed mechanism. Table 5.5 shows the detection accuracy obtained for normal user behavior (class '1'), known attack (class '-1') and novel attack (class '-2') detection on CDMC 2012 dataset using proposed combination of S-IDS and AEC.

Table 5.5: Detection Accuracy on CDMC 2012 Dataset for Known, Novel DoS and DDoS Attack using Proposed Combination of S-IDS and AEC

Class	Testing Records	Correctly Classified	Miss-Classified	Detection Accuracy (%)
1	57,155	55,979	1,176	97.94
-1	70,053	69,174	879	98.75
-2	149	124	25	83.22

Table 3.11, Table 3.12 and Table 3.13 shows the; average, standard deviation and median of numeric attributes when records of CDMC 2012 training dataset are discretized into 3-bins using triangular fuzzy membership function.

It is observed from Table 3.11, Table 3.12 and Table 3.13 that, average, standard deviation and median values of attribute-2, attribute-8, attribute-9, attribute-10, attribute-11, attribute-12 and attribute-13 of class'-2' are different from class'1' and class '-1'.Thus, AEC classifies most of the records representing class '-2' with posterior probability less than predefined threshold 'Th' and result aggregator marks these records as novel attacks. Detection accuracy of 83.22% is achieved for class '-2' (i.e. novel attack).

5.3.3 Experiments using dataset created in laboratory

Network connection records representing normal user behavior, HTTP Flood attack and Slow Read attack in training dataset are used to train the combination of S-IDS, AEC and records belonging to class Slow Write are eliminated from training process. Network connection records labeled as Slow Write attack in testing dataset are used to represent novel attack and test the proposed methodology. Table 5.6 shows the detection accuracy obtained for detection of known, novel DoS and DDoS attacks on testing dataset created in laboratory.

Table 5.6: Detection Accuracy on Dataset Created in Laboratory for Known and Novel DoS/DDoS Attacks using Proposed Combination of S-IDS and AEC

Class	Testing Dataset Records	Correctly Classified Records	Mis-Classified Records	Detection Accuracy (%)
Normal	11,73,148	11,43,585	29,563	97.48
HTTP Flood	17,27,438	16,72,678	54,760	96.83
Slow Read	2,87,134	2,81,592	5,542	98.07
Slow Write	2,45,971	2,42,010	3,961	98.39

Table 3.17, Table 3.18 and Table 3.19 shows the average, standard deviation and median of numeric attributes when records of training dataset created in laboratory are discretized into 3-bins using triangular fuzzy membership function. It is observed from these tables that, average, standard deviation and median values of all attributes of Slow Write class are different from HTTP flood attack, normal user behavior and standard deviation values of all attributes of Slow Write attack are different than Slow Read attack. AEC classifies the records representing Slow Write attack with posterior probability less than predefined threshold 'Th' and result aggregator marks these records as novel attacks. As a result detection accuracy of 98.39% is achieved for Slow Write (i.e. novel) attack.

5.4 Conclusion

Combination is S-IDS and ADE is proposed to detect known, novel DoS and DDoS attacks with an assumption that, novel attacks differs significantly from normal user behavior and known attacks. Proposed method also generates the approximate signatures of novel attacks. The detection accuracy of 99.16%, 83.22% and 98.39% is obtained on KDD 99, CDMC 2012, and dataset representing attacks generated in laboratory respectively. Proposed method analyzes all the network connections and network connection records for detection of DoS and DDoS attacks. Under the presence of heavy network traffic; it becomes computationally very expensive and difficult to process all network connection records in real time. In order to reduce the load on IDS under heavy network traffic; some mechanism is required to filter out the error free network connection records from intrusion detection process.

Chapter 6

Real-Time Light Weight Distributed Intrusion Detection System

With an exponential growth in computing power of machines and decreases in their cost; amount of network traffic generated is also increased. It is very difficult to process such a huge volume of network traffic for attack detection in real-time using single machine. This chapter describes the methodology adopted for implementing real-time light weight distributed IDS for detection of DoS and DDoS attacks.

6.1 Introduction

To reduce the impact of attacks on online systems; these attacks must be detected in real-time. Processing extremely huge volume of traffic in real-time is not possible using conventional single machine based IDS. We proposed a light weight distributed implementation of IDS for detecting DoS and DDoS attacks in real-time. This methodology is based on following assumptions:

i. Normally desktops, laptops and other computing machines in any organization are underutilized (average CPU utilization is 20/25%). Thus, remaining CPU cycles can be used for other processing without affecting the ongoing activities on these machines.

ii. In general, most of the communications by the computing machine over network are attack free; however, every communication is analyzed by IDS to detect intrusive activities. A mechanism is required to exclude the attack free traffic from being analyzed by IDS.

6.2 Proposed Real-time Light Weight Distributed IDS

In general, every organization consists of number of departments. Machines in each department are connected over the network and all the departments are connected with each other and IDS. Figure 6.1 shows placement of machines and IDS in an organization. In order to implement the Light Weight IDS, we introduced the concept of Local Aggregator in each department of the organization.

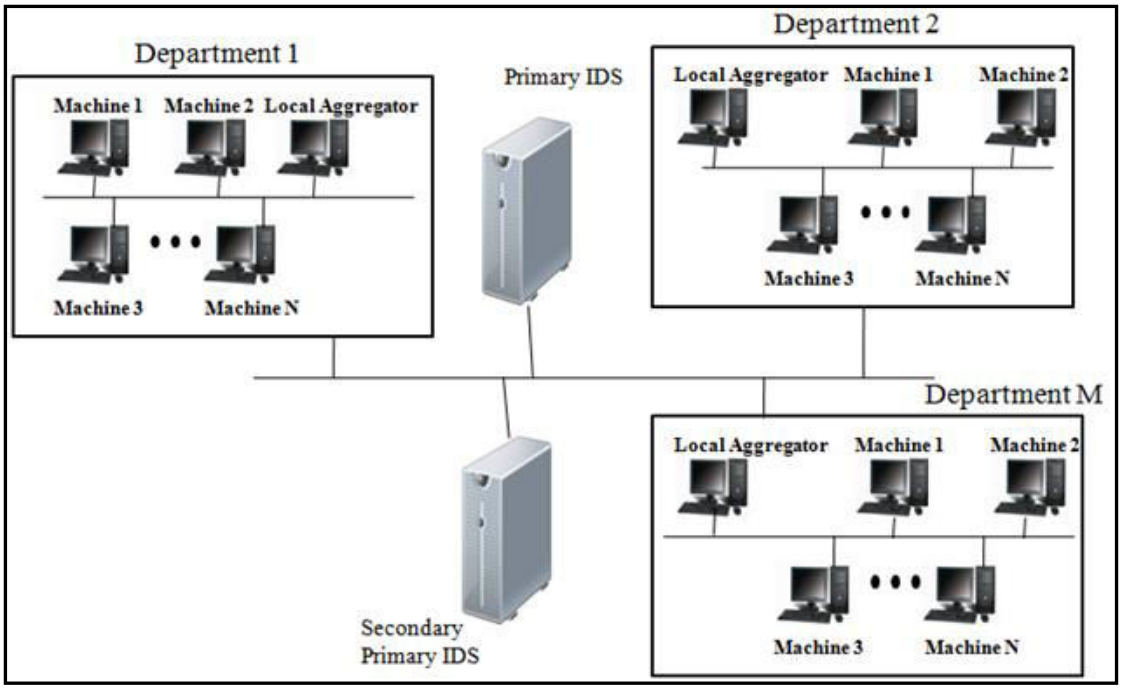


Figure 6.1: Placement of Local Aggregator, Primary IDS and Secondary IDS in Organization

N-IDS captures the traffic of entire network and converts it into network connection records for attack detection. Under heavy network traffic; it becomes difficult for N-IDS machine to detect attacks in real time. To solve this problem; in proposed approach every machine of the organization converts the network traffic originating from and destined to that machine into network connection records and forward these records to the Local Aggregator machine present within the department. Records received at Local Aggregator are aggregated and forwarded to the IDS. IDS receives these network connection records from all the Local Aggregator machines and divide them into 'N' number of non-overlapping subsets. Then, it forwards these subsets to underutilized machines of the organization for attack detection. The local underutilized machines compares the received network connection records with the Signature database available with them and sends the result back to IDS. Results received from all the local machines are merged together in the IDS to update the Signature database.

The proposed system consists of five building blocks.

i. Network traffic collection by Local Machine
ii. Network traffic collection by Local Aggregator
iii. Aggregation and Distribution of records by Primary and Secondary IDS

iv. Intrusion Detection by Local Machine

v. Signature Management by Local Aggregator

i. Network traffic collection by Local Machine

Each machine within a department has been configured to collect the network traffic on the machine using unused CPU cycles of the machine. Sniffer, Traffic Aggregator and Record Transmitter installed on each machine collects and aggregates the network traffic as shown in Figure 6.2.

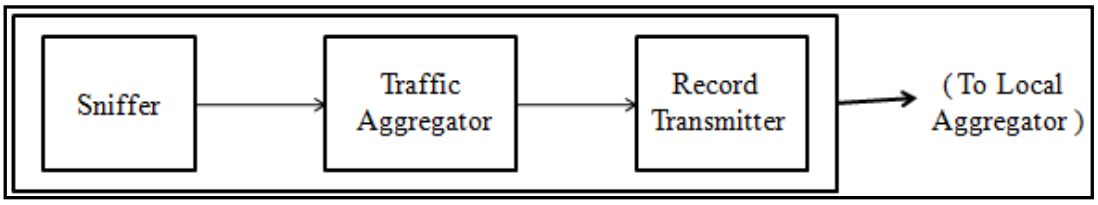


Figure 6.2: Network Traffic Collection Components Installed on Each Machine of the Department

a) Sniffer: It sniffs the traffic originating from or destined to the local machine. This fetched traffic is given as input to traffic aggregator using unused CPU cycles of the machine.

b) Traffic Aggregator: It converts the network traffic into network traffic records. These records are further processed by Record Transmitter. If the Traffic Aggregator identifies a packet originating from machine in which source address (IP/MAC address) does not match with the machine address then such a traffic is considered as spoofing activity. The Traffic Aggregator notifies all such activities to Record Transmitter. All these activities are carried out using unused CPU cycles of the machine.

c) Record Transmitter: It transmits network traffic records containing errors and spoofing activity notifications to Local Aggregator Machine for further processing using unused CPU cycles of the machine.

ii. Network traffic collection by Local Aggregator

Figure 6.3 shows the arrangement of Record Receiver, Record Aggregator and Record Transmitter within Local Aggregator.

