Payload-based Anomaly Detection in HTTP Traffic

A Thesis submitted for the degree of

Doctor of Philosophy

By

Aruna Jamdagni

In

Faculty of Engineering and information Technology

School of Computing and Communications

UNIVERSITY OF TECHNOLOGY, SYDNEY AUSTRALIA

SUBMITTED NOVEMBER 2012

UNIVERSITY OF TECHNOLOFY, SYDNEY

SCHOOL OF COMPUTER AND COMMUNICATIONS

The undersigned hereby certify that they have read this thesis entitled "**Payload-based Anomaly Detection in HTTP Traffic"** by **Aruna Jamdagni** and that in his opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.

Principal Supervisor	Co-Supervisor
Prof. Xiangjian (Sean) He	Dr. Priyadarsi Nanda

CERTIFICATE OF AUTHORSHIP/ORIGINALITY

I certify that the work in this thesis has not been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signature of Author

Abstract

Payload-based Anomaly Detection in HTTP Traffic

Internet provides quality and convenience to human life but at the same time it provides a platform for network hackers and criminals. Intrusion Detection Systems (IDSs) have been proven to be powerful methods for detecting anomalies in the network. Traditional IDSs based on signatures are unable to detect new (zero days) attacks. Anomaly-based systems are alternative to signature based systems. However, present anomaly detection systems suffer from three major setbacks:

(a) Large number of false alarms,
(b) Very high volume of network traffic due to high data rates (Gbps), and
(c) Inefficiency in operation.

In this thesis, we address above issues and develop efficient intrusion detection frameworks and models which can be used in detecting a wide variety of attacks including web-based attacks. Our proposed methods are designed to have very few false alarms. We also address Intrusion Detection as a Pattern Recognition problem and discuss all aspects that are important in realizing an anomaly-based IDS.

We present three payload-based anomaly detectors, including Geometrical Structure Anomaly Detection (GSAD), Two-Tier Intrusion Detection system using Linear Discriminant Analysis (LDA), and Real-time Payload-based Intrusion Detection System (RePIDS), for intrusion detection. These detectors perform deep-packet analysis and examine payload content using *n*-gram text categorization and Mahalanobis Distance Map (MDM) techniques. An MDM extracts hidden correlations between the features within each payload and among packet payloads. GSAD generates model of normal network payload as geometrical structure using MDMs in a fully automatic and unsupervised manner. We have implemented the GSAD model in HTTP environment for web-based applications.

For efficient operation of IDSs, the detection speed is a key point. Current IDSs examine a large number of data features to detect intrusions and misuse patterns. Hence, for quickly and accurately identifying anomalies of Internet traffic, feature reduction becomes mandatory. We have proposed two models to address this issue, namely two-tier intrusion detection model and RePIDS.

Two-tier intrusion detection model uses Linear Discriminant Analysis approach for feature reduction and optimal feature selection. It uses MDM technique to create a model of normal network payload using an extracted feature set.

RePIDS uses a 3-tier Iterative Feature Selection Engine (IFSEng) to reduce dimensionality of the raw dataset using Principal Component Analysis (PCA) technique. IFSEng extracts the most significant features from the original feature set and uses mathematical and graphical methods for optimal feature subset selection. Like two-tier intrusion detection model, RePIDS then uses MDM technique to generate a model of normal network payload using extracted features.

We test the proposed IDSs on two publicly available datasets of attacks and normal traffic. Experimental results confirm the effectiveness and validation of our proposed solutions in terms of detection rate, false alarm rate and computational complexity.

Acknowledgement

This research would not have been possible without the guidance and the help of many people. First and foremost, my utmost gratitude to my supervisor, Prof. Xiangjian He, for his excellent guidance, support and steadfast encouragement that I will never forget. His comments and suggestions during preparation of this thesis have been extremely valuable. Without his support and supervision, I could not have come this far. I would thank him for his helpfulness and to have been by far more than a simple supervisor.

I would like to express my gratitude to my research co-supervisors, Dr. Priydarsi Nanda and Dr. Ren Liu, for their friendly guidance and unfailing support. Their encouragement has kept me moving ahead at a critical time. Without their help, I would not have been able to complete this thesis.

I appreciate the financial assistance of Australian Postgraduate Award (APA) provided by Australian government and Top-up scholarship provided by the Commonwealth Scientific and Industrial Research Organisation (CSIRO).

Many thanks to my Employer University of Western Sydney and Prof. Simeon Simoff, Dean, School of Computing and Engineering, who gave me time off from work. I will be always grateful for that.

I also appreciate Dr. Qiang Wu and Dr. Wenjing Jia for providing helpful suggestions. My special thanks to collaborator and my good friend Thomas Tan for brain storming discussions. My friends: Thomas Tan and Sheng Wang, they are always helpful whenever I have questions not only on research but also on other matters. It would have been a lonely lab without them.

Last but not the least, I would like to express my love and gratitude to my family members, especially my daughter Divya, my husband Rishi, my sister Meera and brother in-law Satya for their endless love, understanding and encouragement to work on this thesis.

“What we are is God's gift to us. What we become is our gift to God.”

Eleanor Powell

Dedicated to Dear God

Table of Contents

List of Tables

Table of Figures

Acronyms and Abbreviations

ABS	Anomaly Based System
DARPA	Defense Advanced Research Projects Agency
DDoS	Distributed Denial of Service
DoS	Denial of Service
IDES	Intrusion Detection Expert System
IDS	Intrusion Detection System
GATECH	Georgia Institute of Technology
GSAD	Geometrical Structure Anomaly Detection System
GSPM	Geometrical Structure Payload Model
HIDS	Host-based Intrusion Detection System
HTTP	Hyper Text Transport Protocol
IFSEng	Iterative Feature Selection Engine
IDPS	Intrusion Detection and Prevention System
KDD	Knowledge Discovery in Databases
LDA	Linear Discriminant Analysis
LDM	Linear Discriminant Module
McPAD	Multi classifier Payload Based Anomaly Detection
MD	Mahalanobis Distance
MDM	Mahalanobis Distance Map
MIT	Massachusetts Institute of Technology
MS-SQL	MiscroSoft Structured Query Language
NIDS	Network Intrusion Detection System
PA	Parallel Analysis
PAYL	Payload Based Anomaly Detection System
PCA	Principal Component Analysis
PC	Principal Component
RePIDS	Real-time Payload-based Intrusion Detection System
R2L	Remote to Local

SBS	Signature Based System
SRI	Stanford Research International
SVM	Support Vector Machines
TC	Text Categorization
U2R	User to Root

Authors Publications for the Ph.D

Published papers

Journal Papers

1. **A. Jamdagni**, Z. Tan, P. Nanda, X. He, R. Liu, "RePIDS: a Multi Tier Real-Time Payload-Based Intrusion Detection System," *Computer Networks* (ERA **Tier A**), Elsevier, accepted (Final) on 8 October 2012.
2. **A. Jamdagni,** Z. Tan, P. Nanda, X. He, R. Liu, "Mahalanobis Distance Map Approach for Anomaly Detection of Web-Based Attacks," *Journal of Network Forensics*, 2(2), 25-39, 2011.

Conference papers

3. Z. Tan,, **A. Jamdagni**, P. Nanda, X. He, R. Liu, "Network Intrusion Detection based on LDA for payload feature selection," IEEE Globecom 2010 Workshop on Web and Pervasive Security (WPS 2010), Miami, USA, 2010, pp.1590-1594.
4. Z. Tan, **A. Jamdagni**, X. He, P. Nanda, R. Liu, W. Jia, W. Yeh, "A Two-Tier System for Web Attack Detection Using Linear Discriminant Method," Information and Communications Security, LNCS, Vol. 6476/2010, pp.459-471.
5. **A. Jamdagni,** Z. Tan, X. He, P. Nanda, R. Liu, "Mahalanobis Distance Map Approach for Anomaly Detection of Web-Based Attacks," in 8th Australian Information Security Management Conference. November 2010. Perth.

6. **A. Jamdagni,** Z. Tan, P. Nanda, X. He, R. Liu, "Intrusion detection using GSAD model for HTTP traffic on web services," in IWCMC' 10 Proceedings of the 6th International Wireless Communications and Mobile Computing Conference 2010. France: ACM.
7. **A. Jamdagni,** Z. Tan, R. Liu, P. Nanda, X. He, "Pattern Recognition Approach for Anomaly Detection of Web-based Attacks," in the Seventh Annual CSIRO ICT Centre Science and Engineering Conference, November 2010.
8. **A. Jamdagni,** Z. Tan, P. Nanda, X. He, R. Liu, "Intrusion Detection Using Geometrical Structure," in 4th International Conference on Frontier of Computer Science and Technology (FCST 2009), Shanghai, China, December 17-19, 2009, pp. 327-333.
9. **A. Jamdagni,** Z. Tan, R. Liu, P. Nanda, X. He, "A Framework for Geometrical Anomaly Detection Model," in the Sixth Annual CSIRO ICT Centre Science and Engineering Conference, November 2009.

CHAPTER 1

INTRODUCTION

Securing a computer system has traditionally been a battle of wits: the penetrator tries to find the holes, and the designer tries to close them.

Gosser

The growth of Internet and local area networks provide quality and convenience to human life. According to the latest statistical analysis [1-3], it is estimated that Internet connects over 1.1 billion users worldwide, and thousands of sub-networks. Internet has adopted a large number of new applications, such as on line banking, online gaming, and Internet telephony, and social networks platforms such as facebook, twitter and LinkedIn. It is evident that Internet has had an enormous impact on the everyday life of people and on the worldwide economy as well.

Internet technologies, on one hand, provide large number of on-line services to the end users. On the other hand, they attract the attention of hackers and provide a platform for them to attack systems in the network. The trust in the Internet and its services is increasingly undermined by network attacks. In 1998, the Computer Emergency

Response Team (CERT) at Carnegie Mellon University reported 3,734 security incidents worldwide. In the latest version of the Cyber Security Risks Report [3], the number of vulnerabilities increased approximately 10% from 7,260 in 2009 to over 7,900 in 2010.

Thus, maintaining information security and securing computer systems and networks are essential. To prevent these security compromises, layers of defense, such as proxies, filters, anti-virus scanners and firewalls, are used. Since these traditional prevention mechanisms are imperfect, Intrusion Detection Systems (IDSs) are used to monitor local area networks, and computer systems for security compromises. The role of IDSs is to detect malicious activities in near real-time and raise an alert.

In general, IDSs are classified into two broad categories. Anomaly-based systems compare attack-free data to network traffic where anomalous events are identified as deviations from the normal. Misuse-based systems match signatures or unique character strings to known attacks. Present anomaly intrusion detection systems suffer from a large number of false alarms and poor efficiency in operation, and cannot be deployed in high speed networks and applications. In this thesis, we address these issues and develop payload-based intrusion detection schemes which can be used efficiently in detecting a wide variety of attacks including web-based attacks.

1.1 Motivations: Need for Information Security

There is a continuous increase in attacks upon information security infrastructures. The legacy network based attacks have largely been replaced by more sophisticated web application attacks. According to HP DVLab 2010 Top Cyber Security Risk Report released on April 4 [4-6], “Web-based attacks jumped from only a tenth of all attacks at

the beginning of 2010 to more than 70 percent of all attacks by the end of the year". Hence, securing web applications have become exceptionally important as the information processed by web applications has become critical to corporations, customers, organizations and countries.

Although other protocols are being used for attacks on networks and computers, attacks against the HTTP protocol have become the most dominant. In the first quarter of 2010, HTTP attacks accounted for about half the total number of attacks [4-6].

1.1.1 Reasons of Network Threats

Based on research review, increase in network threats originates from various reasons. We have identified three key reasons of increase in threats and attacks on networks. In the following subsection, we describe these reasons of increase in network threats.

- The proliferation of Web-based plug-ins has shifted traditional attack methods to web-based attacks.

According to Dausin, manager of advanced security intelligence for HP DVLabs:

"We're seeing a huge explosion in Web application attacks, and the attackers are not just using one or two vulnerabilities, they're sending a barrage of malicious requests, trying every tool they have at their disposal."

- Increasing automation and sophistication of network attacks [7].

While early computer attacks have been manually crafted for specific targets, now sophisticated and inexpensive attack toolkits (estimated at $2,400) are widely available in the market. These toolkits have amazing range of functionality, including network

surveillance, polymorphic shell-codes and distributed propagation. For example, the "Slammer worm" can infect ten thousands of hosts in a couple of minutes [8] rendering regular security systems defenseless in protecting network computers. Such capabilities make malicious software and network attacks attractive for illegal business.

- Application developers either have very little knowledge of security or are unaware of complexity of the developed network application software.

Due to the pressure of business competition, software developers put their networks on high risk, because developed applications are seldom tested for vulnerabilities. In addition, when new applications are implemented on the network, the native network infrastructure is seldom capable of detecting software vulnerability and attacks. This potentially leaves application infrastructure vulnerable and easily disrupted by an attack.

According to the "2011 Top Cyber Security Risks Report":

"*Web application vulnerabilities now comprise about half the total number of newly discovered security vulnerabilities and the organization web-sites are constantly at risk of being defaced and made unusable from various attacks."*

Having said this, security of computer systems has become a major concern with critical public infrastructures relying on computers and the Internet. Classic security measures, such as encryption, authentication and policy managements, are widely deployed for protecting networked computers. While such preventive measures significantly strengthen security, they cannot generally stop the possibility of network attacks. As new attacks appear every day, intrusion prevention measures like firewalls and cryptographic protocols are not just sufficient in ensuring the security of the

networked systems. Thus, intrusion detection systems are needed to detect new attacks and defend networks from all kinds of attacks launched against either stand-alone computers or entire computer networks.

An IDS can detect scanning and probing attacks by analyzing network packet headers or by monitoring network traffic connection attempts and session behaviors. Those viruses and worms that propagate at high speed on the Internet can be detected by analyzing the rate of scanning and probing methods. Both viruses and worms exploit known vulnerabilities in the computer operating systems, application software, device drivers and services. Furthermore, there are malicious activities which do not show abnormality in network connections and protocol behaviors but carry malicious contents and cannot be detected by using packet header analysis and traffic flow statistic approaches [9, 10]. In response, effective techniques based on payload analysis are needed to detect the presence of such malicious activities in the networks.

A number of researchers have focused on payload-based anomaly detection. Approaches that have been studied including specification-based anomaly detection [11].

In our work, we use Mahalanobis Distance Map (MDM) approach, a pattern recognition technique used in image processing and develop payload-based IDS. We focus on design and learning approach to efficiently train payload-based detector models on the normal and attack free data for any application, service, network or host. This trained model can then be used to identify “abnormal” or suspicious traffic. Although the proposed model is useful in detecting a wide range of exploits against many, if not

all services and ports, our focus is mainly to identify web based attacks based on HTTP protocol.

1.2 Challenges for Payload Based Anomaly Detection

Although anomaly detection has emerged as a promising technology and appears to hold great future, it is extremely difficult to achieve. There are still great deals of challenges associated with payload-based anomaly intrusion detection systems. These challenges are:

- Traffic profile is changing constantly and new applications are emerging every day, so it is difficult to construct a model of "normality".
- Nature of anomalies keeps changing over time. It is not possible to know the "a priori" knowledge of attack to efficiently identify new attacks and how the attacker is going to attack network and network resources.
- The dilemma between detection rate and false alarms is the major problem [12]. Improvement in the detection rate results in increase in the false positive rate.
- Encryption and tunnelling hide access to data contained into application layer header and payloads.
- Attacks present in the encrypted payload data are considered normal from the network layer point of view.
- A very high volume of network traffic due to high data rates (Gbps) and accuracy issues in existing algorithms performance results in bottleneck.
- High sensitivity to packet loss data and fragmentation and segmentation issues are most likely to pose security concerns for IDS.

- Most of the IDSs perform poorly in defending themselves from attacks [12].

1.3 Research Objectives

Most detection systems in use today are network- and signature-based IDSs. Due to the popularity of Internet, there is an increasing risk in breach of network security. In addition, new types of attacks are appearing continuously and attacks against network services can cause great harm. No signatures have been produced and deployed for these so called zero-day attacks, so developing flexible and adaptive security oriented approaches is a severe challenge. Our work addresses challenges associated with next generation of intrusion detection (discussed in Section 1.2). This thesis concentrates on the following research objectives:

- Develop an efficient algorithm to detect zero-day attacks and the variants of existing attacks using network packet payload anomaly detection technique.
- Implement and evaluate a working prototype of our proposed IDS.
- Evaluate payload features using feature selection techniques which provide ability for real-time operation in payload-based intrusion detection system.
- Achieve good accuracy in detecting truly anomalous events, with low false positive rates.

1.4 Research Approach

We compute a “normal” model of content for an available service by considering correlation between payloads or groups of payloads during a training phase, and then use

this learned "normal" model to detect abnormal, never-seen before content. These suspicious contents coming through network connections may or may not be attacks.

For efficient operation of intrusion detection system, optimal feature set is computed. Then, this reduced feature set is used to discriminate normal and abnormal contents quickly and accurately.

1.4.1 Design Objectives

We model payload content to satisfy various challenges and research objectives of our anomaly detection system. We consider the following design objectives:

- deployment of automatic "hands-free" intrusion detection system with little or no human intervention,
- being applicable for broad application domain to any service or system,
- sequential nature of data, typically coming in a streaming fashion,
- accuracy in detecting truly anomalous events, with low false positive rates,
- resistance to mimicry attack,
- ability to operate in real-time, and
- efficiency of algorithm to operate in high bandwidth environments.

The ideal requirements of the intrusion detection system are 100% detection of the attack and normal network packets, with 0% false positive rate and false negative rate, able to detect attacks in real-time and be adaptive to dynamic profile of network traffic. In this work, we aim to achieve this ideal level or near ideal level performance by devising efficient IDS. We conduct several experiments using DARPA 99 dataset and

Georgia Institute of Technology attack dataset (real network traffic) to demonstrate how close we may reach this ideal.

1.4.2 Design Approach

To meet all objectives in Subsection 1.4.1 is a very challenging task. Modeling the network traffic profile using payload anomaly detection approach, these objectives can be achieved. The modeling technique proposed in this thesis includes correlations between the payload features and also among the network packets, which provides some information about the payload structure. A geometrical structure model is created for application layer packet payload (*n*-gram window size for normal traffic content) using MDM. We consider "clear text" content, and do not address the issue of encrypted content of the network traffic. We believe that our technique can be used for encrypted content applied at the point of decryption and delivered to the targeted application software.

We consider HTTP traffic for analysis and experimental evaluations of our model. Since most web traffic contents are usually public and pose fewer privacy restrictions, it is easy to obtain web traffic data. We believe that the algorithms and technology presented in this thesis can be applied to other content based traffic too. We have also chosen to limit our study to web traffic since the number of attacks against the known vulnerabilities is rising continuously in respect of severity and frequency [13], and also historically web services have been a common target of previous worm attacks [14-16].

1.5 Contributions to Thesis

An attacker often follows a sequence of events, which are highly correlated and the sequences have dependencies among them. Furthermore, the attacker can also hide individual events within a large number of normal events such that the events cannot be recognised as harmful events. Additionally, events consist of multiple features which are monitored continuously. These features are also highly correlated and must not be analysed in isolation. Existing anomaly-based intrusion detection systems consider the events individually, thereby, discarding any correlation between features and also sequential events, which results in a poor model. Hence, we introduce efficient intrusion detection frameworks and methods which consider a group of events and analyze multiple features assuming dependence among the features.

1.5.1 Framework for Payload-based Anomaly Intrusion Detection

In Chapter 3 of this thesis, we will introduce our novel framework for building network intrusion detection systems, which is known as "Geometrical Structure Anomaly Detection" (GSAD) model to detect anomaly in the application layer payload.

1.5.2 Implementation and Evaluation of proposed ptototype

In Chapter 3 of this thesis, we will experimentally demonstrate that the GSAD model can successfully detect inbound attack and worm packets with high accuracy rate and a low false positive rate, and will compare performance of GSAD against the state-of-the-art payload-based systems. In our framework, we use geometrical structure present in the payload by correlation between the payload features in order to decrease the number of false alarms and increase the attack detection coverage.

1.5.3 Payload Feature Selection Using Linear Discriminant Analysis Technique for Network Intrusion Detection

Network monitoring is one of the common and widely applied methods for detecting malicious activities in an entire network. However, GSAD model uses a large number of features to discriminate normal and malicious (attack) packets that are flowing in the network. As a result, GSAD is computationally expensive. In Chapter 4, we introduce the feature selection algorithm using Linear Discriminnant Analysis (LDA) technique. The selected features provide strong correlations between anomalous behavior and malicious activity. These features are used to develop normal traffic sensor profiles to detect anomaly in the network traffic. This simplifies computational complexity and also reduces training and testing time in anomaly detection.

1.5.4 Cumulative Profile (Signature) Generation

Frequency of new attack is increasing and so is the frequency of polymorphic attacks. In such situation, it is often very difficult to keep signature database up to date with all possible signatures. Moreover, the size of the signature database will also increase. In Chapter 4, we propose to generate one common signature for group of similar type of attacks. This will help in reducing the number of signatures for similar type of attacks. Additionally, this will reduce the resources and detection time.

1.5.5 Framework for Real-time Intrusion Detection Using Principal Component Analysis Feature Selection Technique

In chapter 5, we present a framework for real-time intrusion detection system (RePIDS). This framework integrates Principle Components Analysis (PCA) and Mahalanobis

Distance Map to detect normal and malicious behavior of network traffic. PCA is used to reduce the dimensionality (number of the features) of the dataset but retains original variability in the dataset. Mathematical and graphical methods are used independently for the selection of dominant principal components (optimal features). We further experimentally demonstrate that the RePIDS can successfully detect inbound attack and worm packets with high accuracy rate and a low false positive rate. We also compare its performance against the state-of-the-art payload-based systems.

This dissertation presents evidence to show the validity of the following hypotheses:

- It is possible and feasible to process the values associated with identified (i.e., noisy) attributes to extract useful features. Models built using these features exhibit reduced false positive rates and higher true detection rates.
- The newly extracted features enable behavioral modeling of application traffic that is not possible when using packet header values.
- It is possible to generate a common signature for similar types of attacks.
- Models built using dominant features exhibit real-time data processing characteristic.

1.6 Thesis Organization

The rest of the thesis is organized as follows.

Chapter 2 discusses taxonomy of intrusion detection system and related work in intrusion detection, worm detection and collaborative security.

Chapter 3 describes the Geometrical Structure Anomaly Detection (GSAD) anomaly detection sensor, the framework and detection techniques employed in GSAD, and

demonstrate how well it can detect attacks. Furthermore, we present the importance of HTTP protocol and web-based attacks. We demonstrate implementation of GSAD model in HTTP environment, and confirm high detection rate and low false alarm rate through experimental results.

Chapter 4 discusses Linear Discriminant Approach (LDA) and its use in feature selection and evaluation of LDA based intrusion detectors for detection of attacks. We also present our discussion on the generation of one common signature for a group of similar type of attacks. We present a preliminary work in this direction.

In Chapter 5, we discuss various feature selection techniques used in the anomaly detection. We focus on selection of optimal features in a packet payload using Principal Component Analysis (PCA) technique, and mathematical and graphical techniques for real-time anomaly detection. This chapter also presents a novel real-time intrusion detection system and matrix to evaluate computational complexity and time complexity.

Chapter 6 concludes the dissertation and presents future research work that extends our research.

CHAPTER 2

Taxonomy of Intrusion Detection Systems and Related work

Introduction

Intrusion detection has been at the centre of research in the last decade due to rapid increase of sophisticated attacks on computer system. Naive attackers can launch powerful attacks which can bring down an entire network [5]. Hence, detecting intrusion in networks and applications has become one of the most critical tasks to prevent their misuse by attackers. Typically, intrusion detection refers to a variety of techniques for detecting attacks in the form of malicious and unauthorized activities. To identify the shortcoming of different approaches for intrusion detection, we explore the strategies used for dealing with security vulnerabilities and review related research in the intrusion detection. We view intrusion detection as a pattern recognition problem and introduce our proposed model. Our model uses Mahalanobis Distance Map, a pattern reorganization technique for building intrusion detection system. In this chapter, we describe taxonomy of intrusion detection systems including brief information on

strategies used to alleviate security problems. Then, we present intrusion detection as a pattern recognition problem. Finally, we present review on intrusion detection techniques.

This chapter is organised as follows. In Section 2.1, we list common strategies used in dealing with security vulnerabilities. In Section 2.2, we present taxonomy of intrusion detection systems outlining their role and requirements. Then, we discuss the intrusion detection problem as a pattern recognition problem in Section 2.3. In Section 2.4, evaluation matrices for intrusion detection method are discussed. Section 2.5 covers literature review on the related research work. We conclude this chapter in Section 2.6.

2.1 Strategies for Threat Mitigation

Intrusion detection system is a software tool used to detect unauthorized access to a computer system or network. In 1972, Anderson [17] identified the need for intrusion detection and proposed a threat model. He identified various types of threats and the sources of these threats. He classified threats as internal penetrations, external penetrations and misfeasance, and developed a security monitoring system which detected anomalies in user behavior. Researchers have used multilayer infrastructures for attack detection and prevention. For example, the use of proper policies and physical access restriction (traditional locks and other physical security) can prevent attacks at the physical layer. Different strategies are used to deal with security policies. These strategies are classified into six categories [18]. A summary of threat mitigation strategies is given in Table 2.1

Table 2.1: Summary of attack mitigation techniques

Strategies	Purpose	Actions taken
Attack Deterrence	Discourage attackers and prevent them to do something.	Tag, information systems hardware and software with electronic IDs. Traces true sources of attacks.
Attack Prevention	Prevent attackers.	Block on-line attacks before they reach to targets.
Attack Deflection	Deflect attacker to reveal attack.	Create sites to trap attackers deliberately.
Attack Avoidance	Make resources unusable.	Protect information using Cryptography.
Attack detection	Detect attacks.	Monitor and analyse network traffic activities to generate alerts.
Attack reaction and Recovery	Allow systems to continue operating	Regular backups of critical data.

These strategies are discussed briefly in the following section.

1. **Attack Deterrence** Several technical and legal measures have been undertaken to discourage mongers from tampering with computer systems. Trails built on computers serve incriminating evidence of contributions as measures of securing hosts. They refer as a fear of tracing the true source of an attack. Attackers are discouraged from deploying attacks on computer systems. Attackers use spoofed sources IP addresses to launch attacks (IP Address Spoofing allows people to log onto a website with a different IP address).
2. **Attack Prevention** The aim of attack prevention is to prevent an attack by blocking it before the attack can reach the target. Attack prevention mechanism can be viewed roughly as an intrusion detection mechanism with a preventive response. Intrusion Detection and Prevention System (IDPS) is configured to automatically block

suspected attacks without any intervention required by an operator. IDPS has the advantage of providing real-time corrective action in response to an attack. An example of attack prevention systems is a firewall [19], which filters unwanted traffic at the network level based on certain rules and policies. Such traffic filtering is implemented at the border gateways of networks and sometimes at the network layer of the individual host machines.

3. **Attack Deflection** This refers to tricking an attacker by making the attacker believe that the attack was successful. Though, in reality, the attacker was trapped by the system and deliberately made to reveal the attack. For example, *honey pot* is a trap that lures attackers away from production systems [20]. Honey pot runs special software to emulate services, applications and protocols, and contains information specifically created to trick the attacker.
4. **Attack Avoidance** Attack avoidance strategies identify and remove vulnerabilities from software before they are deployed in a security-critical environment .The aim of attack avoidance strategies is to make the resource unusable by an attacker even though the attacker is able to illegitimately access that resource. For example, cryptography [21] is used to protect information in computer systems.
5. **Attack Detection** Attack detection refers to detecting an attack while the attack is still in progress or to detect an attack which has already occurred in the past. Detecting an attack is significant for two reasons. Firstly, the system must recover from the damage caused by the attack. Secondly, it allows the system to take measures to prevent similar attacks in the future. Research in this area focuses on building intrusion detection systems. An intrusion detection system monitors and

analyses network traffic activity and alerts an operator to potential vulnerabilities and attacks.

6. **Attack Reaction and Recovery** Once an attack is detected, the system must react to such attack and perform the recovery mechanisms as defined in the security policy.

2.2 Taxonomy of Intrusion Detection Systems

Intrusion detection is the act of detecting actions that attempt to compromise the confidentiality, integrity and availability of a system/network. Classifying intrusion detection systems help to better understand their capabilities and limitations. Debar et al. [22] were the first to introduce a systematic and taxonomic approach of intrusion detection systems (later revised in [23]). Axelsson [12] provided a comprehensive survey and taxonomy of intrusion detection systems, which addressed some aspects in more depth, namely the detection principles.

Figure 2.1 presents taxonomy (classification) of intrusion detection systems. We introduce the basic definitions and discuss typical advantages and disadvantages by comparing different approaches used in IDSs.

Traditionally, intrusion detection systems are classified according to two characteristics: data sources (protected system type) and the detection methods. Based on the sources of data being audited and used to design its detection model, an intrusion detection system can be either host-based or network-based. Whereas, based on the detection method, an intrusion detection system can be a signature-based detection system, an anomaly-based detection system or a hybrid/compound detection system. According to Time of Audit, a system can be classified into real-time and off-time,

whereas according to System Structure, an IDS can be classified as distributed system and centralised system. Based on the behavior of a system, an IDS can be classified as active IDS and passive IDS. In this section, we discuss taxonomy of intrusion detection systems more in Figure 2.1.

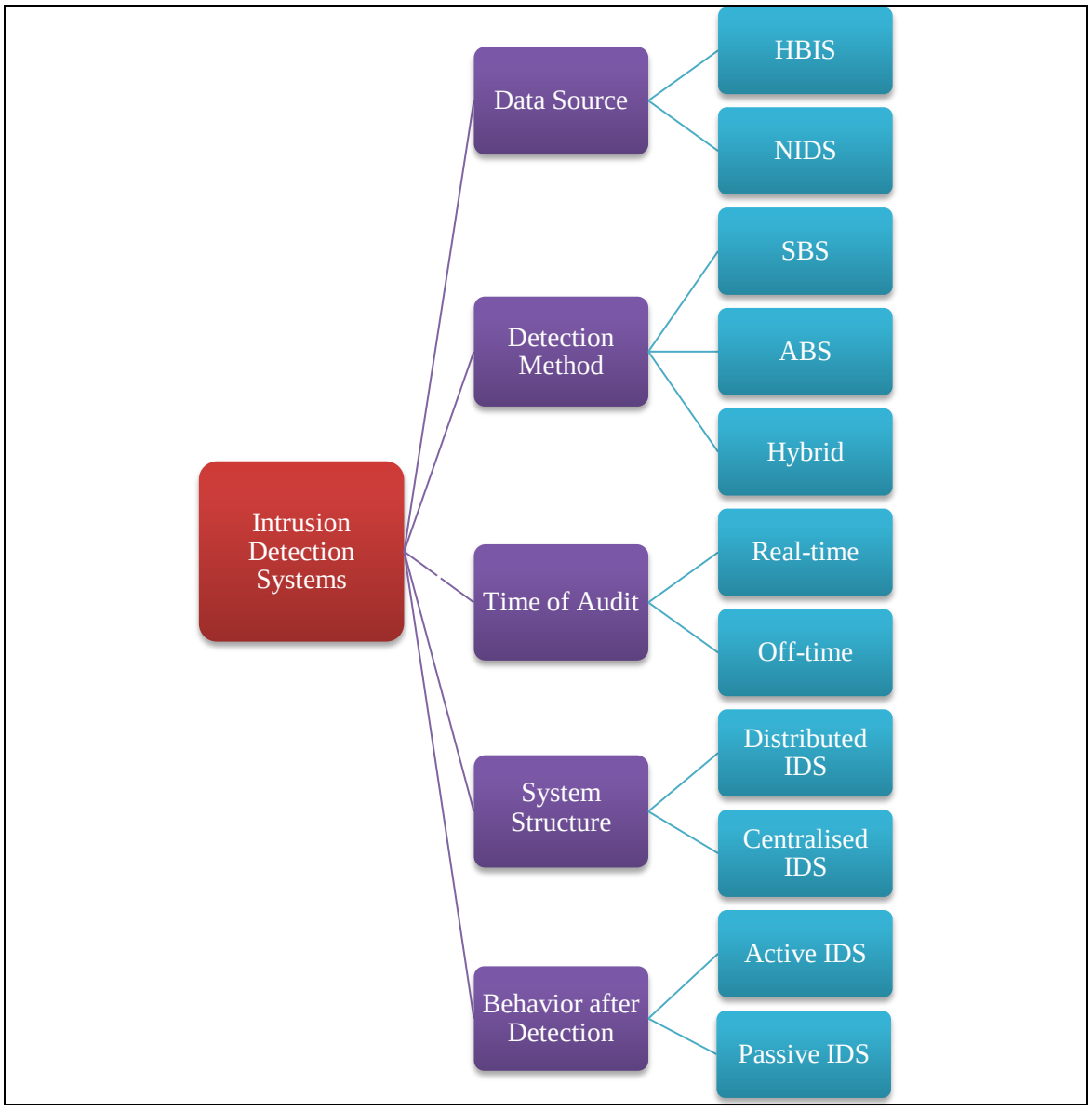


Figure 2.1: Taxonomy of intrusion detection system

2.2.1 Intrusion Detection Systems Based on Data Sources

Based on the sources of data being audited and used to design its detection model, Denning [24] classifies intrusion detection systems into host based and network based intrusion detection systems. A brief discussion on each of them is given below.

Host-based Intrusion Detection System

A Host-based Intrusion Detection System (HIDS) monitors a single host (or a single application) and analyses the audit pattern generated at the operating system or that of a particular application. An intrusion detection system protecting an individual host collects individual log produced by the host [25]. The audit pattern contains more specific information than the network level audit patterns, which can be used to detect an attack more reliably. However, the main drawback of HIDS is that it is difficult to manage a large number of host-based systems. HIDS themselves can be the victims of an attack. Web application firewall known as ModSecurity [26] is an example of HIDS.

