MAXIMUM ENTROPY ESTIMATED PAYLOAD BASED INTRUSION DETECTION SYSTEM
"THE ME-PAYL"

by

Derya ERHAN

B.S., Electrical-Electronics Engineering, Hacettepe University, 2003

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Electrical and Electronics Engineering

Boğaziçi University

2007

ACKNOWLEDGEMENTS

Firstly, I would like to thank to my thesis supervisor Prof. Dr. Emin Anarım for his kind interest, technical advices and his patience.

Additionally, I would like to thank to Assist. Prof. Dr. Frédéric Kerem Harmancı for his kind interest, technical advices and his patience.

Finally, I would like to thank to my husband Emre ERHAN for his patience.

This work is supported by the State Planning Organization of Turkey under the Next Generation Satellite Networks Project, DPT 03K 120250.

ABSTRACT

MAXIMUM ENTROPY ESTIMATED PAYLOAD BASED INTRUSION DETECTION SYSTEM 'THE ME-PAYL"

Computer Networks can be considered as an important component of today's human life. Since data and information of various organizations are transferred through private and public networks such as the global internet, special attention is being paid to the security parameters of these networks. In order to increase the security of these networks, tools such as firewalls and intrusion detection systems are used.

The process of monitoring the events occurring in a computer system or network and analyzing them for sign of intrusions is known as Intrusion Detection System.

In this thesis a payload based intrusion detection system using the maximum entropy principle, the Me-PAYL is proposed. The starting point is the PAYL method. A network anomaly detection technique that uses sniffed data of the network and based on maximum entropy and relative entropy methods is developed. Advantages of maximum entropy approach are combined with PAYL model to obtain more efficiency.

The proposed method, Me-PAYL is tested with DARPA 1999 Intrusion Detection Evaluation (IDEVAL) Dataset, which is the largest dataset available with whole payloads.

When comparing results of PAYL and Me-PAYL with tests on the IDEVAL dataset, it can be seen that the Me-PAYL method is much more efficient than the PAYL method.

ÖZET

MAKSİMUM ENTROPİ KESTİRİMLİ PAYLOAD TABANLI SALDIRI TESPİT SİSTEMİ
"THE ME-PAYL"

Bilgisayar ağları günümüz insan yaşamının önemli bir parçası olarak görülebilir. Çeşitli kuruluşların veri ve bilgileri özel ve genel ağlar tarafından iletilmekte olduğundan bu ağların güvenlik parametrelerine olan giderek artan özel bir ilgi oluşmuştur. Bu ağların güvenliğini artırmak için güvenlik duvarı ve saldırı tespit sistemleri gibi araçlar kullanılır.

Saldırıların belirtilerini bulabilmek için bir bilgisayar sistemi veya ağdaki olayları izlemek ve analiz etmek saldırı tespit sistemi olarak adlandırılır.

Bu tezde maksimum entropi prensibini kullanan payload tabanlı bir saldırı tespit sistemi, Me-PAYL önerilmiştir. Başlangıç noktası PAYL metodudur. Ağdan izlenmiş verileri kullanan maksimum entropi ve göreli entropyiye dayalı bir ağ tabanlı tespit sistemi geliştirilmiştir. Daha fazla verim almak için maksimum entropi yaklaşımının avantajları PAYL modeli ile birleştirilmiştir.

Önerilen metot, Me-PAYL, bütün payloadlar içeren kullanılabilir en büyük veri kümesi olan, DARPA 1999 Intrusion Detection Evaluation (IDEVAL) veri kümesi ile test edilmiştir.

PAYL ve Me-PAYL metotlarının IDEVAL veri kümesi ile olan test sonuçlarını karşılaştırdığımızda, Me-PAYL metodunun çok daha verimli olduğu görülebilir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS / ABBREVIATIONS

$d(x, \bar{y})$	Maholonobis Distance Between Two Vectors x and y
f_i	i^{th} Feature Function
$D_{\tilde{P}\|P}(w)$	Kullback-Leibler Distance
$\tilde{P}(w)$	Empirical Distribution of Byte Classes
$P(w)$	Maximum Entropy Estimation
$P_0(w)$	Initial Density Model
$\bar{x}(w)$	Byte Frequency of Empirical Distribution
Ω	Set of Byte Classes
Λ	Set of Parameters
$Var(X)$	Variance of X

ACK	Acknowledgement
AFRL	Air Force Research Laboratory
AR	Auto-Regressive
ARIMA	Autoregressive Integrated Moving Average
ARP	Address Resolution Protocol
ASCII	American Standard Code for Information Interchange
B	Base rate
BSM	Solaris Basic Security Module
BSM	Solaris system call
C_{ID}	Intrusion detection capability
CPM	Change-Point Monitoring
CPU	Central Processing Unit
CUSUM	Cumulative Sum
DARPA	Defense Advanced Research Projects Agency
DDoS	Distributed Denial of Service
DNS	Domain Name Service
DoS	Denial of Service

EOL	End of Letter
FFT	Fast Fourier Transform
FIN	Finish
FN	False Negative
FP	False Positive
FTP	File Transfer Protocol
HIDS	Host based Intrusion Detection System
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IDEVAL	Intrusion Detection Evaluation
IDS	Intrusion Detection System
IP	Internet Protocol
IT	Information Technologies
KDD	Knowledge-Discovery in Databases
ME	Principle of Maximum Entropy
Me-PAYL IDS	Maximum Entropy Estimated Payload Based Intrusion Detection System
MIB	Management Information Base
MIT	Massachusets Institute of Technology
NIDS	Network based Intrusion Detection System
NPV	Negative Predictive Value
OSPF	Open Shortest Path First
PAYL	Payload Based Anomaly Detector
PCA	Principal Components Analysis
POSEIDON	Payl Over Som for Intrusion DetectiON
PPV	Positive Predictive Value or Bayesian Detection Rate
RIP	Routing Information Protocol
ROC	Receiver Operating Characteristic
RST	Reset
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SOM	Self Organizing Map
SSH	Secure Shell Protocol

SYN	Synchronize
TCP	Transmission Control Protocol
TELNET	Terminal Network Protocol
TN	True Negative
TP	True Positive
UDP	User Datagram Protocol
URG	Urgent Pointer

1. INTRODUCTION

In this chapter, general information about the concept of computer intrusions and intrusion detection systems and datasets used in this thesis work will be given. Also this chapter includes the purpose and the motivation of this thesis.