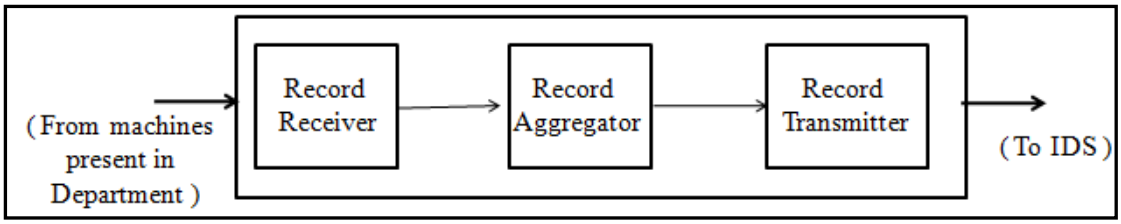


Figure 6.3: Network Traffic Collection Components Present Within Local Aggregator

a) **Record Receiver:** It receives error containing records and spoofing activity notifications, if any, from all the machines in the department and forwards these records to Record Aggregator.

b) **Record Aggregator:** It aggregates network traffic records, spoofing activity notifications and forwards these records to Record Transmitter.

c) **Record Transmitter:** It is responsible for transmission of aggregated records and spoofing activity notifications to Primary IDS. If primary IDS is not working then these records are transmitted to Secondary IDS.

iii. Aggregation and Distribution of records by Primary and Secondary IDS

The conventional IDS is modified to perform aggregation of network traffic records and distribution of the same to local machines at various departments for further analysis. Eight components responsible for aggregation and distribution of network traffic records, result aggregation and zombie machines detection are shown in Figure 6.4.

a) **Record Receiver:** It receives error containing records and spoofing activity notifications, if any, from Local Aggregator Machines and forwards these records to Record Aggregator.

b) **Record Aggregator:** It aggregates network traffic records received from Record Receiver and stores these records into Error Record Database.

c) **Machine Load Identifier:** It detects CPU utilization of all the machines present in the organization and creates a list of machines in ascending order of their CPU utilization (i.e. machines having higher number of unused CPU cycles will be on top of the list). Selects first 'N' machines from the list and forwards their IP addresses to Error Record Distributor.

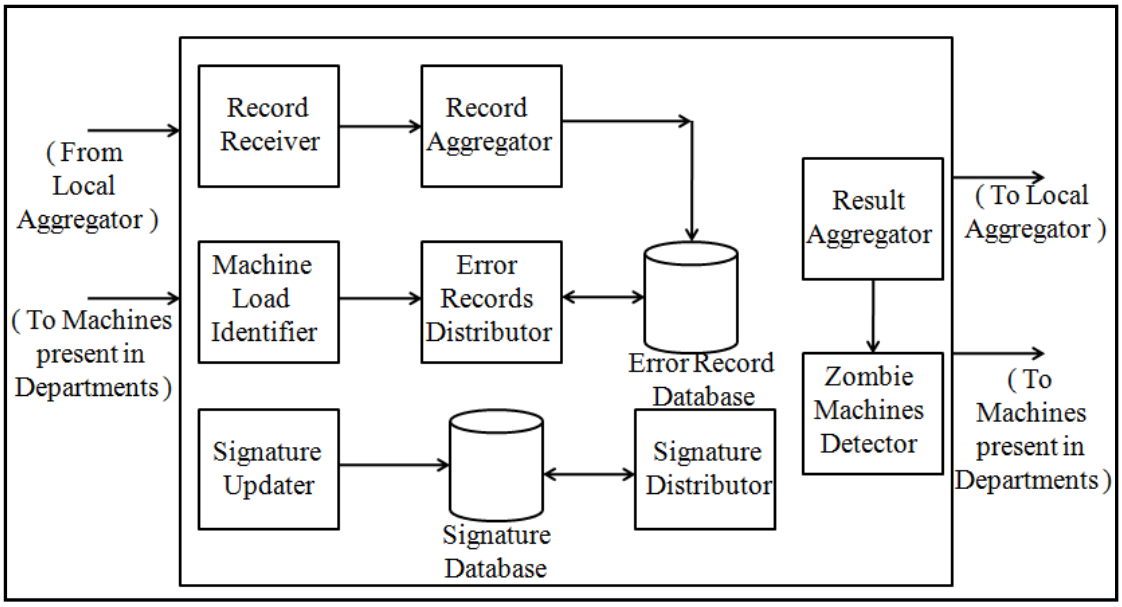


Figure 6.4: Components Present Within Primary and Secondary IDS

d) **Error Record Distributor:** It divides the available error record set into 'N' equal size-non overlapping subsets and forwards that to 'N' machines specified by Machine Load Identifier for intrusion detection.

e) **Result Aggregator:** It receives the result of intrusion detection process from the machines to which error records were transmitted for analysis. It sends the list of machines within the organization which were source of attack and target of attack to the Zombie Machine Detector.

f) **Zombie Machines Detector:** It receives the list of possible zombie machines involved in the attack, target(s) and service(s) from the Result Aggregator and label them as Black Listed machines and updates the appearance count of each machine in the list. When this count crosses a preset threshold value T, the respective machine is labeled as Zombie machine.

g) **Signature Updater:** It is responsible for adding new signatures representing modified normal user behavior and new attacks.

h) **Signature Distributor:** It is responsible for sending the signatures for intrusion detection to all the machines present in the organization through Local aggregator machine.

i) **Signature Database:** It contains the signatures of network traffic representing normal user behavior and attacks.

iv. Intrusion Detection by Local Machine

Intrusion Detection module, Signature Updater and Error Record Receiver on local machine detects attacks using unused CPU cycles of the machine. The arrangement of these modules in local machine is shown in Figure 6.5.

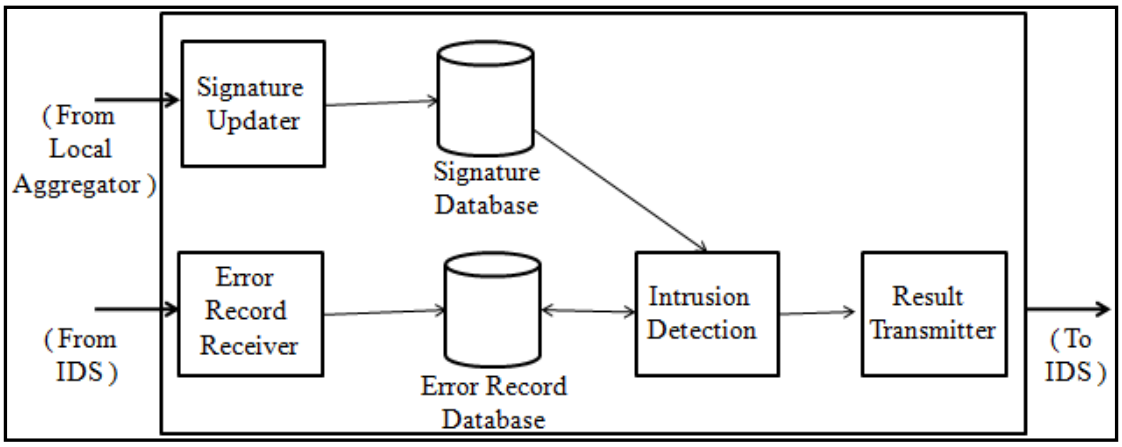


Figure 6.5: Intrusion Detection Components Present on Each Machine of the Department

a) **Error Record Receiver:** It receives error records from Primary or Secondary IDS and stores them into Error Record database using unused CPU cycles of the machine.

b) **Intrusion Detection:** It detects records which represents attack using unused CPU cycles of the machine.

c) **Result Transmitter:** It is responsible for transmitting the result of Intrusion Detection process back to Primary IDS or Secondary IDS using unused CPU cycles of the machine.

d) **Signature Updater:** It updates the signature database whenever modified normal user behavior or new attack signatures are received from Local Aggregator machine using unused CPU cycles of the machine.

v. Signature Management by Local Aggregator

Each Local Aggregator machine contains Signature Updater and Signature Database which are responsible for management of attack signatures and normal user behavior signatures. Figure 6.6 shows process of Signature Management by Local Aggregator.

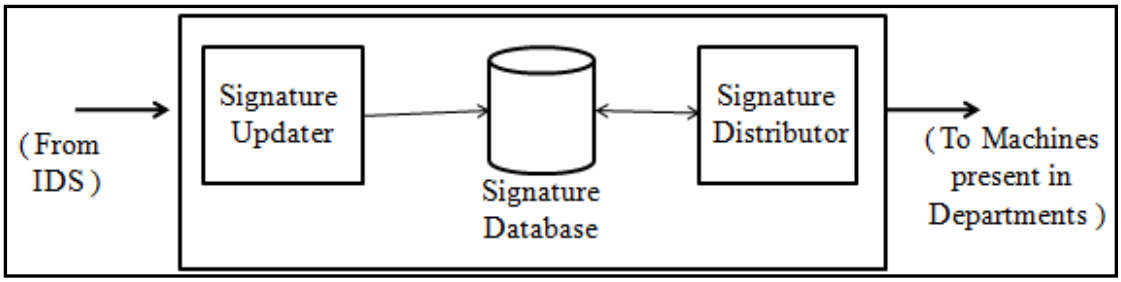


Figure 6.6: Signature Management by Local Aggregator

a) **Signature Updater:** It adds new signatures representing modified normal user behavior and new attacks. These signatures are received from Primary IDS or Secondary IDS.

b) **Signature Distributor:** It sends the signatures to all the machines in the department for updating Signature Database.

c) **Signature Database:** It contains the signatures of network traffic representing normal user behavior and attacks.