Network-based Intrusion Detection System

Network-based Intrusion Detection System (NIDS) monitors network segment and collects audit patterns flowing on this segment. These collected audit patterns at the network level are analysed for attack patterns and these attack patterns are use to protect a single host or an entire network [27, 28]. NIDS can analyse different types of data, namely packet header data, packet payload data, or both. The main advantages of NIDS approach are that: it is possible to monitor data and events without affecting host performance, and a single NIDS can be used to monitor an entire network without the

need for installing dedicated IDS on each host. It can detect attacks that are not visible from a single host and can correlate attacks against multiple hosts.

However, the attack detection capability of NIDS is limited. This is because it is hard to infer the contextual information directly from the network audit patterns. Furthermore, the audit patterns may be encrypted rendering them unusable by the intrusion detector at the network level. In addition, large amount of audit patterns at the network level may also affect the attack detection accuracy based on two reasons. Firstly, a significant portion of the total incoming patterns may be allowed to pass into the network without any analysis. Secondly, in high speed networks, it may be practical to analyse only the summary statistics collected at regular time intervals.

Due to increasing severity of attacks from the Internet, NIDSs are employed in almost all large-scale IT infrastructures. Snort [29] is a typical example of NIDS.

2.2.2 Intrusion Detection System Based on Detection Method

This classification is based on the model it uses for intrusion detection. Based on the detection model, an intrusion detection system can be a signature-based detection system, an anomaly-based detection system or a hybrid/compound detection system, where the strengths of one model are exploited to cover the weaknesses of another. An overview of these approaches is given below.

Signature-based Systems

Signature detection is also known as misuse detection, or detection by appearance. Signature-based intrusion detection technique detection depends on a predefined set of attack signatures. It looks for specific patterns, and signatures, present in the incoming

packets and/or command sequences, and uses pattern matching approaches to detect attacks. When a match is found, an alert is raised.

The advantage of the signature-based detection approach is that, a system based on this approach can detect known attacks fairly accurately with a low false positive rate. They can protect computer/network immediately upon installation and are usually fast. The major drawback of the signature detection approaches is that they have limited attack detection capability, since they cannot detect new (i.e. zero-day) or polymorphic attacks, i.e., the variants of the attacks. They have high false negative alarm rate [30, 31]. Signature-based intrusion detection system typically requires signatures to be defined for all the possible attacks that an attacker may launch against a network. Human interaction is required to keep signature database up-to-date and to analyse each attack to develop the signature. The response time for new attacks is limited to a timescale of hours or days, whereas attacks by self-replicating programs (viruses or worms) can appear and spread in seconds [32]. Hence, maintaining state information of signatures is an important task in the signature detection system.

Thus, an attacker has a window of opportunity to gain control of the system or application under attack. It makes a signature-based detection system less suitable for protecting a web-based service, because of ad-hoc and dynamic nature of web traffic. Some examples of signature-based intrusion detection systems are Snort, anti-virus products such as McAfee AntiVirus plus, Kaspersky, etc.

Anomaly-based Systems

Anomaly-based Intrusion Detection System is also known as detection by a behavior system. The approach used by the anomaly-based system is entirely different than the approach used by the signature-based system to detect an anomaly in the network/system. An anomaly-based system detects behaviors on a computer or computer network that are not normal. An anomaly-based system first creates a base-line profile of the normal system/network, or program activity. It then compares the profile of an incoming event against the base-line profile. A significant deviation from the known normal behavior is identified as malicious. An anomaly-based intrusion detection system assumes that the intrusion attempts are rare and they have different characteristics from normal behavior. A statistical model of normal behavior is created from the training data. When an instance that does not match the created model (learned from the training data) appears, the system raises an alarm.

The main strength of an anomaly detection system is that it has the capability to detect new (zero-day) and polymorphic attacks. Novel attacks can be detected as soon as they take place. Anomaly-based system does not need a-priori knowledge of the application/system and possesses better detection ability than a signature-based system. Since anomaly-based system is based on custom profiles, it is very difficult for an attacker to know with certainty what activity it can carry out without triggering an alarm. Hence, anomaly-based system is suitable for the protection of web applications. However, anomaly detection system also suffers from several weaknesses. For anomaly-based systems to be effective, complete knowledge of normal behaviour of a system/network is required. Creating normal traffic profile is also very challenging for

the reason that finding a proper representation of training data, which shows the normal behavior of data, is a difficult task. Moreover, because of dynamic nature of network data, maintenance of a normal profile is difficult and time consuming. In view of the fact that user and network behavior are not always known beforehand and since an anomaly detection system is looking for anomalous events rather than attacks, it has high false alarms. Not all anomalous events are malicious. Furthermore, an attacker can train anomaly detection system gradually to accept malicious behavior as normal. A good review of anomaly intrusion detection systems can be found in [33] and [34]

Signature-based network intrusion detection system is preferred choice over anomaly-based network intrusion detection system because it is simpler to configure and maintain, despite the occurrence of high false negatives. Most intrusion detection systems in use today are network- and signature-based systems. However, the popularity of the Internet and increasing risk in breach of network security, anomaly-based network intrusion detection system approach is becoming more popular. In our research, we mainly focus on anomaly-based network intrusion detection systems.

2.2.3 Hybrid Intrusion Detection System

A hybrid system uses the partial knowledge of both, i.e., normal and attack information to detect attacks. Thus, they have a better performance, resulting in fewer false alarms and improved attack detection rates. Hybrid systems generally use machine learning approaches.

The hybrid system proposed in [35], combines misuse and anomaly detection to find attacks in logged HTTP request. They resolved the conflicts between signature-based

systems and anomaly-based systems to provide the best accuracy. They used manual methods to identify normal or anomaly web requests. This heavy reliance on human also limits the usefulness of their system. Because of many weaknesses, this system remains unpopular and is restricted from its commercial use.

In [36], authors proposed a hybrid intrusion detection system, which is based on Conditional Random Fields (CRSs). They integrated layered framework with the conditional random fields to build anomaly hybrid intrusion detection system. Normal and abnormal traffic features are used for the training of the system and conditional random fields are used to label every feature in the observation.

2.2.4 Data Audit Time

Intrusion detection system can be divided into real-time intrusion detection system and off-line intrusion detection system based on whether the data analysis is done in real-time or afterward.

Real-time Intrusion Detection System

A real-time intrusion detection system detects an attack as soon as an attack is commenced. However, in practice, it is very difficult to build such a system under the constraints of a low false alarm rate and high detection rate. Snort, an anomaly signature-based intrusion detection system, employs in real-time environment and detects the known attacks accurately with low false positives.

Off-time Intrusion Detection System

Off-time intrusion detection system works differently from real-time intrusion detection system. The audit data logs are collected in a central repository and patterns are analysed for intrusions at a predefined time interval. Such systems cannot provide any immediate response to intrusion and can only perform the recovery task once an attack is detected.

2.2.5 System Structure

Based on the system structure, intrusion detection system can be classified into two categories, centralized intrusion detection system and distributed intrusion detection system. We present a brief description on them below. Both centralized and distributed intrusion detection systems may use host- or network-based data collection methods, or a combination of both.

Centralised Intrusion Detection System

Centralized intrusion detection system determines the global state of the network. Centralised intrusion detection system collects data either from single source or multiple sources for processing and analysing data centrally. The location where the actual analysis is carried out is independent of the location of the sensor.

Distributed Intrusion Detection System

A distributed intrusion detection system is one where data is collected and analysed in multiple hosts and decisions are made locally. The advantage of a distributed system for intrusion detection is that immediate response mechanism can be activated based upon local decisions. However, it is expensive since the number of analyser is proportional to

the number of monitored components. Furthermore, distributed intrusion detection system is less accurate due to lack of global knowledge.

2.2.6 Action after Intrusion Detection

Based on the actions that an intrusion detection system takes after it detects an attack, intrusion detection system can be classified as active intrusion detection system and passive intrusion detection system.

Active Intrusion Detection System

An active Intrusion Detection Systems (IDS) is also known as Intrusion Detection and Prevention System (IDPS). Intrusion Detection and Prevention System (IDPS) blocks suspected attacks automatically without any intervention required by an operator. Intrusion Detection and Prevention System (IDPS) has the advantage of providing real-time corrective action in response to an attack.

Passive Intrusion Detection System

Intrusion detection system is considered to be a passive-monitoring system. It warns administrator of suspicious activity happening in network, and generates alarms. It does not take any action to stop the attack.

2.3 Pattern Recognition Approach for Intrusion Detection

In the following section, we describe anomaly detection as a pattern recognition problem. Anomaly detection approach usually consists of two phases: a training phase and a testing phase. In pattern recognition studies, the learning phase constructs a classifier from example data, which is the same as the training of the model on training

dataset. During the recognition phase, the classifier classifies new data patterns into pattern classes which are similar to testing phase, where the learned profile is applied to new data. Therefore, anomaly detection problem can be seen as a pattern recognition problem.

The pattern recognition task can be subdivided into the following four steps. Step one performs data acquisition. Here, data is collected from various sources and sent to the first stage for pre-processing, where cleaning of data is performed and data is separated into groups. In the second step, the features are extracted and selected. These selected features represent the pattern of data. In the third step, selected features are used to choose the model type. Finally, the classification and analysis of the results are performed in the fourth step. Figure 2.2 shows a generic pattern recognition process.

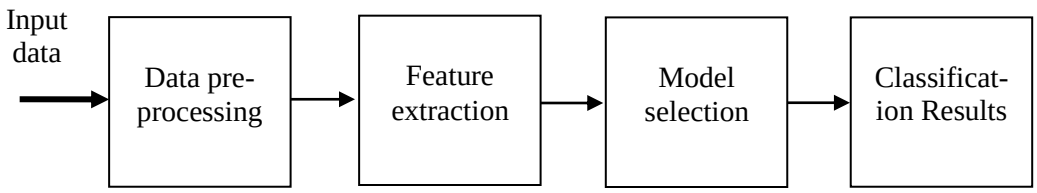


Figure 2.2: Generic pattern recognition process

Following section describes process for intrusion detection using pattern recognition technique.

Network intrusion detection aims at distinguishing the attacks on the network and Internet from normal use of the Internet. This is a typical classification problem, and so intrusion detection can be seen as a pattern recognition problem, whose goal is to distinguish normal behaviors and anomalies. Intrusion detection task can be formulated as a pattern recognition task.

Figure 2.3 illustrates the design process for network intrusion detection using the pattern recognition technique. Traffic data is first processed in order to identify network connections between hosts. In the network, the term “connection" refers to a sequence of data packets related to a particular service between a pair of hosts, e.g., the transfer of a web page via the http protocol. Each network connection represents network data and can be defined as a “pattern" to be classified. Features are extracted from the collected data. These features are used by a pattern recognition technique to describe the patterns.

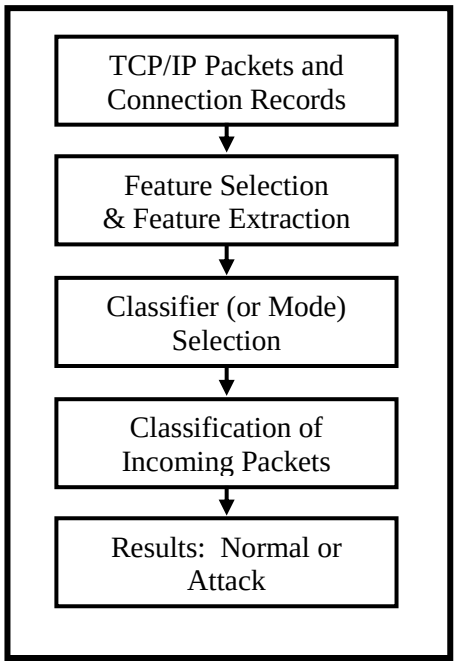


Figure 2.3: Pattern recognition process for intrusion detection

We present a brief explanation of each step and then discuss the techniques used to implement the design process.

a) N-grams Text Categorization Method

Text Categorization or Text Classification (TC) [37] is a fundamental process of classifying content-based documents to pre-defined categories. It allows automated handling of enormous streams of documents in electronic form. Text categorization techniques are adopted to convert each process to a vector and calculate the similarity between two process activities. Since there is no need to learn individual process profiles separately, the calculation involved is largely reduced.

Initially, Forrest et al. [38] introduced the concept of *n*-gram text categorization to build a program profile. He used the sequence of system calls to characterize the behavior of a program running on computer systems. It then became a popular alternative to build user profiles in intrusion detection [39].

b) N-gram Text Categorization Techniques for Intrusion Detection

Liao and Vimuri [40] used text categorization techniques in anomaly intrusion detection. They used frequencies of system calls executed by a program to characterize the program's behavior. Text categorization techniques are adopted to convert each process to a vector. They use the *k*-nearest neighbor classifier to classify new program behavior into either normal or intrusive class. They draw an analogy between a text document and the sequence of all system calls issued by a process, i.e., program execution. Table 2.2 illustrates the similarity in some respects between text categorization and intrusion detection.

Table 2.2: Analogy between text categorization and intrusion detection [39]

Terms	Text categorization	Intrusion Detection
N	total number of documents	total number of processes
M	total number of distinct words	total number of distinct system calls
n_i	number of times i-th word occurred	number of times i-th system call issued
f_{ij}	frequency of i-th word in document j	frequency of i-th system call in process j
D_{ij}	j-th training document	j-th training process
X	test document	test document

We use a pattern recognition technique to detect network intrusions and the *n*-gram text categorization technique to extract important features, which can discriminate normal and malicious patterns accurately. To improve the performance of the detection process, correlation between the features and correlation among the packets is used in the design of intrusion detection system. We propose a novel intrusion detection approach using geometrical structures concepts, which is used for the detection of a human face in the image. Detailed discussion of the framework and its mathematical model is presented in Chapter 3 of this thesis.

c) Classifier Selection

The design goal of an anomaly detection system is to generate a model that accurately describes normal behavior of the system. The network intrusion detection system should be able to classify network connections (network data) between two hosts for a required service, as normal or anomaly. A classifier based on a supervised pattern recognition technique for the training needs data labeled as normal and abnormal, and uses a two class classifier model. Since the number of normal samples is much bigger than the

number of anomaly samples, the network data (training data) is not balanced data. Hence, it is difficult to generate a classifier that can represent true normal behavior and true anomaly behavior of network data. Moreover, the classifier with low training samples will show weak classification ability. Under these constraints, it is not appropriate to select two- (or multi-) class classification model approaches and it may be appropriate to use a one-class classifier model approach.

In anomaly intrusion detection, we assume that the number of normal events is much bigger in comparison to malicious events. Hence, we select one-class classifier model for the classification of samples that are mostly represented in a class. The rest of the samples belonging to the least represented one are left out.

2.4 Performance Evaluation of Intrusion Detection System

The main aims of the intrusion detection systems are to maximize the true-positive rate and minimize the false-positive rate of a proposed technique. Consequently, the performance of the intrusion detection method depends on two basic measures: the number of attacks detected (i.e., the true-positive rate) and the number of normal events classified as attacks (i.e., the false-positive rate). Confusion matrix is given in Table 2.3, which presents the possible outcomes of an individual classification performed by an intrusion detection system.

Table 2.3: Confusion matrix

	Predicted Normal	**Predicted Attack**
True Normal	True Negative (TN)	False Positive (FP)
True Attack	False Negative (FN)	True Positive (TP)

The true positive rate (TPR) and false positive rate (FPR) for intrusion detection system can be calculated as

$$\text{TPR} = \frac{\text{TP}}{\text{TP+FN}}, \tag{2.1}$$

$$\text{FPR} = \frac{\text{FP}}{\text{TN+FP}}. \tag{2.2}$$

Intrusion detection system is an example of imbalanced classes, i.e., attack instances are not equal in number in the training and the testing datasets, and can bias the performance of the system [41]. Therefore, we use precision, recall and *F*-Value as measures for testing the performance of a system, which do not depend on the size of the test dataset and thus calculate unbiased performance of the system. Precision, Recall and *F*-Value can be evaluated using the following equations:

$$Precision = \frac{\text{TP}}{\text{TP+FP}}, \tag{2.3}$$

$$Recall = \frac{\text{TP}}{\text{TP+FN}}, \tag{2.4}$$

$$F - Value = (1 + \beta^2) * Recall * \frac{Precision}{\beta^2(Recall+Precision)}, \tag{2.5}$$

here, β corresponds to the relative importance of precision versus recall and is usually set to 1. It is easy to see that the Detection Rate (DR) is equivalent to the "recall" rate in information retrieval systems, while the False-Positive (FP) rate is somehow the inverse of "precision".

A common technique for visualization of these quantities is *Receiver Operating Characteristic ROC curves* [42], which show true-positive rate on the *Y*-axis and false-positive rate on the *X*-axis for different value of thresholds. The concept of ROC curves gives single numerical measure for the performance of an intrusion detection method: the A*rea Under the ROC Curve* (AUC) which integrates the true positive rate for a particular detection method.

2.5 Related Research Works

The field of intrusion detection is broad, and over the last two decades research has been devoted to the design and evaluation of effective intrusion detection methodologies. The concept for intrusion detection starts from the seminal work of Anderson [17] and Denning [24], which has laid the foundations for the design of numerous detection systems. Denning proposed a general framework for detecting attacks against computer systems by modelling normal behavior patterns generated by users of the system. Since then, a number of intrusion detection systems were designed and deployed as surveyed by Mukherjee et al. [43]. Later, they were evolved as Host-based Intrusion Detection System (HIDS).

As discussed in Section 2.2, the standard taxonomy for intrusion detection systems involves classifying each system according to several distinguishing characteristics: detection methodology, either misuse-based or anomaly-based; detection domain, network, host, application or some combination; and detection models, manually specified or automatically generated. The use of anomaly detection techniques in the context of network intrusion detection (ANIDS) is considered as a promising method of

identifying and understanding novel attack behaviors. An anomaly detection approach performs detection of patterns in two steps, namely training and testing. In the training process, normal (or abnormal) behavior of the system is characterised and a corresponding model is built. This can be performed either automatically or manually, depending upon the type of anomaly network intrusion detection considered. Whereas, in the testing phase, once the normal model for the system is available, it is compared with the observed traffic. If the deviation found exceeds a given threshold, an alarm will be triggered [44].

In this section, we review the detection techniques at two different levels. At the first level, we go over the existing intrusion detection techniques from a general perspective, covering host-based, network-based and some type-specific detection techniques. At the second level, we concentrate on work which utilizes the techniques similar to what we will use in our research.

2.5.1 Review from the Perspectives of Intrusion Detection Techniques

According to Patcha and Park [45], and Garcia-Teodore et al. [46], anomaly detection techniques can be classified into three main categories, namely, statistical based, data-mining based (or knowledge-based) and machine learning-based. Chandola et al. [33] also reviewed anomaly detection methods and discussed several application domains including credit card fraud, image processing and computer security. In this section, we present a brief review on a number of architectures and methods that have been proposed for anomaly detection.

1. Statistical-based Anomaly Network Intrusion Detection Techniques

In the statistical-based technique, the network traffic activity is captured and a profile representing its behavior is created. This profile is typically based on metrics such as traffic rate, number of packets for each protocol, and audit record distribution measure. Two datasets of network traffic are considered. One corresponds to currently observed profile over time, and the other is for the previously trained statistical profile. As the network events occur, the current profile is detected and then compared against the normal profile. An anomaly score is generated, which represents the degree of similarity for a specific event. When the score exceeds a certain threshold, an alarm is raised.

Danning and Neumann [24] used univariate models to model the parameters as independent Gaussian random variables. Ye et al. [47] proposed multivariate models. Their model considered correlations between two or more metrics and showed a better level of data discrimination.

Statistical anomaly detection systems have number of advantages. Firstly, these systems do not require prior knowledge of security flaws and/or the attacks themselves. As a result, such systems have the capability of detecting "zero-day" attacks or the very latest attacks. In addition, statistical approaches can provide accurate notification of malicious activities occurring over a long periods of time.

However, statistical anomaly detection schemes also have many drawbacks. Statistical anomaly detection systems can be trained by skilled attackers to accept abnormal behavior as normal. It can also be difficult to determine thresholds that balance the likelihood of false positives with the likelihood of false negatives. In

addition, statistical methods need accurate statistical distributions but not all behaviors can be modeled using purely statistical methods. Furthermore, most of these schemes rely on the assumption of a quasi-stationary process, which is not always realistic.

Haystack [48] is one of the earliest examples of a statistical anomaly-based intrusion detection system. In the early 1980, scientists at the Stanford Research Institute (SRI) developed Intrusion Detection Expert System (IDES) [49], which continuously monitored user behaviour and detected suspicious events as they occurred. They also developed an improved version of intrusion detection expert system called the Next-generation Intrusion Detection Expert System (NIDES) [50]. NIDS can operate in real-time for continuous monitoring of user activity or can run in a batch mode for periodic analysis of the audit data. Unlike intrusion detection expert system, which is an anomaly detection system, NIDS is a hybrid system that has an upgraded statistical analysis engine. By having the benefit of real-time detection ability, this system has high false alarm rate.

Kruegel et al. in [51] showed that it was possible to use payload byte distribution and then combined this information with extracted packet header features for intrusion detection. In this approach, the resultant ASCII characters were sorted by frequency and then aggregated into six groups. However, this approach leads to a very coarse classification of the payload.

Intrusion detection systems discussed in the previous section are host-based intrusion detection systems. In other words, they do not have ability to defend a network in a global term. Since then, a great amount of research has been undertaken towards the

design and development of network-based intrusion detection systems. Maxion et al. [52] proposed a network-based intrusion detection systems but they did not consider dynamic behavior of network traffic and the intrusion detection systems did not fit in an experimental environment.

Mahoney et al. [53, 54], described several methods that addressed the problem of detecting anomalies in the use of network protocols by inspecting packet headers. They used DARPA 1999 dataset for their experiments. The main aim of these methods was to apply a learning technique to automatically obtain profiles of normal behavior for protocols at different layers. Mahoney et al. proposed PHAD (Packet Header Anomaly Detector) [53], LERAD (LEarning Rules for Anomaly Detection) [54] and ALAD (Application Layer Anomaly Detector) [55]. PHAD monitors 33 attributes as basic features from the Ethernet, IP and transport layer packet headers. ALAD models incoming server TCP requests. ALAD uses source and destination IP addresses and port numbers, opening and closing TCP flags, and the list of commands (the first word on each line) in the application payload. Depending on the attributes, ALAD builds separate models for each target host, port number (service), or host/port combination. LERAD also models TCP connections. Although the dataset is multivariate network traffic data containing fields extracted from the packet headers, both ALAD and LERAD methods break down the multivariate problem into a set of univariate problems and sum the weighted results from range matching along each dimension. While the advantage of this approach is that it makes the technique computationally efficient and effective at detecting network intrusions, breaking multivariate data into univariate data has significant drawbacks especially for detecting attacks.

More recently, analytical studies on anomaly detection systems have been conducted. Lee and Xiang [56] used several information-theoretic measures, such as entropy and information gain, to evaluate the quality of anomaly detection methods, determine system parameters, and build models. These metrics help to understand the fundamental properties of audit data.

Statistical Packet Anomaly Detection Engine (SPADE) [57] is implemented as a Snort [29] pre-processor plug-in. In SPADE, the basic features are used to build a normal traffic distribution model for the monitored network. Traffic distributions are maintained in real-time by tracking joint probability measurements. During detection, packets are compared to the probability distribution to calculate an anomaly score. Highly anomalous packets are retained.

2. Machine Learning-based Anomaly Detection

Machine learning techniques are based on establishing an explicit or implicit model that analyses patterns and classifies them into the normal or malicious categories. A noteworthy characteristic of these schemes is the need for labeled data to train the behavioural model. This puts severe demands on resources.

In many cases, the applicability of machine learning principles coincides with the statistical techniques, although the former focuses on building a model that improves its performance on the basis of previous results. A machine learning ANIDS has the ability to change its execution strategy as it acquires new information. The main drawback of this technique is their resource expensive nature. Many machine learning techniques are

used in intrusion detection research field. A brief review on these techniques is given here.

System call and sequence analysis are widely used techniques for anomaly detection. Forrest et al. [38] established an analogy between the human immune system and an intrusion detection system, and proposed a methodology that involved analysing system call sequences of a program to build a normal profile.

More recent research based on call sequence approach is presented in [58], [59]. In [58] authors proposed an anomaly detection method based on statistical inference and an α-stable first-order model. Whereas in [59], authors used sequences and the parameters of the system calls executed by a process to identify anomalous behavior of the system.

Bayesian Network (BN) [60] is a graphical model that encodes probabilistic relationships among variables of interest. This is used in combination with statistical schemes. High computation and results depend on behavior of the target system.

Principal Component Analysis (PCA) [61] is a technique that is used to reduce the complexity of a dataset. It is not a detection scheme. In mathematical terms, PCA is a technique where n correlated random variables are transformed into $d < n$ uncorrelated variables. This makes it possible to express the data in a reduced form, thus facilitating the detection process [62]. Shyu et al. [63] proposed an anomaly detection scheme, where PCA was used as an outlier detection scheme and was applied to reduce the dimensionality of the audit data.

Markov model/A Hidden Markov model [64] is a statistical model where the system being modeled is assumed to be a Markov process with unknown parameters. The challenge is to determine the hidden parameters from the observable parameters.

Recently Zhiling et al. [65] proposed an automated mechanism for node-level identification. They used Principal Component Analysis (PCA) and Independent Component Analysis (ICA) technique for feature extraction, and outlier detection technique to discriminate expected normal behavior from abnormal behavior.

More recently, research in the field of back track attack is proposed in [66] and [67]. In [67], author proposed a Flexible Deterministic Packet Marking approach to find real source of Internet attackers. FDPM uses flexible flow based marking scheme to detect DDoS attacks launched on the Internet. In [66], two effective anomaly-based detection matrices are proposed to identify attack earlier.

3. Data Mining Approach for Network Intrusion Detection

Data mining is the ability to take data as input, and pull out patterns from the input data or deviations which may not be seen easily to the naked eye. It is also known as knowledge discovery. Data mining has been used for host-based and network-based intrusion detection as well as anomaly-based and misuse-based intrusion detection. In this section, we focus on the data mining applications on network-based IDS.

Lee and Stolfo [68] explored the application of different data mining techniques in the area of intrusion detection. In [69], Lee and Stolfo used multiple data mining techniques including classification, association rules and frequent episodes to build a framework for intrusion detection. They also introduced a feature construction system

for the classification, which categorized the connection based features into low-cost and high-cost features in terms of their computation time. Thus, different features were selected by classification model. The classification methods were basically rule-based algorithm such as RIPPER. Lee and Stolfo further extended their previous work in [70], where they applied association rules and frequent episodes to network connection record to obtain additional features. RIPPER was applied on the labeled attack traffic to learn the intrusion pattern. Barbara [71] proposed ADAM, which used applied association rules.

Bridges and Vaughn [72] used traditional rule-based expert system for misuse detection. They also contributed to anomaly detection using fuzzy logic and Genetic Algorithms (GA). They created fuzzy association rules from the normal dataset, and also built a set of fuzzy association rules from the new unknown dataset, and then compared the similarity between the two groups of rules. If the similarity is low, it indicates a possible intrusion in the new dataset. As stated by Bridges et al. [72], the concept of security is fuzzy in itself. In other words, the concept of fuzziness helps to smooth out the abrupt separation of normal behaviour from abnormal behaviour. Dickerson et al. [73] developed the Fuzzy Intrusion Recognition Engine (FIRE) using fuzzy sets and fuzzy rules.

Genetic Algorithm (GA) [74] has been used for tuning the membership function of the fuzzy sets and to select the most relevant features. Basically, GA is used to give rewards to a high similarity of normal data and reference data, and penalize a high similarity of intrusion data and reference data. The major advantage of GA is its flexibility and robustness as a global search method.

Clustering technique was used for finding patterns in unlabeled data with many dimensions. It is gaining popularity in the context of intrusion detection [75]. The main advantage of clustering is the ability to learn from and detect intrusions in the audit data, without requiring the explicit descriptions of various attack classes/types. As a result, the amount of training data that needs to be provided to the anomaly detection system is also reduced. The outlier detection scheme has been widely used for anomaly detection. An outlier can be identified using statistic features, distance, density and clustering techniques.

Mahoney and Chan extracted the features from the packet headers and clustered these features to build normal profiles. They classified connection which did not fall in any cluster as outlier. Taylor and Alves-Foss used less features. They extracted features from packet headers that were used to build the clusters. Each feature is treated as a variable, and each connection was abstracted to a point with multiple variables (features). The nearest neighbour algorithm was used to compute the distance for the outlier detection.

Other data mining approaches, such as neural network [76], were also explored for intrusion detection. Using neural network approach for intrusion detection, the neural network learns to predict the behavior of various users and daemons in the system. The main advantage of neural networks is their tolerance to imprecise data and their ability to infer solutions from data without having prior knowledge of the data. However, neural network based solutions have several drawbacks. Firstly, they may fail to find a satisfactory solution. Secondly, neural networks can be slow and expensive to train. Ramadas et al. [77] presented the anomalous network-traffic detection with self organizing maps (ANDSOM). The ANDSOM module creates a two dimensional self

organizing map for every network service that is being monitored. Anomaly detection schemes also involve other data mining techniques such as Support Vector Machines (SVM). Because data mining techniques are data driven and do not depend on previously observed patterns of network/system activity, some of these techniques have been very successful in detecting new kinds of attacks. However, these techniques often have very high false positive rates, resulting in a major challenge for the data mining approaches when applied to the real data.

2.5.2 Review from the Perspective of Payload-based Intrusion Detection System

The use of anomaly detection techniques in the context of network intrusion detection (ANIDS) is considered as a promising method of identifying and understanding novel attack behaviours. Network intrusion detection system can extract information from the packet header, packet payload or both. Header information is not helpful in detecting attacks against vulnerable applications (since the connection that carries the attack is established in a normal way). On the other hand, payload information is most suitable to identify attacks against vulnerable applications. Symantec corporations [78] reported that 69% of vulnerabilities were caused by web services, and it was reported in [1-3] that 75% of cyber attacks occurred at the application layer. Thus, organizations rely more heavily on payload-based anomaly intrusion detection for the protection of their networks. In this section, we concentrate on work which utilizes the techniques similar to what we will use in our research.

We present a brief review on the design techniques used by PAYL, SOM, POSEIDON and ANAGRAM intrusion detection systems.

Kuevo Kohonen [79] produced a low-dimensional map of high-dimensional data. The advantage of Kohonen's Self Organising Map (SOM) is the ability to add new inputs into patterns that it has already discovered. However, its original function was to compress data.

The concept of payload-based network anomaly detection was first proposed by Kruegel et al. [15]. They used payload to generate model and grouped 256 ASCII characters present in the payload into six segments that was 0, 1-3, 4-6, 7-11, 12-15 and 16-255. Chi-square test was used to compare the model.

In 2004, Wang and Stolfo [80] proposed PAYL (PAYLoad intrusion detection system), a state-of-the-art system, which used the combination of type, length and distribution to detect anomalous events. They developed the system using Byte Frequency Distribution (BFD) and 1-gram payload modelling. *n*-gram analysis is a primary means of intrusion detection system used to examine payload content. It is a language parser and a method to predict the next sequence in a dataset. BFD is the total number of *n*-gram occurrences, the values that are identified in sampling of payload data. PAYL uses the BFD and standard deviation to compute an anomaly score, which defines the similarity between attacks. Simplified Mahalanobis distance measure was used to compare new incoming traffic to the model. The system was evaluated against the 1999 Lincoln Lab IDEVAL dataset. The overall detection rate was close to 60% with a false positive rate less than 1%. PAYL uses whole payload. However, PAYL does not

consider relative position of different bytes inside the payload into account so that the structure of the payload is not modelled.

Bolzoni et al. [81] proposed POSEIDON, a two tier payload based anomaly intrusion detection system. In this system, payload length and frequency distribution were replaced with an artificial neural network known as a SOM. In the POSEIDON architecture, SOMs was used for pre-processing of packet payload and PAYL was used as a basis for intrusion detection. The SOMs mapped high-dimensional data points onto a single or multi-dimensional grid. The aim of the SOM was to identify similar payloads for a given destination address and port. SOM improved detection accuracy.

Unfortunately, rule-based and statistical-based systems are supervised learning systems. They require manual updating from network administrators. Hence, the ideal approach is to employ unsupervised learning, which does not require human interaction and to have the systems initially setup and then run autonomously. Anomaly-based intrusion detection systems are examples of unsupervised learning techniques.

To model the structure of payload, Wang and Stolfo proposed ANAGRAM [82]. ANAGRAM uses *n*-grams extracted from payloads using a sliding window of length *n* to create unique signatures. Wang et al. [80] used value of $n \geq 2$ to extract byte sequence information in a 256^n dimensional feature space. Due to the exponential growth in memory overhead and required training set size as *n* increases, the authors utilized Bloom filters to record *n*-grams observed from packet payloads during the training phase. They used supervised learning process to model normal traffic by storing *n*-grams of normal packets into one bloom filter and modeled attack traffic by storing *n*-grams

from attack traffic into a separate bloom filter. During the detection phase, packet payloads were scored according to the proportion of *n*-grams observed that were not contained in the Bloom filter. The major difference between binary *n*-gram analysis and 1-gram analysis is that the latter has limitations and can be easily replicated using different forms of mimicry tactics.

Unfortunately, due to the exponential growth in feature space as *n* increases, it is more difficult to construct an accurate model because of the curse of dimensionality and possible computational problem. Perdisci et al. [83] proposed McPAD (Multi classifier Payload-based Anomaly Detector), a state-of-the-art system. They created 2ν-grams and used a sliding window to cover all sets of 2 bytes, ν positions apart in a network traffic payload. Since each byte could have values in the range 0-255, and n=2, the dimensionality of the feature space was very high (256^2=65,536). The high dimensionality of the feature space was then reduced using a clustering algorithm. They combined multiple classifiers using a simple majority voting rule to make their model hard against polymorphic and polymorphic blended attacks.