1.1. Computer Security

Computer security is defined as the protection of computing systems against threats to confidentiality, integrity, and availability [1]. Confidentiality (or secrecy) means that information is disclosed only according to policy. Integrity means that information is not destroyed or corrupted and that the system performs correctly. Availability means that system services are available when they are needed.

Security threats come from different sources such as natural forces (such as food), accidents (such as fire), failure of services (such as power) and people known as intruders. There are two types of intruders: the external intruders who are unauthorized users of the machines they attack, and internal intruders, who have permission to access the system with some restrictions.

In general speaking; an intrusion is a violation of the security policy of the system. This includes a deliberate unauthorized attempt to access information, manipulate information, or render a system unreliable or unusable. An attacker can gain illegal access to a system by fooling an authorized user into providing information that can be used to break into a system. An attacker can deliver a piece of software to a user of a system which is actually a Trojan horse containing malicious code that gives the attacker system access. Bugs in trusted programs can be exploited by an attacker to gain unauthorized access to a computer system. There are legitimate actions that one can perform, can lead to system failure. An attacker can gain access because of an error in the configuration of a system. In some cases it is possible to fool a system into giving access by misrepresenting oneself. An

example is sending a TCP packet that has a forged source address that makes the packet appear to come from a trusted host.

Intrusions are classified as six types.

- Attempted breakins, which are detected by typical behavior profiles or violations of security constraints.
- Masquerade attacks, which are detected by a typical behavior profiles or violations of security constraints.
- Penetration of the security control system, which are detected by monitoring for specific patterns of activity.
- Leakage, which is detected by a typical use of system resources.
- Denial of service, which is detected by a typical use of system resources.
- Malicious use, which is detected by a typical behavior profiles, violations of security constraints, or use of special privileges. [2]

As mentioned in the previous section there is a large variety of attacks exists in computer networks. Attacks can also be categorized in two areas as passive and active attacks. Passive attacks are aimed at gaining access to penetrate the system without compromising IT (Information Technology) resources. Active attacks result in an unauthorized state change of IT resources. In terms of the relation intruder-victim, attacks are categorized as, internal and external attacks. Internal attacks come from own enterprise's employees or their business partners or customers. External attacks come from outside of the enterprise network, frequently via the Internet. Attacks are also identified by the source category, namely those performed from internal systems (local network), the Internet or from remote dial-in sources [3].

The traditional prevention techniques such as user authentication, data encryption, avoiding programming errors and firewalls are used as the first line of defense for computer security. If a password is weak and is compromised, user authentication cannot prevent unauthorized use, firewalls are vulnerable to errors in configuration and ambiguous or undefined security policies. They are generally unable to protect against malicious mobile code, insider attacks and unsecured modems. Programming errors cannot be

avoided as the complexity of the system and application software is changing rapidly leaving behind some exploitable weaknesses. Intrusion detection is therefore required as an additional wall for protecting systems. Intrusion detection is useful not only in detecting successful intrusions, but also provides important information for timely countermeasures [2].

1.2. Intrusion Detection

The process of monitoring the events occurring in a computer system or network and analyzing them for sign of intrusions is known as Intrusion Detection System (IDS) [4].

Intrusion detection is classified into two types: misuse intrusion detection and anomaly intrusion detection. Misuse intrusion detection uses well defined patterns of the attack that exploit weaknesses in system and application software to identify the intrusions. These patterns are encoded in advance and used to match against the user behavior to detect intrusion. Anomaly intrusion detection uses the normal usage behavior patterns to identify the intrusion. The behavior of the user is observed and any deviation from the constructed normal behavior is detected as intrusion. Normal usage patterns can be constructed with various methods using data that indicate some normal behavior of normal activity.

Intrusion detection systems can also be categorized according to their origin of detection. These are network intrusion detection system which detects intrusion from whole network data; and host intrusion detection systems which detect intrusions by examining activities of host systems in network. There is also hybrid systems which use both the network data and data obtained from hosts.

In this thesis a network anomaly detection intrusion detection system is proposed. A method that models normal usage from payload of TCP packets, and detect deviations of TCP payloads in test data from constructed model, is developed.

1.3. DARPA 1999 Intrusion Detection Evaluation (IDEVAL) Dataset

As different methods and various new systems are developed for intrusion detection, it is clearly advantageous to have a means of evaluating the success of these systems in detecting attacks. The best environment for testing and evaluation of an intrusion detection system is the actual environment in which it will be used. However, often it is hard to obtain reliable data from operational networks on which to test intrusion detection systems. Because of this fact these systems are tested in a simulated environment. The ability to perform accurate testing and evaluation in a simulated environment requires high-quality data that is similar to the traffic (including attacks) that one finds on operational networks. In general, this data is difficult to acquire because it contains private information and reveals potential vulnerabilities of the networks from which the data is collected. These factors led to DARPA sponsorship of MIT Lincoln Laboratory's intrusion detection evaluation dataset [5].

Before 1999 evaluation, MIT Lincoln Laboratory generated 1998 Dataset. The DARPA 1998 Intrusion Detection Evaluation was an initial attempt to perform a comprehensive technical evaluation of intrusion detection technology. This evaluation had limited goals. It was designed to evaluate only DARPA funded intrusion detection technology, and not complete deployable intrusion detection systems or commercial systems. It was also designed to measure false alarm rates using background traffic similar to that on one Air Force base and to measure detection rates of remotely-initiated attacks against UNIX hosts [6].

There are several improvements in 1999 evaluation. Also the 1999 evaluation is the most complete dataset with full payload publicly available for experimental use [7]. Because of these reasons 1999 evaluation dataset is used in this thesis for benchmarking maximum entropy estimation payload based intrusion detection system (Me-PAYL IDS) method.