6.3 Experimental Setup and Results

The experimental setup was established in a laboratory to launch DDoS and Spoofing attacks. Figure 6.7 shows the logical placement of machines at different locations.

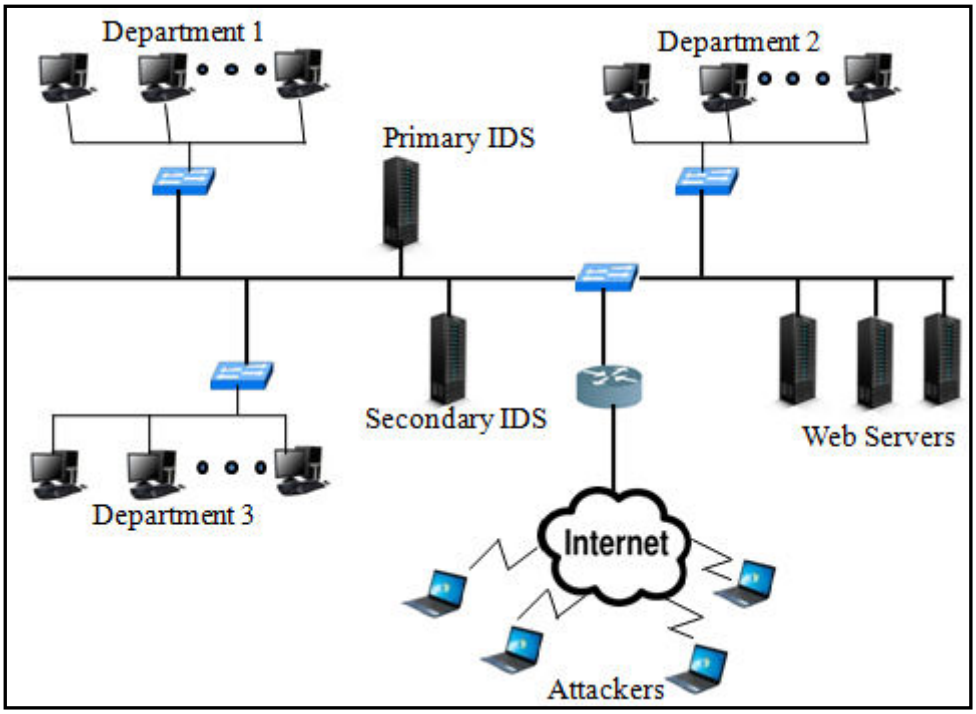


Figure 6.7: Experimental Setup for Real-time Light Weight Distributed Intrusion Detection System

The attacks can be generated within the organization and from outside of the organization. Three case scenarios are generated for launching the attacks:

i. DDoS attack using zombie machines within the organization
ii. DDoS attack using zombie machines located outside the organization
iii. Spoofing attack within the organization

For simplicity we considered three departments, each having 20 machines. Out of which one machine is used as Local Aggregator, five machines are used as zombie machines to launch DDoS attack and three machines are used to launch spoofing attack within the organization. Attacks generated outside the organization are launched by twenty machines connected via Internet to the organization. i5 workstation having 8 GB RAM are used as Web Servers, Primary IDS and Secondary IDS.

Scenario 1: DDoS attack using zombie machines within the organization

Fifteen zombie machines present within the organization are used to launch the DDOS attacks against Web Servers present within the organization. A java program is written to create HTTP Flood, Slow Read, Slow Write attacks within the organization. Table 6.1 gives distribution of records generated during zombie attacks launched within the organization. Table 6.2 gives detection accuracy for normal user behavior, HTTP flood Attack, Slow Read attack and Slow Write attack.

Table 6.1: Distribution of Records Generated During Zombie Attack Launched Within the Organization

Sr. No.	Type	Number of Records
1.	HTTP Flood	4,14,856
2.	Slow Read	16,478
3.	Slow Write	14,526
4.	Normal	1,22,476

Table 6.2: Detection Accuracy When Zombie Attack is Launched Within the Organization

Sr. No.	Type	Number of Records	Correctly Classified Records	Mis- Classified Records	Detection Accuracy (%)
1.	HTTP Flood	4,14,856	3,93,984	20,872	94.97
2.	Slow Read	16,478	15,834	644	96.09
3.	Slow Write	14,526	13,955	571	96.06
4.	Normal	1,22,476	1,18,413	4,063	96.68

All the attacks launched using zombie machines present within the organization represent the HTTP traffic. HTTP Flood attack, Slow Read attack and Slow Write attack represents the DDoS attacks which establishes connection with target HTTP server and consumes the resources required for HTTP connection in order to make the server unavailable for legitimate users. During these attacks, initial connections are established and completed as normal user connections. As a result error indicating attributes of network connection records representing these connections does not indicate any error and these records are filtered out from Intrusion Detection process by Record Transmitter component of zombie machine present within the organization. This reduces the detection accuracy of these attacks. All fifteen zombie machines used for launching the attack are detected as zombie machines.

Scenario 2: DDoS attack using zombie machines located outside the organization

Twenty machines connected to organization via Internet are used to launch HTTP Flood, Slow Read, Slow Write attack against Web Servers present within the organization. Table 6.3 gives distribution of records generated during zombie attacks launched from outside the organization. Table 6.4 gives the detection accuracy for normal user behavior, HTTP Flood attack, Slow Write attack and Slow read attack.

Table 6.3: Distribution of Records Generated During Zombie Attack Launched From Outside the Organization

Sr. No.	Type	Number of Records
1.	HTTP Flood	5,28,781
2.	Slow Read	19,697
3.	Slow Write	16,924
4.	Normal	80,543

Table 6.4: Detection Accuracy When Zombie Attack is Launched from Outside the Organization

Sr. No.	Type	Number of Records	Correctly Classified Records	Mis- Classified Records	Detection Accuracy (%)
1.	HTTP Flood	5,28,781	5,03,025	25,756	95.13
2.	Slow Read	19,697	18,999	698	96.46
3.	Slow Write	16,924	16,359	565	96.66
4.	Normal	80,543	78,395	2,148	97.33

As discussed in experimental results of scenario 1; during HTTP Flood attack, Slow Wrote attack, Slow Read attacks, initial connections are established and completed as normal user connections. As a result average, standard deviation and median values of

network connection records representing these initial intrusive connections are similar to average, standard deviation and median values of network connection records representing normal user behavior. Thus, these initial connections of HTTP Flood attack, Slow Read attack and Slow Write attack are classified as Normal user behavior. This reduces the detection accuracy of these attacks.

Scenario 3: Spoofing attacks within the organization

Nine machines within the organization are used to launch the IP spoofing and MAC spoofing attacks. These attacks are created using JAVA programs. Table 6.5 gives distribution of records generated during spoofing attacks launched within the organization.

Table 6.5: Distribution of Records Generated During Spoofing Attacks and Detection Accuracy

Sr. No.	Type	Number of Records	Correctly Classified Records
1.	IP Spoofing	1,240	1,240
2.	MAC Spoofing	1,370	1,370

During IP spoofing and MAC spoofing attacks, attacker generates network packets with forged IP or MAC address. Traffic aggregator component present within each local machine, matches the IP and MAC address of packets originating from that machine with actual IP and MAC address of that machine. Any mismatch is considered as a spoofing activity; as a result all the spoofing attacks are detected by Traffic Aggregator component.

6.4 Conclusion

This chapter describes the error detection based light weight methodology for detecting DoS and DDoS attacks in real-time using unused CPU cycles of the computing machines in the organization. Distribution of data processing tasks of IDS among the machines in the organization allows real-time attack detection without affecting the detection precision. Detection of spoofing activities within the organization helps to find the exact source of zombie machines. However, if huge flood of legitimate connection requests is generated within a small span of time; it will be detected as a DDoS attack. Thus, there is a need of intelligent detection mechanism which can identify the difference between DDoS attack and genuine flood of legitimate requests.

Chapter 7

Scalable Intrusion Detection System

Social media, on line news and search engines are now integral parts of human life to get and share the information. Their contents describes; what people are thinking, looking for and so on. This chapter describes the process of differentiating the legitimate flood of connection requests from DoS and DDoS attacks in real time by analyzing the data present on social media, news RSS feeds and keywords searched on search engines.

7.1 Introduction

When very large number of users starts using a particular service over network (e.g. shopping during online mega sale) or access a particular data present over the network (e.g. news about death of Michael Jackson) this results into a huge flood of network traffic towards servers in short period of time. Figure 7.1 and Figure 7.2 describes how Michael Jackson's death resulted into high flood of legitimate traffic.