Rieck and Laskov [84] also extracted language features in the form of high-order *n*-grams from connection payloads. They compared high order *n*-grams and words in connection payloads using vectorial similarity measures such as kernel and distance functions.

Correlating alerts is another important aspect of intrusion detection. PAYL primarily uses String Equality (SE), Longest Common Substring (LCS), and Longest Common

Subsequence (LCSeq). These techniques correlate attacks using ingress/egress signature matching.

POSEIDON uses ATLANTIDES and PANACEA, to correlate alerts and to classify attacks. Similar to PAYL, ATLANTIDES [85] correlates alerts using ingress/egress technique. However, a major difference is the system correlates attacks based on user requests that employ higher-level applications. The system is engineered to reduce false positives. Whereas, PANACEA [86] correlates alerts using Repeated Incremental Pruning to Produce Error Reduction (RIPPER) and Support Vector Machine (SVM). RIPPER uses IF-THEN rules to predict a class and SVM classifies input features.

However, all these systems have not considered correlations between the payload features and among the payloads. In contrast, we propose a novel approach to develop model for packet payload to detect anomalies in the packet payloads. Each network connection between a pair of hosts will be viewed as an object in an image (to be recognized through image processing), and each image will be viewed as a pattern to be classified as normal or anomalous traffic class based upon the given information about the connections. This model includes the correlation between various payload features and increases the detection accuracy. We use Mahalanobis Distance Map (MDM) technique to determine the hidden correlations between payload features and to calculate the difference between normal and anomaly traffic of network. For feature reduction, we propose to use Linear Discriminant Analysis (LDA) and Principle Component Analysis (PCA) techniques. This will reduce the computational complexity. In addition, it will improve time to train and test the model and improve detection accuracy as well.

2.6 Conclusions

In this chapter, we have presented various security strategies used to mitigate attacks and protect the system from inside and outside attack as well from mis-configuration of the system. We have further discussed the intrusion detection problem as a pattern recognition problem and discussed a design process to build an intrusion detection system. We have then presented the taxonomy for intrusion detection systems. We have shown how an intrusion detection system can be classified on the basis of the data source that it analyses and the detection model that employs, and their strength and limitations. We have discussed here the performance evaluation matrices for an intrusion detection method using *F*-Value and *ROC* curves.

We have presented literature review in the perspective of techniques used for the design of intrusion detection system, particularly to anomaly-based network intrusion detection systems and also in the perspective of payload-based anomaly intrusion detection system, which is the main focus of this research work.

Anomaly-based systems have been extensively researched but there are still open issues that limit the application of an anomaly-based intrusion detection system in real environments, despite its advantages over a signature-based system. Moreover, header based anomaly systems are unable to detect attacks against vulnerable applications (since the connection that carries the attack is established in a normal way). There is a need to design accurate payload-based intrusion detection system to protect the network from web-based attacks. We will address this issue in the next chapter and provide a frame work for payload-based intrusion detection system, which can detect attacks more

accurately. Chapter 3 and Chapter 4 discuss proposed solutions to these problems, developed through this research.

CHAPTER 3

GSAD: Geometrical Structure Anomaly Detection System

Introduction

As mentioned in Chapter 2, several approaches have been proposed to mitigate computer attacks in a network. Intrusion detection system appears to be one of the most effective solutions for defending networks against malicious users. Sophisticated IDSs generally fall into two categories: misuse detection (or signature detection) and anomaly detection. Network based anomaly detection can be applied at a packet header level, packet payload level or both to process the network traffic. An IDS that simply analyses packet header information cannot adequately secure a network from malicious attacks. The alternative is to perform deep-packet analysis and explore the payload of each incoming packet. Examining the payload content is an important aspect of network security, particularly in today's volatile computing environment, where web applications are common attack targets.

SOM, POSEIDON, PAYL, Anagram and McPAD are examples of the payload-based anomaly intrusion detection systems. Most of these research except Anagram and McPAD use 1-gram analysis procedure in building a statistical model for certain types of data based on the byte frequency and do not have structural information of the payload features. However, all of these IDSs have high false positive rates since dependencies and correlations of the features are intrinsically neglected. The goal of this research is to develop a novel payload-based anomaly detector that improves the detection rate with a relatively lower false positive rate by using an image processing technique. We intend to use the Mahalanobis Distance Map (MDM) approach, which is a pattern recognition technique, to identify patterns of packet payloads. It is based on the idea that the geometric structures of all human beings are similar although they wear clothes of different colours. Our work is motivated by this idea and the MDM is used to discriminate normal payload from anomalous payload. MDM is promising in extracting the hidden correlations between features and the correlations among network packet payloads. It also partially captures structural information of payload. These correlations and structural information help improve the detection performance and reduce false positive rate.

In the previous chapter, we have reported the relevant work of anomaly detection approach. However, this approach has many issues. We discuss these issues in the following section.

High number of false alarms, especially false positive alarms, is the key issue of the anomaly detection system. This high false positive rate is due to the fact that an anomaly detector considers unseen normal behavior also as an anomaly. Anomaly detection

systems are highly dependent on the quality of the training data. Training data, which may contain attack samples, can leave an anomaly detection system non-functional because the attacks will be learned as normal traffic and IDS will never produce an alert related to them. Additionally, the lack of sufficient training samples degrades the ability of an anomaly detection system to correctly identify attacks. Anomaly detection algorithms have high overhead, preventing their deployment in high bandwidth environments. New attacks and services appear every day and IDS must be able to cope up with the new environment. Finally, anomaly detection systems have poor descriptive power regarding the types of attacks they detect.

The chapter is divided in two parts. In the first part, we present detailed information on our proposed Geometrical Structure Anomaly Detection (GSAD) model. In the second part, we first discuss the implementation of the GSAD model in the HTTP environment. The HTTP environment is selected for implementation in our work because a considerable amount of Internet traffic is made up of HTTP traffic. Then, we compare its performance against the state-of-the-art PAYL model. We focus on successful HTTP requests that use the GET method and contain the query component. Malicious inputs can be sent to a web application using the parameter-value portion of the query.

The more detailed organisation of this chapter is listed as follows. Section 3.1 provides a description of GSAD, an IDS based on Mahalanobis Distance Map (MDM) approach. Experimental results and discussion are presented in Section 3.2. Section 3.3, provides a brief introduction of HTTP and some examples on web-attack. In Section 3.4, we present the implementation of GSAD in the HTTP environment. Section 3.5

describes the experimental setting and dataset used on which GSAD has been evaluated. Experimental results and analysis are given in Section 3.6. Finally, we summarize our work in Section 3.7.

In contrast, we propose Geometrical Structure Anomaly Detector (GSAD), a novel payload-based IDS, to detect intrusion in the network. This model uses an image processing technique that has been used for human face recognition [87]. Each network connection between a pair of hosts is viewed as an object in an image (to be recognized through image processing), and each image is viewed as a pattern to be classified as normal or anomalous traffic based upon the given information about the connections. Similar to other anomaly detection systems, GSAD models the normal behavior of the network traffic rather than the malicious ones. Moreover, the most significant contribution of GSAD is the integration of MDM (geometrical structure) approach and payload-based anomaly detection systems.

3.1 GSAD-Geometrical Structure Anomaly Detection System

In this section, we elaborate on our new approach. Firstly, we present the framework of our GSAD system. Then, we discuss the modules in the framework, namely Payload feature classifier, Payload feature analyst, Payload geometrical structure module (PGSM), Attack recogniser and Alarm Acknowledgement.

3.1.1 Framework of the Proposed Intrusion Detection System

We present the framework of the GSAD, a payload based anomaly detector derived from an image processing technique. The complete framework of our proposed intrusion

detection system (GSAD) has three stages as shown in Figure 3.1. In this figure, solid arrows indicate data flows inside the GSAD model. The Payload Feature Analyst module and the Payload Geometrical Structure module together form the *Geometrical Structure Payload Model* (GSPM). A comprehensive description of each module is given below.

The first stage of GSAD consists of payload classification and data preparation. For data preparation, raw data are collected from the network input, e.g., tcpdump file, which provides a list of connections. Incoming network traffic is filtered according to the type of application and payload length.

In the second stage of the framework, payload features are analyzed using *n*-gram text categorization technique, which converts the network traffic packet payloads into a series of feature vectors. These feature vectors describe the patterns of the incoming traffic. Correlation between the payload features is calculated and an MDM is created for normal network traffic as a normal profile, which is used for the classification of the new incoming network traffic in the next stage.

In the third stage, Mahalanobis Distance [88] criterion is used to measure the dissimilarity between the pre-developed normal profile and the profile of a new incoming network packet. Score value is calculated, which is used for classification of normal and malicious payloads and for the activation of the alarm. Detailed description of each module is given in the following subsections.

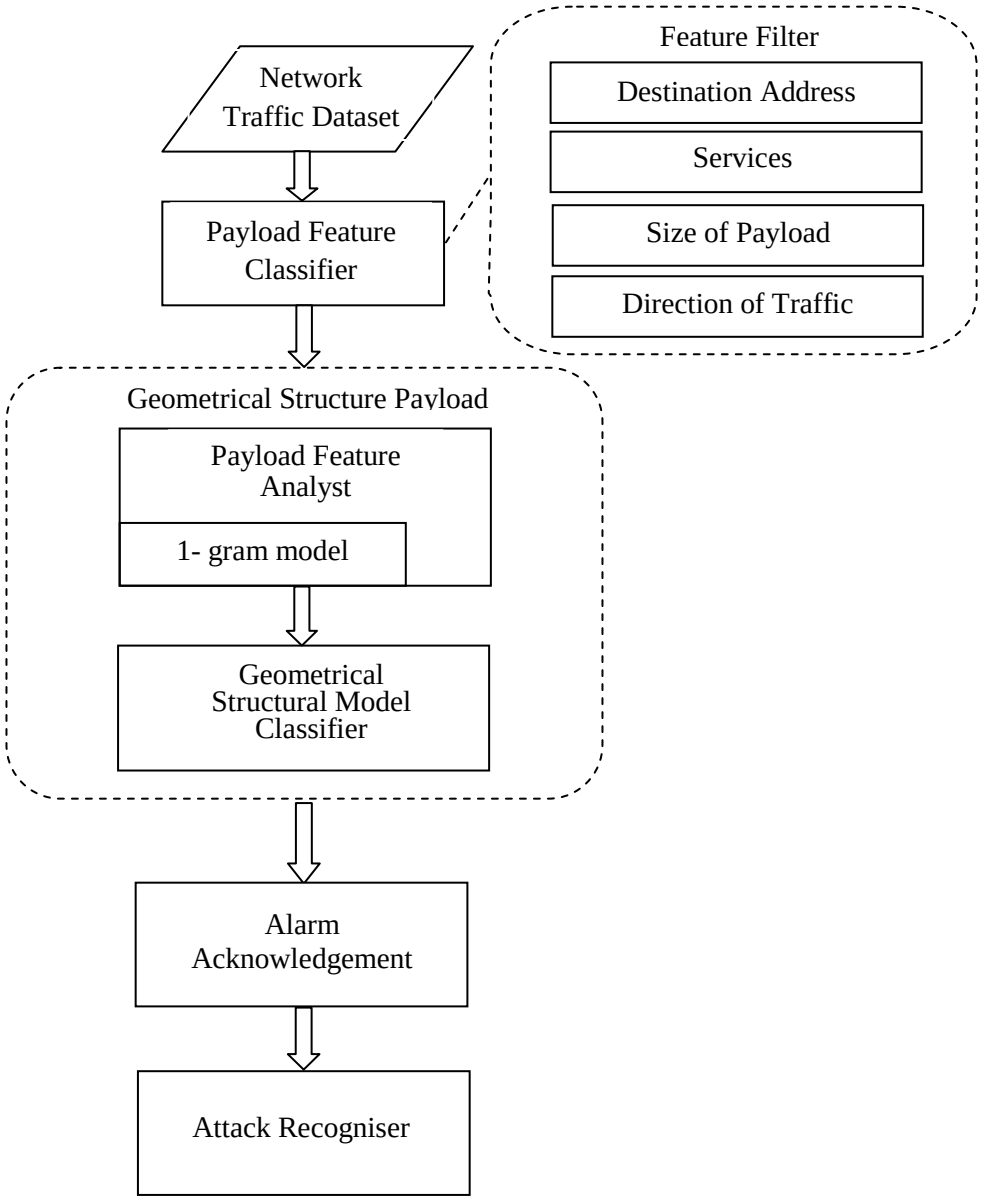


Figure 3.1: Framework of Geometrical Structure Anomaly Detection System

3.1.2 Framework Modules

In this section, we provide a step-wise description and technical details of all modules contained in our proposed IDS framework.

a) Payload Feature Classifier

Payload feature classifier is the first stage of the framework, where different datasets are prepared. When a packet is received from GSAD, the payload is extracted. We group network traffic into various categories using Wireshark [89], which is a traffic analyzer and separates the network traffic based on type of services, destination address, payload length and direction of network traffic flow. The source of network traffic can be real network (for real-time operation) or collected tcpdump files. The prepared dataset is used by the next stage of this intrusion detection system.

b) Payload Feature Analyst

The payload feature analyst is the first key constituent of the Geometrical Structure Payload Model (GSPM). For feature extraction, text categorization technique [40] is used, which is responsible for payload feature analysis and feature construction. It extracts raw features using *n*-gram (i.e., the sequences of *n* consecutive bytes in the payload) text categorization technique (n=1 in our case) from the packet payload and converts observations into a series of feature vectors. Each payload is represented by a feature vector in a *256*-dimensional feature space. The mathematical model for *1-gram feature construction* is discussed as follows.

c) 1-gram Feature Construction

The 1-gram payload model is a payload based statistical model, which does not take network packet header features into account. In addition to average frequency of each ASCII character (0-255), it calculates the mean value and the standard deviation of each

feature's frequency and correlations between these features. Each payload is represented by a feature vector in a *256*-dimensional feature space using

$$f_i = \frac{O_i}{\sum_{j=1}^{256} O_j}, \tag{3.1}$$

where O_i is the occurrence of *i*-th *n*-gram. The overall value of the relative frequencies is given by

$$\sum_{j=1}^{256} f_j = 1. \tag{3.2}$$

Thus, a packet payload is then denoted by a relative frequency vector $q = [f_1 f_2 \cdots f_{256}]^T$, which represents a pattern in the network payload in a 256-dimensional feature space. Here, *T* stands for 'transpose' of a matrix. We assume that there is a network traffic dataset with *n* network packets. The mean value and standard deviation of each byte's frequency are described in Equations 3.3 and 3.4 respectively.

$$\bar{X} = [\bar{x}_1\ \bar{x}_2\ \cdots \bar{x}_{256}]^T, \tag{3.3}$$

$$\bar{\sigma} = [\bar{\sigma}_1\ \bar{\sigma}_2\ \cdots \bar{\sigma}_{256}]^T, \tag{3.4}$$

where $\bar{x}_i$ and $\bar{\sigma}_i$ are the mean value and the standard deviation of each feature's frequency and given by

$$\bar{x}_i = \frac{1}{n}\sum_{k=1}^{n} x_{i,k}\ (1 \le i \le 256), \tag{3.5}$$

$$\bar{\sigma}_i = \sqrt{\frac{1}{n}\sum_{k=1}^{n}(x_k - \bar{x}_i)^2}\ (1 \le i \le 256). \tag{3.6}$$

The mean value and standard deviation vectors, $\bar{X}$and $\bar{\sigma}$, are stored in a model *M*. The network traffic dataset consists of traffic generated by the various network services. Therefore, we need to filter network traffic for a selected service based on the following

features: size of payload, destination address, services and direction of traffic flow. Then, models are developed according to types of service and extracted group of features.

d) Payload Geometrical Structure Model

Payload Geometrical Structure is the second key constituent of GSPM, used for payload analysis. It includes techniques to determine where and how to separate data input prior to the classification or pattern recognition processes. The Mahalanobis Distance Map approach [87] develops geometrical structure models of the payloads. The following subsection presents a practical application of g*eometrical structure model* in payload-based anomaly detection.

e) Geometrical Structure Model

The Geometrical Structure Model (GSM) is a new concept in intrusion detection system. Network traffic profile is generated using Mahalanobis Distance Map (MDM) which captures complex non-linear correlations of the data. By using MDM, the hidden correlations between the features (256 ASCII characters) present in the payload of length *L* and the correlations among packets are obtained as follows.

$$\sum_i = (x_i - \mu)(x_i - \mu)^T (1 \le i \le 256), \tag{3.7}$$

$$d_{(i,j)} = \frac{(x_i - x_j)(x_i - x_j)^T}{\sum_i + \sum_j} (1 \le i, j \le 256), \tag{3.8}$$

$$D = \begin{bmatrix} d_{(1,1)} & d_{(1,2)} \cdots & d_{(1,256)} \\ d_{(2,1)} & d_{(2,2)} \cdots & d_{(2,256)} \\ \vdots & \vdots \ddots & \vdots \\ d_{(256,1)} & d_{(256,2)} \cdots & d_{(256,256)} \end{bmatrix}, \tag{3.9}$$

Where x_i represents the *i*-th feature in the feature vector q, μ denotes the average of each feature, $d_{(i,j)}$ defines the Mahalanobis distance between the *i*-th feature and the *j*-th feature, $\sum_i$ is the covariance value of each feature, and finally *D* is the MDM, the image of a network packet (the pattern of a network packet payload). *D* is a 256×256 image that represents the distance from one block to another block. Distance map *D* is used to generate the network traffic profiles (normal and attack) of the training and test data. These profiles are used for the classification of incoming network traffic.

The above basic formulas are used in the GSM model to process a large amount of sample network traffic with normal behaviors. To include the variations in different normal payloads, we consider all normal traffic as a group. The distance maps of normal behaviors for all traffic in the groups are calculated using Equations 3.7 and 3.8 and shown in Equation 3.9. We assume that the group has *m* normal packets inside. Then, the distance maps of individual normal packets are: D_1^{nor}, … , D_m^{nor}, and the averages and variances for all elements of the distance maps are computed by Equations 3.10 and 3.11.

$$\bar{d}_{nor(i,j)} = \frac{1}{m}\sum_{k=1}^{m} d_{nor(i,j),k}\ , \tag{3.10}$$

$$\sigma^2_{nor(i,j)} = \frac{1}{m}\sum_{k=1}^{m}\left(d_{nor(i,j),k} - \bar{d}_{nor(i,j)}\right)^2, \tag{3.11}$$

Where $1 \le i,j \le 256$ and $d_{nor(i,j),k}$ is the (i,j) element of distance map D_k^{nor}. $\bar{d}_{nor(i,j)}$ and $\sigma^2_{nor(i,j)}$ are all kept in a model M_{nor} for further evaluation.

In the attack recognition phase, an incoming packet follows the same preprocessing procedure (as discussed in Equations 3.7 to 3.11) to construct its Mahalanobis Distance Map:

$$D_{obj} = \left[d_{obj(i,j)}\right]_{256\times256}. \quad (3.12)$$

f) Attack Recogniser

Ideally, the attack recognition phase is to identify the difference between the normal and abnormal patterns. In this study, GSAD uses Mahalanobis distance criterion to measure the dissimilarity between the developed profile and new incoming traffic profile and recognize the attacks. It compares each incoming packet payload profile against the pre-developed base-line normal payload profile to calculate weight factor score. Weight *w* is calculated using Equation 3.13 to detect an intrusive activity

$$w = \sum_{i,j=1}^{256,256} \frac{\left(d_{obj(i,j)} - \bar{d}_{nor(i,j)}\right)^2}{\sigma^2_{nor(i,j)}}, \quad (3.13)$$

If the weight factor *w* exceeds the threshold, the incoming packet is considered as an intrusion. This weight factor score value is used for the payload classification and for the activation of the alarm.

g) Selection of threshold value

While setting the threshold is entirely subjective, ultimately it should be set to capture all attacks (ideally). Standard deviation of the observed samples is an appropriate criterion used to determine the threshold value. A series of experiments are conducted for standard deviation (δ) varying from $\pm\delta$ to $\pm3\delta$ to determine an appropriate value of threshold. We consider a threshold value between -3δ and +3δ in our experiments for achieving optimal detection rates and low false positive alarm rates. Assuming the distribution is normal, three standard deviations account for 99% of the sample

population are studied. The incoming request is considered as an attack or a threat if the weight factor is more than +3δ or less than -3δ.

h) Alarm Acknowledgement

In this module, the attack alarm will be generated if the score of a test packet payload is larger than the threshold and is then reported to the administrator. Otherwise, it will consider the packet as a normal.

3.1.3 Base-line Profile Generation

Training of the GSAD model is required to generate base-line profile of network packet (application) behavior and evaluate whether intrusion detection systems truly identify known and unknown attacks. To do this, anomaly-based IDSs will use normal data to learn the behavior of network packets during training and generate a normal profile. In the training stage, the distance maps of all sample images are constructed using Equations 3.8 and 3.9. Let us assume that there are *m* normal packet images. Then, the average distance $\overline{d}_{nor(i,j)}$ of element *(i,j)* is computed using Equation 3.10 and the average covariance of *m* packets is calculated by Equation 3.11. This will generate a normal profile of network packets for an application. This profile is saved as a base-line profile (GSAD model) and is used for the testing purpose of each new incoming packet.

3.1.4 Model Testing

Testing is a critical process necessary to evaluate the capability of an IDS and to identify intrusions. In GSAD framework, as shown in Figure 3.1, testing is performed by the Attack Recognizer module. Whenever a new payload *p* is received, we generate a profile

(as described in Section 3.1.3). Then, we calculate weigh factor *w* and compare it with the pre-determined threshold value. If value of '*w*' is greater than the threshold value for payload *p*, then payload *p* is classified as an intrusion and an alarm will be activated.

3.2 GSAD Evaluation

In this section, we report experimental results. We present the results based on the accuracy of GSAD first. We evaluate GSAD on DARPA 1999 dataset [90]. Although the DARPA 1999 dataset is criticized by McHugh [7] for many of its weaknesses, it is the only publicly available dataset which is considered as a benchmark dataset to test intrusion detection systems.

In the following subsections, we first present our experimental environment and brief information on the dataset, and then we discuss the training and testing in our model.

3.2.1 Experimental Setup

GSAD is written entirely in Matlab. We conduct experiments on a computer with two 3.33 GHz 8MB cache Quad Core Xeon CUPs and 48GB DDR3-1333 ECC memory. This is a shared computational environment, which is used for heavy mathematical calculation and modelling experimentation. However, performance of our model is heavily influenced by the number of processes running simultaneously.

3.2.2 DARPA 1999 Dataset

The DARPA 1999 IDS dataset was collected at MIT Lincoln Labs to evaluate intrusion detection systems. Entire network traffic was recorded in tcpdump format. The dataset consists of three weeks of training data and two weeks of test data. In the training data,

there are two weeks of attack-free data and one week of data with labeled attacks. These attacks are grouped into five classes as scan or probe, DoS, R2L, U2R and data. A Series of experiments on DARPA 1999 [90] dataset are conducted to evaluate the performance of our proposed model.

For our experiments, we consider inbound TCP traffic, extracted from the week 1, week 2 and week 3, which was captured between routers and victims. We use Wireshark [89], a packet analyzer, to extract TCP traffic. First 150 bytes of packet payload are used for experiments and to identify various attacks coming through port 80 and port 25.

3.2.3 Experimental Results and Analysis

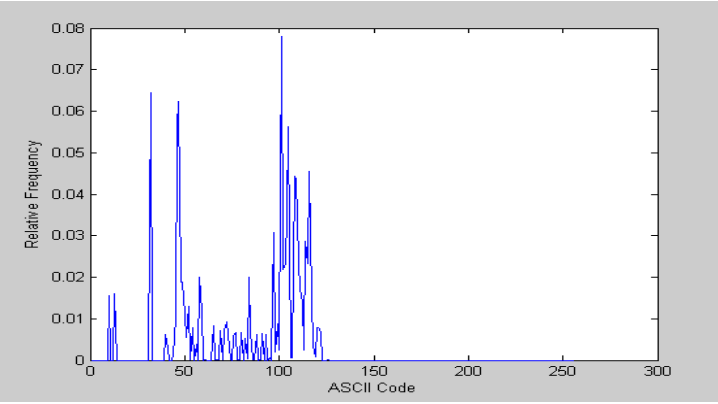
In the first part of our experiments, we represent a payload by a feature vector as described in Subsection 3.1.2. Then, we compute Mahalonobis distance between features, and generate MDM, which represents pattern in the payload (as discussed in Subsection 3.1.2), and generate a MDM profile as described in Subsection 3.1.3 and save it as a base-line model. We train our model on the DARPA dataset using week 1 and week 3 attack-free data. The model is then evaluated using week 2 data which contain 43 instances of 15 different attacks. Then, we evaluate the accuracy of GSAD in detecting three types of attacks [90],[91] namely Crashiis attack, Back attack and Mailbomb attack. A brief explanation of these attacks is given in the following section.

- For port 80, the attacks are often malformed HTTP requests and are very different from normal requests. For instance, *Crashiis Attack is a* Denial of Service attack against the NT IIS web server. The attacker sends a malformed GET request via

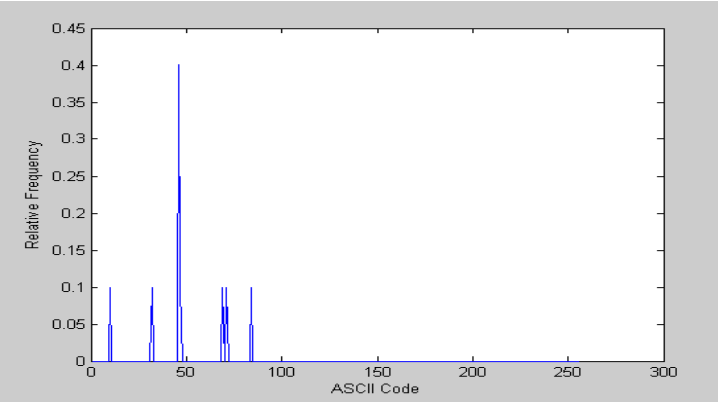
telnet to port 80 on the NT victim. The command "GET ../.." crashes the web server (and sometimes crashes the ftp and gopher daemons as well).

- *Back Attack is a* DoS attack against Apache web server, where client sends an HTTP requests “GET //////////….” with more than 6000 slashes, these requests will slow down the server and be unable to process other requests.
- *Mailbomb Attack is a s*imple attack where an attacker floods a user’s mailbox with thousands of junk mails.

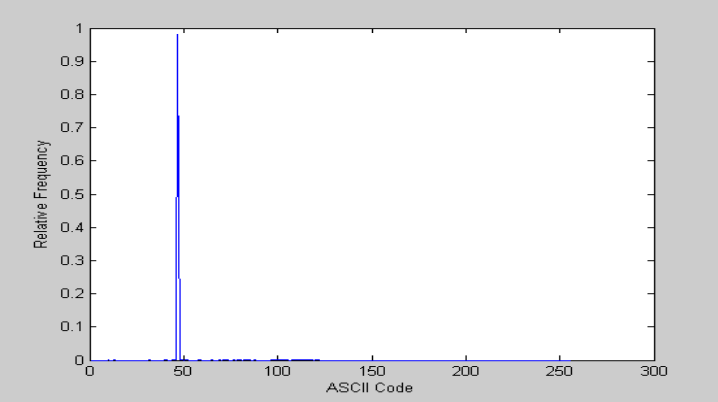
Figures 3.2(a)-(c) show the average relative frequency of each byte for normal HTTP payload, Crashiis attack payload and back attack payload respectively. In Figures 3.2(a)-(c), payload features (ASCII characters) are plotted on *X*-axis and relative frequency of each byte in the payload on *Y*-axis. The MDM (geometrical structure model) of normal HTTP payload, Crashiis attack payload and attack payload are given in Figures 3.3(a)-(c) respectively. MDM presents the correlations between the features. The MDMs in Figures 3.3(a)-(c) demonstrate that the correlations between normal features are different from the correlations between attack features.



(a) normal HTTP payload



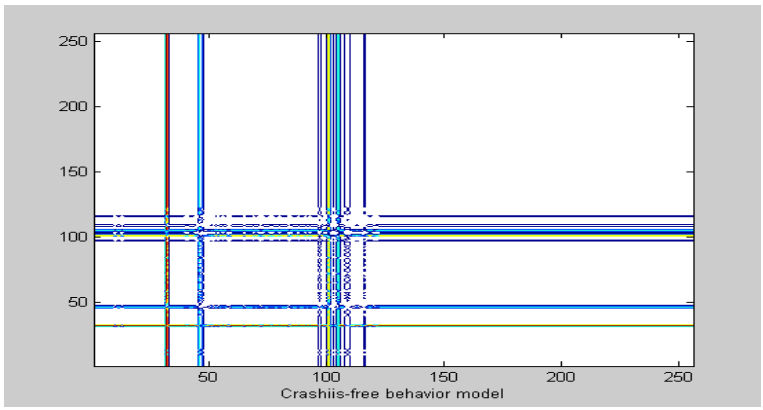
(b)



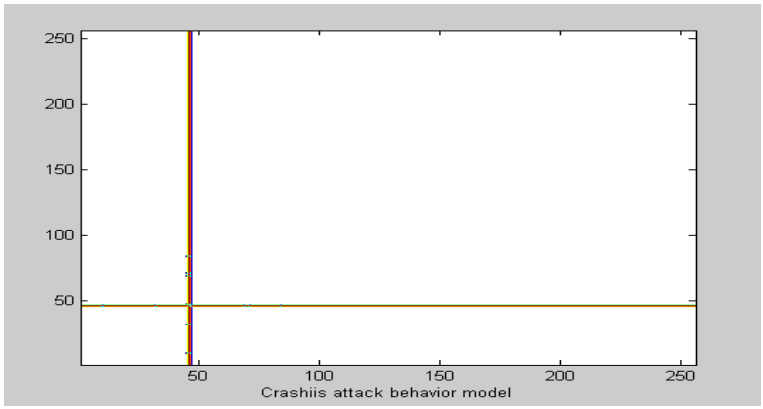
(c)

Figure 3.2: Average relative frequency of each byte. (a) normal HTTP payload; (b) crashiis attack payload; (c) back attack payload

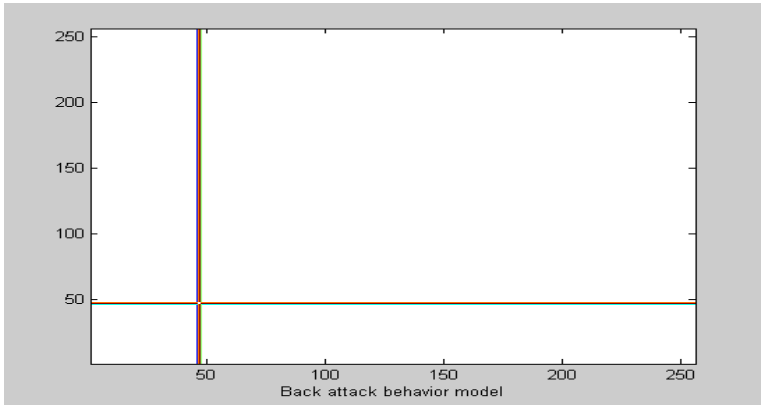
It can be seen from Figures 3.2(a)-(c) that the average relative frequencies of bytes appearing in the normal payload and various attack payloads are very different. For the Crashiis attacks, the "." character has the highest frequency whereas other characters share equal frequencies. Relatively, the statistical nature of the back attack is totally different and it has a perfect match with the signature. Around 98 percent of characters in the back attack packets are "/".



(a)



(b)

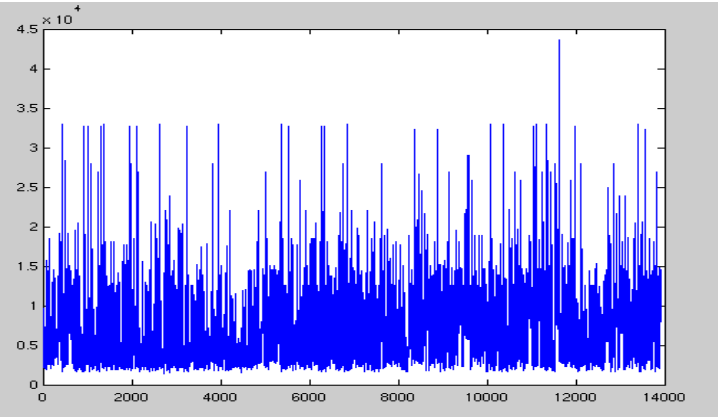


(c)

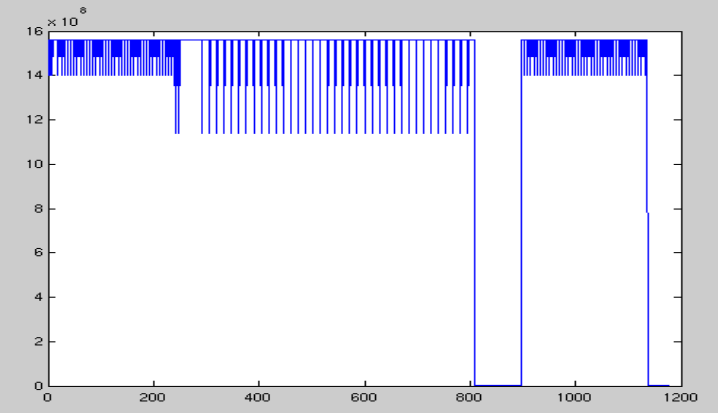
Figure 3.3: Average MDM images. (a) normal HTTP payload; (b) crashiis attack payload; (c) back attack payload

Results reported in Figures 3.3(a)-(c) demonstrate differences between the normal MDM image and various attack MDM images. Therefore, we have strong evidences to distinguish various attacks from normal packets.