1.3.1. Properties of DARPA 1999 Intrusion Detection Evaluation dataset

The 1999 evaluation was a blind off-line evaluation, as in 1998, but modified based on suggestions from 1998 and also with major extensions to enhance the analysis and cover more attack types. Figure 1.1 shows a block diagram of the 1999 test bed. Major changes for 1999 are the addition of a Windows NT workstation as a victim, the addition of an inside tcpdump sniffer machine, and the collection of both Windows NT audit events and inside tcpdump sniffing data for inclusion in archival data provided to participants.

As shown in this Figure 1.2 there are new Windows NT workstations added to support NT attacks, new inside attacks, and new stealthy attacks designed to avoid detection by network-based systems tested in 1998. Inside attacks and inside sniffer data to detect these attacks were added due the dangers posed by inside attacks. Stealthy attacks were added due to an emphasis on sophisticated attackers who can carefully craft attacks to look like normal traffic. In addition, two new types of analyses were performed. First, an analysis of misses and high-scoring false alarms was performed for each system to determine why systems miss specific attacks and what causes false alarms. Second, participants were optionally permitted to submit attack forensic information that could help a security analyst identify important characteristics of the attack and respond. This identification information included the attack category, the name for old attacks, ports/protocols used, and IP addresses used by the attacker. Another major change in 1999 was a focus on determining the ability of systems to detect new attacks without first training on instances of these attacks. 1999 evaluation was designed to evaluate enhanced systems which can detect new attacks and to analyze why systems miss new attacks. Many new attacks were thus developed and only examples of a few of these were provided in training data [6].

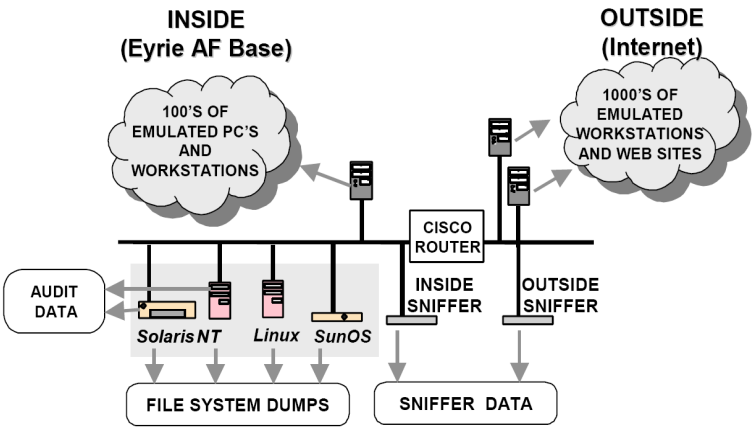

Figure 1.1. Block diagram of DARPA 1999 testbed [6]

The inside of the simulated Eryie Air Force base shown in Figure 1.2 contains four victim machines which are the most frequent targets of attacks in the evaluation (Linux 2.0.27, SunOS 4.1.4, Sun Solaris 2.5.1, Windows NT 4.0), a sniffer to capture network traffic, and a gateway to hundreds of other inside emulated PCs and workstations. The outside simulated internet contains a sniffer, a gateway to hundreds of emulated workstations on many other subnets and a second gateway to thousands of emulated web servers.

Data collected to evaluate intrusion detection systems include network sniffing data from the inside and outside sniffers. We use inside sniffed data for testing of our method. Solaris Basic Security Module (BSM) audit data collected from the Solaris host, Windows NT audit event logs collected from the Windows NT host, nightly listings of all files on four victim machines, and nightly dumps of security-related files on all victim machines.

Custom software automata in the test bed simulate hundreds of programmers, secretaries, managers, and other types of users running common UNIX and Windows NT application programs. In addition, custom Linux kernel modifications provided by the AFRL allow a small number of actual hosts to appear as if they are thousands of hosts with different IP addresses. Figure 1.3 shows the average number of connections per day for the

most common TCP services. As can be seen, web traffic dominates but many other types of traffic are generated which use a variety of services.

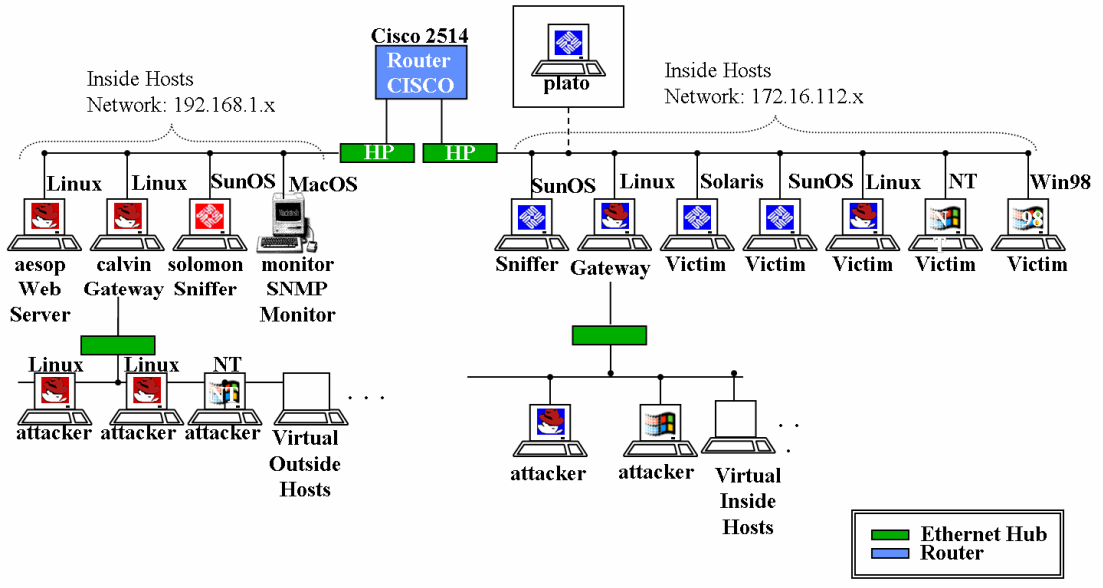

Figure 1.2. Simulation topology of DARPA 1999 IDS evaluation dataset.