Figure 7.1: Death of Michael Jackson Affects the Internet Performance

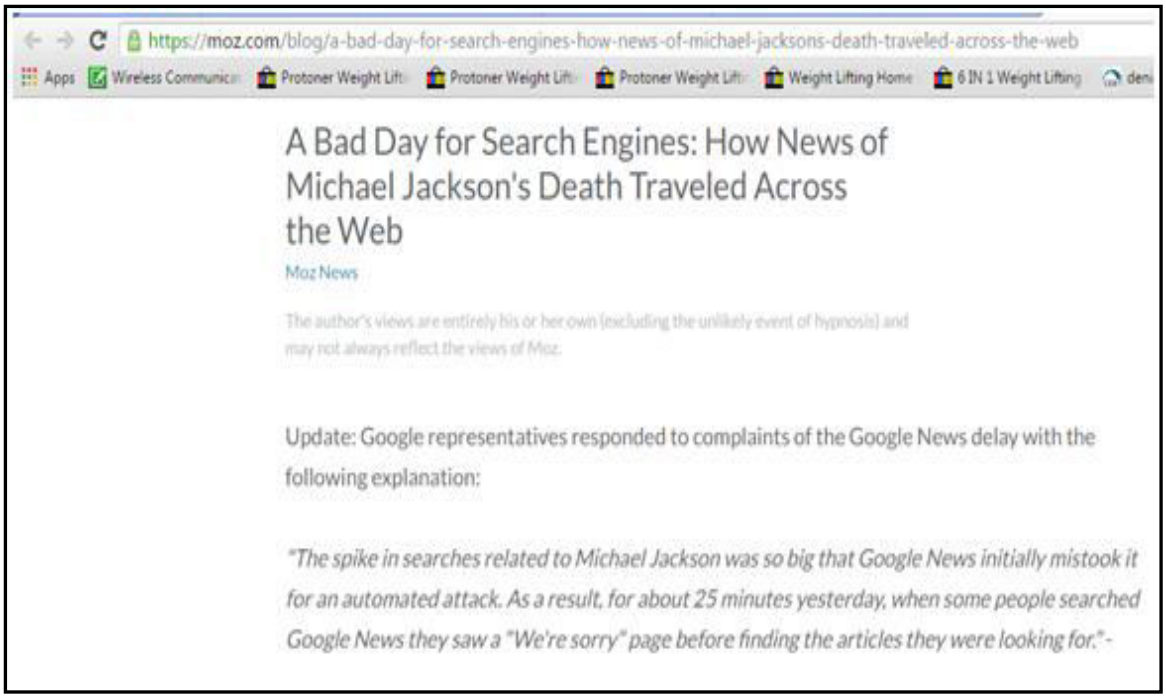


Figure 7.2: Death of Michael Jackson: A Bad Day for Search Engines

In general IDS treat this flood legitimate traffic in short span of time as a DoS/DDoS attack. To protect the system from this attack, IDS either blocks the source IP addresses of connection requests or uses captcha as shown in Figure 7.3 to differentiate legitimate user from zombie attacker (Mehra et al., 2011).

Figure 7.3: Captcha Used to Differentiate Legitimate User from Zombie Attacker

7.2 Proposed Scalable Intrusion Detection System

The hosting network servers do not have the capability to handle heavy flood of traffic generated in a short time span. The IDS treats such requests as DDoS attack leading to breakdown of services. Proposed system describes the process of handling such situations by implementing Scalable Intelligent Network Intrusion Detection System using Cloud Servers. To process flood of requests; virtual servers and IDS are created on the cloud which is situated outside the organization and network traffic is redirected to these servers. The system consists of five processes as given below:

i. Process for creation of Hot Logs
ii. Process for scaling up available network servers
iii. Process for scaling down available network servers
iv. Process of communication between Network servers and Network Intrusion Detection System
v. Process of differentiating genuine flood of network traffic from Distributed Denial of Service attack

Process 1 and 4 are continuously executed after fixed interval, whereas process 2 and 3 are executed as and when servers detects heavy network traffic.

i. Creation of Hot Logs

The process of creation of Hot Logs consists of three sub processes as shown in Figure 7.4.

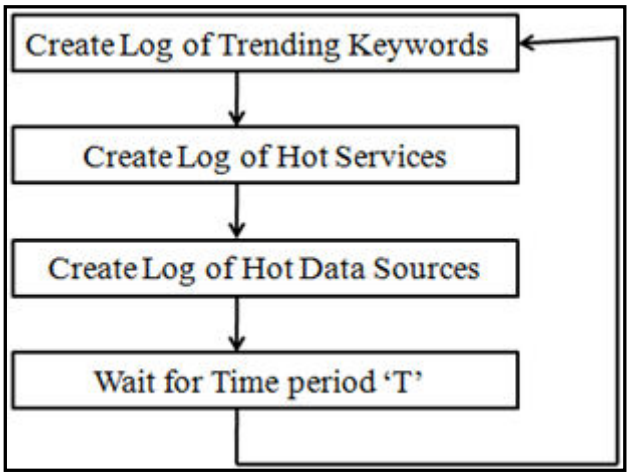


Figure 7.4: Creation of Hot Logs

a) **Creating Log of Trending Keywords:** This is the first step in creating Hot Logs. Trending keywords present in News RSS feed, top searched trending keywords on search engines, top trending keywords on social media are combined together to create a Log of Trending Keywords. Figure 7.5 shows the process of creating Log of Trending Keywords. This log is then used to find the Hot Services and Hot Data Sources present on the Network Servers hosted by organization.

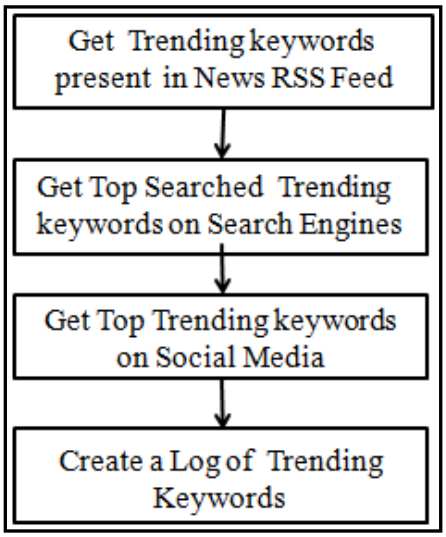


Figure 7.5: Creation of Log of Trending Keywords

b) **Creating Log of Hot Services**

The process of creating Log of Hot Services is shown in Figure 7.6.

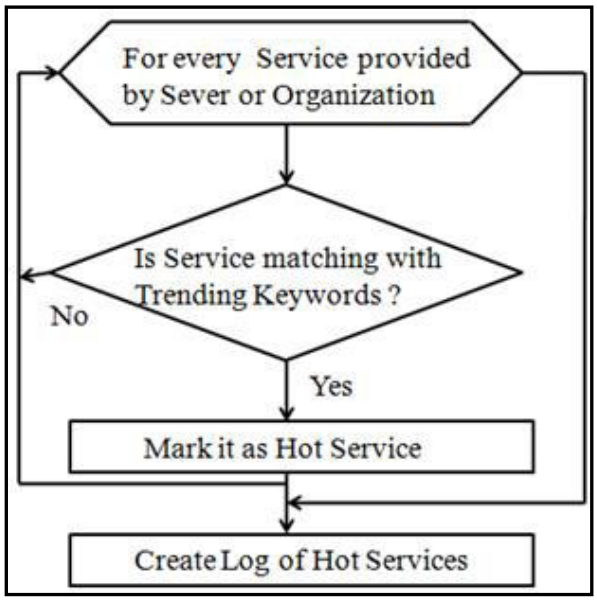


Figure 7.6: Creation of Log of Hot Services

List of keywords describing the services hosted by the organization are maintained on the servers. For example, An organization hosts a service which gives current prices of electronic gadgets available on different online shopping sites. Keywords describing this service may be mobile price, laptop price, minimum price, etc. If keyword(s) describing the service hosted by organization matches with the trending keywords(s), then that service is marked as Hot Service and Other services are considered a normal services.

c) Creating Log of Hot Data Sources

The process of creating Log of Hot Data Sources is shown in Figure 7.7.

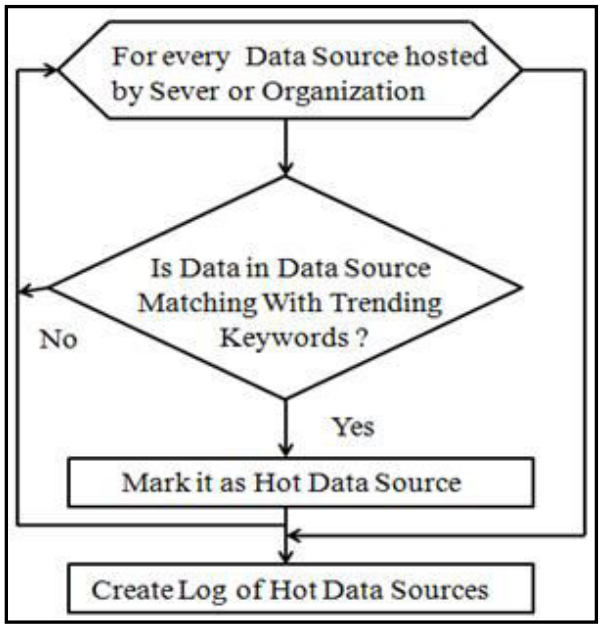


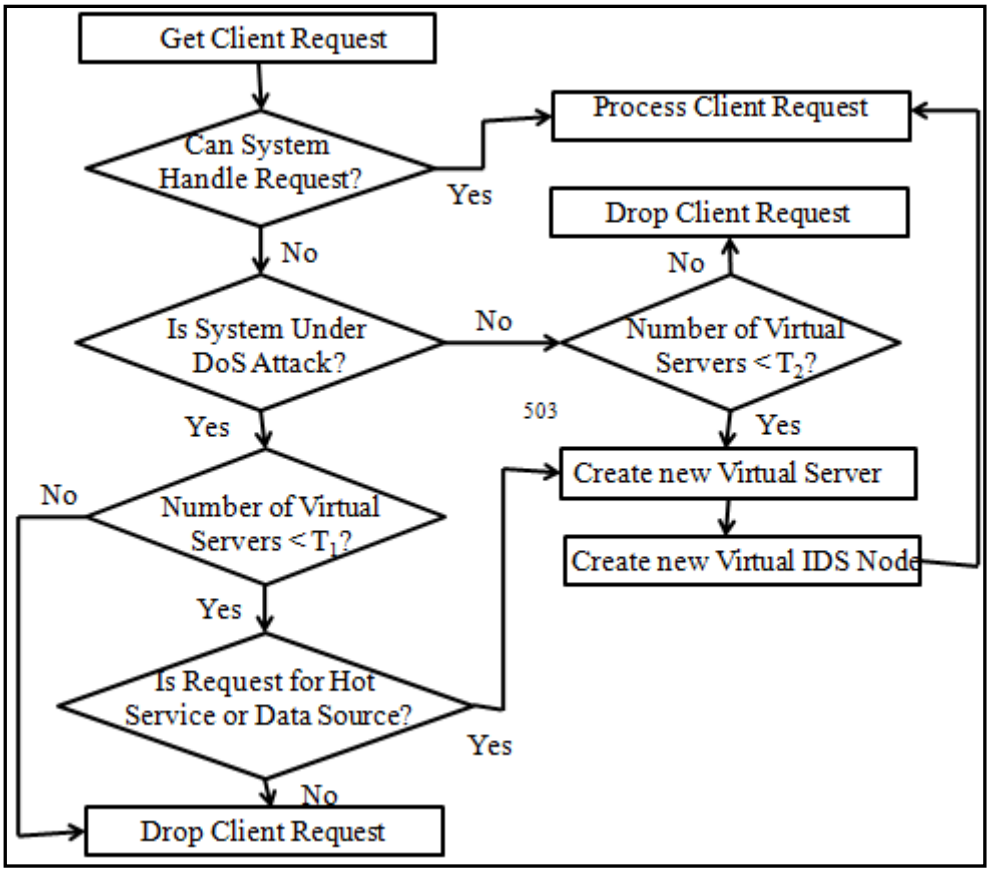
Figure 7.7: Creation of Log of Hot Data Sources