We use weight factor scores to classify normal and attack packets. Results for the weight factor scores for normal HTTP request packet and Back attack packets are presented in Figure 3.4. In the Figure 3.4, *X*-axis represents the number of test packets and *Y*-axis represents weight factor score values. This is clear from Figure 3.4 that the GSAD model is able to detect different types of attacks without any prior knowledge of the attacks.



(a)



(b)

Figure 3.4: Weight factor scores. (a) normal HTTP request packets; (b) back attack packets

From Figures 3.4(a)-(b), we can conclude that the weight factor score for normal packets is much smaller than the weight factor scores of back attack packets.

We use the Receiver Operating Characteristic (ROC) curve method to evaluate the performance of GSAD model on DARPA 1999 dataset. The ROC curve shows relationship between false positive rate and detection rate. ROC curve is shown in Figure 3.5. To show more clarity of results, the *X* axis and *Y* axis values are shown in the range [0.8e-03, 1.5e-03] and [0.65, 1] respectively. Our model could achieve 100 percent detection rate with a very low false positive rate of 0.087 percent on DARPA 1999 dataset.

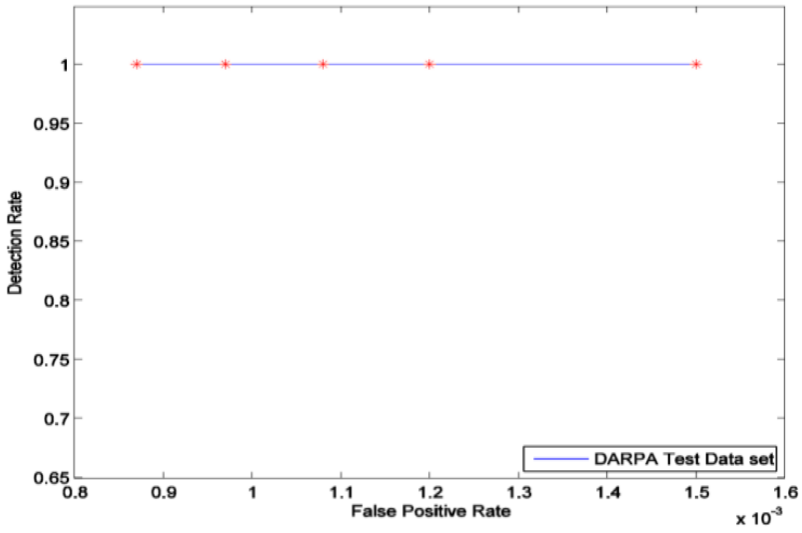


Figure 3.5: ROC Curve for the accuracy of the GSAD model

Experimental results illustrate good performance of our GSAD model in detecting various types of attacks and discriminating normal and attack packets. This is clear from the geometrical structure models which explain the correlation among 256 ASCII characters and also correlation among the packets.

To further investigate the attack detection accuracy and detecting a variety of web-based attacks, we implement our framework in HTTP environment. In the next section, we will discuss the implementation of GSAD model in HTTP environment to detect web-based attacks coming through HTTP protocol in the network.

Part II: GSAD Model in HTTP Environment

With the popularity of the internet, more and more systems are subject to attack by intruders. 23 million attacks have been reported in HPDV Lab survey report [5]. Specifically, web applications continue to pose one of the biggest risks to company networks, often due to vulnerabilities present in the systems. Thus, securing a network server is an important but difficult task. This has motivated us to extend our proposed GSAD model for anomaly detection on HTTP requests sent to web servers. In this section, we focus on the detection of intrusions/attacks coming in the network through HTTP traffic and investigate detection accuracy of our GSAD model in HTTP environment.

3.3 HTTP and Examples on Attacks

3.3.1 HTTP

HTTP is a stateless, application-level protocol described by the Internet standard RFC 2616 [92]. The protocol operates on a client-server mode. While a HTTP request is a string, it is also structured. HTTP has become the universal transport protocol for almost all kinds of web-server applications. HTTP passes through most of these firewalls, with little or no trouble at all. Hence, web application developers started using HTTP as a

transport protocol for their new software. Some of the examples for this include tunneling secure shell connections, Microsoft RPC for accessing Exchange (email) servers, etc. The creation of all these new services over the HTTP protocol creates additional opportunities for the intruders [93]. They also create a lot of variations in the characteristics of the HTTP requests. The vast majority of requests use GET, but other methods, such as HEAD and POST, exist, and extensions to the HTTP standard define more.

A web application generates an output in response to a user request, which is a string containing a number of parameter names and their respective parameter values. RFC 2616 [92] defines the structure and the syntax of a request with parameters (Figure 3.6).

```
GET  /modules.php?name=New&file=Article&sid=25  HTTP/1.1
```

Figure 3.6: A Typical HTTP (GET) request with parameters

The method used in this example is GET. The fields of our interest are: the presence of a path, a number of parameter names and their respective values (in Figure 3.6, the parameter names are 'name','file' and 'sid', and their respective values are: "New", "Article" and "25"). The set of parameters is finite. A value can be any string, though not all of the strings will be accepted. Since no type is defined, the semantics of each parameter is implicitly defined within the context of the web application and such parameters are usually used in a consistent manner (i.e., their syntax is fixed).

3.3.2 HTTP Attack Examples

As we know that attackers introduce bugs into code, the diversity of attacks against HTTP is high. This diversity implies that knowing one attack provides no assistance and information about the structure of the next one. Here, we present some of the examples on HTTP attack. Indeed, most existing attacks are only one request long. HTTP is a stateless protocol that contains structure useful in separating high- and low-variability portions of the request.

One famous attack is “Nimda worm”, as shown in Figure 3.7, which is present in the resource path [94]. It is capable of affecting both the clients that use any version of Windows as the host operating system and also the servers running Windows NT or 2000.

```
GET /scripts/root.exe?/c+dir HTTP/1.0
Host: www
Connection: close
```

Figure 3.7: Nimda attack

This attack targets a collection of bugs in Microsoft systems, and uses the fact that the default configuration enables the attacker to exploit the vulnerability. This attack has several known variants, all of them exploiting the vulnerability in the Windows operating system.

Another popular attack type is the Cross-site scripting attack [95]. This vulnerability enables the malicious user to use a web application program to inject code, most likely

as client (browser) side scripts, into the web pages viewed by a lot of other users, who then become the victims of this attack. The malicious user causes a legitimate web-server to send a response page to a client's browser that contains malicious script or HTML that the attacker chooses.

Back attack [91], an attack in a resource path, is shown in Figure 3.8.

```
GET
////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////////
```

Figure 3.8: Back attack, 300 /s, extra /s have been deleted to save space

3.4 Implementation of GSAD in HTTP Environment

The GSAD framework proposed in Section 3.2 is very general and can be easily customized by adding domain specific knowledge as per the specific requirements of the network in concern, thereby, giving flexibility in implementation. In this section, we implement our GSAD model in the HTTP environment.

The analysis technique uses the particular structure of HTTP queries that contain parameters. GSAD uses knowledge related to the application layer protocol by mapping HTTP payloads into 256 features space. Each feature represents the occurrence frequency of ASCII character in the HTTP request payload of one of the 256 possible byte values. A simple model of normal HTTP traffic is then constructed using the MDM for these features to recognize intrusive action.

In our experiments, we parse 150 bytes of HTTP GET request payload by using a sliding window of 1 byte length and count the occurrence frequency of each feature in the payload. The HTTP GET request payload is represented by a pattern vector in a 256–dimensional feature space. A profile is created for HTTP GET request payload using Equations 3.1 to 3.12 as explained in Section 3.2. Then, we design a number of experiments based on Figure 3.1 to determine the performance of GSAD in HTTP environment.

3.5 Evaluation in HTTP Environment

In this section, we report the results of our experiments. We first present the results based on the accuracy of GSAD in HTTP environment and then compare it to PAYL [80] and McPAD [83]. We evaluate GSAD on DARPA 99 dataset [96] and Georgia Institute of Technology Attack Dataset (GATECH) [83, 97]. In the following subsections, we first present our experimental environment and brief information on the dataset.

3.5.1 Experimental Setup

The experimental setup used in this section is similar to that explained in Section 3.2.1. Code for the implementation of GSAD in HTTP environment is written using Matlab 2009b.

Assumptions

Following assumptions are made to evaluate the robustness of GSAD model in HTTP environment.

- The attackers may have some information about the IDS deployed in the organization, but do not have access to legitimate traffic from the same network where the IDS is deployed.
- Polymorphic attacks used by the attacker are not specific to normal behavior of the attacked network.
- Attack events are comparatively smaller than the normal events.
- There are no mis-configured abnormal attacks in GATECH attack dataset.
- A targeted attack is crafted to attack specific (often custom) systems/applications within a network.

3.5.2 Datasets

The following subsection describes characteristics of two datasets that we use in our experiments. These two datasets are used by the state-of-the-art payload-based IDSs that we will compare in this paper.

a) DARPA 1999 Dataset

DARPA 1999 dataset is the only publicly available, large and well labeled dataset, and is still the most widely used public benchmark for testing intrusion detection systems. In Section 3.3.2, we have discussed characteristics of this dataset.

b) GATECH Attack Dataset

The GATECH attack dataset is also publicly available and contains traces of real attack traffic. This dataset is a collection of 63 attack requests. Attacks are collected from various online security forums and other online sources. GATECH is a labeled dataset

and has several non-polymorphic HTTP attacks provided by Ingham and Inoue [97] and several polymorphic HTTP attacks generated by Perdisci et al. [83] using both the polymorphic engine CELT and Polymorphic Attacks. The GATECH attack data set is divided into four groups of attacks used as test data, namely Generic attacks, Shell-code attacks, CLET attacks known as Polymorphic attacks and Polymorphic blending attacks. The total number of attacks and attack packets in the dataset are 6,512 and 72,539 respectively. All 66 HTTP generic attacks and 205 total HTTP request attack packets from the attack dataset are used in our experimentation.

3.5.3 Experimental Results and Analysis

In this section, we present a detailed description of the experiments conducted using our GSAD model to detect various attacks coming through HTTP services. The DARPA 1999 dataset is used for constructing a normal profile of HTTP protocol. For attack traffic, the HTTP-related attacks contained in DARPA 1999 dataset and GATECH attack dataset are used.

Experimental Procedures and Methodology

A series of experiments are conducted for the training and testing of the GSAD model for incoming HTTP requests from client to web server(s).

During the training phase, an average normal profile is generated using a geometrical distance algorithm for HTTP GET request, and then the weight factor score and threshold are calculated. During the test phase, similarities between the new incoming HTTP request and the average profile of the HTTP requests are calculated using MDM and a weight factor. The weight factor is used to determine whether the incoming packet

is an attack packet or not. We configure several model settings to optimize the performance of our GSAD model. We introduce a smoothing factor γ to avoid the weight factor becoming infinite due to possibility of the variance $\delta^2_{nor\ (I,j)}$ being equal to zero. The smoothing factor γ reflects the statistical confidence of the sampled training data. The larger the value of γ is, the less confidence of the samples truly representing the actual data is. Analysis of samples during the training phase for different values of γ is performed. We choose 0.0000001 as the value of γ because it is small enough to cause very limited impact on the actual correlation among packets.

Selection of threshold value is very important for the evaluation of IDS as this directly impacts the performance of the IDS. According to Bolzoni & Etalle [81], a lower threshold yields more alarms, significantly raising the false positive rate. In contrary, a higher threshold yields lower alarms and thus would lower the false positives. While setting the threshold is entirely subjective, ultimately it should be set to capture all attacks (ideally). Standard deviation of the observed samples is an appropriate criterion used to determine the threshold value. A series of experiments are conducted with different values of standard deviation (δ) to determine an appropriate value of threshold. We consider a threshold value between -3δ and +3δ in our experiments for achieving optimal detection rates and low false positive alarm rates. Assuming the distribution is normal, three standard deviations account for 99% of the sample population are studied. The incoming request is considered as an attack or a threat if the weight factor is more than +3δ or less than -3δ.

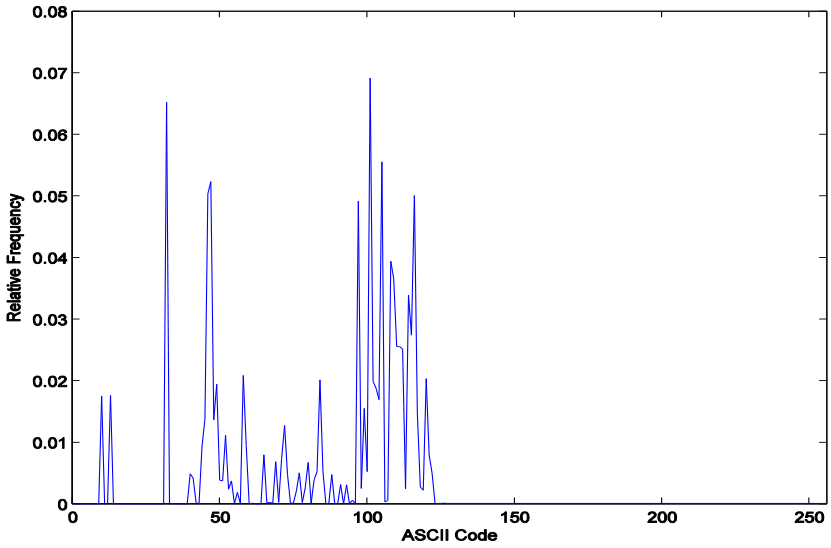
Experimental Results

We conduct experiments on training and test dataset. In the first part of our experiments, we present the model generation for normal HTTP traffic to host marx and hume. Then, we evaluate the accuracy of our GSAD model in detecting various attacks, namely Back attack, Phf attack, Crashiis attack, Generic attacks, Shell-code attacks, Polymorphic attacks and the Polymorphic Blending attacks coming through HTTP services.

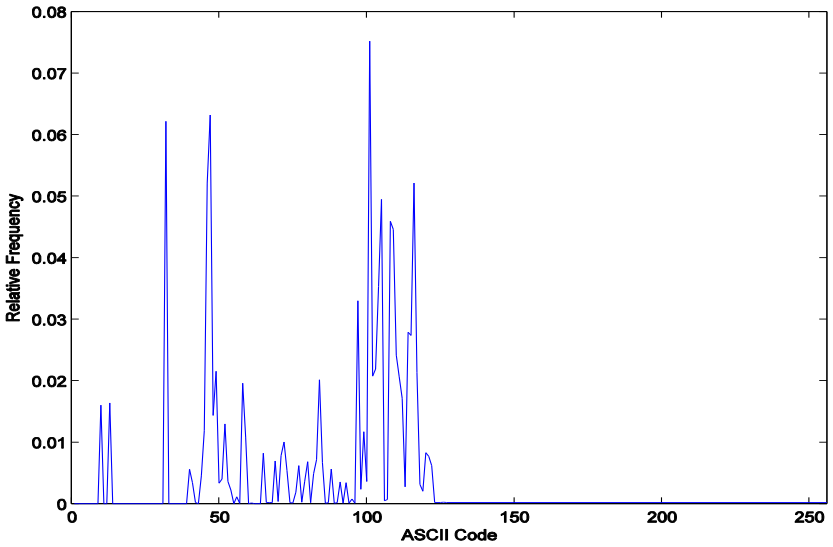
a) Model Training Results on DARPA 1999 Dataset

We extract inbound HTTP request traffic of week 1 and week 5 from DARPA 1999 dataset [90] for the training of the GSAD model. The extracted HTTP traffic packets correspond to normal HTTP requests destined to two different HTTP servers existing in DARPA 1999 data set: marx (Linux Server) and hume (NT Server). The total numbers of packets used for training of the model after filtering are 13,933 and 10,464 for hosts marx and hume respectively.

For a specific host, HTTP GET traffic has very similar behavior. In the experiments, we train the GSAD models on training dataset (10 days normal HTTP GET request traffic) for hosts marx and hume respectively, and generate an average normal profile for the HTTP GET request. For normal profile, we generate character's relative frequency model. Figures 3.9(a)-(b) show the relative frequencies of each character (0-255) in the extracted normal packet payload for the hosts, marx and hume respectively.



(a)

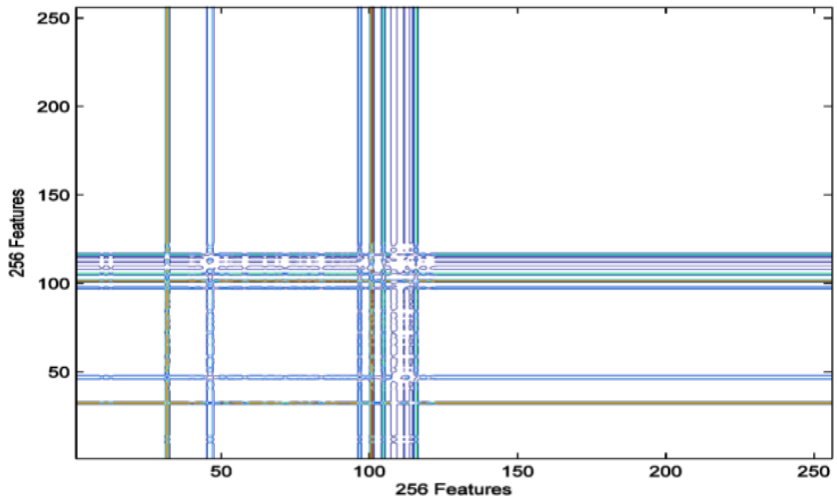


(b)

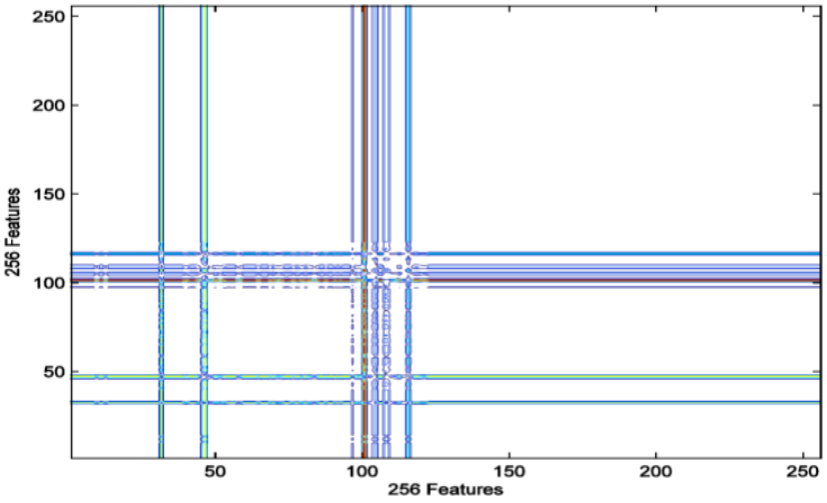
Figure 3.9: Average relative frequency of characters for normal HTTP GET request payloads. (a) marx; (b) hume

Then, an average geometrical structure model using Equations 3.11 to 3.12 is developed. Figures 3.10(a)-(b) show the MDM images of normal HTTP traffic behavior

for hosts marx and hume. MDM computes correlation between the packets and also between the features of the HTTP payloads.



(a)



(b)

Figure 3.10: Average MDM images of Normal HTTP GET request, (a) marx, (b) hume

In Figure 3.10, *X* axis and *Y* axis show the 256 possible features present in a packet payload. The cross point on the figure represents correlation between two features.

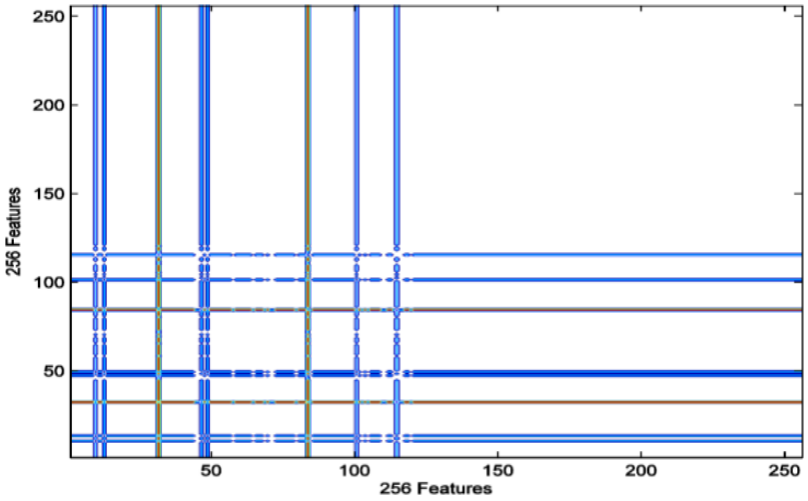
b) Model Testing Results on DARPA 1999 Test Dataset

In DARPA 1999 dataset, total numbers of packets after filtering for testing are 783,443 for marx, and 8431 for hume hosts respectively. In weeks 4 and 5 evaluation dataset, 23 attack instances are detected by our GSAD model.

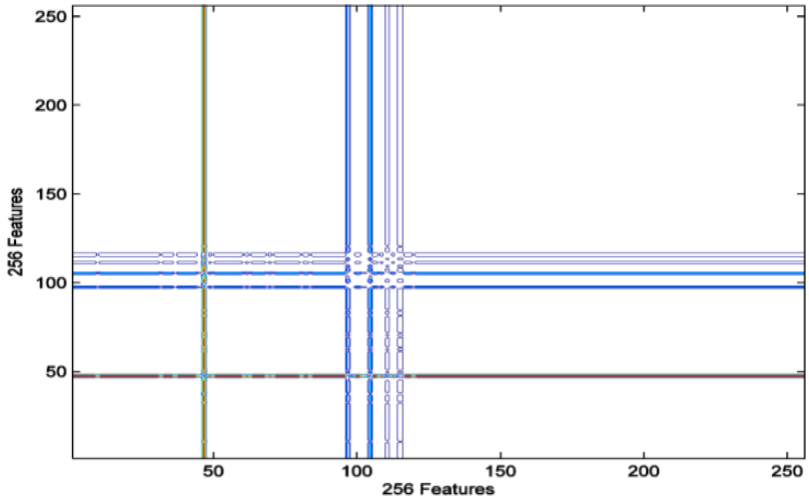
We conducted experiments with the extracted test data from the DARPA 1999 dataset and evaluated the accuracy of our GSAD model in detecting various attacks coming through HTTP services including Apache2 attacks, Back attacks, Crashiis attacks, NTInfoscan attacks and Phf attacks. MDM test results for Apache2 attacks and Phf attacks are shown in Figure 3.11.

The behavior of Apache2 attacks in Figure 3.11(a), the behavior of Phf attacks in Figure 3.11(b) and the normal profile in Figure 3.10 show clear differences in these profiles. Moreover, the correlations between the features of Apache2 attack and Phf attack packets are different from the correlations between the features of normal GET traffic on the hosts marx and hume.

From the MDM images, we can infer visually the differences between the suspected incoming packets and the normal network traffic involving high link speed and large amount of everyday network traffic. However it is not an appropriate solution for network intrusion detection because of human involvements. Furthermore, this also easily overloads the capacity of a network administrator. To overcome this problem, a weight factor score criterion is used.



(a)



(b)

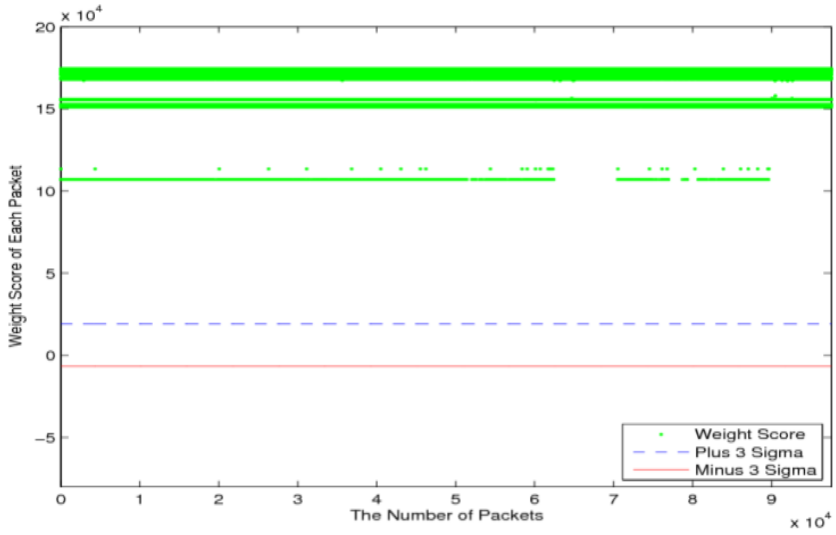
Figure 3.11: MDM images of attack packets. (a) apache2 attack; (b) phf attack

To differentiate abnormal or malicious behaviors from normal behaviors, weight factor score and threshold value are used. We use more than 90,000 Apache 2 attack packets and more than 2500 phf packets to evaluate our model. For our model, we choose a threshold value between -3δ and +3δ as this gives the best results. According to our experiments, the threshold values for hosts marx and hume are [-6.6759e03, 1.9187e04] and [-1.0440e04, 2.1655e04] respectively.

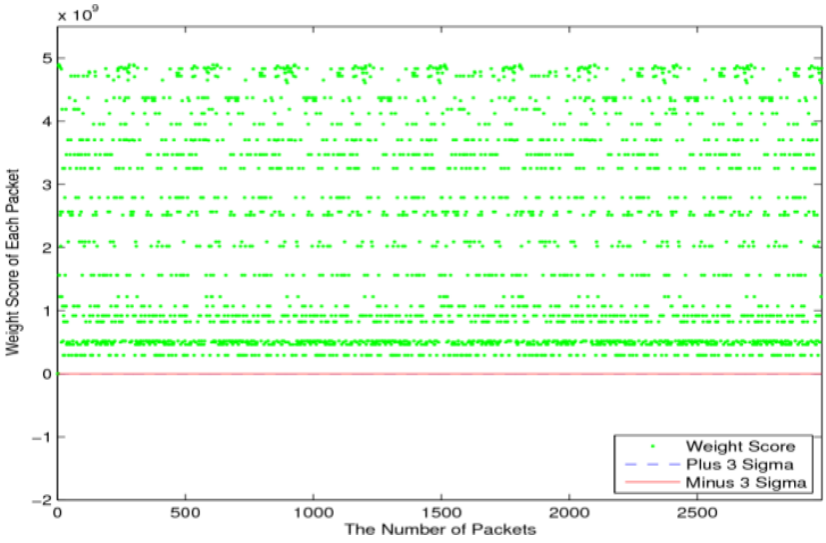
Figure 3.12 shows the weight factor score results for Apache2 attacks and Phf attacks. Here, *X*-axis denotes the number of packets and *Y*-axis represents the weight score values. In Figure 3.12, green color plot shows a higher value of weight factor score in comparison to normal threshold values. This provides a good visualization of the performance of our model.

The red and blue lines in Figure 3.12 are the thresholds of hosts marx and/or hume. They indicate that any packets assigned with a weight factor score beyond them are classified as intrusions.

In Figure 3.12(b), red line and blue line overlap with each other because of the scale of *Y* axis. In our definition of attack detection, an attack is detected as long as one of its attack packets is identified as abnormal. Based on the experiments, we find that all of the attack instances are successfully detected using the GSAD models.



(a) **apache2 attack**



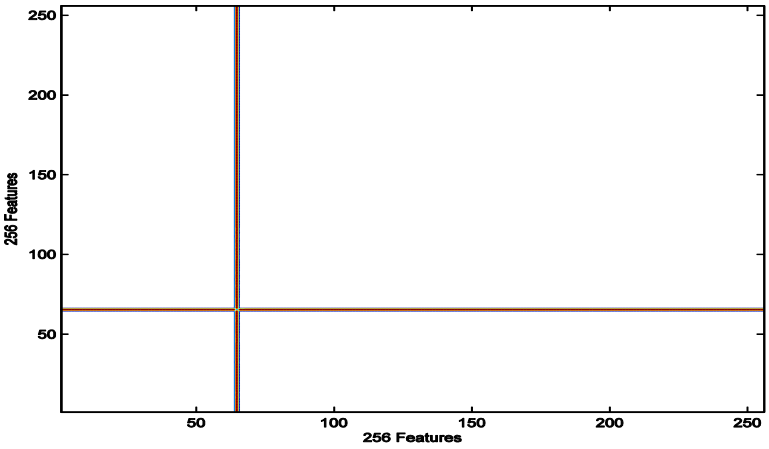
(b) **phf attack**

Figure 3.12: Weight factor scores of attack. (a) apache2; (b) phf

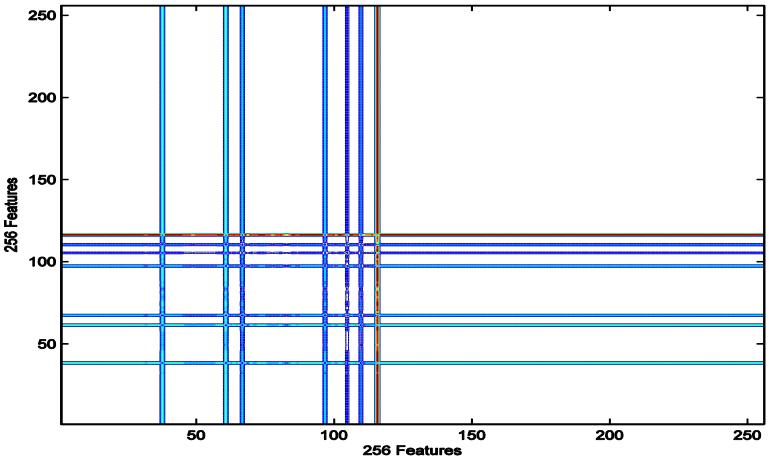
c) Evaluation of GSAD on GATECH Attack Dataset

In this section of experiments, we use a GATECH attack dataset. All HTTP request attack packets from the attack dataset are used in our experiments. We conduct several experiments for various types of attacks using GATECH attack dataset. We evaluate the similarity between the MDM of attack profile with the MDM of normal profile. The incoming request is considered as an attack or a threat if the weight factor is more than +3δ or less than -3δ. Results are encouraging. Results of some attacks are discussed in the following section.

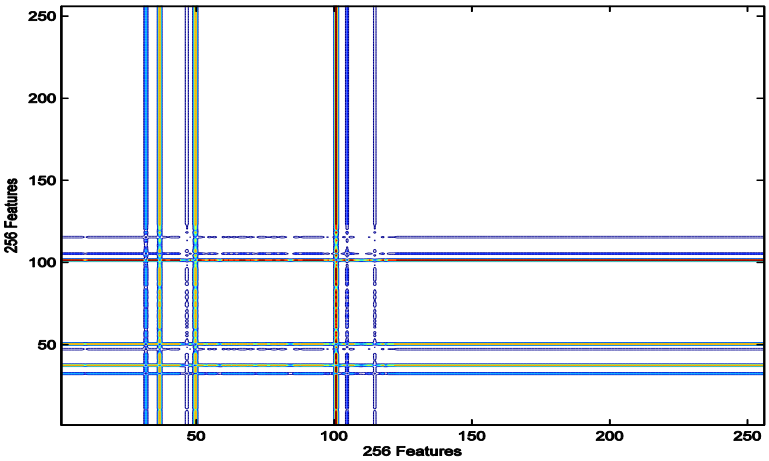
Generic attacks: This dataset consists of 66 HTTP attacks, plus shell-code attacks (these attacks carry executable codes in the payload that exploits vulnerability (MS03-022) in Window Media Service (WMS)). Other attacks cause Information Leakage and Denial of Service (DoS). Our model could detect around 90% of these attacks. Figure 3.13(a)-(d) shows MDM results for some of the Generic attacks.



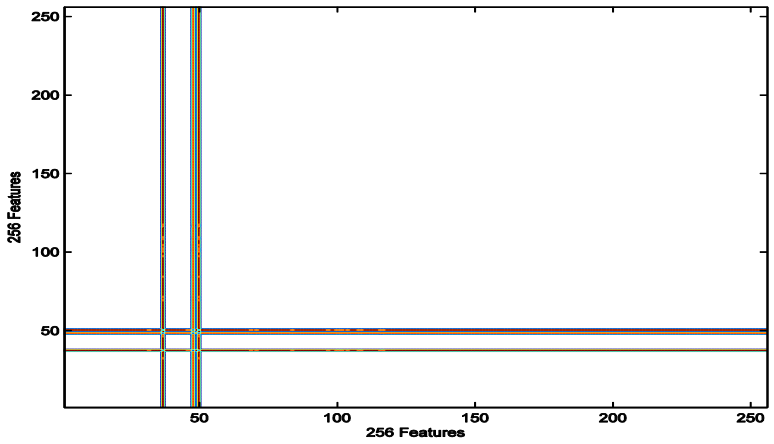
(a) **Remote GET buffer overflow vulnerability**



(b) **Information leakage attack - ADSI path disclosure**



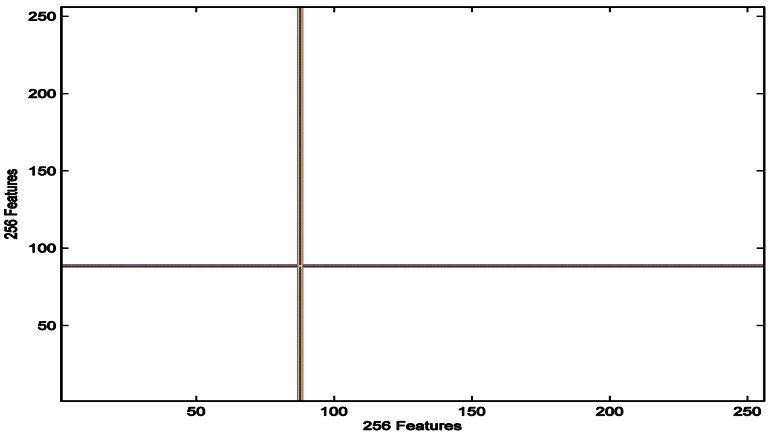
(c) **Input validation error attack**



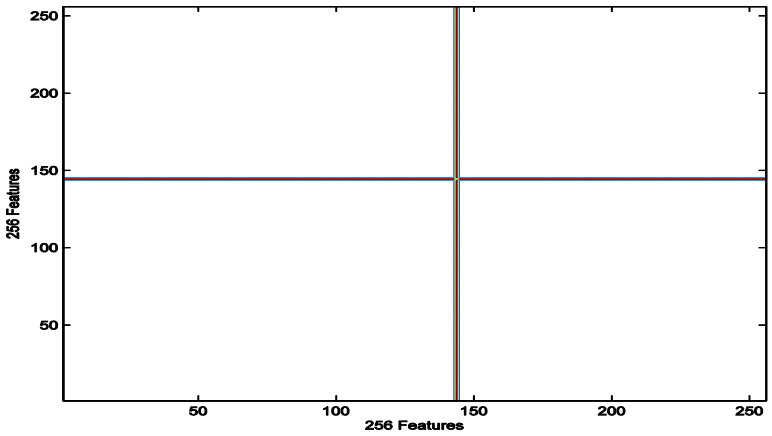
(d) **Input validation error attack - NT index server directory travel**

Figure 3.13: MDMs of generic attacks

Shell-code Attacks: Shell-code attacks are particularly very harmful as they inject executable code and hijack the normal execution of the target application. The data set contains 11 shell-code attacks from the Generic Attack data set, such as Code-Red Worm and Buffer Overflow attacks. Figure 3.14(a)-(b) show MDM results for Code-Red and Get Buffer Overflow Attacks respectively.