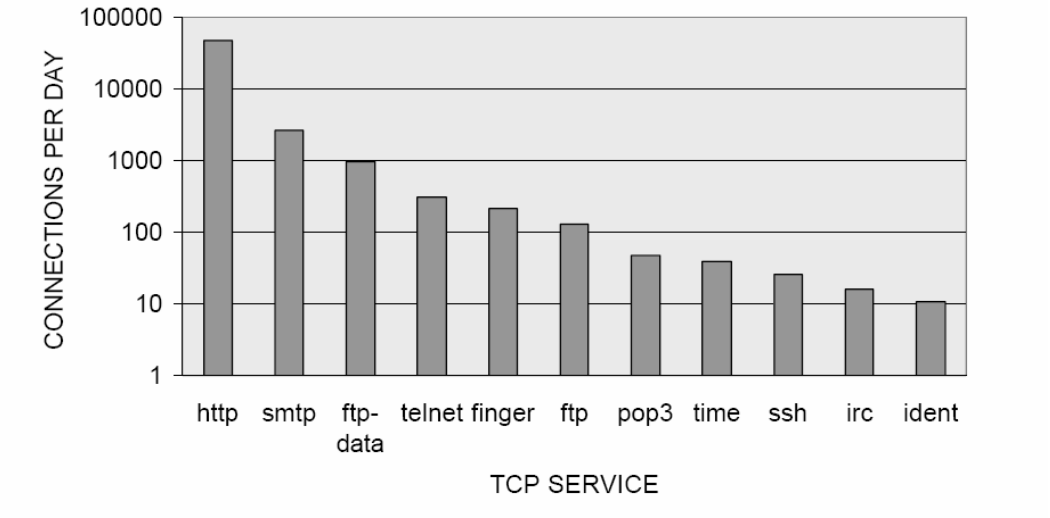

Figure 1.3. Average connections per day for dominant TCP services [5].

User automata send and receive mail, browse web sites, send and receive files using FTP, use telnet and ssh to log into remote computers and perform work, monitor the router remotely using SNMP, and perform other tasks. In addition to automatic traffic, the test bed allows human actors to generate background traffic and attacks when the traffic or attack is too complex to automate. Background traffic characteristics including the overall traffic level, the proportion of traffic from different services, and the variability of traffic with time of day are similar to characteristics measured on a small Air Force base in 1998. The average number of background-traffic bytes transmitted per day between the inside and outside of this test bed is roughly 411 Mbytes per day, with most of the traffic concentrated between 8:00 AM and 6:00 PM. The dominant protocols are TCP (384 Mbytes), UDP (26 Mbytes), and ICMP (98 Kbytes).

Details of attacks in this dataset are explained briefly in next section. Also detailed explanation about all attacks in this dataset can be found in [5]. Our anomaly detection method is responsible only detection of attacks that have information in payloads of packets. There are 97 instances in a total of 201 attack instances that we concern. Also only dominant TCP protocols (can be seen in Figure 1.2) http, smtp, tenet and ftp data are examined. All results and conclusions are depending on this condition.

1.3.2. Attacks in DARPA 1999 dataset

In DARPA 1999 Dataset, 201 instances of 56 attack types were launched against victim UNIX and Windows NT hosts in three weeks of training data and two weeks of test data. These 56 different attack types (shown in Table 1.1 and Table 1.2) were used in the evaluation. Details on attacks including further references and information on implementations are available in [5].

Five major attack categories and the attack victims are shown in Table 1.1 and Table 1.2. Primary victims listed along the top of these tables are the four inside victim hosts, shown in the gray box of Figure 1.1, and the Cisco router. In addition, some probes query all machines in a given range of IP addresses as indicated by the column labeled “all” in Table 1.1.

Table 1.1. Probe and Denial of Service (DoS) attacks [6].

	Solaris	SunOS	Windows NT	Linux	All
Probe (37)	portsweep queso	portsweep queso	ntinfoscan portsweep	lsdomain mscan portsweep queso satan	illegal-sniffer ipsweep portsweep
DoS (65)	neptune pod processtable selfping smurf syslogd tcpreset warezclient	arpoison land mailbomb neptune pod processtable	arppoison crashiis dosnuke smurf tcpreset	apache2 arppoison back mailbomb neptune pod processtable smurf tcpreset teardrop udpstorm	

The upper row of Table 1.1 lists probe or scan attacks. These attacks automatically scan a network of computers or a DNS server to find valid IP addresses (ipsweep, lsdomain, mscan), active ports (portsweep, mscan), host operating system types (queso, mscan), and known vulnerabilities (satan). All of these probes except two (mscan and satan) are either new in 1999 (e.g. ntinfoscan, queso, illegalsniffer) or are stealthy versions of 1999 probes (e.g. portsweep, ipsweep). Probes are considered stealthy if they issue ten or fewer connections or packets or if they wait longer than 59 seconds between successive network transmissions. Stealthy probes are similar to clear probes because they gather similar information concerning IP addresses, vulnerable ports, and operating system types. They differ because this information is gathered at a slower rate and because less but more focused information is gathered from each attack instance. For example, stealthy port sweeps are slow and focus only on ports with known vulnerabilities. The new

“illegalsniffer” attack is different from the other probes. During this attack, a Linux sniffer machine is installed on the inside network running the tcpdump program in a manner that creates many DNS queries from this new and illegal IP address.

Table 1.2. Remote to Local (R2L), User to Root (U2R), and Data attacks. [6]

	Solaris	SunOS	NT	Linux	Cisco
R2L (56)	dict ftpwrite guest httptunnel xlock xsnoop	dict xsnoop	dict framespoof netbus netcat ppmacro	dict imap named ncftp phf sendmail sshtrojan xlock xsnoop	snmpget
U2R (37)	eject fdformat ffbconfig ps	loadmodule	casesen ntfsdos nukepw sechole yaga	perl sqlattack xterm	
DATA (13)	secret		ntfsdos ppmacro	secret sqlattack	

The second row of Table 1.1 contains denial of service (DoS) attacks designed to disrupt a host or network service. New 1999 DoS attacks crash the Solaris operating system (selfping), actively terminate all TCP connections to a specific host (tcpreset), corrupt ARP cache entries for a victim not in others’ caches (arppoison), crash the Microsoft Windows NT web server (crashiis), and crash Windows NT (dosnuke).