If information related to the files, web pages, images, videos, audios hosted on the servers in the organization are extensively searched by internet users on search engines or information related to them is in trending topic on social media and news RSS feed; then those files, web pages, images, audios, videos are marked as hot data sources and others are considered as normal data sources.

ii. Scaling up available network servers

The process for scaling up available network servers shown in Figure 7.8. When load on available servers increases above predefined threshold; this process is executed at regular intervals. After receiving a request from client the proposed system first checks for the available capacity of the local servers for processing

the request. If the local servers are unable to process the request then proposed system check for DoS, DDoS attack. If servers are not under attack and number of currently available servers is less then pre-defined threshold (T2) then new virtual server and corresponding Intrusion Detection System node is created on cloud to handle the request from client otherwise drop the request. If servers are under attack but number of currently available servers are less than pre-defined threshold (T1) and user is requesting for Hot service or Hot Data then create a new virtual server and corresponding Intrusion detection System node on cloud to handle request from client otherwise drop the request.



503

Figure 7.8: Scaling Up Available Network Servers and IDS

iii. Scaling down available network servers

Figure 7.9 shows the process of scaling down available servers. This process checks for the number of available servers (T3) and the number of Servers required to process the requests from Clients (T4). If number of available Servers is more than the required Servers, then, Virtual Servers and corresponding IDS node on the cloud are removed from the system one by one till number of available servers are more than required Servers.

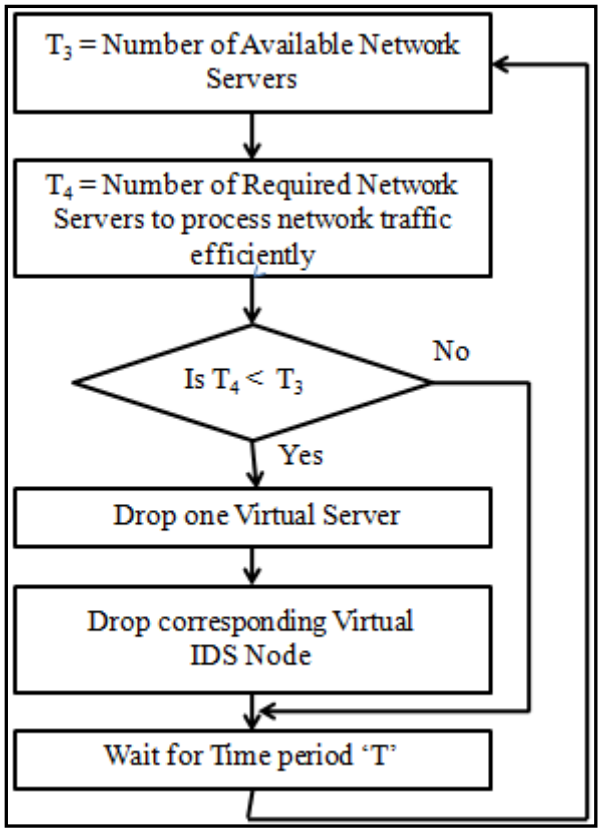


Figure 7.9: Scaling Down Available Network Servers and IDS

iv. Communication between Servers and IDS

Figure 7.10 shows the process of communication between Servers and IDS node. Servers converts the network traffic into network connection records at regular interval 'α' and forwards them to corresponding Intrusion Detection node for attacks detection. A real-time system is a system which responds to externally generated input stimuli within a finite and specified period (say 'β'). If value of 'α' is set in such a way that generated network connection records can be easily processed within (β - α) time period then real time Intrusion Detection is possible.

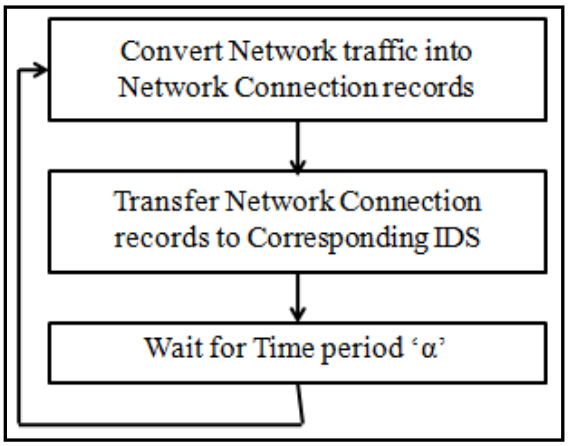


Figure 7.10: Communication Between Servers and Intrusion Detection System

v. Differentiating genuine flood of network traffic from DDoS attack

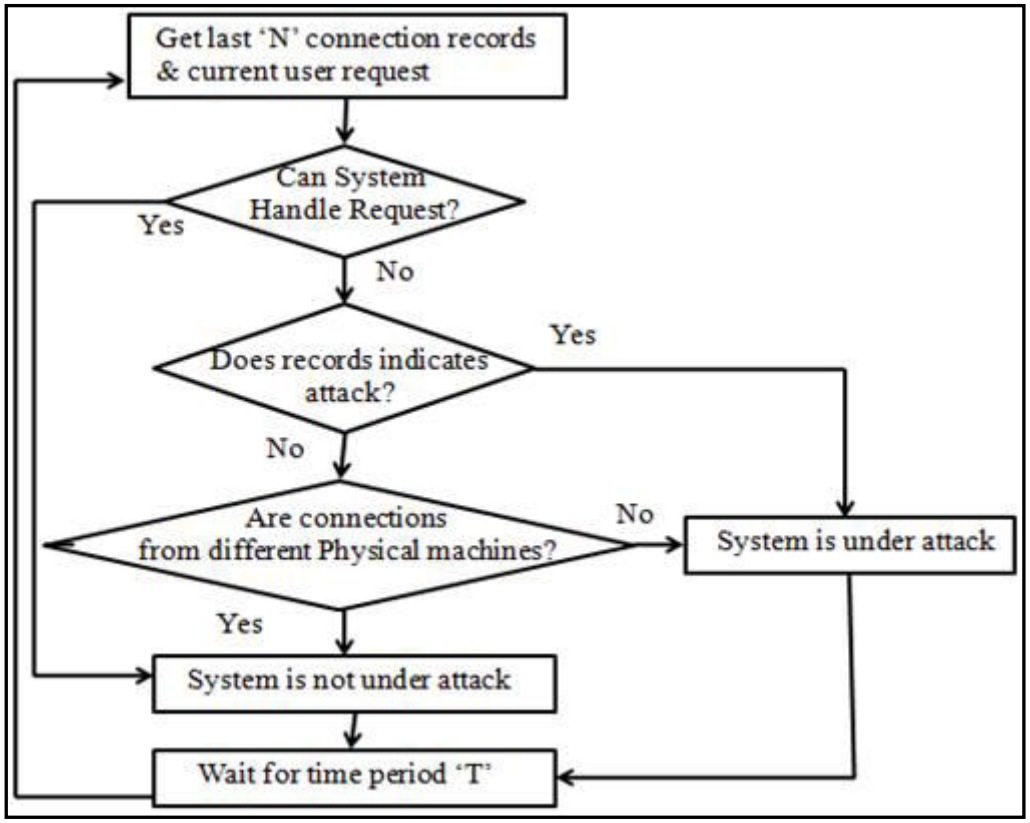


Figure 7.11: Differentiating Genuine Flood of Network Traffic from DDoS Attack

Process of differentiating genuine flood of network traffic from DDoS attack is illustrated in Figure 7.11. The process for differentiating genuine flood of network traffic from DDoS attack is based on a fact that; attacker sends huge number of intrusive connections from zombie machines to launch a DDoS attack. As a result, many connections are originating from same set of machines.

This process checks last 'N' number of network connection records. If last 'N' network connection records are not classified as attack by the IDS and requests are originating from different machines; the system is considered as attack free. If last 'N' network connection records are classified by IDS as attack or network connection requests are originating from same set machines then system is considered to be under DDoS attack.

This process is repeated at regular interval (T). If the waiting time (T) is small then real time Intrusion Detection is possible.

7.3 Experimental Setup and Results

The experimental setup for implementation of proposed Scalable Intrusion Detection System is shown in Figure 7.12.