(a) **Code red worm**



(b) **Get buffer overflow**

Figure 3.14: MDMs of shell-code attacks

CLET Attacks: These attacks are generated from 8 shell-code attacks using polymorphic engine CLET. Polymorphic version of each attack using the payload statistics was computed on each distinct day of traffic from DARPA and GATECH datasets for training CLET polymorphic engine. Overall, 96 polymorphic attacks are present in the data set. Figure 3.15 shows the MDM result for CLET attacks.

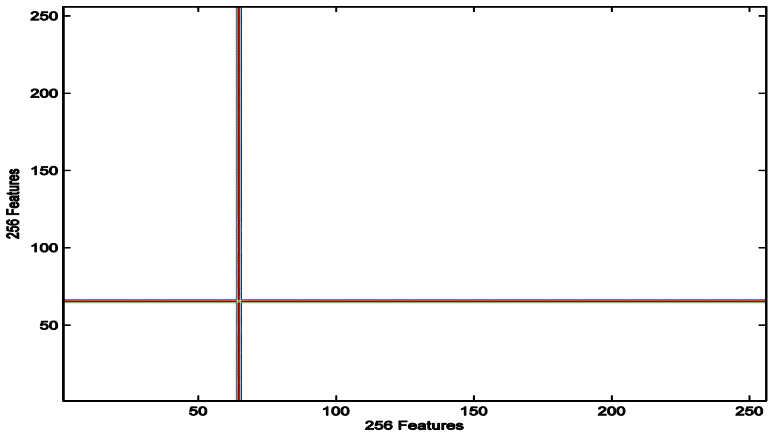


Figure 3.15: MDM of a polymorphic attack-padded attack

Comparing results for various attacks shown in Figures 3.13-3.15, the MDM attack profiles of generic, shell-code and polymorphic attacks with the MDM normal HTTP

request profiles show clear differences in there behaviors. Also the correlations between the features in these attacks are different from the correlations between the features of normal HTTP requests on the hosts marx and hume. The *X* axis and *Y* axis show the 256 possible features (ASCII characters) present in a packet payload. The cross points in the figure represents the correlation between two features.

3.6 Analysis of Results

The results reported in the previous section show that the Geometrical Structure Anomaly Detector (Mahalanobis Distance Map and Weight factor score) can detect new attacks without prior knowledge of the attacks with high accuracy and low false positive rate.

The results obtained for our model are very encouraging. We have achieved nearly 100% detection rates for DARPA 1999 data set with 0.087% false positive rates. For the comparison of GSAD model with PAYL, we use the data set used by Wang and Stolfo [80] for the evaluation of PAYL. The differences in the detection rates are not big but we have achieved very low false positive rates. Table 3.1 shows a comparison in terms of detection rate and false positive rate for PAYL and GSAD on DARPA 1999 dataset. The results for PAYL are taken from [80].

Table 3.1: Performance Comparison

Algorithm	Detection Rate	False Positive Rate
GSAD	100%	0.087%
PAYL	98%	0.1%

Table 3.2 shows comparison of detection rates and false positive rates for PAYL, McPAD and GSAD on GATECH attack dataset. Researchers of these models have also used DARPA 1999 dataset for the training and test of their models. These datasets have similar types of attacks, since the attacks are coming from the same sources (web sites). Table 3.3 gives a summary of experimental results obtained for different algorithms.

Table 3.2: Comparison of GSAD, McPAD and PAYL on GATECH attacks dataset

Algorithm	**Detection Rate**			**False Positive Rates**
	Generic attack	**Shell code attack**	**CLET attack**	
GSAD	90%	100%	100%	0.087%
McPAD	75%	96%	99%	1%
PAYL	90%	99%	95%	1%

Table 3.3: Summary of experimental results for Generic attacks on various datasets

Modeling Approaches	**Detection Rate**	**False Positives Rate**	**Dataset**
PAYL (1-gram)	90%	20%	DARPA 99
MarKov chain (conditional probabilities, $\varepsilon = 10e^{-10}$	95%	39.81%	Arach NIDS
MarKov chain (Syntactical segmentation), $\varepsilon = 10e^{-15}$	95%	4.98%	Arach NIDS
McPAD	75%	1%	GATECH
GSAD	90%	0.087%	GATECH

3.7 Conclusion

In this chapter, we have presented the framework of our geometrical structure anomaly detection (GSAD) scheme for building an effective intrusion detection system. We have presented an approach to network intrusion detection that combines two different techniques: 1-gram text categorization technique and the Mahalanobis Distance Map technique. This approach is based on geometrical structures of the payload features. The important characteristics of our model are that: it considers co-relation between the features and some structural information of the payload to build the behavior profile of the network traffic for intrusion detection. We have compared the performance of our model with the state-of-the-art PAYL model.

In addition to improving the attack detection accuracy and detecting a variety of attacks, we have further implemented GSAD model in the HTTP environment to detect web-based attacks coming through HTTP, at port 80.

Our experiments on the DARPA dataset and GATECH attack dataset show low false positive and high detection rate of our model. Experimental results show better performance when compared against some state-of-the-art payload-based intrusion detection systems, namely PAYL and McPAD for HTTP attacks.

GSAD model uses 256×256 features for profile generation and discriminating normal and anomalous packets and thus requires very heavy computational cost. In Chapter 4 and Chapter 5, we will propose solutions to reduce the computational complexity of the GSAD model. We believe that our model can be used in real-time applications.

CHAPTER 4

Feature Selection and Two Tier Payload Based Intrusion Detection using LDA

Introduction

In the previous chapter, we have described a GSAD (Geometrical Structure Anomaly Detection) detector that models the "normal" attack-free traffic as a geometrical structure and demonstrates its ability to detect network attacks effectively. The detector has been designed to be language-independent and has considered correlations among features and partial structural information of payload. Furthermore, GSAD is applicable to any network service. Various experiments have demonstrated that GSAD can achieve a high detection rate and low false positive rate for worms and exploits.

However, GSAD uses a large number of features to analyse the hidden patterns of packet payload and uses these features to discriminate normal and attack patterns present in the network traffic. It creates computational complexity. It requires large storage and a long time for training and testing. Furthermore, the intrusion detection system has to deal with a huge amount of network traffic, increasing computational complexity and

overhead. This large feature set and voluminous dataset limits the GSAD to off-line applications only. Hence, feature reduction becomes mandatory for efficient operation of the intrusion detection system.

Feature reduction techniques are essential to create an efficient intrusion detection system when taking into account the computational complexity and the classification performance. Selection of an appropriate method that can precisely determine relevance of features to intrusion detection tasks and redundancy between features is a major challenge. There are several methods used by researchers for the selection of header features and relevance analysis in intrusion detection research [98-103]. However, there are very few papers that have considered feature selection according to application-layer payload. GSAD model uses packet payload to detect normal and attack patterns. Although GSAD is proven to be promising in detecting network attacks, it uses 256×256 features to identify a suspicious attack pattern in the packet payload. This is time consuming.

In this thesis, we propose to use Linear Discriminant Analysis (LDA) and Principal Component Analysis (PCA) techniques, for payload feature selection and classification of normal and attack patterns. LDA [104] is a supervised technique which is used for selecting important features in a large set of features, whereas PCA [105, 106] is an unsupervised technique used for analysis of the data, which examines and weights the importance of various components in terms of variance that a component reserves. These two techniques that are used in intrusion detection domain for feature selection of a packet header have achieved good results. However, these techniques are not used for the payload feature selection. We propose to use these two techniques for payload

feature selection and classification of normal and attack patterns. In this chapter, we describe the implementation of LDA technique for the selection of payload features and discriminating normal and malicious patterns in network traffic. Here, we propose our solutions for feature reduction and identify group of features which can enable an efficient use of GSAD system for intrusion detection in the network. The detailed description of PCA technique for packet payload feature selection will be discussed in Chapter 5.

In this chapter, a new method is proposed, which uses LDA technique and difference distance map (DDM) [87] to order the potential features for payload feature selection and to distinguish normal and attack patterns in the network traffic. LDA-based approach reduces the computational complexity dramatically while retaining the high detection rates and provides an elegant solution. The rest of this chapter is organised as follows. Section 4.1 gives brief information on feature selection algorithms. In Section 4.2, we discuss basic concepts of LDA. In Section 4.3, we propose LDA-based intrusion detection system and a detailed explanation of LDA-based feature selection approach for intrusion detection. The experimental results and analysis are given in Section 4.4. In Section 4.5, we present a two-tier intrusion detection system and discuss experimental results and analysis in Section 4.6. We give a brief information on integrated signature generation in Section 4.7. Finally, Section 4.8 concludes our work.

4.1 Feature Selection Algorithms

In machine learning and statistics, feature selection also known as feature reduction, or variable subset selection is used to select a subset of relevant features for building robust

learning models. Intrusion detection system employs feature selection technique for data processing, which reduces the number of features, removing irrelevant, redundant, or noisy data from the dataset. There are two types of feature selection algorithms, namely, filter and wrapper. Filter method is fast and uses correlation based approach for the selection of feature subsets [82][107], but it is not capable to minimize generalization error. On the other hand, wrapper method uses classification algorithm and performs cross validation to identify important features. Every time a feature is added, induction algorithm is used for cross-validation. Due to this, wrapper method is computationally very expensive and does not scale well to large datasets that contain many features and instances [107][108]. To overcome the shortcomings of these two methods, a hybrid approach is used, that incorporates some of the features of wrapper method into a fast filter method for feature selection [107].

In recent years, linear methods [98-99] and [104-105] have played a major role in finding features, that describe classes, build classifiers and assign classes to new feature vectors. Most problems can be formulated as an eigenvalue decomposition equation, which can be solved by many existing algorithms [104-105]. Many linear feature reduction techniques, such as Correlation-based Feature Selection (CFS), Support Vector Machine (SVM), Principal Component Analysis (PCA), Generalized Discriminant Analysis (GDA) and Linear Discriminant Analysis (LDA), are proposed in [100], [101], [102], [103], [105] and [109] to reduce the header features of packets. However, there are very few papers that have considered feature reduction according to application-layer payload.

The early feature reduction approach [51] on payload, developed by Krugel et al., grouped the byte frequency distributions of 256 ASCII characters into six bins, namely 0, 1-3, 4-6, 7-11, 12-15 and 16-255. Wang et al. [82] proposed an Anagram detector, in which Bloom Filter (BF) was used to reduce memory overhead. Nwanze and Summerville proposed a lightweight payload inspection approach [110], where bit-pattern hash functions were employed to map the bytes at the packet payload onto a set of counters which were the selected features used for intrusion detection.

In this chapter, we propose to use the LDA technique for payload feature selection. It attempts to select the discriminating features from Difference Distance Map (DDM) between a normal Mahalanobis Distance Map (MDM) and the MDM of a particular type of attack using LDA.

4.2 Linear Discriminant Analysis

Discriminant Analysis is a statistical method for obtaining a reduced feature set. LDA is one of the commonly used dimensionality reduction and data classification techniques and has been applied in human detection [87], face recognition, speech recognition, bankruptcy prediction, network intrusion detection [111, 112] etc.

Different from PCA, which extracts features that are the most efficient for representation but may not be useful for discrimination, LDA selects an optimal projection matrix to transform a high dimensional feature domain to a lower dimensional feature space while preserving the significant information for data classification. We assume that there are n d-dimensional samples $\{x_1, \ldots, x_n\}$ assigned

to k different classes. Each class C_i, where $i = 1, \ldots, k$, has n_i samples. Projection matrix A_r is found to maximize the between-class scatter matrix S_B

$$S_B = \sum_{i=1}^{k} n_i(\mu_i - \mu)(\mu_i - \mu)^T, \tag{4.1}$$

and minimize the within-class scatter matrix S_w

$$S_w = \sum_{i=1}^{k} \sum_{x \in C_i} (x - \mu_i)(x - \mu_i)^T, \tag{4.2}$$

where μ is the sample mean vector of the whole sample set denoted by

$$\mu = \frac{1}{n} \sum_{j=1}^{n} x_j, \tag{4.3}$$

and μ_i is the sample mean vector for class C_i given by

$$\mu_i = \frac{1}{n_i} \sum_{x \in C_i} x. \tag{4.4}$$

The Fisher criterion is defined as the ratio of the variance between classes to the variance within the classes. Thus, the ratio, J, between the between-class scatter matrix S_B and the within-class scatter matrix S_W can be easily maximized by the projection matrix A_r.

$$J = \frac{{A_r}^T S_B A_r}{{A_r}^T S_w A_r}. \tag{4.5}$$

Once the above optimization problem is solved, the classification decision can be easily made on the low dimensional feature space by projecting the original feature space onto the optimal projection matrix A_r.

4.3 LDA-based Intrusion Detection System

In this section, we elaborate our new approach. We first discuss the framework of an LDA-based intrusion detection system. Then, detailed discussion of each block is given in the following subsection.

4.3.1 Framework of LDA-based Intrusion Detection System

Figure 4.1 presents the framework of proposed LDA-based intrusion detection system. It has four stages, namely Difference Distance Map (DDM) Generation, LDA-based feature selection, profile generation and classification of network traffic.

In GSAD, each payload record is characterized by a large set of features, and involves large computation power and time for classification process. LDA technique is used to improve the computational complexity of GSAD model and visualizes the classification problem as two-class categorization (normal and anomaly classes). LDA is used to select significant features from a Mahalanobis Distance Map (MDM), which is generated by the Geometrical Structure Model (GSM), a key component of the GSAD model, described in Chapter 3. Each MDM is used to explore the correlations among features (ASCII characters) in the packet payload for each single network packet. Then, selected significant features are used in the detection process, which is conducted on a new low-dimensional domain.

To extract the low-dimensional significant features, Difference Distance Maps (DDMs) must be generated to measure the difference between normal traffic and particular types of attack traffic, such as the difference between each pair of <Normal, Phf attack>, <Normal, Back attack> and <Normal, Apache2 attack>. Then, LDA is employed to select the most significant features for each normal and attack pair based on the pre-generated difference distance maps. Finally, all the selected features are integrated into a new significant feature set used for normal profile generation and malicious behavior detection.

It sends an alert signal to administrator if the new incoming packet is identified as an attack packet.

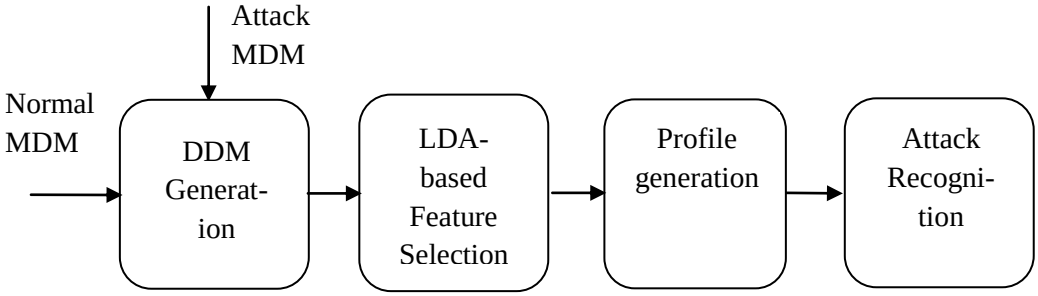


Figure 4.1: Framework of LDA-based intrusion detection system

4.3.2 Framework Modules

a) Difference Distance Map Generation Module

In order to discover the difference between the normal and attack samples, distance maps of all samples images are constructed as discussed in Chapter 3. Let us assume that there are *m* normal sample images and *n* attack sample images. Then, the average distance of element (*i*,*j*) is computed using Equations 4.6 and 4.7, where $\bar{d}_{(i,j)}^{normal}$ is the average of *m* normal samples at (*i*,*j*) and $\bar{d}_{(i,j)}^{attack}$ is the average of *n* attack samples at (*i*,*j*).

$$\bar{d}_{(i,j)}^{normal} = \frac{1}{m}\sum_{k=1}^{m} d_{(i,j)}^{normal,k}, \tag{4.6}$$

$$\bar{d}_{(i,j)}^{attack} = \frac{1}{n}\sum_{k=1}^{n} d_{(i,j)}^{attack,k}. \tag{4.7}$$

Covariance $\sigma_{normal(i,j)}^{2}$ of the (*i*,*j*)-th elements of normal samples and covariance $\sigma_{attack(i,j)}^{2}$ of the (*i j*)-th elements of attack samples are computed using Equations 4.8

and 4.9. The difference at each element (*i,j*), where *i, j* = 1, …, 256 between the MDMs of the normal samples and the attack samples is computed using Equation 4.10.

$$\sigma^2_{normal(i,j)} = \frac{1}{m}\sum_{k=1}^{m}\left(d^{normal,k}_{(i,j)} - \bar{d}^{normal}_{(i,j)}\right)^2, \tag{4.8}$$

$$\sigma^2_{attack(i,j)} = \frac{1}{n}\sum_{k=1}^{n}\left(d^{attack}_{(i,j)} - \bar{d}^{attack}_{(i,j)}\right)^2, \tag{4.9}$$

$$diff_{(i,j)} = \frac{(\bar{d}^{normal}_{(i,j)} - \bar{d}^{attack}_{(i,j)})^2}{\sigma^2_{normal(i,j)} + \sigma^2_{attack(i,j)}}. \tag{4.10}$$

In Equations 4.6-4.9, $d^{normal,k}_{(i,j)}$ represents the (*i, j*)-th element of MDM of the *k*-th normal sample, and $d^{attack,k}_{(i,j)}$ stands for the (*i, j*)-th element of MDM of the *k*-th attack sample. The difference between the normal samples and the attack samples is denoted by $diff_{(i,j)}$. The difference distance map between the normal samples and the attack samples is defined by $Diff = \left[\, diff_{(i,j)}\right]_{256\times256}$. A difference distance map is generated for each pair of normal traffic and a particular type of attack traffic. Because the dimension of the difference distance map is large (256×256), it is very time consuming if the difference map is directly used to differentiate the normal traffic and the attack traffic. Therefore, we use Linear Discriminant method [98] to reduce the number of difference distance map elements (i.e., to reduce the dimension of the difference map).

b) LDA-based Feature Selection

This module is of prime importance. It selects most significant elements of the difference distance map. In the following section, we discuss iterative feature selection process in detail.

A difference distance map generated in the first stage, for each pair of normal traffic and particular type of attack traffic, is used for the selection of significant features.

In the difference distance map, the larger a feature value (i.e., a matrix element results) is, the more important feature is used to discriminate attack traffic from normal traffic. Figure 4.2 shows a flow model for iterative feature selection process.

We first select the most significant r features from the difference distance map. The element locations of these features in the difference distance map determine the element locations in every MDM of a normal or an attack sample to form a corresponding r dimensional distance vector represented by $D_{r,k} = [d_{k(U_{r,1}, V_{r,1})}, d_{k(U_{r,2}, V_{r,2})}, \ldots, d_{k(U_{r,r}, V_{r,r})}]^T$, where $(U_{r,1}, V_{r,1}), (U_{r,2}, V_{r,2}), \ldots, (U_{r,r}, V_{r,r})$ indicate the element locations of the largest r features in the difference distance map, r is ranged from 1 to 256^2 and k indicates the k-th sample. Let $D_{r,k}^{normal}$ and $D_{r,k}^{attack}$ represent the $D_{r,k}$ of the k-th normal sample and the k-th attack sample respectively. Then, the projection vector A_r is computed by

$$A_r = (\sum \bar{D}_r^{normal} + \sum \bar{D}_r^{attack})^{-1}(\bar{D}_r^{normal} - \bar{D}_r^{attack}), \tag{4.11}$$

where $\bar{D}_r^{normal}$ and $\bar{D}_r^{attack}$ are the averages of $D_{r,k}^{normal}$ and $D_{r,k}^{attack}$ respectively, and $\sum \bar{D}_r^{normal}$ and $\sum \bar{D}_r^{attack}$ are the covariances of $D_{r,k}^{normal}$ and $D_{r,k}^{attack}$ respectively.

The number of the elements of the projection matrix A_r is reduced iteratively by determining the projection matrix A_r and removing the lower contribution elements. The positions of the elements are retained. The whole process is carried out iteratively until the number of significant features reaches the pre-set value. Then, the final projection matrix A_r is determined. Once the projection vector is finalized, the corresponding final

set of features is considered as the most significant features. The matrix A_r is employed to transform the multi-dimensional feature vector as a one dimensional score value for each sample.

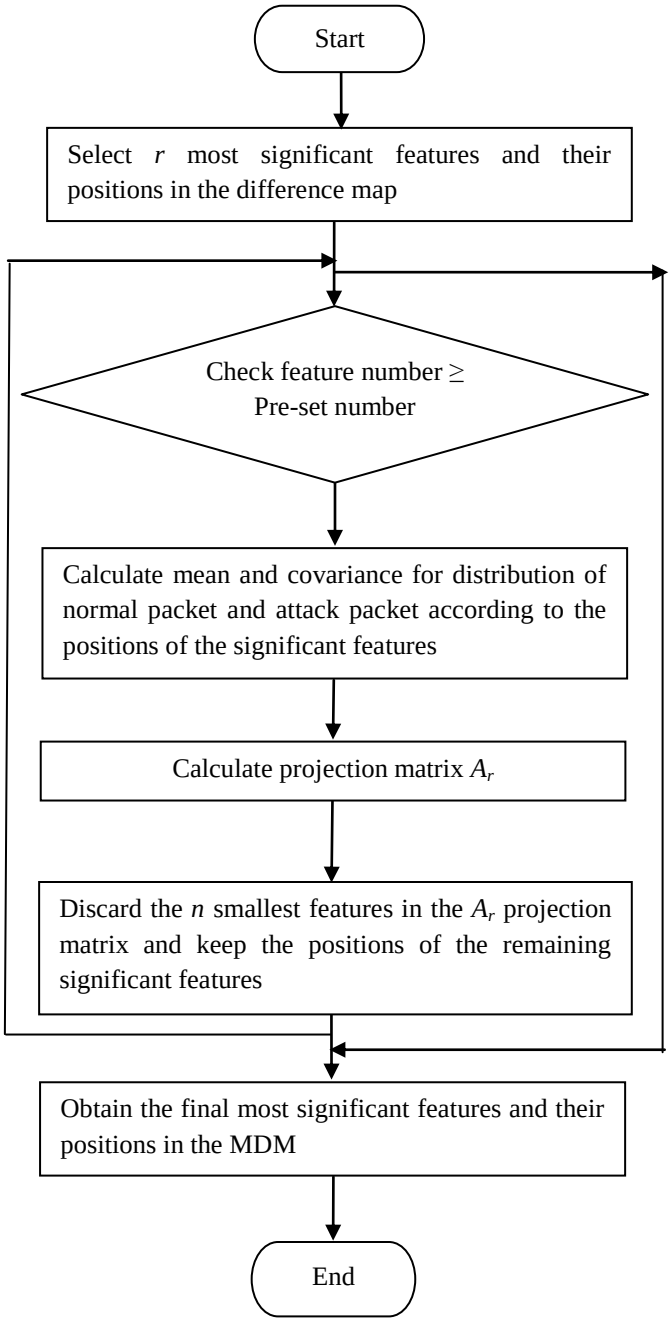


Figure 4.2: Flow model for feature selection process

c) Normal Profile Generation

The normal profile is utilized to detect the similarity between the normal behavior and new incoming packet. It is developed by using the normal training samples and the selected significant feature set. In this section, we explain how to perform the development of the normal profile.

Mean values of the significant *r* features of all normal training samples and a detection threshold are the basic components of the normal profile. Given a set of normal training samples $X = \{x_1, \ldots, x_m\}$, which have been applied in the feature selection phase, and the significant feature set $F_k = [f_k(U_1, V_1), f_k(U_2, V_2), \ldots\ldots\ldots, f_k(U_r, V_r)]^T$, in which (U_1, V_1), (U_2, V_2), …, (U_r, V_r) indicate the locations of the significant *r* features and *k* indicates the *k*-th sample. The mean values are denoted by

$$\overline{F} = \frac{1}{m}\sum_{k=1}^{m} F_k, \tag{4.12}$$

and they are stored in the normal profile used for comparing with any new incoming packet. Threshold is another important component to consider. Without an appropriate criterion, it is hard to achieve a satisfactory detection performance. The larger the threshold value is, the less false positive alarm is generated. On the other hand, a smaller threshold will in turn create a higher detection rate.

In this research, we select a threshold through a distribution analysis of the Euclidean distance between each normal training sample and the mean value $\overline{F}$. The Euclidean distance from the *k*-th normal training sample to the mean value $\overline{F}$ is computed by

$$ED_k = \sqrt{\sum_{i=1}^{r}(f_{k(U_i,V_i)} - \overline{f_{(U_i,V_i)}})^2}, \tag{4.13}$$

where $\overline{f_{(U_i,V_i)}}$ is the (U_i,V_i)-th element of $\overline{F}$. The standard deviation of the Euclidean distance from the *k*-th normal training sample to the mean value $\overline{F_r}$ of the normal training samples is

$$\delta = \sqrt{\frac{1}{m-1}\sum_{k=1}^{m}(ED_k - \overline{ED})^2}\,, \tag{4.14}$$

where $\overline{ED} = \frac{1}{m}\sum_{k=1}^{m} ED_k$. We assume that the distance ED_k is of normal distribution, so three standard deviations account for 99% of the sample population.

d) **Attack Recognition Process**

Similar to normal profile development process, for any new incoming packet, the GSM is applied to generate the MDM of the packet. Then, the most significant *r* features are collected from the MDM. Projection matrix A_r, and the *r* selected significant features are used to calculate the score value for each input network packet. The score value of an incoming packet is computed by

$$Score = A_r \times D_r^{input}, \tag{4.15}$$

where A_r is the projection matrix obtained in the training stage and D_r^{input} is the feature vector with *r* elements of the distance map of the input network packet. When the score is larger than a pre-calculated threshold, the input network packet is identified as an attack packet, otherwise it is identified as normal network packet.

To calculate the classification threshold, all training data consisting of normal and attack data are fed into the testing procedure respectively. Corresponding scores obtained based on Equation 4.15 are clustered into two classes on a *d* dimensional data domain. A

threshold is selected using the LDA optimizing criterion using Equation 4.5 which finds out the maximum ratio of between-class difference S_B and within-class difference S_W.

4.4 Experimental Results and Analysis

In this section, we first present experimental results. Then, an analysis of our experimental results is given. We evaluate the LDA feature selection approach on the DARPA 1999 IDS dataset [90], which contains tcpdump network traffic recorded for five weeks. Week 1 and week 3 data are attack-free data, and the other three-week data contain both normal and attack traffic, as explained in Chapter 3. This dataset consists of scan or probe, DoS, R2L, U2R and data. The dataset is divided into subsets. Each subset comprises of data records of specific attack type and normal traffic. LDA is used on the training dataset to obtain the important features for the classification process. The experiments are performed in two steps.

- Iterative feature selection is the first step of our experiments. We use heuristic search method to select the number of elements as a starting point. Then, LDA approach is used to determine optimal feature set.
- In the second phase of our experiments, the obtained optimal feature set is used to calculate profile of normal payload, which is then used to classify new incoming packet payload as normal or anomaly.

4.4.1 Experimental Results

To verify the performance of the proposed algorithm, HTTP traffic is used. LDA-based IDS is trained and tested with the inbound HTTP GET request traffic carrying payload

extracted on week 4 and week 5. We use the same dataset which is used in Chapter 3, Section 3.6. The total numbers of packets after filtering are 783,443 packets for marx (Linux server with IP address 172.16.114.50), and 8431 packets for hume (NT server with IP address 172.16.114.100) hosts respectively. Then, we further filter the normal and attack HTTP GET request packets, and divide them into normal and attack datasets respectively. We randomly choose 300 normal packets and 900 attack packets for feature selection, and choose 4000 packets for normal model training. Finally, we randomly select 1000 normal packets and 3000 attack packets for testing. The types of attacks are Apache2 attacks, Back attacks and Phf attacks.

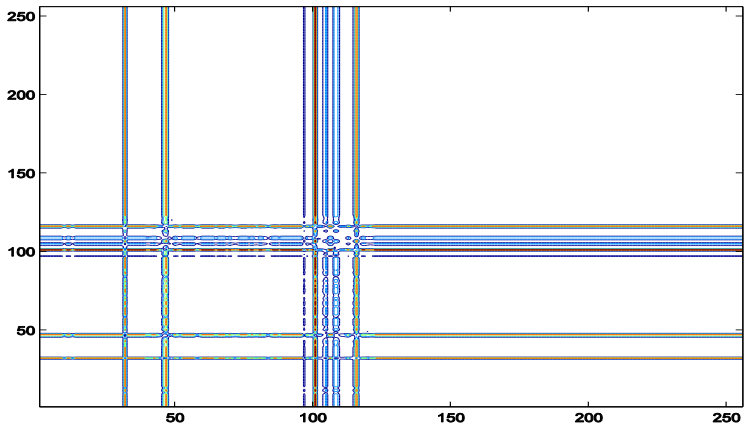
To select the most significant features, we first conduct experiments to choose a step size for iterative feature selection. We randomly choose 2600 normal HTTP request packets and 2600 Phf attack packets and calculate MDM using Equations 4.6 to 4.9. The MDM represents the correlations between features and is symmetric along the diagonal. We discard upper half of MDM matrix, reducing number of features. Furthermore, we neglect features that have small value in the difference distance map. We assume that these features have little influence on traffic classification. We manually select 2700 features as initial starting point and use LDA-based iterative feature selection technique, discussed in Section 4.3.2 to determine optimal feature set, detection rate, false positive rate and threshold values for two step sizes, namely 10 and 20 respectively. Results for step sizes 10 and 20 are given in Table 4.1.

Table 4.1: Performance of Phf attack for various selected features

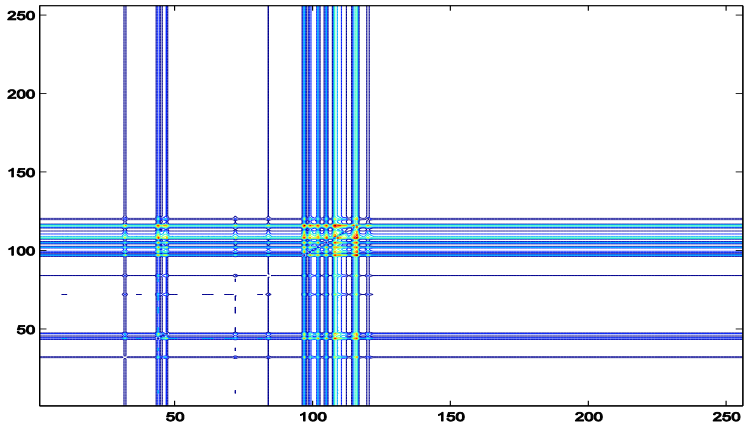
Selected Features	Threshold		Detection Rate (%)		False Positive Rate (%)	
	Step=10	*Step=20*	*Step=10*	*Step=20*	*Step=10*	*Step=20*
100	8.6059	8.462	96.69	99.76	0	0
200	8.6277	8.5289	96.23	99.57	0	0
300	8.631	8.542	96.23	99.46	0	0
400	8.634	8.261	96.23	99.19	0	0

We randomly select 300 normal packets and 900 attack packets for each type of attack to select significant features using the feature selection process, discussed in Section 4.3.2. The normal model of LDA-based IDS is trained on the 4000 normal training packets. It can be observed from the results in Table 4.1 that the detection rate is high, and is 99.76% for step size of value 20 when 100 features are selected.

Figures 4.3(a)-(b) show the average MDMs of normal HTTP request packet and Phf attack packet used in the experiments. The average difference distance map (DDM) between normal packets and Phf attack packets is represented in Figure 4.4. The difference distance map (DDM) shows a clear difference between normal packets and phf attack packets.