Detection method, attack signature and methods of generation of attacks in dataset are explained in [5]. From given information about attacks, it can be found the attack

instances that are expected to be detected with payload based IDS methods. The attacks and their categories according to detection methods are listed in Table 1.3.

In table 1.3 fist row contains the attacks that is expected to be detected with Me-PAYL method. In attack descriptions in [5], if the attack can be detected from sniffed data it is assumed to be detectable with payload based IDS method. From 201 attack instances in whole dataset, 97 of them can be detectable with payload based methods [7]. Detections of Me-PAYL method for all 97 instances of attacks can be found in section 5.3.

Second row of Table 1.3 shows attacks which are not considered in performance analysis; and third row contains the reason of this. For example attacks arppoison, Ping of Death and selfping are ICMP attacks. These attacks contain no information about themselves in payloads of generated packets. This means they can not be detected with payload based methods that examine TCP payloads of packets.

Since Me-PAYL method is a network IDS it can not detect attacks that do not create network traffic. Also insidesniffer which is passive attack can not be detected by network any network IDS. Because passive attacks do not make any change in network traffic.

1.4. Motivation

There are many IDS systems available and most of them are primarily signature-based detectors. Although these are effective at detecting known intrusion attempts and exploits, they fail to recognize new attacks and carefully crafted variants of old exploits. A new generation of systems is now appearing based upon anomaly detection. Anomaly Detection systems model normal or expected behavior in a system, and detect deviations of interest that may indicate a security breach or an attempted attack.

Some attacks exploit the vulnerabilities of a protocol; other attacks seek to survey a site by scanning and probing. These attacks can often be detected by analyzing the network packet headers, or monitoring the network traffic connection attempts and session behavior. Other attacks, such as worms, involve the delivery of bad payload (in an otherwise normal connection) to a vulnerable service or application. These may be

detected by inspecting the packet payload (or the ill-effects of the worm payload execution on the server when it is too late after successful penetration). Network anomaly IDS methods, that uses header information or misuse IDS systems can not be useful on detecting a zero-day worm propagation in network. In such cases payload based anomaly detection methods are much more practical.

The purpose of this thesis is to find out an effective payload based network anomaly detection method. Starting point of this work is PAYL model proposed in [7].

To improve detection capability of PAYL model as an addition to PAYL modeling algorithm maximum entropy principle is used. A network anomaly detection technique that uses sniffed data of network and based on maximum entropy and relative entropy methods is developed. Advantages of maximum entropy approach are combined with PAYL model to obtain more efficiency.

1.5. Thesis Outline

This thesis work contains combination of two methods to form a maximum entropy based payload model intrusion detection system (Me-PAYL). These are PAYL method and maximum entropy principle.

The thesis is organized in the following fashion. Chapter two makes introduction to Anomaly Detection Methods and give information about Payload Based IDS model. In chapter three useful information that will help reader to understand some terms like TCP/IP, maximum entropy modeling and payload based IDS method. In chapter four details about modeling and detection method will be given. Also with chapter four and chapter five all modeling and detection processes can be understood clearly. Chapter five also contains results in tables and figures to help understanding performance and efficiency of proposed IDS model. In the last chapter evaluation of results will be done. Also Me-PAYL model will be compared with PAYL [7] and POSEIDON [8] models.

Table 1.3. Categorized attacks of DARPA 1999 Dataset.

Detectable With Me-PAYL	Not Detectable With Me-PAYL	Explanation
Apache2 Back casesen Crashiis Eject Fdformat Ffbconfig framespoofer Ftpwrite guessftp Httptunnel Mailbomb Mscan ncftp netbus netcat_setup NTinfoscan Perl Phf ppmacro Process Table Ps queso Satan sechole secret Sendmail sshtrojaninstall warezclient warezmaster Xterm Dictionary yaga Loadmodule anypw	arppoison Ping of Death (POD) selfping Smurf Teardrop Ipsweep	These attacks include ICMP packets that don't contain any TCP Payload.
	Land SYN Flood (Neptune) Syslogd tcpreset Nmap resetscan	These attacks can be detected with header information.
	insidesniffer	Not Detectable
	Udpstorm Imap Named Xlock Xsnoop ls_domain Saint	These attacks contain non TCP packets or with TCP packets with destination port other than TCP 21, 23, 25 and 80.
	ntfsdos	Local attack(Don't create network traffic)
	Guest	Can be detected from audit log

2. ANOMALY DETECTION METHODS

In this section, the most commonly used network anomaly detection methods are reviewed. In decision of the anomaly detection method to survey in this work, these papers are examined. The methods described are rule-based approaches, pattern matching, Wavelet Models, and statistical analysis models.

2.1. Rule Based Models

Early work in the area of fault or anomaly detection was based on expert systems. In expert systems, an exhaustive database containing the rules of behavior of the faulty system is used to determine if a fault occurred. Rule-based systems are too slow for real-time applications and are dependent on prior knowledge about the fault conditions on the network [9]. Furthermore, the identification of relevant criteria for the different faults will, in turn, require a set of rules to be developed. In addition, using any functional approximation scheme, such as back propagation, causes an increase in computation time and complexity. The number of functions to be learned also increases with the number of faults studied [9].

Alhamaty et al concentrate on finding a solution to the intrusion detection main attacks of fragmentation information packets. Main idea is to check TCP packet integrity so as not to restrict the check attack special signature. This work focuses on the packet that whether packet rightly is fragmented or not, until by change attack signature [10].

2.2. Pattern Matching Models

The efficiency of this pattern matching approach depends on the accuracy of the traffic profile generated. Given a new network, it may be necessary to spend a considerable amount of time building traffic profiles. In the face of evolving network topologies and traffic conditions, this method may not scale gracefully [9]. The methods of data analysis

and pattern recognition presented are the basis of a technology study for an automatic intrusion detection system that detects the attack in the reconnaissance stage [11].