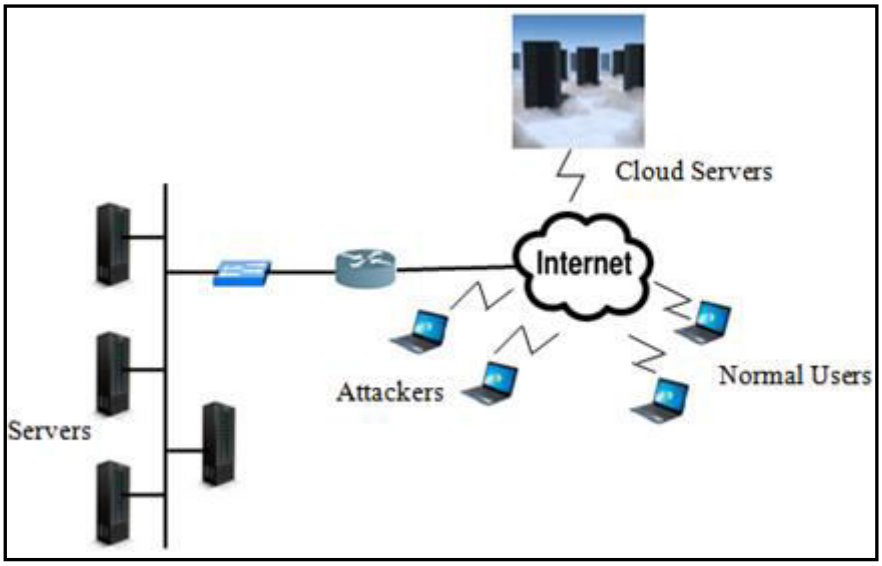


Figure 7.12: Experimental Setup for Scalable Intrusion Detection System

Three file sharing server are implemented to serve five concurrent requests each for file transfer. Threshold for creating number of virtual servers in the presence of DoS, DDoS attack (T1) is set to seven and threshold for creating number of virtual servers in the presence of genuine flood (T2) is set to twelve. Files containing information of Roger Federer, Sachin Tendulkar and Virat Kohli are marked as Hot data sources. User requests for hot data sources are considered as hot requests and requests for other files are considered as normal requests.

Two scenarios are used to test the proposed methodology:

iv. Flood of genuine user requests
v. Genuine user requests in the presence of DDoS attack

Scenario 1: Flood of genuine user requests

Fifty five machines connected to the servers over network are used to send requests for files related to Roger Federer, Sachin Tendulkar and Virat Kohli (i.e. Hot data sources) to the servers and five machines are used to send requests for other files. Table 7.1 shows the details of requests handled during the flood of genuine requests. During the genuine flood of requests; twelve virtual servers are created gradually to handle the user request. There are fifteen servers (twelve virtual and three physical) to

process the requests. These fifteen servers can process maximum 75 concurrent requests; as each server can process 5 concurrent requests. When concurrent requests are increased beyond 75; these requests are discarded by the servers. As a result, some requests during the genuine flood are not handled by server. As shown in Table 7.1, 91.14% hot requests and 89.37% normal requests are handled by proposed scalable architecture.

Table 7.1: Requests Handled During Flood of Genuine Requests

Sr. No.	Type of Request	Number of Requests	Number of Handled Requests	Handled Requests (%)
1	Hot Request	1400	1276	91.14
2	Normal Request	160	141	89.37

Scenario 2: Genuine user requests in the presence of DDoS attack

Thirty five machines connected to the servers over network are used to launch the DDoS attack by sending a flood of requests for files to the servers. Eighteen machines are used to send the requests for hot files and seven machines are used to send the requests for other files. Table 7.2 shows the details of genuine hot requests and genuine normal requests handled during the DDoS attack.

Table 7.2: Genuine Requests Handled During DDoS Attack

Sr. No.	Type of Request	Number of Requests	Number of Handled Requests	Handled Requests (%)
1	Genuine Hot Request	900	647	71.89
2	Genuine Normal Request	120	82	68.33

During the DDoS attack; seven virtual servers are created gradually to handle the user requests. There are ten servers (seven virtual and three physical) to process the requests. This cluster of ten servers can process maximum 50 concurrent requests. When servers are under DDoS attack and they are unable to process hot request; they creates one virtual server to handle five extra user connections, however, machine involved in DDoS attacks consumes few of these extra connections. As a result, less number of hot requests are handled. As shown in Table 7.2, 71.89% of hot requests are handled during DDoS attack.

When servers are under DDoS attack and they are unable to process normal request; this request is discarded instead of creating a new virtual server to handle it. As a

result, less number of normal requests are handled. As shown in Table 7.2, 68.33% of normal requests are handled during DDoS attack.

7.4 Conclusion

This chapter presents a process to differentiate legitimate flood of connection requests from DoS and DDoS attacks using analysis of information available on social media, news RSS feeds and search engine query trends. Social media data analysis can be used for proactive precaution by creating extra cloud servers to host hot services and hot data sources. Using cloud servers to handle the huge connection requests and data processing for attack detection; allows design of scalable system for real time intrusion detection at low cost.

Chapter 8

Conclusion and Future Work

This report presents various methods for known, novel DoS and DDoS attack detection. The KDD 99, CDMC 2012 and dataset generated in the laboratory representing HTTP flood, Slow read, Slow Write attacks have been used for experimentation.

Researchers have carried out huge amount of research over last two decades for designing IDS. Brief review of solutions proposed by researchers for implementing different categories of IDS and their limitations are presented.

An approach to design an Adaptive Ensemble of Classifiers to detect known DoS and DDoS attacks is described. It adopts the changes in normal user behavior, DoS and DDoS attack strategies and according changes the base classifiers within it to improve the detection accuracy. The detection accuracy obtained is 99.81%, 97.40% and 96.44% respectively for KDD 99, CDMC 2012 and dataset generated in laboratory.

This report then extends the use of Adaptive Ensemble of Classifiers in combination with Signature detection based IDS for detection of known, novel DoS and DDoS attacks. Signature detection based IDS detects known DoS, DDoS attacks, whereas, Adaptive Ensemble of Classifiers is used to identify patterns of novel DoS and DDoS attacks. These patterns are further used to generate the signatures of novel DoS and DDoS attacks. The novel attack detection accuracy of 99.16%, 83.22%, 98.39% is obtained on KDD 99, CDMC 2012 and dataset generated in laboratory respectively.

To process heavy network traffic in real time Distributed IDS has been designed. It filter outs the network connection records from attack detection process by analyzing the error indicating attributes of network connection records. Then, it processes the remaining network connection records using unused CPU cycles of computing machines within the organization. This reduces the processing power, resource requirement of attack detection system and enables real time attack detections. The detection accuracy of 95.39%, 95.48% and 100% is obtained for detection of attacks originating within the organization, attacks originating outside the organization and

spoofing activities within the organization respectively. Detection of spoofing activities within the organization helps to find the exact source of zombie machines.

Finally, a methodology has been suggested to differentiate legitimate flood of connection requests from DoS and DDoS attacks using analysis of information available on social media, news RSS feeds and search engine query trends. The infrastructure required to process the network traffic and attack detection is scaled up and scaled down gradually by analyzing the change in network traffic volume. Using virtual servers to scale the required infrastructure; allows design of scalable system for real time intrusion detection at low cost.

Future Work:

Training datasets for IDS are highly imbalanced datasets. As a result, for attacks like 'Back' which have very less number of representative records in training dataset; detection accuracy is low. The methodology can be developed to improve the detection accuracy of these attacks.

Adaptive Ensemble of Classifier considers any network connection as novel attack; if it is deviating significantly from normal user behavior and known attacks. This assumption may fail some times; as normal user behavior may change over a period of time or abruptly. The mechanism can be proposed to differentiate such deviations from DoS and DDoS attacks.

DDoS Attacks similar to HTTP Flood attack, Slow Read attack and Slow Write attack which establishes connection with target HTTP server and consumes the resources required for HTTP connection in order to make the server unavailable for legitimate users. During these attacks, initial connections are established and completed as normal user connections. These connections are not detected as attack by IDS system. The mechanism can be proposed to detect these initial connections as intrusive connections.

Research Publications

Papers Published in International Journals

1. Vijay D. Katkar, S. G. Bhirud (2014) "HIDS for Detection of Novel Distributed Denial of Service attacks using Fuzzy Mining, One Pass Apriori & Ensemble of Naïve Bayesian Classifiers" European Journal of Scientific Research 119(2): 214-223
2. Vijay D. Katkar, S. G. Bhirud (2013) "Detection of Distributed Denial of Service attacks using Ensemble of Fuzzy Logic and Naïve Bayesian Classifiers" European Journal of Scientific Research 105(1): 166-174
3. Vijay D. Katkar, S. G. Bhirud (2012) "Novel DoS/DDoS Attack Detection and Signature Generation" International Journal of Computer Applications 47(10): 18-24
4. Vijay D. Katkar, S. G. Bhirud (2010) "Novel Architecture for Intrusion-Tolerant Distributed Intrusion Detection System using Packet Filter Firewall and State Transition Tables" International Journal of Computer Applications 8(11): 29-32

Papers Published in International Conferences

1. Vijay D. Katkar, S. G. Bhirud (2011) "Light weight approach for IP-ARP spoofing detection and prevention" Second Asian Himalayas International Conference on Internet (AH-ICI): 1-5
2. Vijay D. Katkar, S. G. Bhirud (2011) "SYN flood attack prevention using main-memory database management system" Second Asian Himalayas International Conference on Internet (AH-ICI): 1-6

Patents Filed

1. Title: Light weight distributed intrusion detection system.
   Application Number: 2029/MUM/2015
2. Title: Scalable Intelligent Network Intrusion detection system using cloud server
   Application Number: 201621001495

Bibliography

Aburomman A.A. And Reaz M.B.I. (2016) "A Novel SVM-KNN-PSO Ensemble Method for Intrusion Detection System" Applied Soft Computing (38): 360-372.

Al-Jarrah O. Y., Alhussein O., Yoo P.D., Muhaidat S., Taha K. and Kim K. (2016) "Data Randomization and Cluster-Based Partitioning for Botnet Intrusion Detection" IEEE Transactions on Cybernetics 46(8): 1796-1806

Baker Z.K. And Prasanna, V.K. (2005), "A Computationally Efficient Engine for Flexible Intrusion Detection" IEEE Transactions On Very Large Scale Integration (Vlsi) Systems 13(10): 1179-1189.

Bandre S.R. And Nandimath J.N. (2015), "Design Consideration of Network Intrusion Detection System using Hadoop And GPGPU" International Conference On Pervasive Computing (ICPC): 1-6.

Bankovic Z., Stepanovic D., Bojanic S. And Nieto-Taladriz O. (2007) "Improving Network Security using Genetic Algorithm Approach" Computers and Electrical Engineering 33(5): 438-451.

Barker I. (2015) The evolution of DDoS attacks in 2015 http://www.itproportal.com/ 2015/12/02/the-evolution-of-ddos-attacks-in-2015/.