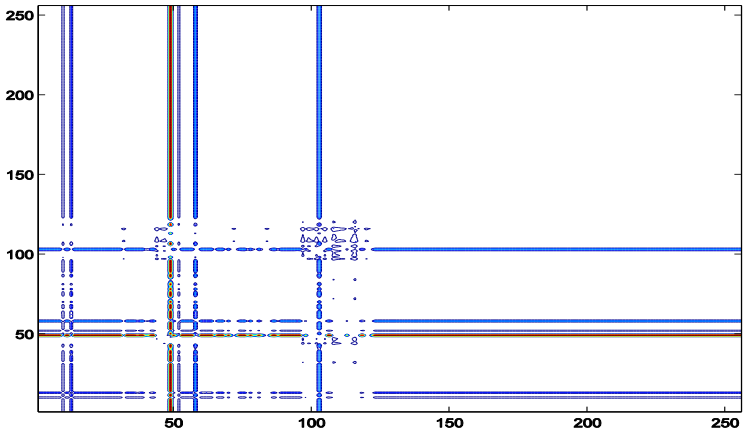


(a)



(b)

Figure 4.3: Average MDMs. (a) normal HTTP request; (b) phf attack packets



.

Figure 4.4: Difference distance map between Normal HTTP and Phf attack packets

To validate our results, we conduct several experiments to extract the optimal number of significant features to best separate normal packets from attack packets for Back attacks, Apache 2 attacks, Phf attacks and Crashiis attacks. We use step size of value 20 for our experiments. The optimal results are achieved with 100 features selected using iterative feature selection process for each of the three types of attacks except Crashiis attack. This is because Crashiis attack carries a small packet payload and has very few features. Thus, we can conclude that LDA-based feature selection technique successfully transforms the original 256×256 dimensional feature domain to a relatively very low dimensional feature space and preserves the most significant information for the final classification. We integrate all selected features into one feature vector, which is used for the normal profile generation.

In the test stage, we evaluate the performance of LDA-based IDS on the testing dataset containing both the normal packets and the attack packets. The test results are shown in Table 4.2, which shows the performance of our one-tier model. A detailed analysis of the results is given in the next subsection.

Table 4.2: Confusion Matrix for LDA-based IDS using integrated feature set

Predicted Actual	Normal	Attack
Normal	987	13
Attack	142	2858

4.4.2 Analysis of Results

We evaluate the performance of our one-tier LDA-based model. From Table 4.2, we further calculate detection rate and false positive rate. Our model could detect 95.3% of attacks correctly with low false positive rate of 1.3%. The results in Table 4.2 reveal that the 300 optimally selected significant features can well differentiate various attack packets from the normal packets except Crashiis attack. The poor detection rate for Crashiis attack packets is related with the size of attack payload. Crashiis attack carries a small packet payload and has very few features, and some relevant features are removed in the feature selection process.

To overcome the problem of small packet payload and to further improve the performance of LDA-based intrusion detection system, we propose a two-tier intrusion detection system. A detailed description of two-tier IDS is given in the following section.

4.5 Two-Tier Intrusion Detection System

The LDA-based feature selection approach, discussed in Section 4.3 is proven efficient in reducing the computational complexity while retaining the high detection rates. However, we have considered only three types of attacks, namely Apache2, Back and Phf in the experiments. We have excluded the Crashiis attack due to the small packet payload size, which bias the overall detection performance and increase the false positive and the negative alarm rates.

We propose a two-tier IDS, which uses payload length criterion to separate the small size payloads from the normal size payloads. In the two-tier model, tier one is a

statistical based detector responsible for the detection of the small size payload attacks, and tier two is LDM-based detector applied to identify the other attacks.

We first describe the framework of two-tier IDS. Then, we evaluate the IDS on DARPA 99 attack dataset to detect various payload size attacks. Finally, we discuss the experimental results with analysis.

4.5.1 Framework of Two-Tier System

The framework of two-tier intrusion detection system is given in Figure 4.5. The system consists of four key components, namely Filter, Statistical Signature Based Detector, LDM Based Detector and Alert Generator. The solid arrow indicates the incoming network traffic, and the dotted arrow stands for the analysis decisions made by the detectors.

Under the HTTP environment, we make use of the length of packet payload as the filtering criterion because the normal HTTP packet has a very low probability to carry a very short payload. Therefore, the Filter component pre-processes the non-zero incoming HTTP Get request packets. Then, pre-processed request packets are grouped together based on the length criterion. If the length of any payload is less than the length criterion, the packet will be forwarded to the Statistical Signature Based Detector on the first tier. Otherwise, the packet will be passed to the second tier detector, i.e., the LDM Based Detector.

The detector analyzes the received packet and makes the final decision. Then, the Alert Generator will decide to raise an alarm or not based on the detection result given by the detector.

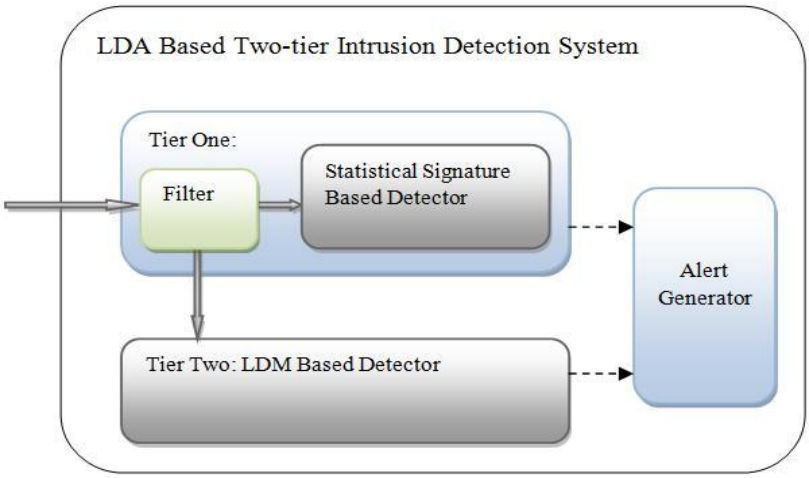


Figure 4.5: Framework of LDM based two-tier intrusion detection system

Tier-one: Statistical Signature Based Detector

As the first tier detector, Statistical Signature Based Detector only processes the small packet payloads. In this case, the observed HTTP Get request packets are highly suspicious, and the anomaly patterns carried by the attacks are easy to learn from the character relative frequencies. This is because these attacks have very high frequencies on some particular ASCII characters in the payloads, which is unusual and is not going to happen in the normal cases. Thus, we can develop the statistical signatures for these types of attacks.

To develop the attack signatures, the techniques discussed in Chapter 3 are used to parse and to extract the character relative frequencies from the labeled training attack packet payloads. The patterns of the character relative frequencies are stored as the signatures and are applied to identify the corresponding attacks in the future.

In the attack recognition phase, any new incoming packet is processed using the same techniques mentioned above to generate character relative frequency profile. The profile is compared with each known statistical signature, and the attack is identified as long as the profile is matched with one of the known statistical signatures.

Tier-two: LDA-based Detector

If the length of HTTP Get request packet payload is larger than the pre-set length criterion, the packet will be forwarded to the LDA-based Detector. The proposed LDA-based feature selection approach is used to extract a low-dimensional feature space for profile development and attack detection. The processes of normal profile development and attack recognition are discussed in detail in the following subsections.

a) LDA-based Feature Selection

The methodology used in two tier IDS to extract significant features is similar to the iterative feature extraction process, explained in Section 4.3.2. To extract significant features, difference distance maps need to be generated to measure the difference between normal traffic and particular types of attack traffic, such as the difference between each pair of <Normal, Phf attack>, <Normal, Back attack> and <Normal, Apache2 attack>. The difference distance map between the normal samples and the attack samples is defined by $Diff = \left[diff_{(i,j)} \right]_{256\times256}$ and is calculated using Equations 46 to 4.10 described in Section 4.3.2.

Then, LDA is employed to select the most signification features for each normal and attack pair based on the pre-generated difference distance maps. For the selection of the most significant features, we randomly choose normal training samples and various

attack training samples from the labeled samples set. A generated difference distance map is used for the significant feature selection. We first select the most significant *r* features from the difference distance map. Then, the optimal value of projection vector A_r is computed. Once the projection vector is finalized, the corresponding final set of features is considered as the most significant features.

b) Normal Profile Development

To measure the similarity between any new incoming packet and normal packets, the characteristics of the normal packets need to be extracted to develop a normal profile, which has been discussed in Section 4.3.2. In this section, we briefly explain the generation of the normal profile.

Mean values of the significant *r* features of all normal training samples and a detection threshold are the basic components of the normal profile. The mean values of the significant *r* features of all normal training samples are calculated by Equation 4.12. Each feature is represented by an index and location pair, such as (U_1, V_1), (U_2, V_2), …, (U_r, V_r), to indicate the locations of the significant *r* features.

To achieve a satisfactory detection performance, a threshold is selected through a distribution analysis of the Euclidean distance between each normal training sample and the mean value of the significant features. The Euclidean distance from the k^{th} normal training sample to the mean value $\overline{F}$ is obtained by Equation 4.13.

The standard deviation of the Euclidean distances from the k^{th} normal training sample to the mean value $\overline{F}$ of the normal training samples is calculated using Equation 4.14.

We assume that the distance ED_k is of normal distribution, so three standard deviations account for 99% of the sample population.

c) Attack Recognition

In the attack recognition process, the values of the most significant *r* features are generated and used to form a feature vector *F*. An incoming packet is considered as an attack or a threat if and only if the Euclidean distance from *F* to $\overline{\boldsymbol{F}}$ is greater than +3δ or smaller than -3δ, where δ is the standard deviation computed by Equation 4.14.

4.6 Experimental Results and Analysis

To evaluate the effectiveness of the proposed two-tier system, a series of experiments are conducted on the DARPA 1999 IDS dataset and compared with the outcomes of LDA-based IDS. In the following subsections, we present the experimental results and the analysis.

4.6.1 Experimental Results

DARPA 1999 IDS dataset is used for evaluation of our proposed system. We focus on the detection performance of the proposed IDS on HTTP traffic.

In the experiments, we use the same conditions as discussed in Chapter 3 to filter the interested HTTP Get request traffic from the week 4 and week 5 data of the DARPA 1999 dataset, and the extracted packets are grouped into normal and attack sample sets respectively. We randomly choose a certain number of extracted normal packets and attack packets from the sample sets for the training of the model, and the rest of sets are used for testing. The attack packets contain Crashiis attack, Phf attack Apache2 attack

and Back attack. The proposed two-tier system is trained and tested with the selected inbound HTTP Get request traffic carrying non-zero payload as discussed in Section 4.3.

All four types of attacks are used for the LDA-based IDS to obtain the significant feature set. The proposed two-tier system, however, uses Phf attack, Apache2 attack and Back attack only, and we exclude the Crashiis attack. This is because Crashiis attack is the only attack carrying a small packet payload with respect to the length criterion using in our experiments.

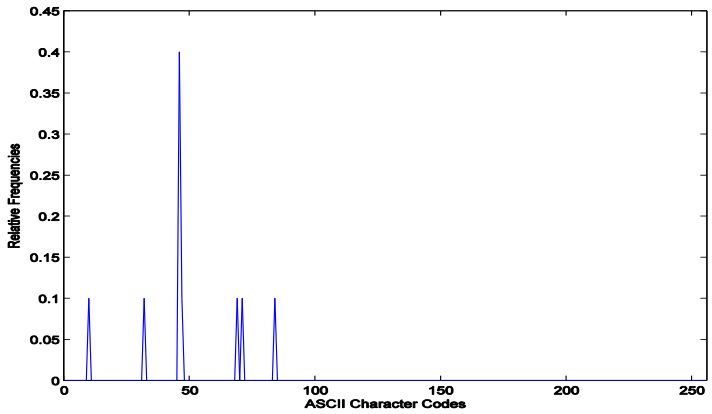


Figure 4.6: Character relative frequencies of crashiis attack

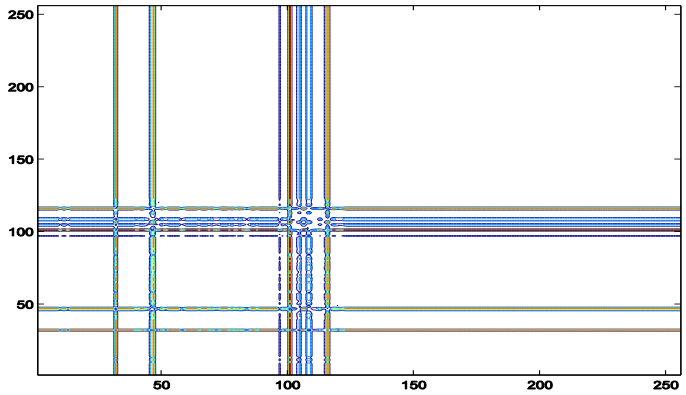
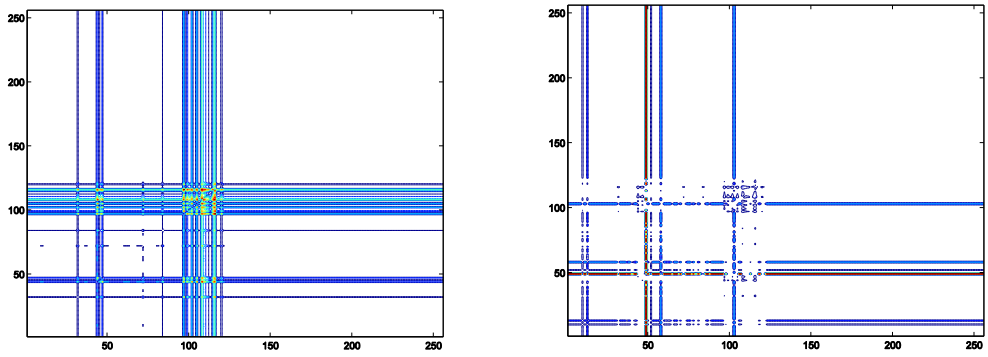


Figure 4.7: Average Mahalanobis distance map of normal HTTP Get request packets

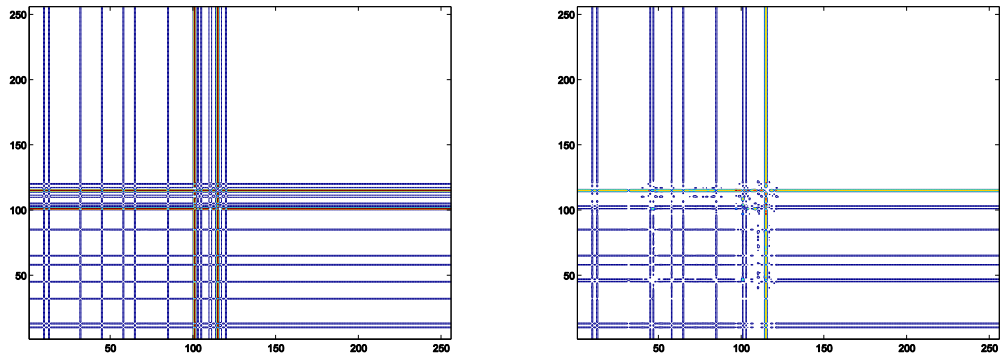
Thus, in the proposed two-tier system, the pattern of the character relative frequencies of Crashiis is used as the statistical signature for the tier-one detector. Figure 4.6 shows the character relative frequencies of Crashiis attack.

To obtain the optimal feature set for Phf and Apache2 attack, we use Figure 4.7, and Figures 4.8(a) and 4.9(a) to generate the difference distance maps as shown in Figures 4.8(b) and 4.9(b) respectively. The same method is used to obtain the optimal feature sets for the other types of attacks.



(a) (b)

Figure 4.8: Average Mahalanobis distance map. (a) phf attack packets; (b) difference distance map between normal HTTP and phf attack packets



(a) (b)

Figure 4.9: Average Mahalanobis distance map. (a) apache2 attack packets, (b) difference distance map between normal HTTP and apache2 attack packets

Experiments are conducted to extract the optimal number of significant features to best separate normal packets from attack packets. The optimal result is found to be 100 features selected by LDA for each of four types of attacks. Then, the normal profiles of the LDA-based IDS and the proposed two-tier system are developed based on the integrated 381 and 300 significant features respectively.

In the test stage, the LDA-based IDS and the proposed two-tier system are evaluated on the testing sample sets containing both the normal packets and the attack packets. All the test samples are used for the testing of LDA-based IDS. However, in the proposed two-tier system, the test samples are assigned to the detectors on different tiers according to the length. In tier-one, the detector uses the character relative frequencies of any assigned new incoming packet payload to compare with the pre-generated signatures in order to identify the suspicious intrusive activity. In tier-two, the detector evaluates the similarity between any new incoming packet and the normal profile using Euclidean distance and the decision is made by comparing the distance with the pre-set threshold (i.e. $\pm 3\delta$).

The experimental results of the LDM-based IDS and the proposed two-tier system are shown in Tables 4.3 and 4.4 respectively. Table 4.3 presents the performance of LDA-based IDS using features extracted from four types of attacks. Table 4.3 gives a comparison between the results obtained for the normal profiles developed using different numbers of training samples, i.e., 300, 700 and 4000 samples.

Table 4.3: Performance of LDA-based IDS for four types of attacks

Test samples	300 training samples		700 training samples		4000 training samples	
	Classify correctly	Mis-classify	Classify correctly	Mis-classify	Classify correctly	Mis-classify
Normal	96.83%	3.17%	97.1%	2.9%	99.07%	0.93%
Apache2 attack	100%	0%	86.94%	13.06%	0%	100%
Back attack	100%	0%	100%	0%	100%	0%
Phf attack	100%	0%	100%	0%	100%	0%
Crashiis attack	**6.67%**	**93.33%**	**5.64%**	**94.36%**	**4.1%**	**95.9%**

As can be seen from the table, the percentage of correct classification of normal samples is improved as the number of training samples increases. Back attack and Phf attack remain constant in all cases and have 100% correct classification rates. In contrast, the trend of correct classification of Apache2 attack and Crashiis attack is reverse. In the case of 4000 training samples, the classification of Apache2 attack drops down to 0%. This behaviour shows that this set of training samples has some discrepancy in dataset and integrated iterative feature set has removed some important features.

The results in Table 4.3 show the LDA-based IDS is unable to classify Crashiis attack correctly, and has misclassification rates higher than 93% consistently in all three training scenarios.

In Table 4.4, the performance of two-tier system using features extracted from three types of attacks is given. It compares the results obtained for the normal profiles

developed using the same numbers of training samples as Table 4.3. The difference is that the normal profiles for tier-two detector are built up on three types of attacks (Apache2 attack, Back attack and Phf attack) instead of all the four types.

Table 4.4: Performance of two-tier system using features from three types of attacks

Test samples		300 training samples		700 training samples		4000 training samples	
		Classify correctly	Mis-classify	Classify correctly	Mis-classify	Classify correctly	Mis-classify
Tier-two (LDM-based detector)	Normal	96.62%	3.38%	96.81%	3.19%	98.5%	1.5%
	Apache2 Attack	100%	0%	100%	0%	86.94%	13.06%
	Back Attack	100%	0%	100%	0%	100%	0%
	Phf Attack	100%	0%	100%	0%	100%	0%
Tier-one (Statist-ical signature detector)	Crashiis Attack	100%	0%	100%	0%	100%	0%

As can be seen from Table 4.4, the proposed two-tier system achieves encouraging performance in all the cases except the detection of Apache2 attack using the normal profile developed by 4000 training samples. DARPA 1999 dataset is used for the generation of normal profile and has variability in the dataset, which over generalizes trained model. Also number of Appache2 attack in DARPA 1999 dataset is very large, this could be one of the reasons why performance for Apache2 attack is poorer in comparison to other types of attacks. However, compared with the LDA-based IDS, the proposed two-tier system is proven more promising. It outperforms the LDA-based IDS

in detecting Crashiis attack. Benefiting from two-tier architecture, we are able to classify all the Crashiis attack samples. The detailed analysis is given in the next subsection.

4.6.2 Analysis of Results

The results in Tables 4.3 and 4.4 reveal that the 300 training samples can provide sufficient knowledge for both the LDA-based IDS and the proposed two-tier system to achieve good overall detection performance. In this section, the information contained in these two tables is further analyzed using Detection Rate (DR) and False Positive Rate (FPR).

Table 4.5 shows the comparison of the number of features, the detection rates and the false positive rates for LDA-based IDS, two-tier IDS and GSAD model.

Table 4.5: Comparison of IDSs

Systems	Number of features	Detection rate (%)	False positive rate (%)
Two-tier system	300	100	3.38
LDA-based IDS	381	99.8	3.17
GSAD model	65536	100	0.087

The results show that the proposed two-tier system has 100% detection rate and 3.38% false positive rate, which is slightly higher than LDA-based IDS. Two-tier system performs less number of calculations in comparison to LDA-based system. It can successfully classify Crashiis attack, and it uses less number of features in comparison to LDA-based IDS for the attack classification.

Compared with the GSAD model, the two-tier system achieves 100% detection rate. Although it has a higher false positive rate, the system successfully transforms the original 65536 dimensional feature space in GSAD model to a relatively very low dimensional feature space. It integrates various attack signatures while preserving the most significant information for the final detection. It not only significantly reduces the computational complexity of the detection process (attack signature comparison operation) but also reduces computational time.

In the following, we give two Receiver Operating Characteristic (ROC) curves for the LDA-based IDS and the proposed two-tier system in Figures 4.10 and 4.11, which show the relationships between detection rates and false positive rates to the corresponding systems.

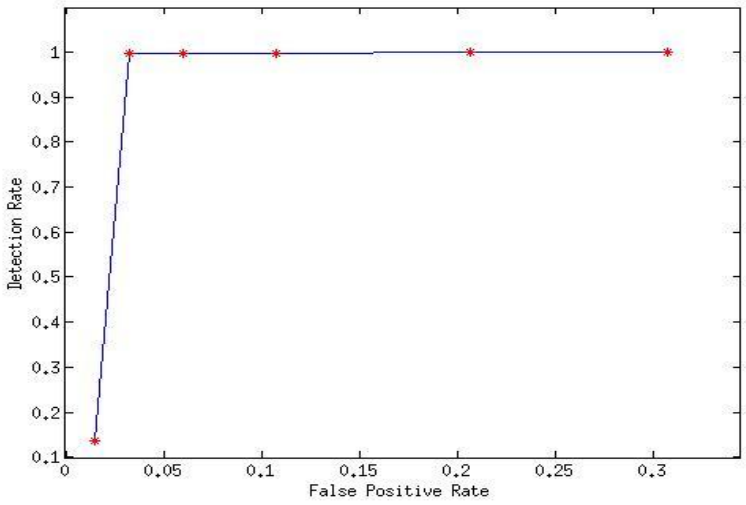


Figure 4.10: ROC curve of LDA-based IDS

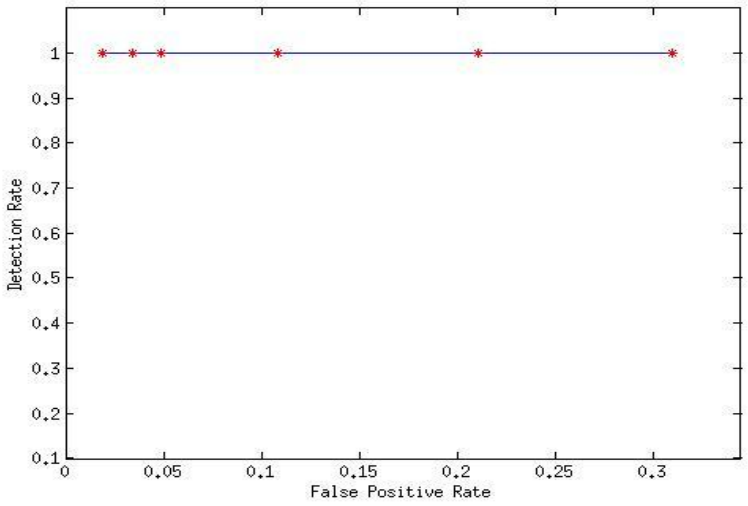


Figure 4.11: ROC curve of a two-tier IDS

As shown in Figure 4.10, the detection rate of LDA-based IDS increases significantly from 13.7% to 99.82% when the false positive rate is set to be around 3.38%. Then, the detection rate keeps going up slowly to 99.8%. Contrastively, the ROC curve of the two-tier system in Figure 4.11 is more stable, and it always stays at 100%.

Despite the ROC curve of LDA-based IDS finally reaches to nearly 100% detection rate, the detection performance of the LDA-based IDS in fact is significantly influenced by the number of small payload (i.e. Crashiis attack) appearing in our test sample set. The test sample set used in this paper is heavily dominated by the Apache2 attack (97576 test samples), and the small payload attack (i.e. Crashiis attack) only contributes a very small portion (195 test samples) to the test sample set.

Therefore, even around 93.33% of the Crashiis attack packets are classified incorrectly by the LDA-based IDS shown in Table 4.3, its overall detection rate did not drop dramatically. Hence, the ratio of the attacks in a test sample set bias the detection performance of LDA-based IDS. However, our two-tier system does not have this issue.

4.7 Common Profile (Signature) for Integrated Feature Set

Signature generation commonly refers to automatic generation of signatures, once an attack has been detected by an anomaly-based system. Chung and Mok [113] demonstrated that it was possible to generate signatures that match normal traffic. We use the same idea and develop one common signature for the normal traffic that can classify three different types of attack, namely, Phf, Apache2 and Back attacks. We have combined optimal features extracted for three attacks, namely Phf, Apache2 and Back attacks. Then, we have removed all common features from the combined feature set. This profile is used for the traffic classification. We have evaluated new integrated signature in Section 4.4 and Section 4.6 on DARPA 1999 dataset and calculated detection rate and false positive rate.

4.8 Conclusion

In this chapter, we have proposed a novel LDA-based feature selection approach to reduce the computational time and number of discriminating features of payload based anomaly IDS. The approach not only extracts a set of low-dimensional features but also preserves most signification information for data classification.

We have proposed a two-tier IDS to detect various attacks. The system processes the incoming packets based on the payload length of the packet. Tier-one uses the statistical signature approach for the classification of small payload attack packets, and tier-two uses LDA-based approach for the classification of the other attack packets.

The proposed two-tier model has been evaluated using DARPA 1999 IDS dataset for the HTTP traffic. It has achieved encouraging results with 100% detection rate and

3.38% false positive rate, and it can classify the Crashiis attack successfully, which is not able to be identified by the LDA-based IDS. Compared to GSAD model, it transforms a high dimensional feature space to a very low dimensional feature space, and have efficiently reduced the training and detection time.

Note that the amount of selected significant features may grow to a large number when more types of attacks are considered. This is because more sets of significant features will be selected with respect to the increasing number of types of attacks. However, this approach is able to generate one common signature for three different attack types, and reduces number of signatures required to classify patterns. The optimal feature set can be used to generate the single signature for a group of attacks. This will reduce the computation time for signature comparison for those selected attacks.

CHAPTER 5

RePIDS: a Multi Tier Real Time Payload Based Intrusion Detection System

The increase in network bandwidth has facilitated a large number of services to be provided over a network. In order to operate in high speed networks, intrusion detection systems are either signature-based or anomaly-based. Signature-based systems perform pattern matching and are limited to detect attacks with known signatures. On the other hand, anomaly intrusion detection systems can detect new attacks. Unfortunately, they are prone to false positives. In addition, network intrusion detection systems operate at the periphery of the networks, and are overloaded with large amount of network traffic. Thus, network intrusion detection systems have problems with handling heavy traffic and they lack the ability to process data in real-time as well.

In addition, 75% of cyber attacks occur at the application layer and 69% of vulnerabilities are caused by web services [78, 114]. This shows that, attackers are trying to exploit vulnerabilities at the application level, where sensitive data is stored. Header-based systems are not suitable to detect attacks intended for application level. On the other hand, payload-based systems can identify attacks trying to exploit

vulnerabilities at the application level. Thus, organisations rely heavily on payload-based intrusion detection for the protection of their networks. This poses significant challenges to build efficient network intrusion detection systems to detect a wide variety of attacks in real-time with acceptable reliability.

5.1 Introduction

The literature review on Network-based Intrusion Detection Systems (NIDSs), presented in Chapter 2, indicates that most previous research works in anomaly detection do not mention about data preprocessing techniques and traffic feature selection criteria used in NIDS. Intrusion detection algorithms are used directly on the raw data of the network. For practical applications, data preprocessing is one of the most important stages in the development of detection algorithm, and it directly impacts the accuracy and capability of the classification algorithm.

As presented in Chapter 4, current IDSs examine a large number of data features to detect intrusions or misused patterns. Some of the features may be redundant or have little contribution to the detection process. In Chapter 4, we also proposed to use the Linear Discriminant Analysis (LDA) technique to identify important features in building a payload-based anomaly intrusion detection system. We have demonstrated through experiments that the LDA-based system is able to reduce a large feature set to a small feature set [115], and discriminate normal and malicious network traffic. The LDA-based intrusion detection system is computationally efficient and effective. However, LDA is a supervised technique for constructing network IDS and needs labeled dataset as normal traffic and malicious (attack) traffic. Moreover, it is very hard and expensive

to create and analyse a labeled dataset of traffic from a real network. Furthermore, payload attacks are computationally expensive to detect because they require deeper searches into network sessions and also look for large number of payload features to discriminate normal packets and anomalous packets in the network traffic. Such challenge motivates us to use unsupervised technique to construct important and suitable features, which characterize behavioral patterns of network traffic and build a real-time payload-based intrusion detection system using suitable features. The Principal Component Analysis (PCA) [105] approach is used to construct important and suitable features from network traffic data which can distinguish normal and abnormal activities on a network. PCA helps reduce dimensionality by providing a linear mapping of *n*-dimensional feature space to a reduced *m*-dimensional feature space (according to dominant principal components) to boost the detection performance, so it is suitable for real-time applications.

Although PCA has been applied on header-based intrusion detection [116]118] to achieve sensible feature reduction, no work has been done on the data pre-processing using PCA for payload feature selection. Nwanze et al. [119] discussed modelling of packet payload using data mining technique based on PCA. However, they ignored the main idea of PCA and did not use the projection of original data on a new lower dimensional feature space. Furthermore, they did not consider correlations between features.

In this chapter, we propose a 3-tier Iterative Feature Selection Engine (IFSEng) for feature subset selection, which addresses the issues related to the quality of feature set. We also propose a Real-time Payload-based Network Intrusion Detection System

(RePIDS), which aims to detect payload-based attacks on a network in real-time. 3-tier IFSEng and Mahalanobis Distance Map (MDM) are the key components of RePIDS, which facilitate effective and efficient detection of attack packets in the network traffic. We have demonstrated through experiments in Chapter 3 that the MDM approach is promising in extracting the hidden correlations between features and the correlations among network packet payloads. MDM also partially captures structural information of payload which helps improve the detection performance and reduces false positive rate.

We evaluate our model on two datasets, namely DARPA 1999 [96] and Georgia Institute of Technology (GATECH) [97] datasets, and compare the detection performance (*F*-Value) and computational complexity of our proposed real-time payload-based IDS with two state-of-the-art payload-based IDSs, namely, PAYL [80] and McPAD [83]. Furthermore, we compute processing speed of our proposed model and compare it with the processing speed of a real scenario of medium size enterprise network.

This chapter is organised as follows. Section 5.2 provides an overview of two payload-based state-of-the-art systems. In Section 5.3, we present detail description of the framework of RePIDS and its mathematical model. Experimental results and their analysis are given in Section 5.4. Section 5.5 demonstrates the evaluation results of RePIDS in terms of computational complexity, and compares RePIDS with the state-of-the-art PAYL and McPAD intrusion detection systems. We also compare processing speed of RePIDS with the processing speed of a real scenario of medium size enterprise network with a gateway speed of 1GB. We conclude this chapter in Section 5.6.

5.2 State-of-Art Systems

In the following section, we review the main characteristics of PAYL and McPAD, which are the most relevant to our work in payload-based anomaly detection.

a) PAYL

PAYL is a payload-based anomaly detector proposed by Wang and Stolfo [74], which employs 1-gram analysis, an *n*-gram ($n \geq 1$) text-classification technique for clustering packets based on payload data length. The concept of *n*-gram text categorization can be found in Subsection 5.3.2. In PAYL, intrusions are detected by analysing the distribution of bytes inside the HTTP payload. The pre-processing of packet payload using 1-byte sliding window creates a feature vector containing the relative frequency count of each of the 256 possible 1-gram (byte) in the payload. Simplified Mahalanobis distance measure was used to compare new incoming traffic against the model. The relative position of different bytes inside the payload is not taken into account, so that the structure of the payload is not modeled. To model the structure of the payload, a value of $n \geq 2$ should be considered.

To include the structural information of the payload, Wang and Stolfo proposed ANAGRAM [76]. A value of $n \geq 2$ was used to extract byte sequence information. Supervised learning was employed to model normal traffic and attack traffic by storing *n*-grams of normal packets and attack packets into two separate Bloom Filters (BFs). A new incoming payload is categorized as anomalous if a major deviation exists in the *n*-grams of incoming payload with respect to *n*-grams of normal filter value. The representation of the payload by *n*-gram analysis generates a features space of size

256^n. It is easy to see that as the value of *n* increases, the size of feature space increases exponentially. This is the reason why in a real scenario a value of *n* greater than 2 is not used.

b) McPAD

Perdisci et al. proposed a Multi classifiers Payload Anomaly Detector (McPAD) [75]. McPAD is a payload anomaly detector based on ensemble of one-class SVM classifier. McPAD measures the occurrence frequency of pair of bytes that are ν position apart and creates 2*v*-grams by using a sliding window in a payload. When a packet is received from McPAD, the payload is extracted to perform feature extraction. After the extraction of features, the same payload results are represented in *m* different feature spaces by using a 2*v*-grams sliding window. The dimensionality of each feature space is then reduced using a clustering algorithm. The payload is processed by *m* different classifiers each working in a different space. Finally, the outputs of the classifiers are combined in a fusion stage. 2*v*-gram feature extraction technique may include partial structural information of the payload features.

In the following section, we present detailed description of our proposed model.