2.3. Multiscale and Multidimensional Analysis

These wavelet-based scaling analysis tools are incredibly useful for describing and detecting certain kinds of properties of one-dimensional functions, measures, or random processes. They can compute summary statistics about scale-dependent properties, local scaling behavior, and even extremely localized information about the local regularity of network traffic. Because these methods can be implemented in an on-line fashion, thet use them to monitor network links, either at one main link or at many access points. However, these tools do have some serious drawbacks when it comes to the next step in network measurements. A. C. Gilbert works only with information local to a network. Gilbert cannot address distributed network measurements at all [12]

An approach for real-time network monitoring in terms of numerical time-dependant functions of protocol parameters was suggested in [11]. Gudkov et al have applied complex systems theory for information flow analysis of networks, the information traffic is described as a trajectory in multi-dimensional parameter-time space with about 10-12 dimensions. The network traffic description is synthesized by applying methods of theoretical physics and complex systems theory, to provide a robust approach for network monitoring that detects known intrusions, and supports developing real systems for detection of unknown intrusions [11].

2.4. Wavelet Models

The link loads and traffic matrices are simply related by a linear equation $b = Ax$ The vector b contains the link measurements, and A is the routing matrix. They wish to infer x, which contains the unknown traffic matrix elements written as a vector. Tomographic inference techniques seek to invert this relationship to find x. Two basic solution strategies to network anomography: (i) early inverse, and (ii) late inverse. Early inverse approaches may appear more intuitive. The early inverse approach tackles the problem in two steps. The first is the network tomography step, where OD (Origin Destination) flow data at each

interval j are inferred from the link load measurements by solving the ill-posed linear inverse problem. Given the estimated OD flow data xj at different time points j, in the second step, anmaly detection can then be applied to the xj [12].

In [12] Zhang et al have showed that the ARIMA (Autoregressive Integrated Moving Average) methods, FFT (Fast Fourier Transform) and Wavelet anomography approaches have superb performance the number of false negatives is very low. This indicates that very few important traffic anomalies can pass undetected by these approaches. The PCA based approaches, however, identify about half of the anomalies [12].

In [13], Huang et al apply signal processing techniques in intrusion detection systems, and develop and implement a framework, called Waveman, for real time wavelet-based analysis of network traffic anomalies. Then, they use two metrics, namely percentage deviation and entropy, to evaluate the performance of various wavelet functions on detecting different types of anomalies like Denial of Service (DoS) attacks and portscans. Results show that Coiflet and Paul wavelets perform better than other wavelets in detecting most anomalies considered in this work [13].

Inspired by the methods that use the selfsimilarity property of data network traffic as normal behavior and any deviation from it as the anomalous behavior, In [14], Rawat et al have proposed a method for anomaly based network intrusion detection. Making use of the relations present among the wavelet coefficients of a self-similar function in a different way, method determines the possible presence of not only an anomaly, but also its location in the data. They provide the empirical results on KDD (Knowledge-Discovery in Databases) data. Hurts parameter was used to perform anomaly detection [14].

In [15], Kim et al suggest a technique for traffic anomaly detection based on analyzing correlation of destination IP addresses in outgoing traffic at an egress router. This address correlation data are transformed through discrete wavelet transform for effective detection of anomalies through statistical analysis. Results from trace-driven evaluation suggest that proposed approach could provide an effective means of detecting anomalies close to the network [15]. Based on statistical bounds on normal traffic patterns of the correlation signal of destination addresses, sudden changes can be used to detect

anomalies in traffic behavior. A correlation calculation is using a simple data structure. These correlation data are processed through coefficient selective discrete wavelet transform for effective and high-confidence detection [16].

2.5. Image Based Anomaly Detection

In [17] NetViewer was introduced a network measurement approach that can simultaneously detect, identify and visualize attacks and anomalous traffic in real-time by passively monitoring packet headers. Kim et al propose to represent samples of network packet header data as frames or images. With such a formulation, a series of samples can be seen as a sequence of frames or video, revealing certain kinds of attacks to the human eye. This enables techniques from image processing and video compression to be applied to the packet header data to reveal interesting properties of traffic. They show that "scene change analysis" can reveal sudden changes in traffic behavior or anomalies. They also show that "motion prediction" techniques can be employed to understand the patterns of some of the attacks. They show that it may be feasible to represent multiple pieces of data as different colors of an image enabling a uniform treatment of multidimensional packet header data [17].

Their approach passively monitors packet headers of network traffic at regular intervals and analyzes the aggregate data for anomaly detection. Our approach generates images of the packet header data for both visualization and for effective processing of the collected data. During network anomalies or attacks, the usage pattern of network may change and the peculiarities could become visible in the traffic images. When anomalies are detected, further analysis can characterize the anomalies by their nature into several categories and help in mitigating the attacks [17].

2.6. Change Point Detection

Wang et al present a simple mechanism, called Change-Point Monitoring (CPM), to detect denial of service (DoS) attacks. The core of CPM is based on the inherent network protocol behaviors and is an instance of the Sequential Change Point Detection. To make the detection mechanism insensitive to sites and traffic patterns, a nonparametric

Cumulative Sum (CUSUM) method was applied, thus making the detection mechanism robust, more generally applicable, and its deployment much easier [18].

CPM compares the observed sequence with the profile that represents the user's normal behavior and detects any significant deviation from the normal behavior. The key difference of CPM from others is that CPM exploits the inherent network protocol behaviors, instead of traffic patterns, for detecting network anomalies. The objective of Change-Point Detection is to determine if the observed time series is statistically homogeneous and, if not, to find the point in time when the change happens [18].

2.7. PCA

Labib et al have proposed a multivariate statistical method called Principal Component Analysis is used to detect Denial-of-Service and Network Probe attacks using the 1998 DARPA (Defense Advanced Research Projects Agency) data set. Visualization of network activity and possible intrusions is achieved using Bi-plots, which are used as a graphical means for summarizing the statistics. The principal components are calculated for both attack and normal traffic, and the loading values of the various feature vector components are analyzed with respect to the principal components. The variance and standard deviation of the principal components are calculated and analyzed. A brief introduction to Principal Component Analysis and the merits of using it for detecting the selected intrusions are discussed [19].

2.8. Statistical Analysis Models.

Using online learning and statistical approaches, it is possible to continuously track the behavior of the network. Statistical analysis has been used to detect both anomalies corresponding to network failures, as well as network intrusions [9].