Bulajoul W., James A. and Pannu M. (2015) "Improving Network Intrusion Detection System Performance Through Quality Of Service Configuration And Parallel Technology" Journal of Computer and System Sciences 81(6): 981-999.

Cabrera, J. B., Gutierrez, C. and Mehra, R. K. (2008) "Ensemble Methods for Anomaly Detection and Distributed Intrusion Detection in Mobile Ad-Hoc Networks" Information Fusion 9(1): 96-119.

Catania C. A., Brombergb F. and Garino C. G. (2012) "An Autonomous Labeling Approach to Support Vector Machines Algorithms for Network Traffic Anomaly Detection" Expert Systems with Applications 39(2): 1822-1829.

Chavan S., Shah K., Dave N., Mukherjee S., Abraham A. and Sanyal S. (2004), "Adaptive Neuro-Fuzzy Intrusion Detection Systems" IEEE International Conference

on Information Technology-Coding and Computing: 70-74.

Chebrolu S., Abrahama A. and Thomas J. P. (2005) "Feature Deduction and Ensemble Design of Intrusion Detection Systems" Computers and Security 24(4): 295-307.

Chen C.M., Chen Y.L. and Lin H.C. (2010) "An Efficient Network Intrusion Detection" Computer Communications 33(4): 477-484.

Chen T., Zhang X., Jin S. and Kim O. (2014) "Efficient classification using parallel and scalable compressed model and its application on intrusion detection" Expert Systems with Applications 41(13): 5972-5983

Christoph G.G., Jackson K.A., Neuman M.C., Siciliano C.L.B., Simmonds D.D., Stallings C.A. and Thompson J.L. (1995) "Unicorn: Misuse Detection for Unicos" IEEE ACM SC95 Conference Supercomputing: 56-79.

Creech G. and Hu J. (2014) "A Semantic Approach to Host-Based Intrusion Detection Systems using Contiguous and Discontiguous System Call Patterns" IEEE Transactions On Computers 63(4): 807-819.

Cronin B. and Wang X. (2013) "Hardware Acceleration of Regular Expression Repetitions in Deep Packet Inspection" IET Information Security 7(4): 1751-8709.

Dangeloa G., Palmieri G., Ficco M. and Rampone S. (2015) "An Uncertainty-Managing Batch Relevance-based Approach to Network Anomaly Detection" Applied Soft Computing 36: 408-418.

Das A., Nguyen D., Zambreno J., Memik G. and Choudhary A. (2008), "An FPGA-based Network Intrusion Detection Architecture" IEEE Transactions on Information Forensics and Security 3(1): 118-132.

Denning D. E. (1987) "An Intrusion-Detection Model" IEEE Transactions on Software Engineering 13(2): 222-232.

Depren O., Topallar M., Anarim E. and Ciliz M.K. (2005) "An Intelligent Intrusion Detection System (IDS) for Anomaly and Misuse Detection in Computer Networks" Expert Systems with Applications 29(4): 713-722.

Dickerson J.E. and Dickerson J.A. (2000) "Fuzzy Network Profiling for Intrusion Detection" IEEE 19th International Conference of the North American Fuzzy Information Processing Society: 301-306.

Elbasiony R.M., Sallam E.A., Eltobely T.E. and Fahmy M.M. (2013) "A Hybrid

Network Intrusion Detection Framework based on Random Forests and Weighted K-Means" Ain Shams Engineering Journal 4: 753-762.

Endler D. (1998) "Intrusion Detection: Applying Machine Learning to Solaris Audit Data" IEEE 14th Annual Computer Security Applications Conference: 268-279.

Erdem O. (2016) "Tree-based String Pattern Matching on FPGA" Computers and Electrical Engineering 49: 117-133.

Erfani S.M., Rajasegarar S. and Karunasekera S. (2016) "High-Dimensional and Large-Scale Anomaly Detection using a Linear one-class SVM with Deep Learning" Pattern Recognition 58: 121-134.

Eric Y.K., Chan H.W., Chan K.M., Chan P.S., Chanson S.T., Cheung M.H., Chong C.F., Chow K.P., Hui A.K.T., Hui L.C.K., Ip S.K., Lam C.K., Lau W.C., Pun K.H., Tsang Y.F., Tsang W.W., Tso C.W., Yeung D.Y., Yiu S.M., Yu K.Y., and Ju W. (2006) "Intrusion Detection Routers: Design, Implementation and Evaluation using an Experimental Testbed" IEEE Journal on Selected Areas in Communications 24(10): 1889-1900.

Filippi E., Costa M. and E. Pasero (1994) "Multi-Layer Perceptron Ensembles for Increased Performance and Fault-Tolerance in Pattern Recognition Tasks" IEEE World Congress on Computational Intelligence 5: 2901-2906.

Fiore U., Palmieri F., Castiglione A. and Santis A.D. (2013) "Network Anomaly Detection with the Restricted Boltzmann Machine" Neurocomputing 122: 13-23.

Franois J., Aib I. and Boutaba R. (2012) "Firecol: A Collaborative Protection Network for the Detection of Flooding DDoS Attacks" IEEE ACM Transactions on Networking 20(6): 1828-1841.

Fung C.J., Zhang J. and Boutaba R. (2012) "Effective Acquaintance Management based on Bayesian Learning for Distributed Intrusion Detection Networks" IEEE Transactions on Network and Service Management 9(3): 320-332.

Garca K.A., Monroy R., Trejo L.A., Mex-Perera, C. and Aguirre E. (2012) "Analyzing Log Files for Postmortem Intrusion Detection" IEEE Transactions on Systems, MAN, and Cybernetics 42(6): 1690-1704.

Garcia-Teodoro P., Diaz-Verdejo J., Tapiador J. and Salazar-Hernandez R. (2015), "Automatic Generation of HTTP Intrusion Signatures by Selective Identification of

Anomalies" Computers and Security 55: 159-174.

Gaspary L.P., Sanchez R.N., Antunes D.W. and Meneghetti E. (2005) "A SNMP-based Platform for Distributed Stateful Intrusion Detection in Enterprise Networks" IEEE Journal on Selected Areas in Communications 23(10): 1973-1982.

Geetha K. and Sreenath N. (2014) "SYN flooding attack - Identification and analysis" International Conference on Information Communication and Embedded Systems (ICICES): 1-7

Govindarajan M. and Chandrasekaran R. (2011) "Intrusion Detection Using Neural Based Hybrid Classification Methods" Computer Networks 55(8): 1662-1671.

Hansen L.K. and Salamon P. (1990) "Neural Network Ensembles" IEEE Transactions on Pattern Analysis and Machine Intelligence 12(10): 993-1001.

Honan B. (2015) DDoS attacks take down RBS, Ulster bank, and Natwest online systems. http://www.csoonline.com/article/2955693/cyber-attacks-espionage/ddos-attacks-take-down-rbs-ulster-bank-and-natwest-online-systems.html

Hu J., Yu X., Qiu D. and Chen H.H. (2009) "A Simple and Efficient Hidden Markov Model Scheme for Host-based Anomaly Intrusion Detection" IEEE Network 23(1): 42-47.

Hu W., Gao J., Wang Y., Wu O. and Maybank S. (2014) "Online Adaboost-based Parameterized Methods for Dynamic Distributed Network Intrusion Detection" IEEE Transactions on Cybernetics 44(1): 66-82.

Huang S., Wang B., Qiu J., Yao J., Wang G. and Yu G. (2016) "Parallel Ensemble of Online Sequential Extreme Learning Machine Based on Mapreduce" Neurocomputing 174(1): 352-367.

Hutchings B.L., Franklin R. and Carver D. (2002) "Assisting Network Intrusion Detection with Reconfigurable Hardware" 10th Annual IEEE Symposium On Field-Programmable Custom Computing Machines: 111-120.

Jamdagni A., Tan Z., He X., Nanda P. and Liu R.P. (2013) "Repids: A Multi Tier Real-Time Payload-based Intrusion Detection System" Computer Networks 57(3): 811-824.

Jiang G. (2016) CVE-2016-4117: Flash Zero-Day exploited in the wild « threat research Blog https://www.fireeye.com/blog/threat-research/2016/05/cve-2016-4117-flash-zero-day.html

Jiang S., Song X., Wang H., Han J.J. and Li Q.H. (2006) "A Clustering-based Method for Unsupervised Intrusion Detections" Pattern Recognition Letters 27(7): 802-810.

Jiang W., Song H. and Dai Y. (2005) "Real-time Intrusion Detection for High-Speed Networks" Computers and Security 24(4): 287-294.

Kemmerer R.A. and Vigna G. (2005) "Hi-DRA: Intrusion Detection for Internet Security" Proceedings of the IEEE 93(10): 1848-1857.

Khreich W., Granger E., Miri A. and Sabourin R. (2012) "Adaptive ROC-based Ensembles of HMMs Applied to Anomaly Detection" Pattern Recognition 45(1): 208-230.

Kim G., Lee S. and Kim S. (2014) "A Novel Hybrid Intrusion Detection Method Integrating Anomaly Detection with Misuse Detection" Expert Systems with Applications 41: 1690-1700.

Kosoresow A.P. and Hofmeyr S.A. (1997) "Intrusion Detection via System Call Traces" IEEE Software 14(5): 35-42.

La H.E.D. Hoz E.D.L., Ortiz A., Ortega J. and Prieto B. (2015) "PCA Filtering and Probabilistic SOM for Network Intrusion Detection" Neurocomputing 164(21): 71-81.

Li M., Li J. and Zhao W. (2008) "Simulation Study of Flood Attacking of DDOS" International Conference on Internet Computing in Science and Engineering: 286-293

Li P., Wu X., Hu X. and Wang H. (2015) "Learning Concept-Drifting Data Streams with Random Ensemble Decision Trees" Neurocomputing 166: 68-83.

Li Q.H., Xtong J.J. and Yang H.B. (2003) "An Efficient Mining Algorithm for Frequent Pattern in Intrusion Detection" IEEE 2nd International Conference on Machine Learning and Cybernetics: 138-142.