5.3 RePIDS: Real-time Payload Based Network Intrusion Detection System

In this section, we elaborate on our new approach. RePIDS is an anomaly detector based on payload, PCA [61] and MDM [117]. PCA selects important and suitable features from network traffic data, and MDM extracts the hidden correlations between features and the correlations among network packet payloads, which are used for the

classification of network traffic. Firstly, we present the framework of our real-time payload-based intrusion detection system. Then, we discuss the modules in the framework, namely data preparation module, *n*-gram text categorization module, 3-tier Iterative Feature Selection Engine (IFSEng), profile generation and traffic classification.

5.3.1 Framework of Real-Time Intrusion Detection System

The complete framework of our proposed intrusion detection system has four stages as shown in Figure 5.1. They are data preparation, data pre-processing, model generation and anomaly detection.

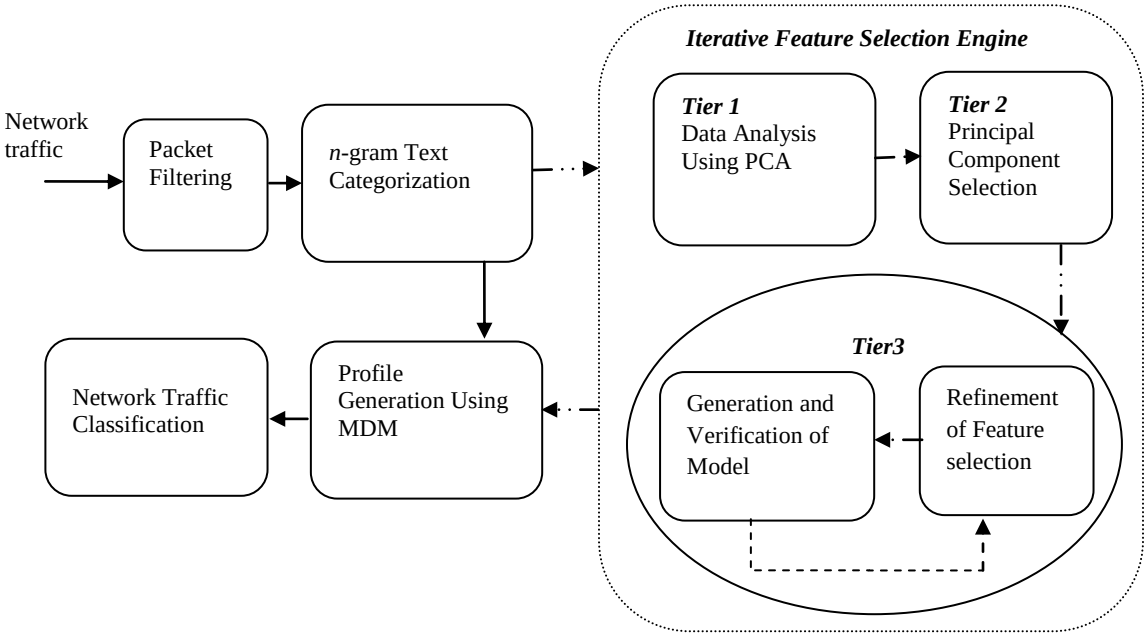


Figure 5.1: Framework for real-time payload based intrusion detection system

The first stage of this IDS consists of data preparation and *n*-gram text categorization [40]. For data preparation, the incoming network traffic is filtered according to type of application and payload length, and *n*-gram text categorization converts network traffic packet payloads into a series of feature vectors. These feature vectors describe the patterns of incoming traffic in a high dimensional feature space.

In the second stage, a 3-tier IFSEng, detailed in Subsection 5.3.2, is used for feature subset selection. Each tier performs a specific task. At tier 1, PCA technique [59] is used to analyse network traffic. At tier 2, selection of dominant Principal Components (PCs) (which is a subsets of important features) is performed using various methods as shown in [118] and [119]. And tier 3 refines the optimal feature subset (PCs) and evaluates the discriminative power of the feature subset to represent packet payloads. MDM shown in [117] (to be further discussed in Subsection 5.3.2) is used to capture more complex non-linear correlations among the selected features, and construct a distance map which represents a network traffic profile.

In the third stage of the framework, the finally selected PCs, (i.e., output of IFSEng) are used to build a normal traffic profile. An MDM is created for normal network traffic as a normal profile, which is used for classification of new incoming network traffic in the next stage.

In the last stage, Mahalanobis Distance criterion is used to measure the similarity between the pre-developed normal profile and the profile of a new incoming network packet. The packet is classified as a normal or an attack packet depending upon the

amount of deviation of its profile from the normal profile. Detailed description of each module is given in the following subsections.

5.3.2 Framework Modules

In this section, we provide a step-wise description and technical details of all modules contained in our proposed IDS framework.

a) Data Preparation Module

Data preparation is the first stage of the framework, where different datasets are prepared. We group network traffic into various categories using Wireshark [89], which is a traffic analyser, and separates the network traffic based on type of services, destination address, payload length and direction of network traffic flow. The source of network traffic can be real network (for real-time operation) or collected tcpdump files. The prepared dataset is used by next stage of intrusion detection system.

b) N-gram Text Categorization Module

n-gram Text Categorization is responsible for payload feature analysis and feature construction as discussed in Section 3.1.2 of this thesis. It extracts raw features using *n*-gram text categorization technique (here n=1) from the packet payload and converts observations into a series of feature vectors. Each payload is represented by a feature vector, which represents a pattern in the network payload in a *256*-dimensional feature space.

c) 3-tier Iterative Feature Selection Engine

The 3-tier Iterative Feature Selection Engine (IFSEng) consists of "Data Analysis Using PCA" (tier 1), "Principal Component Selection" (tier 2), and "Refinement of Feature Selection, Generation and Evaluation" (tier 3) modules.

At t***ier 1***, PCA [61] is used to analyse the original dataset. As a linear mathematical system, PCA is developed based on eigenvector-based multivariate analysis. It attempts to efficiently represent data by converting a set of observations into a new orthonormalized coordinate system, where the data are maximally decorrelated. The axes (eigenvectors or principal components) that contain greater variations (eigenvalues) make more contributions to the data representation. The first few most contributing axes are usually used to construct a new lower dimensional feature space to give efficient representations for the data.

PCA is applied on network traffic dataset, $Q = [q_1\, q_2 \cdots q_m]$, where m is the number of observations and each observation $q_i (1 \leq \mathrm{i} \leq m)$ is denoted by a *256*-dimensional feature vector $q_i = [f_1^i f_2^i \cdots f_{256}^i]^T$. First, mean-shift is conducted on the dataset for all the observations to make PCA work properly. The mean shifted dataset is represented by

$$Q_{sh} = \begin{bmatrix} q_1 - \bar{q} \\ q_2 - \bar{q} \\ \vdots \\ q_m - \bar{q} \end{bmatrix}^T, \tag{5.1}$$

where $\bar{q} = \frac{1}{m}\sum_{i=1}^{m} q_i$. Then, the principal components are obtained by analysing the sample covariance matrix C_Q of the data set given in Equation 5.2.

$$C_Q = \frac{1}{m-1} Q_{sh} {Q_{sh}}^T. \tag{5.2}$$

Using eigen-decomposition, the covariance matrix C_Q can be decomposed into a matrix W and a diagonal matrix λ. They satisfy the condition, $\lambda W = C_Q W$. The columns of the matrix W stand for the eigenvectors (called the principal components) of the covariance matrix C_Q, and the elements along the diagonal of the matrix λ are the ranked eigenvalues associated with the corresponding eigenvectors in the matrix W.

PCA only demonstrates the contribution of different components of a feature space in terms of data representation. It does not determine the number of principal components that should be retained. Thus, some other supplementary techniques are applied at Tier 2 to decide the optimal number of components to be retained based on the analysis results from PCA.

At t***ier 2***, several techniques, such as cumulative energy [61], scree test [118, 120] and parallel analysis criteria [119], help achieve one of the main goals of good pre-processing principal that is to retain as much relevant information as possible. Cumulative energy, scree test and parallel analysis criteria are utilized independently to select corresponding k_1, k_2 and k_3 principal components (the eigenvectors of matrix W) respectively. The selected k_1, k_2 and k_3 principal components are three subsets of k (which equals to 256 in our case), principal components contained in matrix W. These mathematical and non-mathematical criteria are used to verify the outcomes of each others. The subsets of principal components represent reduced feature spaces, which provide the best presentations determined by the criteria for a packet payload. By projecting the feature vector $q_i = [f_1^i f_2^i \cdots f_{256}^i]^T$ onto these selected reduced feature spaces, the dimension of the feature vector can be reduce significantly to smaller values,

namely k_1, k_2 and k_3. At the meanwhile, the criteria guarantee that the reduced feature vector can correctly represent the packet payload. A brief explanation of individual criteria is given below.

Cumulative Energy. An energy associated with a component is represented by the corresponding eigenvalue. The greater an eigenvalue is, the larger energy the corresponding component (eigenvector) has. Suppose $(\lambda_1, u_1),\ (\lambda_2, u_2), \ldots, (\lambda_k,\ u_k)$ are k eigenvalue-eigenvector pairs decomposed from the covariance matrix C_Q. The cumulative energy of the first k_1 components is defined by the sum of the energies across the components from 1 through k_1, and it is computed using Equation 5.3.

$$CE = \sum_{j=1}^{k_1} \lambda_j \,, \tag{5.3}$$

where $k_1 \in \{1, \cdots, k\}$ can be determined subject to the objective function given in

$$\frac{CE}{\sum_{j=1}^{k} \lambda_j} \geq \alpha \,. \tag{5.4}$$

In Equation 5.4, α is the ratio of variation in the subspace to the total variation in the original space. This objective function intends to obtain a value of k_1 as small as possible while achieving a reasonably high value of *CE* on a percentage basis.

Scree Test is a graphical method, first proposed by Cattell [118] in 1966. More explanations of scree test are given by Nelson [121]. In a scree plot all eigenvalues are plotted against all (k) principal components (eigenvectors) in the descending order. In the scree plot, we look for the k_2-th point, where sharp decrease in eigenvalue levels off (the scree). This point is identified as an 'elbow'. After the k_2-th point, the remaining ($k-k_2$) principal components (eigenvectors) are ignored and not used in the model. This is based on the arguments that the most significant components extract a large

proportion of the variances from the covariance matrix, while the remaining insignificant ($k - k_2$) ones are associated with similar low value variances. The criticisms of scree test criterion are that there is no sharp transition where the scree begins, and the decision is not robust and reproducible. Alternatively, parallel analysis criterion is used to verify the selection of principal components (feature subset).

Parallel Analysis (PA) is a modification of Cattell's scree test. PA [117] alleviates the component indeterminacy problem and determines which variable loadings are significant for each component. This operation is repeated twice and the obtained eigenvalues for each component are used to calculate means and Standard Deviations (SD) in the two iterations. From the means and standard deviations, the 95 percentile values are obtained (95 percentile = mean + 1.65 SD). If the eigenvalue of a component exceeds the 95 percentile of the simulated values, then the component is retained.

At t***ier 3***, feature refinement and evaluation module is used. In the refinement stage, we extend the range of the selected principal components, obtained from tier 2, on both the upper and lower sides. Then, we observe the discriminative power of the subsets of principal components to represent packet payloads. Lastly, we select the final $k_{final} \in \{k_1, k_2, k_3\}$ feature principal components through iterative evaluation of normal training model using *F*-Value (as discussed in Section 2.4 of this thesis) using Equation 5.5.

$$F\text{-Value} = (1 + \beta^2) * Recall * \frac{Precision}{\beta^2(Recall + Precision\ )}, \tag{5.5}$$

A low value of precision means a higher degree of false positives and vice versa. A lower value of recall represents a higher degree of false negatives and vice versa. In Equations 5.5, β corresponds to the relative importance of precision versus recall and is

usually set to 1. On one hand, when precision and recall have equal weights and close to 1, the model can achieve *F*-Value close to 1, which indicates good performance meaning that the classifier has 0% false alarms and 100% detection of attacks. On the other hand, *F*-Value close to 0 indicates poor performance. Thus, the *F*-Value of a classifier is desired to be as high as possible.

The selected k_{final} principal components are the ones which facilitates the classifier to achieve the greatest *F*-Value among the candidates k_1, k_2 and k_3. Then, selected k_{final} principal components are used in the profile generation, which is briefly discussed in the following Subsection.

d) Profile Generation Using Mahalanobis Distance Map

Network traffic profile is generated using Mahalanobis Distance Map (MDM) which captures complex non-linear correlations of the data. By using MDM (as described in Section 3.1.2 of this thesis), we obtain the hidden correlations between the features of projected feature vector $\left[x_1 x_2 \cdots x_{k_{final}}\right]$, by projecting the original feature vector $q = [f_1 f_2 \cdots f_{256}]^T$ onto the k_{final} dimensional feature subspace $\left[u_1 u_2 \cdots u_{k_{final}}\right]$ (outcome of IFSEng); and the correlations among packets as follows.

$$\sum a = (x_a - \mu)(x_a - \mu)^T \left(1 \le a \le k_{final}\right), \tag{5.6}$$

$$d_{(a,b)} = \frac{(x_a - x_b)(x_a - x_b)^T}{\sum a + \sum b} \left(1 \le a, b \le k_{final}\right), \tag{5.7}$$

$$D = \begin{bmatrix} d_{(1,1)} & d_{(1,2)} \cdots & d_{(1,k_{final})} \\ d_{(2,1)} & d_{(2,2)} \cdots & d_{(2,k_{final})} \\ \vdots & \vdots \quad \ddots & \vdots \\ d_{(k_{final},1)} & d_{(k_{final},2)} \cdots & d_{(k_{final},k_{final})} \end{bmatrix}, \tag{5.8}$$

where x_a represents the *a*-th projected feature in the projected feature vector, μ denotes the average of each projected feature, $d_{(a,b)}$ defines the Mahalanobis distance between the *a*-th projected feature and the *b*-th projected feature, $\sum a$ is the covariance value of each projected feature, and finally *D* is the MDM (the pattern of a network packet). Distance map *D* is used to generate the network traffic profiles (normal and attack) of the training and test data. These profiles are used for the classification of incoming network traffic.

e) Traffic Classification

Mahalanobis distance is the criterion used to measure the similarity between the developed profile and new incoming network traffic profile. Weight *w* is calculated using Equation 5.9 to detect an intrusive activity.

$$w = \sum_{a,b=1}^{k_{final}} \frac{\left(d_{obj(a,b)} - \bar{d}_{nor(a,b)}\right)^2}{\sigma^2_{nor(a,b)}}, \tag{5.9}$$

where $\bar{d}_{nor(a,b)}$ and $\sigma^2_{nor(a,b)}$ are the average and variance of the (*a*,*b*)-th element in the distance map $D_{nor} = \left[d_{nor(a,b)}\right]_{k_{final} \times k_{final}}$ of the normal profile, and $d_{obj(a,b)}$ is the (*a*,*b*)-th element of the distance map $D_{obj} = \left[d_{obj(a,b)}\right]_{k_{final} \times k_{final}}$ of the new incoming packet. We calculate threshold value as explained in Chapter 3 framework section. If the weight factor exceeds the calculated threshold, the input packet is considered as an intrusion.

5.4 Experimental Results and Analysis

In the following subsections, we first present experimental setup and brief information on the dataset and types of attacks. We then discuss training and test of our model. Finally, we present experimental results and analysis.

A Series of experiments on DARPA 1999 [96] dataset and GATECH attack dataset [97] are conducted to evaluate the performance of our proposed model. These two datasets are used by the state-of-the-art payload-based IDSs that we will compare in this paper.

5.4.1 Experimental Setup

The experimental setup as discussed in Chapter 3 is used to conduct experiments. We use Matlab 2009b for the simulation of RePIDS.

We experiment with several different threshold (δ) values. For evaluation of our model, we use threshold (δ) equal to ±3σ, where σ represents single standard deviation value, because this value will give us good results for detection rate, false positive rate and *F*-Value.

5.4.2 Datasets

a) Training (Normal Traffic) Dataset

We extract week 1 and week 3 inbound 'HTTP request' traffic from DARPA 99 dataset for the training of our model. The extracted normal traffic corresponds to two different HTTP servers, as discussed in Chapter 4 of this thesis. The total numbers of packets

used for training of the model after filtering are 13,933 and 10,464 for hosts *marx* and *hume* respectively.

b) Test (Attack + Normal Traffic) Datasets

In order to test the performance of our proposed model in detecting known attacks and new attacks, we use attacks contained in DARPA 1999 dataset and GATECH attack dataset. The labeled test data is further pre-processed to form two test datasets, which contain instances that do not appear in our training dataset. For our experiments, we focus on attacks coming through HTTP service only.

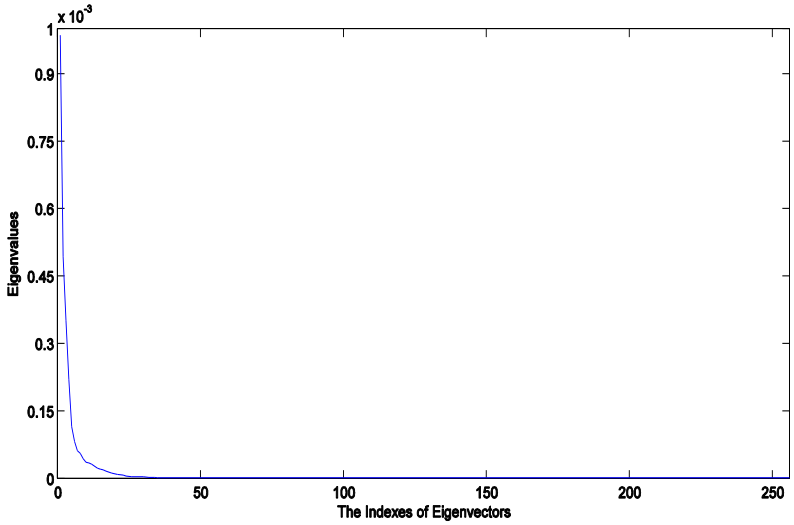
As explained in Chapters 3 and 4 of this thesis, HTTP-based attacks are mainly from the HTTP GET/POST requests to web servers. There are several HTTP-based attacks provided by DARPA 1999 dataset, namely Apache2 attack, Crashiis attack, back attack and Phf attack. The GATECH attack dataset has several non-polymorphic HTTP attacks provided by Ingham and Inoue [97] and several polymorphic HTTP attacks created using CLET engine generated by Perdisci et al. [83]. The attacks, namely Generic attack, Shell-code attack and CLET attack (polymorphic attack), are placed in different groups, and each group has attacks of the same category for the presentation of results. All HTTP request attack packets are used in our experiments.

5.4.3 Model Training and Testing Process

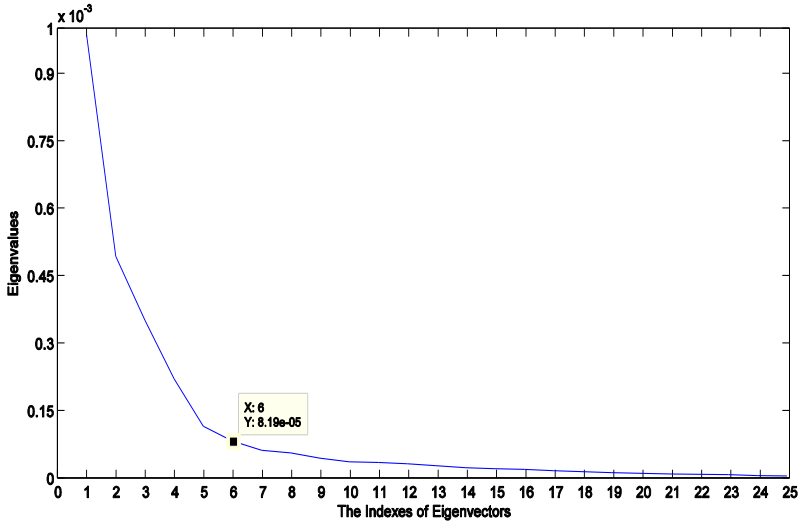
The experimental approach involves following procedure for training and testing of model:

1. We use 185 bytes of HTTP GET request payload for training and testing of model. *n*-gram text categorization module discussed in Subsection 5.3.2, represents HTTP GET request payload by a feature vector *q* in a *256*–dimensional feature space.
2. As discussed in Subsection 5.3.2 of this thesis, tier 1 of IFSEng uses the PCA technique to analyse raw data, by projecting raw data on a reduced feature space. Then, tier 2 of IFSEng performs selection of dominant Principal Components using cumulative energy, scree test and parallel analysis criteria on the outcome of PCA. First, cumulative energy criterion is applied for the selection, in which we consider 93 percent of cumulative energy level for Equation 5.6. k_1=7 is obtained, which means that the first 7 principal components are selected as the best subspace to represent the data by cumulative energy criterion. Then, we use scree test to draw scree plot as a variance captured by a given principal component and to select another set of principal components. Figure 5.2(a) shows full scree plot, where we use *k* (*k* = 256 in our case) principal components (*X*-axis) of a particular dataset and the corresponding variances, namely eigenvalues (*Y*-axis) to draw a scree plot, and the PCs are sorted in descending order with respect to the values of the corresponding variances. In Figure 5.2(a), we look for an 'elbow', a flattening out of the curve. To provide better vision, we magnify the scree plot and show the first 25 principal components in Figure 5.2(b). It can be observed from Figure 5.2(b) that there is a sharp decrease of variance in the front part of the plot and then it starts flattening out after the 6-th principal component. In Figure 5.2(b), we can observe 'elbow' somewhere in the range from 6 to 9 principal components and the first k_2 = 6 principal components are able to capture about 92 percent of the variance. After

the k_2-th point, the remaining (k - k_2) principal components capture only around 8 percent of the total variance and are ignored.



(a)



(b)

Figure 5 2: Scree test plot. (a) full scree plot; (b) enlarged scree plot with first 25 eigenvectors

We use k_2 = 6 as a dominant principal components in our case. However, from Figure 5.2(b), we have observed a range of principal components from 6 to 9, and it is not very clear that what is the most appropriate value of k_2. To overcome this

ambiguity, we use parallel analysis criterion as described in the following and to verify the selection of k_2.

We verify the outcome of scree plot by using parallel analysis criterion as discussed in Subsection 5.3.2 on the same dataset. The result of parallel analysis also suggests a selection of first 7 principal components, which is the same as what has been obtained using cumulative energy criterion.

The results of three feature selection criteria are given in Table 5.1.

Table 5.1: Principal Component (PC) selection

PC Selection Method	Cumulative Energy (0.93)	Scree Test	Parallel Analysis
Number of PCs	7	6	7

3. Although these are the dominant principal components, further refinement of dominant principal components is needed to be done at tier 3 of IFSEng (as presented in Subsection 5.3.2) because of the ambiguity in these results. In addition, training model generation and evaluation at tier 3 is performed using *F*-Value metric defined in Equation 5.7. The MDM represents the correlations among the features obtained from the projection of the original feature vector onto the finally selected principal components and among packets. These principal components help represent normal behavior profile in the low dimensional feature space.
4. For testing, we project the extracted feature vector of an incoming packet payload onto the reduced feature space (the finally selected principal components) and use

Mahalanobis distance dissimilarity criterion to detect intrusive behaviors. The performance of RePIDS in detecting attacks is evaluated using *F*-Value.

In the experimentation, the 10 days normal 'HTTP GET request' traffic from DARPA 1999 dataset is used. The normal traffic is randomly divided into three subsets. One of the subsets is selected randomly and used for training the model. The remaining two subsets are reserved for the test of the model.

In the testing stage, an attack is detected as long as one of its attack packets is identified as abnormal. We conduct our experiments using the features obtained from the projection of original feature vectors onto the optimal principal components determined by the IFSEng for various types of attacks (Apache2, Phf, Crashiis and Back attacks) present in DARPA 99 attack dataset. We further evaluate our model on GATECH attack dataset, which is comprised of Generic, Polymorphic (CLET) and Shell-code attacks.

5.4.4 Results and Analysis

Experimental results are explained in two steps. In the first step of the experiments, we obtain an optimal subset of principal components. Then, we design a number of experiments according to Figure 5.1, showing the RePIDS framework, to determine the performance of our proposed model when using various subsets of principal components varying from 5 components to 9 components. Experiments are also conducted for different values of threshold varying from 2σ to 3.5σ. Results are presented in Table 5.2 for various feature subsets and using 3.5σ as the optimal value of threshold.

Table 5.2 shows the variation of FP, TN, TP and FN rates along the change of the number of principal components.

To obtain the optimal number of principal components, *F*-Value is calculated for each feature subspace (principal components) using Equation 5. 7. The variation of *F*-Value with the number of principal components is shown in Figure 5.3.

Table 5.2: Performance scores corresponding to the number of Principal Components

	5 PCs	6 PCs	7 PCs	8 PCs	9 PCs
False Positive (FP) Rate	1.37%	0.67%	0.85%	1.31%	1.99%
True Negative (TN) Rate	98.63%	99.33%	99.15%	98.69%	98.01%
True Positive (TP) Rate	98.70%	99.50%	100%	100%	99.97%
False Negative (FN) Rate	1.30%	0.50%	0	0	0.03%

The results in Figure 5.3 show that the best *F*-Value is achieved with 7 principal components. In other words, the feature subspace of 7 principal components has good representation, discriminative power and high accuracy. The increase and decrease of the eigenvectors both dilute the performance of RePIDS.

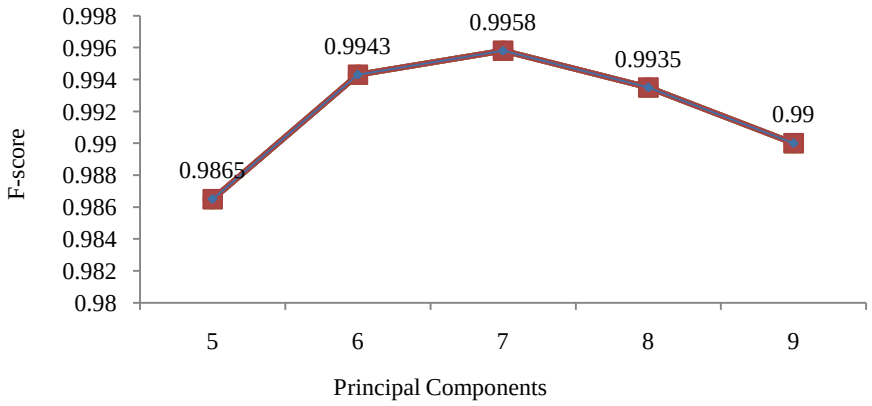


Figure 5.3: Trend of *F*-Value

It can be concluded that PCA and the three selection criteria help reduce the dimensionality of dataset from 256 to 7. The amount of information extracted using IFSEng is high in the selected 7-dimensional feature space, which helps create more accurate normal traffic profiles using MDM that is used for traffic classification.

To demonstrate how MDM presents the correlations between the features, the MDMs of normal HTTP payload and some attack payloads are generated using projected features on the optimal 7-dimensional feature space.

0	0.001406625	0.001449804	0.001332988	0.001463112	0.001270879	0.001241186
0.001406625	0	0.000289528	0.000305565	0.000268982	0.000231624	0.000208517
0.001449804	0.000289528	0	0.000239652	0.000214287	0.000194018	0.000163789
0.001332988	0.000305565	0.000239652	0	0.000287999	0.000198282	0.000158613
0.001463112	0.000268982	0.000214287	0.000287999	0	0.00016282	0.000170964
0.001270879	0.000231624	0.000194018	0.000198282	0.00016282	0	9.17989 exp-05
0.001241186	0.000208517	0.000163789	0.000158613	0.000170964	9.17989 exp-05	0

Figure 5.4: MDM of normal HTTP Payload.

0	7.211042exp-05	3.686978 exp-06	0.00237153	0.000102354	0.00078712	0.00072289
7.211042 exp-05	0	5.214637 exp-05	0.00163562	0.00033709	0.00132127	0.00034557
3.686978 exp-06	5.214637 exp-05	0	0.00225582	0.00012659	0.00085651	0.00065863
0.00237153	0.00163562	0.00225582	0	0.00344129	0.00586325	0.00048361
0.00010235	0.00033709	0.00012659	0.00344129	0	0.00032706	0.00135888
0.00078711	0.00132127	0.00085651	0.00586325	0.00032706	0	0.00300482
0.00072289	0.00034557	0.00065864	0.00048362	0.00135888	0.00300482	0

(a)

0	0.000677245	0.00081015	0.00032632	0.00019996	0.00095956	0.00014843
0.000677245	0	0.00022798	0.00036073	0.00038006	0.00045121	0.00033855
0.00081015	0.00022798	0	0.00029764	0.0003492	0.00030296	0.00032801
0.00032632	0.00036073	0.00029764	0	8.31436139	0.00032254	0.00011205
0.00019996	0.00038006	0.0003492	8.31436139	0	0.00033113	8.634902 exp-05
0.00095956	0.000451205	0.00030296	0.00032254	0.00033113	0	0.00055453
0.000148432	0.00033855	0.00032801	0.00011205	8.634902 exp-05	0.00055453	0

(b)

0	0.05178815	0.04735877	0.04525517	0.03765384	0.03965582	0.05104155
0.051788147	0	0.03508168	0.05975747	0.05529712	0.05478485	0.03144298
0.047358766	0.03508168	0	0.03686035	0.0250256	0.0571498	0.0332321
0.045255171	0.05975747	0.03686035	0	0.05269052	0.05324839	0.05400761
0.037653843	0.05529712	0.0250256	0.05269052	0	0.03450803	0.04522816
0.039655825	0.05478485	0.0571498	0.05324839	0.03450803	0	0.04336399
0.051041546	0.03144298	0.0332321	0.05400761	0.04522816	0.04336399	0

(c)

Figure 5.5: MDMs. (a) apache2 attack; (b) crashiis attack; (c) phf attack payloads

Figures 5.4-5.5 shows the MDMs of normal HTTP payload and some attack payloads, respectively. It can be seen from Figures 5.4-5.5 that the MDM is a symmetric matrix and the values of the elements along its diagonal are all equal to zeros. This is because the correlation of a feature to itself is always zero. MDMs also demonstrate that the correlations between normal projected features are different from the correlations between attacks projected features. Besides, the 7-dimensional space is able to help differentiate normal payload and various attack payloads efficiently and accurately.

Figure 5.4 shows the MDM of normal HTTP payload (normal profile), and Figures 5.5(a)–(c) show the MDMs of the attack profiles for Apache2, Crashiis and Phf attacks.

Although we can directly compare the normal profile (model) and attack profiles (MDMs) to confirm the differences between normal and various attack payloads, it is a time-consuming task. Having MDM profiles for training dataset and a new incoming packet, the weight score *w* is calculated. If the deviation in weight score *w* is greater than the pre-selected threshold, then the incoming packet is classified as an attack packet.

Moreover, to evaluate the robustness of RePIDS in recognizing unknown attacks (Generic, Shell-code and Polymorphic (CLET) attacks), we conduct experiments on

GATECH attack dataset using the same setup. Table 5.3 reports the FP rate, TN rate, TP rate, FN rate and *F*-Value on optimal 7-dimensional space.

Table 5.3: Performance score

Performance score	7 eigenvectors
False positive (FP) rate	0.86%
True negative (TN) rate	99.15%
True positive (TP) rate	96.29%
False negative (FN) rate	3.71%
F- value	0.976

It can be concluded from Table 5.3 that RePIDS has a high detection rate, a low false positive rate and a low false negative rate. The *F*-Value achieved is 0.976, which confirms that the model can detect attacks with high accuracy and demonstrates its good performance.

In conclusion, the proposed RePIDS is able to detect novel attacks very well, with a high *F*-Value (0.976) and a low FP rate.

5.5 Comparison of RePIDS

In this section, comparisons between RePIDS and the state-of-the-art PAYL and McPAD anomaly based intrusion detection systems are presented. Then, we further compare throughput of our proposed model with that of real scenario of a medium sized enterprise network.

5.5.1 Detection Performance

In order to provide a reasonable comparison for these payload-based IDSs, the detection performance of RePIDS, PAYL and McPAD anomaly based intrusion detection systems is first compared. Thus, we use the results of false positive rate and detection rate from [83]. From Figures 5.6-5.7 in [83], we estimate average detection rates for generic, shell-code and polymorphic attacks. We use false positive rate of 1% to calculate *F*-Values for PAYL and McPAD on GATECH attack dataset respectively. As mentioned in [83], their results for DARPA 1999 dataset are similar to those for GATECH attack dataset. Table 5.4 shows the comparison of *F*-Values for PAYL, McPAD and RePIDs on DARPA 1999 dataset and GATECH attack dataset. From Table 5.4, we can conclude that RePIDS shows better *F-Value* in comparison with PAYL and McPAD on DARPA 99 and GATECH attack datasets.

Table 5.4: Performance comparison

	RePIDS	PAYL*	McPAD*
DARPA 99	0.9958	0.969*	0.953*
GATECH	0.976	0.969	0.953

** F-Values for DARPA 99 dataset and GATECH attack dataset for PAYL and McPAD have been derived from [75].*

5.5.2 Complexity Analysis

In this section, we provide an analysis of the computational complexities of the algorithms used in RePIDS, PAYL and McPAD. Only the computation involved in the test phase is taken into account in the analysis, due to the training of the algorithms can be performed off-line, which does not affect efficiency of the algorithms in detection.

Given a payload P of length n and a fixed value of v, the occurrence frequencies of 1-gram and $2v$-grams can both be computed in $O(n)$. The numbers of extracted features in these algorithms are constant regardless of the actual values of n and v (2^8 features extracted by RePIDS and PAYL, and 2^{16} features extracted by McPAD).