Qingtao et al has presented a method of detecting network anomalies by analyzing the abrupt change of time series data obtained from Management Information Base (MIB) variables. The method applies the Auto-Regressive (AR) process to model the abrupt

change of time series data, and performs sequential hypothesis test to detect the anomalies [20].

2.9. Payload Based Models

In [7] a method PAYL, which is based upon analyzing and modeling normal payloads that are expected to be delivered to the network service or application is proposed. These normal payloads are specific to the site in which the detector is placed. The system first learns a model or profile of the expected payload delivered to a service during normal operation of a system. Each payload is analyzed to produce a byte frequency distribution (histogram) of those payloads, which serves as a model for normal payloads. After this centroid model is computed during the learning phase, an anomaly detection phase begins.

The anomaly detector captures incoming payloads and tests the payload for its consistency (or distance) from the centroid model. The distance metric used is the Mahalanobis distance metric, here applied to a finite discrete histogram of byte value (or character) frequencies computed in the training phase. Any new test payload found to be too distant from the normal expected payload is deemed anomalous and an alert is generated. The aim of this work is to detect the first occurrences of a worm either at a network system gateway or within an internal network from a rogue device and to prevent itspropagation.

Bolzoni et al proposed a model that aims to increase the efficiency of model PAYL model. Bolzoni et al have proposed POSEIDON which has an architecture that combines a Self-Organizing Map, and a modified PAYL system. POSEIDON, like most network intrusion detection systems, is packet-oriented. This architecture presents two main advantages: firstly, POSEIDON can identify and block an attack while it is taking place (intrusion prevention). Secondly, connection-based systems are computationally more expensive, in particular they require a huge amount of memory resources to keep all the segments to analyze. This makes connection-based system more suitable for off-line analysis [8].

Bolzoni et al have proposed APHRODITE which is an architecture designed to reduce false positives in network intrusion detection systems. APHRODITE works by detecting anomalies in the output traffic, and by correlating them with the alerts raised by the NIDS working on the input traffic. Benchmarks show a substantial reduction of false positives and that APHRODITE is effective also after a "quick setup", i.e. in the realistic case in which it has not been "trained" and set up optimally. APHRODITE works as follows: when the NIDS raises an alert, the correlation engine checks whether the communication that raised this alert also causes an anomaly in the output (detected by the OAD (Output Anomaly Detector). If this is the case, the alert is considered a true positive and APHRODITE forwards it to the IT professionals, otherwise, it is discarded as a false positive [21]. They have tested APHRODITE together with both POSEIDON and Snort on the traffic of weeks 4 and 5 of DARPA.

2.10. Maximum Entropy Estimated Models

In [22], a network anomaly detection technique based on maximum entropy and relative entropy techniques is developed. The work exploits the idea of behavior based anomaly detection. First packets are divided into classes along multiple dimensions. A maximum entropy baseline distribution of the packet classes in attack free traffic is determined by learning a density model from a set of pre-labeled training data. The empirical distribution of the packet classes under observation is then compared to this baseline distribution using relative entropy as the metric. If the two distributions differ, the packet classes primarily responsible for the difference, contain packets related to an anomaly.

The maximum entropy approach described in [22] exhibits many advantages. First, it provides the administrators a multi-dimensional view of the network traffic by classifying packets according to a set of attributes carried by a packet. Second, it detects anomalies that cause abrupt changes in the network traffic, as well as those that increase traffic slowly. A large deviation from the baseline distribution can only be caused by packets that make up an unusual portion of the traffic. If an anomaly occurs, no matter how slowly it increases its traffic, it can be detected once the relative entropy increases to a certain level. Third, it provides information about the type of the anomaly detected. The method requires

only a constant amount of memory and consists solely of counting the packets in the traffic, without requiring any per flow information.

2.11. Comments on Related Works

Signature based models and patter matching models can not detect newly designed attacks because of lack of information of new attacks types. They need continuous updates to catch the up to date attacks. Using case-based reasoning for describing fault scenarios also suffers from heavy dependence on past information. Furthermore, the identification of relevant criteria for the different faults will, in turn, require a set of rules to be developed. In addition, using any functional approximation scheme, such as back propagation, causes an increase in computation time and complexity. The number of functions to be learned also increases with the number of faults studied.

Header information is mainly useful to recognize attacks aiming at vulnerabilities of the network stack implementation or probing the operating system to identify active network services. On the other hand, payload information is most useful to identify attacks against vulnerable applications (since the connection that carries the attack is established in a normal way) [7]. Without pretending to be globally better than other types of anomaly based systems, payload-based systems have importance of their own, as they are particularly suitable for detecting popular attacks such as those on the HTTP protocol, and worms (see Wang and Stolfo [23] and Costa et al. [24] for a discussion). Notably, PAYL and the system of Kruegel et al. [25] are mainly payload-based, while PHAD [26] is partly payload based.

3. THEORETICAL BACKGROUND

3.1. Introduction

To understand the methodology of this thesis some of the terms should be understood totally. For this purpose in this chapter a general information to understand the methodology of this thesis will be given.

Initially to understand what the term payload means, the TCP/IP model should be known. Since payload of transport layer protocol is used for modeling, the term payload with TCP packet structure will be explained.

This thesis work is a combination of PAYL algorithm and maximum entropy estimation model, detailed information about these two methods will be given. So in the second section of this chapter, the starting point of Me-PAYL method; the PAYL algorithm will be explained. After maximum entropy estimation algorithm used in this thesis will be given.

3.2. CID: Intrusion Detection Capability

A fundamental problem in intrusion detection is what metric(s) can be used to objectively evaluate an intrusion detection system in terms of its ability to correctly classify events as normal or intrusion. The intrusion detection process can be examined from an information-theoretic point of view. Intuitively, we should have less uncertainty about the input (event data) given the IDS output (alarm data). In [27] a new metric called Intrusion Detection Capability, C_{ID} is proposed. C_{ID} is simply the ratio of the mutual information between IDS input and output, and the entropy of the input. C_{ID} has the desired property that:

1. It considers all the important aspects of detection capability naturally, i.e., true positive rate, false positive rate, positive predictive value, negative predictive value, and base rate;
2. It objectively provide an intrinsic measure of intrusion detection capability;
3. It is sensitive to IDS operation parameters.