Li T. and Feng X.N. (2015) "Novel Heuristic Dual-Ant Clustering Algorithm for Network Intrusion Outliers Detection" International Journal for Light and Electron Optics 126: 494-497.

Liao Y. and Vemuri V.R. (2002) "Use of K-Nearest Neighbor Classifier for Intrusion Detection" Computers and Security 21(5): 439-448.

Lunt T.F. (1989) "Real-time Intrusion Detection" 34th IEEE Computer Society International Conference: Intellectual Leverage: 348-353.

Lunt T.F. and Jagannathan R. (1988) "A Prototype Real-time Intrusion-Detection Expert

System" IEEE Symposium on Security and Privacy: 59-66

Mabu S., Chen C., Lu N., Shimada K. and Hirasawa K. (2010) "An Intrusion-Detection Model based on Fuzzy Class Association Rule Mining using Genetic Network Programming" IEEE Transactions on Systems, MAN, and Cybernetics 41(1): 130-139.

Maci-Prez F., Mora-Gimeno F.J., Marcos-Jorquera D., Gil-Martnez-Abarca J.A., Ramos-Morillo H. and Lorenzo-Fonseca I. (2011) "Network Intrusion Detection System Embedded on a Smart Sensor" IEEE Transactions on Industrial Electronics 58(3): 722-732.

Magalhaes R.M. (2003) Host-based IDS vs Network-based IDS (part 1) http://www. windowsecurity.com/articles- tutorials/intrusion_detection/Hids_vs_Nids_Part1.html.

Maggi F., Matteucci M. and Zanero S. (2010) "Detecting Intrusions through System Call Sequence and Argument Analysis" IEEE Transactions on Dependable and Secure Computing 7(4): 381-395.

Mahoney M.V. and Chan P.K. (2003) "An Analysis of the 1999 DARPA/Lincoln Laboratory Evaluation Data for Network Anomaly Detection" Lecture Notes in Computer Science 2820: 220-237.

Mayo K. and Newcomb P. (2009) The birth of the world wide web: An oral history of the Internet http://www.vanityfair.com/news/2008/07/internet200807

Mchugh J. (2000) "Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed By Lincoln Laboratory" ACM Transactions on Information and System Security 3(4) 262-294.

Mehmood Y., Shibli M.A., Kanwal A. and Masood R. (2015) "Distributed Intrusion Detection System using Mobile Agents in Cloud Computing Environment" Conference on Information Assurance and Cyber Security: 1-8.

Mehra M., Agarwal M., Pawar R. and Shah D. (2011) "Mitigating denial of service attack using CAPTCHA mechanism" International Conference and Workshop on Emerging Trends in Technology: 284-287

Meng W., Li W. and Kwok L.F. (2014) "EFM: Enhancing the Performance of Signature based Network Intrusion Detection Systems using Enhanced Filter Mechanism" Computers and Security 43: 189-204.

Meng Y., Li W. and Kwok L.F. (2013) "Towards Adaptive Character Frequency-based

Exclusive Signature Matching Scheme and its Applications in Distributed Intrusion Detection", Computer Networks 57: 3630-3640.

Miller P. and Inoue A. (2003) "Collaborative Intrusion Detection System" 22nd International Conference of the North American Fuzzy Information Processing Society: 519-524.

Modia C.N., Patela D.R., Patelb A. and Rajarajanb M. (2012) "Integrating Signature Apriori based Network Intrusion Detection System (NIDS) in Cloud Computing" Procedia Technology 6: 905-912.

Mylavarapu G., Thomas J. and TK A.K. (2015) "Real-Time Hybrid Intrusion Detection System using Apache Storm" IEEE High Performance Computing and Communications: 1436-1441.

Ouyang M.G., Wang W.N. and Zhang Y.T. (2002) "A Fuzzy Comprehensive Evaluation based Distributed Intrusion Detection" First International Conference on Machine Learning and Cybernetics: 281-284.

Palermo E. (2015) 10 worst data breaches of all time. http://www.tomsguide.com/us/biggest-data-breaches,news-19083.html

Pao D., Or N.L. and Cheung R.C. (2013) "A Memory-Based NFA Regular Expression Match Engine for Signature-based Intrusion Detection" Computer Communications 36: 1255-1267.

Park J., Iwai K., Tanaka H. and Kurokawa T. (2014) "Analysis of Slow Read DoS attack" International Symposium on Information Theory and its Applications: 60-64

Patcha A. and Park J.M. (2007) "Network Anomaly Detection with Incomplete Audit Data" Computer Networks 51(13): 3935-3955.

Pegram, B. (2016) 10 surprising Cyber security facts that may affect your online safety [Updated] - Heimdal security Blog. https://heimdalsecurity.com/blog/10-surprising-cyber-security-facts-that-may-affect-your-online-safety/

Peisert S., Bishop M., Karin S. and Marzullo K. (2007) "Analysis of Computer Intrusions using Sequences of Function Calls" IEEE Transactions on Dependable and Secure Computing 4(2): 137-150.

Peng T., Leckie C. and Ramamohanarao K. (2007) "Information Sharing for Distributed Intrusion Detection Systems" Journal of Network and Computer Applications 30(3):

877-899.

Pfleeger, S.L. and Pfleeger, C. (2003) Informit http://www.informit.com/articles/article .aspx?p=31339&seqNum=5

Pontarelli S., Bianchi G. and Teofili S. (2013) "Traffic-Aware Design of a High-Speed FPGA Network Intrusion Detection System" IEEE Transactions on Computers 62(11): 2322-2334.

Prezi (2016a) History of the internet https://prezi.com/q3k1voj9l5xz/history-of-the-internet/

Prezi (2016b) The Internet. https://prezi.com/nt0kdsbj7ijg/the-internet/.

Qu G., Hariri S. and Yousif M (2005) "A new dependency and correlation analysis for features" IEEE Transactions on Knowledge and Data Engineering 17(9): 1199-1207

Safaa H., Choumana M., Artailb H. and Karam M. (2008) "A Collaborative Defense Mechanism Against SYN Flooding Attacks in IP Networks" Journal of Network and Computer Applications 31(4): 509-534.

Schuehler D.V., Moscola J. and Lockwood J.W. (2004) "Architecture for a Hardware-Based, TCP/IP Content-Processing System" IEEE Micro 24(1): 62-69.

Sen J., Sengupta I. and Chowdhury P.R. (2006) "An architecture of a distributed intrusion detection system using cooperating agents" International Conference on Computing & Informatics: 1-6

Shieh S.P. and Gligor, V.D. (1997) "On A Pattern-Oriented Model for Intrusion Detection" IEEE Transactions on Knowledge and Data Engineering 9(4): 661-667.

Singh K., Guntuku S.C., Thakur A. and Hota C. (2014) "Big Data Analytics Framework for Peer-To-Peer Botnet Detection using Random Forests" Information Sciences 278: 488- 497.

Su M.Y., Yu G.J. and Lin C.Y. (2009) "A Real-Time Network Intrusion Detection System for Large-Scale Attacks based on An Incremental Mining Approach" Computers and Security 28(5): 301 309.

Tsai M.K., Lin S.C. and Tseng S.S. (2003) "Protocol Based Foresight Anomaly Intrusion Detection System" IEEE 37th Annual International Carnahan Conference on Security Technology: 493-500

Veteranus M. (2013) Network design: Firewall, IDS/IPS - InfoSec resources. http://

resources.infosecinstitute.com/network-design-firewall-idsips/

Wang G., Hao J., Ma J. and Huang L. (2010) "A New Approach to Intrusion Detection using Artificial Neural Networks and Fuzzy Clustering" Expert Systems With Applications 37(9): 6225-6232.

Wang W., Guyet T., Quiniou R., Cordier M.O., Masseglia F. and Zhang X. (2014) "Autonomic Intrusion Detection: Adaptively Detecting Anomalies over Unlabeled Audit Data Streams in Computer Networks" Knowledge-Based Systems 70: 103-117.

Wang X., Liu B., Jiang J., Xu Y., Wang Y. and Wang X. (2014) "Kangaroo: Accelerating String Matching by Running Multiple Collaborative Finite State Machines" IEEE Journal on Selected Areas in Communications 32(10): 1784-1796.

Wuu L.C., Hung C.H. and Chen S.F. (2007) "Building Intrusion Pattern Miner for Snort Network Intrusion Detection System" Journal of Systems and Software 80(10): 1699-1715.

Xiang-Rong Y., Qin-Bao S. and Jun-Yi S. (2001) "Implementation of Sequence Patterns Mining in Network Intrusion Detection System" IEEE International Conferences on Info-Tech and Info-Net: 19-23

Xu-Sheng G., Jing-Shun D., Jia-Fu W. and Wei C. (2013) "Anomaly Intrusion Detection based on PLS Feature Extraction and Core Vector Machine" Knowledge-Based Systems 40: 1-6.

Yin X.C., Huang K. and Hao H.W. (2015) "DE2: Dynamic Ensemble of Ensembles for Learning Nonstationary Data" Neurocomputing 165: 14-22.

Zaremoodi P., Beigy H. and Siahroudi S.K. (2015) "Novel Class Detection in Data Streams using Local Patterns and Neighborhood Graph" Neurocomputing 158: 234-245.

Zhang J., Li H., Gao Q., Wang H. and Luo Y. (2015) "Detecting Anomalies from Big Network Traffic Data using An Adaptive Detection Approach" Information Sciences 318: 91-110.

Zhang J., Zulkernine M. and Haque A. (2008) "Random-Forests based Network Intrusion Detection Systems" IEEE Transactions on Systems, MAN and Cybernetics 38(5): 649-659.

Zhang R., Qian D., Chert H. and Wu W. (2003) "Collaborative Intrusion Detection

Based on Coordination Agent" Fourth International Conference on Parallel and Distributed Computing" Applications and Technologies: 175-179.

Zhang Y., Wang L., Sun W., Green R.C. and Alam M. (2011) "Distributed Intrusion Detection System in A Multi-Layer Network Architecture of Smart Grids" IEEE Transactions on Smart Grid 2(4): 796-808.