The feature reduction process of the RePIDS can be completed by $2^8*2*7=3584$ simple operations including multiplications and additions. In contrast, McPAD algorithm reduces features by mapping the occurrence frequency distribution of $2v$-grams to the k feature clusters using a simple look-up table and a number of sum operations that is always less than 2^{16} (regardless of the value of k). Therefore, the feature reduction processes of RePIDS and McPAD can be computed in $O(1)$. However, there is no feature reduction is performed in PAYL.

Thus, the complete computational complexities of data pre-processing of the RePIDS, PAYL and McPAD algorithms can be obtained by adding up the computational complexities of the feature extraction and reduction processes. Since RePIDS uses a fixed payload length (185 bytes) to extract the occurrence frequency, the complete computational complexities of data pre-processing is $O(1)$. PALY has a complete computational complexities of data pre-processing equal to $O(n)$, because no feature reduction is required. For McPAD, it has to be repeated m (representing the number of different one class classifiers used to make a decision about each payload P) times every time choosing a different value of v. Hence, the complete computational complexities of data pre-processing of McPAD can be accomplished in $O(nm)$.

Once the features have been extracted and the dimensionality has been reduced to *k*, each payload has to be classified according to each of the *m* classifiers. To classify a payload *P*, RePIDS computes the Mahalanobis distance between the payload and the pre-determined normal profile. Given the number of features equal to 7 as determined and a single classifier used in classification, the computational complexity of the classification process of RePIDS is $O(1)$. Similarly, PAYL uses a single classifier to classify the payload *P* represented by 256 features. Therefore, the classification process of PAYL can be accomplished in $O(1)$ as well. Compared to RePIDS and PAYL, McPAD has *m* classifiers. Each classifier computes the distance between the payload *P* represented by *k* feature clusters and each of the support vector *s* obtained during training. Therefore, the classification of a payload using McPAD can be computed in $O(ks)$. McPAD has to repeat the classification process *m* times and the results are then combined. Thus, the overall classification process of McPAD can be computed in $O(mks)$.

The detailed break-down of the computational complexity of the algorithms is given in Table 5.5.

Table 5.5: Computational complexity of RePIDS, PAYL and McPAD

	RePIDS	PAYL	McPAD
Complexity of data pre-processing	$O(1)$	$O(n)$	$O(nm)$
Complexity of classification	$O(1)$	$O(1)$	$O(mks)$
Overall complexity	$O(1)$	$O(n)$	$O(nm+mks)$

As shown in Table 5.5, the overall computational complexities of RePIDS, PAYL and McPAD are $O(1)$, $O(n)$ and $O(nm+mks)$ respectively. This proves that our RePIDS has the lowest computational complexity in comparison with PAYL and McPAD.

We also evaluate the efficiency of our scheme by comparing the throughput of RePIDS with a similar environment used within a medium size enterprise network with a gateway speed of 1GB. Our throughput comparison is based on number of packets processed through such a network against the packet processing speed of our scheme considering the most ideal parameters. On one hand, the throughput calculated for a medium size enterprise network, considering that ideal parameters is 25600 packets in one second. However, for real-time applications using IDS we expect the throughput to be much less. On the other hand, our proposed scheme could process 33146 packets per second, which is 1.3 times more than the packet processing speed on the enterprise network, indicating our scheme has potential to be implemented in real-time. However such consideration involving real throughput analysis with most ideal network parameters is beyond the scope of this thesis and we intend to extend it for our future work.

Summarizing the overall performance in terms of detection accuracy and computational complexities of algorithms, RePIDS performs better than the state-of-the-art PAYL and McPAD anomaly based intrusion detection systems. Furthermore, in terms of throughput, RePIDS can process more packets per second than the throughput of a medium sized enterprise network with a gateway speed of 1GB. Hence, our model, RePIDS, is expected to be capable of processing packets in real time operation.

5.6 Conclusions

In this chapter, we have proposed an efficient payload-based intrusion detection system (RePIDS) to detect attacks against Web applications through the analysis of HTTP payloads using 3-tier Iterative Feature Selection Engine (IFSEng) and Mahalanobis Distance Map (MDM). Mahalanobis distance criterion is used for classification of network data. The proposed model uses selected, small size of feature subspace to detect generic, shell-code and CLET attacks.

The proposed 3-tier IFSEng is used to select an optimal feature subspace and reduce the dimensionality of the data, which significantly influence the detection efficiency.

RePIDS has been thoroughly tested on the normal traffic of DARPA dataset, and on two different datasets of attacks, namely DARPA 1999 and GATECH datasets. Experimental results indicate that the method is effective in detecting attacks with high detection rates and low false positive rates. RePIDS has achieved high *F*-Value, 0.9958 on DARPA dataset and 0.976 on GATECH dataset respectively.

In particular, we have demonstrated that RePIDS performs better in comparison with the state-of-the-art PAYL and McPAD. In addition, we have also showed that the computational complexity of RePIDS for the classification of new incoming traffic payload is lower than PAYL and much less than McPAD.

Finally, in terms of throughput, RePIDS can process more packets per second than the throughput of a medium sized enterprise network with a gateway speed of 1GB. Hence, our model, RePIDS, is expected to be capable of processing packets in real-time operation.

CHAPTER 6

Conclusion and Future work

To withstand the increasing threat of network attacks, it is required to detect novel attacks as soon as they appear. However, current signature-based intrusion detection systems can only detect known attacks because they depend on the generation of signatures.

In addition, with the popularity of Internet and the increase in number of attack incidents on the Internet, web security is one of the key challenges in computer security research. Moreover, web applications are generally large, complex and highly customized. To protect each web applications, it requires signatures written explicitly for the application. It is not possible to guarantee that the application is completely free of vulnerability and secure as unknown vulnerabilities might exist. Writing signatures is also difficult. Hence, an application protected only by a signature based IDS cannot be completely secure at all. With emerging network technologies, anomaly-based intrusion detection systems are believed to provide a practical solution for identifying known and novel attacks on the networks, and for the protection of web applications. Anomaly-based IDS creates a statistical model of the normal behavior from a set of training data.

If any network activity deviates too far from the pre-developed normal model, then the activity generates an alarm and identifies network activity as a novel attack. Unfortunately, existing anomaly detection approaches suffer from several significant weaknesses. In this chapter, we summarize our thesis and review the contributions, and discuss possible future work.

6.1 Summary

In this thesis, we have addressed the problems of detecting unknown attacks in application layer of network communication. In particular, we have introduced novel frameworks and developed models which address three critical issues that severely affect large scale deployment of payload-based anomaly detection systems in high speed networks. These three issues are:

- Limited attack detection coverage,
- Large number of false alarms, and
- Inefficiency in operation.

This thesis described a number of novel frameworks using network payload for effectively detecting wide variety of payload-based attacks and zero-day attacks, in particular, web-based attacks. We have proposed three anomaly detectors, namely Geometrical Structure Anomaly Detection model (GSAD), two-tier LDA-based detector, and Real-time Payload-based Intrusion Detection System (RePIDS), which can detect novel attacks and protect networks at the application level. We have generated a common profile using optimal features for a group of similar types of attacks.

6.1.1 Geometrical Structure Anomaly Detection Detector

The proposed GSAD anomaly detector uses pattern recognition techniques to identify patterns of packet payloads. GSAD models the payload of network traffic using geometrical structures of the payload features and correlations between the payload features. This approach can detect new attacks without a-priori knowledge of the attacks. *n*-grams Text Categorization and Mahalanobis Distance Map (MDM) approaches are used to develop payload profile. MDM technique determines the hidden correlations between payload features, and includes payload structural information partially, which helps to improve the detection rate and false positive rate. We have implemented the GSAD model in the HTTP environment to detect web-based attacks coming through HTTP service, at port 80, and evaluated it on two datasets, namely DARPA 99 and GATECH datasets. GATECH dataset contains real traces of various attacks coming through HTTP service. The MDMs compute the correlations between the payload features. Deviation between the average MDM profile for training dataset and MDM profile of a new incoming packet has been used to classify the incoming packet into either an attack packet or a normal packet. In Chapter 3 of this thesis, MDM images (geometrical patterns) shown in Figures 3.9-3.14 have confirmed the differences between normal and various attack payloads. In addition, they have demonstrated differences in the correlations between the features of various attack payloads and the correlations between the features of normal packet payload.

6.1.2 Two-tier LDA-Based Detector

We have proposed a novel framework for a two-tier detector, which uses LDA technique and difference distance map (DDM) to order the potential features for payload feature selection and to distinguish normal and attack patterns in the network traffic. Linear discriminate iterative feature selection algorithm is used to select an optimal set of features. The two-tier detector uses the packet payload length criterion to group packets and forward them either to Statistical Signature Based Detector on the first tier or Linear Discriminate Method (LDM) Based Detector on the second tier detector for further analysis. Then, the detector analyzes the received packet and makes final decision to raise an alarm or not. The Receiver Operating Characteristic (ROC) curves for the proposed two-tier system are shown in Figure 4.11. The proposed two-tier system has showed 100% detection rate and 3.38% false positive rate. The LDA-based approach reduces the computational complexity dramatically while retaining the high detection rates and providing a novel lightweight solution for network payload-based attacks detection.

6.1.3 Real-time Payload Based Intrusion Detection System

For real-time operation of payload-based system, we have proposed a novel, efficient real-time payload-based detector, RePIDS, which maintains small footprints in terms of use of resources, computational complexity and packet processing speed. RePIDS uses Principal Component Analysis (PCA) approach, which is an unsupervised technique to construct important and suitable features and select dominant Principal Components by means of cumulative energy, scree test and parallel analysis criteria on the outcome of

PCA. We have built a real-time payload-based intrusion detection system using suitable features. As discussed in Chapter 5, RePIDS has two key components, which are 3-Tier IFSEng and MDM. 3-Tier IFSEng addresses the issues, related to the quality of feature set, and Mahalanobis Distance Map (MDM) extracts the hidden correlations between features and the correlations among network packet payloads. Together, they have facilitated effective and efficient detection of attack packets in the network traffic. RePIDS has achieved high *F*-Value of 0.9958 on DARPA dataset and 0.976 on GATECH dataset respectively. This demonstrates that RePIDS can differentiate normal and attack instances accurately. In particular, we have demonstrated that RePIDS performs better in comparison with the state-of-the-art PAYL and McPAD, and the computational complexity of RePIDS for the classification of new incoming traffic payload is lower than PAYL and much less than McPAD. Furthermore, we have shown that RePIDS can process more packets per second than the throughput of a medium sized enterprise network with a gateway speed of 1GB. All these facts have given substantial evidence that the proposed model, RePIDS, is capable of processing packets in real-time operation.

6.1.4 Single Profile (Signature) for a Group of Similar Types of Attacks

We have also proposed a preliminary solution to create a single profile (signature) for a group of similar types of attacks. Based on research presented in [122], we have used similar concept and developed one common profile (signature) for the normal traffic that could classify three different types of attacks, namely, Phf, Apache2 and Back, and reduce the number of signatures to be compared.

6.2 Thesis Contributions

We recap the thesis contributions here:

- We have identified and addressed three critical issues that may have severely affected deployment of payload-based anomaly detection system in high speed networks.
- We have built GSAD, a payload-based intrusion detection system, which models payload statically using geometrical structure of payload features and language independent *n*-gram ($n = 1$, in our case). We have demonstrated the effectiveness in detecting new and variants of known attacks using DARPA dataset and GATECH dataset, which contain traces of real traffic.
- We have built a two-tier novel detector using LDA technique and Difference Distance Map (DDM) approach to order the potential features for payload feature selection and distinguish normal and attack patterns in the network traffic.
- We have described a preliminary solution to create a single signature for a group of similar types of attacks that could efficiently classify these different types of attacks and can help to reduce the number of signatures to be compared.
- We have built an efficient payload-based intrusion detection system (RePIDS) to detect attacks against Web applications through the analysis of HTTP payloads using 3-Tier Iterative Feature Selection Engine (IFSEng) and Mahalanobis Distance Map (MDM). The proposed model uses selected features from a low dimensional feature subspace to detect generic, shell-code and CLET attacks.

Furthermore, RePIDS is capable of discriminating normal patterns and attack patterns in real-time.

6.3 Future Work

We now outline some interesting avenues for future research.

- We have evaluated the efficiency of our real-time payload-based intrusion detection system by comparing the throughput of RePIDS with a throughput of a medium size enterprise network with a gateway speed of 1GB under similar environment. To further test its effectiveness and real throughput analysis, we need to run RePIDS in real-time and choose the proper network parameters involving real throughput analysis. Furthermore, to improve the performance of RePIDS in real-time, work can be extended and regress simulation can be done using real network test bed.
- We have evaluated our proposed profile (signature) generation scheme as a preliminary solution on DARPA 99 dataset. To further test the effectiveness of this scheme, we need to develop more signatures for similar types of attacks and evaluate those signatures on different datasets. To create a single signature for a group of similar types of attacks that could efficiently classify these different types of attacks and can help to compress the number of signatures to be compared, more tests on GATECH dataset and any other datasets may be performed.
- We have evaluated our two tier model on DARPA 1999 dataset. To further test the effectiveness and performance of this model, we need to test this model on

GATECH attack dataset. And also need to develop more signatures for similar types of attacks.

- RePIDS has limitation in terms of encrypted text. It is used for intrusion detection of unencrypted (plain text) payload data only and does not look into encrypted data. However, it can detect attacks coming through encrypted data when used at the host machine using an appropriate encryption key. Hence, this is an important area of research for payload-based intrusion detection system.
- RePIDS is a real-time centralised intrusion detection system. We believe our work can be extended to distributed network environment. Internet has opened venues for attackers to send malicious packets in the networks. Supervisory Control and Data Acquisition (SCADA) system [123] is one of them and the security of SCADA system is a key issue for current critical infrastructure environment. This is a new research area for use of intrusion detection system in industrial environment. RePIDS is payload-based real-time detector and can protect networks from web-based attacks, as we have demonstrated through experiments. We intend to extend our real-time payload-based intrusion detection system for monitoring and securing of industrial control systems. To secure the control process (network traffic), we will develop a prototype on content-based intrusion detection system integrating Modbus protocol [124], which is an application layer messaging protocol of OSI model.

References

1. Allen, J.H. *Cert system and network security practices*. 2001.
2. Center, C.C., *CERT advisory CA-1993-17 xterm logging vulnerability*. URL http://www. cert. org/advisories/CA-1993-17. html, 1993.
3. Center, C.E.O.C.S.R. and C.M. CyLab, *CYBERSECURITY RELATED INTERNET SOURCES.*
4. Caldwell, T., *Ethical hackers: putting on the white hat.* Network Security. 2011(7): p. 10-13.
5. Gourd, J. *Cyber Storm: The Culmination of an Undergraduate Course in Cyber Security*. in International Conference on Security & Management, SAM 2010. 2010. Las Vegas Nevada, USA: CSREA Press.
6. Gragido, W., *Beyond zero: analysing threat trends.* Network Security. 2011(7): p. 7-9.
7. McHugh, J., *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, 2000. 3(4): p. 262-294.
8. Moore, D., et al. *Internet quarantine: Requirements for containing self-propagating code*. in *INFOCOM*. 2003: Ieee.
9. Christodorescu, M. and S. Jha, *Static Analysis of Executables to Detect Malicious Patterns.* Sciences, New York, 2003. 8(3): p. 169-186.
10. Toth, T. and C. Kruegel. *Accurate buffer overflow detection via abstract pay load execution*, pp. 274-291, RAID 2002: Springer.
11. Sekar, R., et al. *Specification-based anomaly detection: a new approach for detecting network intrusions*. p. 265-274, 2002: ACM.
12. Axelsson, S., *Intrusion detection systems: A survey and taxonomy*. 2000, Technical report.

13. Gupta, K., et al. *Attacking confidentiality: An agent based approach*. in Intelligence and Security Informatics. LNCS 2006: Springer Verlag, London
14. Nascimento, G.M.B.A., *ANOMALY DETECTION OF WEB-BASED ATTACKS.2010.*
15. Kruegel, C. and G. Vigna. *Anomaly detection of web-based attacks*. in 10th ACM conference on Computer and communications security 2003. New York, NY, USA: ACM.
16. http://isc.sans.org/index.php?on=toptrends. *SANS Institute - Internet Storm Center web site.* . 2011 [cited.
17. Anderson, J.P., *Computer security threat monitoring and surveillance*. Technical Report, 1980. p. 56.
18. Kruegel, C., F. Valeur, and G. Vigna, *Intrusion detection and correlation: challenges and solutions*. Vol. 14. 2005: Springer-Verlag New York Inc.
19. Cheswick, W.R., S.M. Bellovin, and A.D. Rubin, *Firewalls and Internet security: repelling the wily hacker*. 2003: Addison-Wesley Longman Publishing Co., Inc.
20. Mell, R., *Intrusion detection systems.* National Institute of Standards and Technology (NIST), Special Publication, 2001. 51.
21. Schneier, B., *Applied cryptography: protocols, algorithms, and source code in C*. 2007: A1bazaar.
22. Debar, H., M. Dacier, and A. Wespi, Towards a taxonomy of intrusion-detection systems. *Computer Networks,* 1999. 31(8): p. 805-822.
23. Debar, H., M. Dacier, and A. Wespi, A revised taxonomy for intrusion-detection systems. *Annals of Telecommunications*, 2000. **55**(7): p. 361-378.
24. Denning, D.E., An intrusion-detection model. Software Engineering, *IEEE Transactions on Software Engineering*, 1987(2): p. 222-232.
25. Vigna, G. and C. Kruegel, *Host-based intrusion detection.* Handbook of Information Security. John Wiley and Sons, 2005.
26. Mischel, M., *Modsecurity 2.5*. 2009: Packt Pub.

27. Ghorbani, A.A., W. Lu, and M. Tavallaee, *Network intrusion detection and prevention: concepts and techniques*, ed. A.A.G.L. Tavallaee. Vol. 47. 2009: Springer-Verlag New York Inc.
28. Vigna, G. and R.A. Kemmerer, *NetSTAT:* A network-based intrusion detection system. *Journal of Computer Security*, 1999. **7**: p. 37-72.
29. Roesch, M. *Snort-lightweight Intrusion Detection for Networks*. in 13th USENIX Conference on System Administration, Seattle, Washington. 1999: Seattle, Washington.
30. Ptacek, T.H., *Insertion, evasion, and denial of service: Eluding network intrusion detection*. 1998, DTIC Document.
31. Tang, Y. and S. Chen, An automated signature-based approach against polymorphic internet worms. *IEEE Transactions on Parallel and Distributed Systems*, 2007. 18(7): p. 879-892.
32. Paxson, V. and M. Handley. *Defending against network IDS evasion*. 1999.
33. Chandola, V., A. Banerjee, and V. Kumar, *Anomaly detection: A survey*. ACM Computing Surveys (CSUR), 2009. 41(3): p. 15.
34. Lazarevic, A., et al. *A comparative study of anomaly detection schemes,* in network intrusion detection. in Third SIAM International Conference on Data Mining. 2003: SIAM.
35. Tombini, E., et al. *A serial combination of anomaly and misuse IDSes applied to HTTP traffic*. in IEEE 20th Annual Computer Security Applications Conference. 2004. Tucson, AZ, USA.
36. Gupta, K.K., B. Nath, and R. Kotagiri, Layered Approach using Conditional Random Fields for Intrusion Detection. I*EEE Transactions on Dependable and Secure Computing.* Vol. 7(1), p. 35 - 49, 2010.
37. Damashek, M., *Gauging similarity with n-grams: Language-independent categorization of text.* Science. 1995. 267(5199): p. 843.
38. Forrest, S., et al. *A sense of self for unix processes*. in *SP '96 IEEE Symposium on Security and Privacy* 1996: IEEE Computer Society Washington, DC, USA.
39. Hofmeyr, S.A., S. Forrest, and A. Somayaji, *Intrusion detection using sequences of system calls.* Journal of computer security, 1998. 6(3): p. 151-180.

40. Liao, Y. and V.R. Vemuri. *Using text categorization techniques for intrusion detection*. in Proceedings of the 11th USENIX Security. 2002: USENIX Association.
41. Dokas, P., et al. *Data mining for network intrusion detection*. 2002.
42. Fawcett, T., *An introduction to ROC analysis.* Pattern recognition letters, 2006. 27(8): p. 861-874.
43. Mukherjee, B., L.T. Heberlein, and K.N. Levitt, *Network intrusion detection.* Network, IEEE, 1994. 8(3): p. 26-41.
44. Estevez-Tapiador, J.M., P. Garcia-Teodoro, and J.E. Diaz-Verdejo, Anomaly detection methods in wired networks: a survey and taxonomy. *Computer communications,* 2004. 27(16): p. 1569-1584.
45. Patcha, A. and J.M. Park, An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer Networks*, 2007. **51**(12): p. 3448-3470.
46. Garcia-Teodoro, P., et al., Anomaly-based Network Intrusion Detection: Techniques, Systems and Challenges. *Computers & Security,* 2009. 28(1-2): p. 18-28.
47. Ye, N., B. Harish, and T. Farley, Attack profiles to derive data observations, features, and characteristics of cyber attacks. *Information-Knowledge-Systems Management,* 2005. 5(1): p. 23-47.
48. Smaha, S.E. *Haystack: An intrusion detection system*. in 4th Aerospace Computer Security Applications Conference. 1988. Orlando, FL: IEEE.
49. Lunt, T.F. *Real-time intrusion detection*. 1989:IEEE
50. Anderson, D., T. Frivold, and A. Valdes, *Next-generation intrusion detection expert system (NIDES): A summary*. 1995: SRI International, Computer Science Laboratory.
51. Kruegel, C., et al., *On the detection of anomalous system call arguments.* Computer Securityâ€“ESORICS 2003, p. 326-343.2003:IEEE.
52. Maxion, R.A. and F.E. Feather, A case study of ethernet anomalies in a distributed computing environment. *IEEE Transactions on Reliability,* 1990. 39(4): p. 433-443.

53. Mahoney, M. and P. Chan, *Detecting novel attacks by identifying anomalous network packet headers.* Florida Institute of Technology Technical Report CS-2001-2, 2001.
54. Mahoney, M. and P.K. Chan, *Learning models of network traffic for detecting novel attacks.* Florida Institute of Technology Technical Report CS-2002-08, 2002.
55. Mahoney, M.V. and P.K. Chan. *Learning nonstationary models of normal network traffic for detecting novel attacks*. in eighth ACM SIGKDD international conference on Knowledge discovery and data mining 2002: ACM.
56. Lee, W. and D. Xiang. *Information-theoretic measures for anomaly detection*. 2001.
57. Biles, S., *Detecting the unknown with snort and statistical packet anomaly detection engine (SPADE).* Computer Security Online Ltd., Tech. Rep, 2003.
58. Maggi, F., M. Matteucci, and S. Zanero, Detecting Intrusions through System Call Sequence and Argument Analysis. *IEEE Transactions on Dependable and Secure Computing*, 2010: p. 381-395.
59. Wattenberg-Simmross, F., et al., Anomaly Detection in Network Traffic Based on Statistical Inference and Alpha-Stable Modeling, *IEEE Transactions on Dependable and Secure Computing*, July 2011: p. 494-509.
60. Heckerman, D., *A tutorial on learning with bayesian networks.* Innovations in Bayesian Networks, 2008: p. 33-82.
61. Jolliffe, I.T. and MyiLibrary, *Principal component analysis*. Vol. 2. 2002: Wiley Online Library.
62. Wang, W. and R. Battiti. *Identifying intrusions in computer networks with principal component analysis*. in First International Conference on Availability, Reliability and Security, *ARES '06* 2006: IEEE Computer Society Washington, DC, USA.
63. Shyu, M.L., *A novel anomaly detection scheme based on principal component classifier*. 2003, DTIC Document.

64. Ye, N., Y. Zhang, and C.M. Borror, *Robustness of the Markov-chain model for cyber-attack detection.* IEEE Transactions on Reliability: 2004. 53(1): p. 116-123.

65. Zheng, Z., Z. Lan, and Y. Li, Toward Automated Anomaly Identification in Large-Scale Systems, *IEEE Transactions on Parallel and Distributed Systems*, 2010: p. 381-395.

66. Xiang, Y., K. Li, and W. Zhou, Low-Rate DDoS Attacks Detection and Traceback by Using New Information Metrics. *IEEE Transactions on Information Forensics and Security*, 2011. 6(2): p. 426-437.

67. Xiang, Y., W. Zhou, and M. Guo, *F*lexible Deterministic Packet Marking an IP Traceback System to Find the Real Source of Attacks. *IEEE Transactions on Parallel and Distributed Systems*, 2009. 20(4): p. 567-580.

68. Lee, W., S.J. Stolfo, and K.W. Mok. *Mining in a data-flow environment: experience in network intrusion detection*. in Fifth International Conference on Knowledge Discovery and Data Mining (KDD), ACM 1999. 1999: ACM.

69. Lee, W. and S.J. Stolfo, *A framework for constructing features and models for intrusion detection systems.* ACM Transactions on Information and System Security (TISSEC), 2000. 3(4): p. 227-261.

70. Lee, W., S.J. Stolfo, and K.W. Mok, *Adaptive intrusion detection: A data mining approach.* Artificial Intelligence Review, 2000. 14(6): p. 533-567.

71. BarbarÃ¡, D., et al., *ADAM: a testbed for exploring the use of data mining in intrusion detection.* Journal of ACM SIGMOD Record: Special Issue, 2001. **30**(4): p. 15-24.

72. Bridges, S.M. and R.B. Vaughn. *Fuzzy data mining and genetic algorithms applied to intrusion detection*. in *NISSC*. 2000.

73. Xin, J., J.E. Dickerson, and J.A. Dickerson. *Fuzzy feature extraction and visualization for intrusion detection*. 2003.

74. Kim, D.S., H.N. Nguyen, and J.S. Park. *Genetic algorithm to improve SVM based network intrusion detection system*. 2005.

75. Portnoy, L., E. Eskin, and S. Stolfo. *Intrusion detection with unlabeled data using clustering*. 2001: Citeseer.

76. Ramadas, M., S. Ostermann, and B. Tjaden. *Detecting anomalous network traffic with self-organizing maps*. in 6th International Symposium on RAID. 2003. Pitsburgh, PA, USA: Springer.
77. Ramadas, M., S. Ostermann, and B. Tjaden, eds. *Detecting Anomalous Network Traffic with Self-organizing Maps*. RAID 2003, LNCS 2820, pp. 36–54, 2003., ed. E.J. G. Vigna, and C. Kruegel 2003. 36–54.
78. Fossi, M., et al., *Symantec Internet Security Threat Report trends for 2010.* Volume XVI.
79. Kohonen, T., *The self-organizing map.* Proceedings of the IEEE, 1990. 78(9): p. 1464-1480.
80. Wang, K. and S.J. Stolfo. *Anomalous payload-based network intrusion detection*. in *RAID, Lecture Notes in Computer Science*. 2004: Springer.
81. Bolzoni, D., et al., *POSEIDON: A 2-Tier anomaly based intrusion detection system*, in Proceedings of the Fourth IEEE International Workshop on Information Assurance. 2006.
82. Wang, K., J. Parekh, and S. Stolfo. *Anagram: A content anomaly detector resistant to mimicry attack.* 2006: Springer.
83. Perdisci, R., et al., *McPAD: A multiple classifier system for accurate payload-based anomaly detection.* Computer Networks, 2009. 53(6): p. 864-881.
84. Rieck, K. and P. Laskov, *Language models for detection of unknown attacks in network traffic.* Journal in Computer Virology, 2007. 2(4): p. 243-256.
85. Bolzoni, D., B. Crispo, and S. Etalle. *ATLANTIDES: An architecture for alert verification in network intrusion detection systems*. in LISA'07 Proceedings of the 21st conference on Large Installation System Administration Conference 2007: Usenix Association.
86. Bolzoni, D. and S. Etalle, eds. *Approaches in anomaly-based network intrusion detection systems*. Intrusion Detection Systems, ed. L. Springer Verlag. Vol. 38. 2008. 1-16.
87. Utsumi, A. and N. Tetsutani. *Human detection using geometrical pixel value structures*. 2002: IEEE.

88. Theodoridis, S., et al., *Introduction to pattern recognition: a matlab approach*. 2009: Academic Pr.
89. Lamping, U. and E. Warnicke, *Wireshark User's Guide*. 2004, Recuperado el.
90. Mahoney, M. and P. Chan. *An analysis of the 1999 DARPA/Lincoln Laboratory evaluation data for network anomaly detection*. in Proceedings of Recent Advances in Intrusion Detection (RAID). 2003: Springer.
91. Kendall, K., *A database of computer attacks for the evaluation of intrusion detection systems*. 1999, Massachusetts Institute of Technology.
92. Fielding, R., et al., *Hypertext transfer protocol--HTTP/1.1*. 1999, RFC 2616, June.
93. Dhamankar, R., et al., *The top cyber security risks.* TippingPoint, Qualys, the Internet Storm Center and the SANS Institute faculty, Tech. Rep, 2009.
94. Balthrop, J., et al., *Technological networks and the spread of computer viruses.* Science, 2004. 304(5670): p. 527-529.
95. Di Lucca, G.A., et al. *Identifying cross site scripting vulnerabilities in web applications*. 2004: IEEE.
96. Lippmann, R., et al., The 1999 DARPA off-line intrusion detection evaluation. *Computer Networks*, 2000. 34(4): p. 579-595.
97. Ingham, K. and H. Inoue. *Comparing anomaly detection techniques for http*. in RAID'07, The 10th international conference on Recent advances in intrusion detection 2007: Springer.
98. Chen, Y., et al., eds. *Survey and taxonomy of feature selection algorithms in intrusion detection system*. Vol. 4318. 2006, Springer. 153-167.
99. Chen, C.M., Y.L. Chen, and H.C. Lin, An efficient network intrusion detection. *Computer communications*. 33(4): p. 477-484.
100. Singh, S. and S. Silakari, *Generalized Discriminant Analysis algorithm for feature reduction* in Cyber Attack Detection System. Arxiv preprint arXiv:0911.0787, 2009: p. 173-180.
101. Shih, H.C., et al. *Detection of Network Attack and Intrusion Using PCA-ICA.* in 3rd International Conference on Innovative Computing Information and Control, 2008. ICICIC '08. 2008. Dalian, Liaoning: IEEE.

102. Venkatachalam, V. and S. Selvan, *Performance comparison of Intrusion detection system classifiers using various feature reduction techniques.* International journal of simulation, 2008. 9(1): p. 30-39.
103. Mahoney, M.V. *Network traffic anomaly detection based on packet bytes*. in Proceedings of the 2003 ACM symposium on Applied computing 2003. New York, NY, USA.
104. McLachlan, G.J. and J. Wiley, *Discriminant analysis and statistical pattern recognition*. 1992: Wiley Online Library.
105. Bishop, C.M. and SpringerLink, *Pattern recognition and machine learning*. 1st ed. Information Science and Statistics. Vol. 4. 2006: springer New York.
106. Ringberg, H., et al., *Sensitivity of PCA for traffic anomaly detection.* ACM SIGMETRICS Performance Evaluation Review, 2007. 35(1): p. 109-120.
107. Isabelle, G. and E. Andre, An introduction to variable and feature selection. *Journal of Machine Learning Research*, 2003. 3(1): p. 1157-1182.
108. Chebrolu, S., A. Abraham, and J.P. Thomas, *Feature deduction and ensemble design of intrusion detection systems.* Computers & Security, 2005. 24(4): p. 295-307.
109. Suebsing, A. and N. Hiransakolwong. *Euclidean-based Feature Selection for Network Intrusion Detection*. in *I*nternational Conference on Machine Learning and Computing, 2009: IACSIT Press.
110. Summerville, D.H., N. Nwanze, and V.A. Skormin. *Anomalous packet identification for network intrusion detection*. 2004: IEEE.
111. Yu, H. and J. Yang, A direct LDA algorithm for high-dimensional data-with application to face recognition. *Pattern Recognition*, 2001. 34(10): p. 2067.
112. Sharma, S., et al. *Feature extraction using non-linear transformation for robust speech recognition on the Aurora database*. 2000: IEEE.
113. Chung, S. and A. Mok. *Allergy attack against automatic signature generation*. 2006: Springer.
114. Düssel, P., et al., *Cyber-critical infrastructure protection using real-time payload-based anomaly detection*. Lecture Notes in Computer Science,Critical Information Infrastructures Security. Vol. 6027. 2010: Springer. p. 85-97.

115. Martinez, A.M. and A.C. Kak, *PCA versus LDA.* Pattern Analysis and Machine Intelligence, *IEEE Transactions on*, 2001. 23(2): p. 228-233.
116. Bouzida, Y. and S. Gombault. *Intrusion detection using principal component analysis*. in Seventh multi-conference on Systemics, Cybernetics and Informatics. 2003. Orlando, Florida, USA: Citeseer.
117. Jamdagni, A., et al. *Intrusion Detection Using Geometrical Structure*. in Fourth International Conference on Frontier of Computer Science and Technology, 2009. FCST '09. 2009. China: IEEE Computer Society Washington, DC, USA
118. Cattell, R.B., *The scree test for the number of factors.* Multivariate behavioral research, 1966. 1(2): p. 245-276.
119. Franklin, S.B., et al., Parallel analysis: a method for determining significant principal components. *Journal of Vegetation Science*, 1995. 6(1): p. 99-106.
120. Cattell, R.B. and S. Vogelmann, *A comprehensive trial of the scree and KG criteria for determining the number of factors.* Multivariate behavioral research, 1977. 12(3): p. 289-325.
121. Nelson, L.R., *Some observations on the scree test, and on coefficient alpha.* Thai Journal of Educational Research and Measurement. 2005. 3(1): p. 1-17.
122. Chung, S. and A. Mok. *Advanced Allergy Attacks: Does a Corpus Really Help?* 2007: Springer.
123. Rrushi, D. and U. di Milano. *SCADA Intrusion Prevention System*. 2006.
124. Xiaoxiang, Z., *Modbus Protocol and Programing.* Electronic Engineer, 2005.