It is also proposed in [27] that C_{ID} is the appropriate performance measure to maximize when fine tuning an IDS. The thus obtained operation point is the best that can be achieved by the IDS in terms of its intrinsic ability to classify input data.

Table 3.1. List of terminology used in this thesis.

Term	Meaning
FP	False positive rate. The chance that there is an alert, A, when there is no intrusion, $\neg I$.
TP	True positive rate. The chance the there is an alert, A, when there is an intrusion, I.
FN	False negative rate. The chance there is no alert, $\neg A$, when there is an intrusion, I.
TN	True negative rate. The chance there is no alert, $\neg A$, when there is no intrusion, $\neg I$.
PPV	Positive predictive value. The chance that an intrusion, I, is present when an IDS outputs an alarm, A.
NPV	Negative predictive value. The chance that there is no intrusion, $\neg I$, when an IDS does not output an alarm, $\neg A$.
B	Base rate. The probability that there is an intrusion in the observed audit data.

Defining an appropriate metric is essential to both practice and research because we need a metric when selecting the best IDS configuration for an operation environment and when comparing different IDSs. The most basic and commonly used metrics are true positive rate (TP), which is the probability that the IDS outputs an alarm when there is an intrusion, and false positive rate (FP), which is the probability that the IDS outputs an alarm when there is no intrusion. Alternatively, one can use false negative rate FN = 1-TP and true negative rate TN = 1-FP.

When we fine tune an IDS (especially an anomaly detection system), for example by setting the threshold of deviation from a normal profile, there may be different TP and FP values associated with different IDS operation points (e.g., each with a different threshold). Clearly, both TP and FP need to be considered when selecting the best IDS operation point and comparing IDSs. The question is then how to use these two metrics together.

A popular approach is to use an ROC (receiver operating characteristic) curve to plot the different TP and FP values associated with different IDS operation points. For example, an ROC curve can show one (operation) point with <TP = 0:99; FP = 0:001> and another with <TP = 0:999; FP = 0:01>, etc. A ROC curve shows the relationship between TP and FP but by itself cannot be used to determine the best IDS operation point. ROC curves may be used for comparing IDSs.

If the ROC curves of two IDSs do not "cross" (i.e., one is always above the other), then the IDS with the top ROC curve is better because for every FP it has a higher TP. However, if the curves do cross, then there is no easy way to compare the IDSs. It is not always appropriate to use the area under ROC curve for comparison because it measures all possible operation points of an IDS. One can argue that comparison should be based on the best operation point of each IDS because in practice an IDS is fine tuned to a particular configuration (e.g., using a particular threshold).

In addition to TP and FP, two other useful metrics are the positive predictive value (PPV), which is the probability that there is an intrusion when the IDS outputs an alarm, and negative predictive value (NPV), which is the probability that there is no intrusion when the IDS does not output an alarm [27]. These metrics are very important from a usability point of view because ultimately, the IDS alarms are useful to an intrusion response system (or admin staff) only if the IDS has high PPV and NPV . Both PPV and NPV depend on TP and FP, and are very sensitive to base rate (B), which is the prior probability of intrusion.

Base rate can be entered as a piece of prior information about the IDS operational environment in the Bayesian equations. Similar to the situation with TP and FP, both PPV

and NPV are needed when evaluating an IDS from a usability point of view, and currently there is no objective method to integrate both metrics.

The metric, called Intrusion Detection Capability, or C_{ID}, is simply the ratio of the mutual information between IDS input and output, and the entropy of the input. Mutual information measures the amount of uncertainty of the input resolved by knowing the IDS output. CID is normalized using the entropy (the original uncertainty) of the input. Thus, the ratio provides a normalized measure of the amount of certainty gained by observing IDS outputs. This natural metric incorporates TP, FP, PPV , NPV and B, and thus provides a unified measure of the detection capability of an IDS. It is also sensitive to TP, FP, and B.

C_{ID} is used as a single unified metric that takes into account all the important aspects of detection capability. These metrics are TP, FP, PPV , NPV , and B. That is, this metric should incorporate existing metrics because they all are useful in their own rights. C_{ID} is sensitive to IDS operation parameters. All the results are shown in this chapter is under maximum C_{ID} condition. Tables that show detected attacks in Appendix A list detections achieved for largest C_{ID} value.

In calculation of C_{ID} three ratios must be known about dataset. These are false positive rate, false negative rate and base rate. The meanings of these three ratios are given in Table 3.1. Details about calculation of C_{ID} can be found in [27].The mathematical details of calculation method will not be explained in this thesis. Instead the method that is followed for calculation of FP, FN and base rates will be explained (see Chapter 5).

3.3. TCP/IP Model

A TCP/IP model is an architectural model, which provides a common frame of reference for discussing Internet communications. It is used not only to explain communication protocols but to develop them as well. It separates the functions performed by communication protocols into manageable layers stacked on top of each other. Each layer in the stack performs a specific function in the process of communicating over a network.

No document officially specifies the model; different names are given to the layers by different documents, and different numbers of layers are shown by different documents. Generally, TCP/IP is described using three to five functional layers. To describe TCP/IP, the common DoD reference model, which is also known as the Internet reference model is chosen. There are versions of this model with four layers and with five layers. The original four-layer version of the model has:

- Layer 1 - Network Access Layer - This layer describes the physical equipment necessary for communications, such as twisted pair cables, the signaling used on that equipment, and the low-level protocols using that signaling.

- Layer 2 - Internet or Internetworking Layer - This layer defines IP addresses, with many routing schemes for navigating packets from one IP address to another.

- Layer 3 - Host-To-Host (Transport) Layer - This is where flow-control and connection protocols exist, such as TCP. This layer deals with opening and maintaining connections, ensuring that packets are in fact received

- Layer 4 - Process Layer or Application Layer - This is where the "higher level" protocols such as SMTP, FTP, SSH, HTTP, etc. operate.

As the reference model indicates, protocols (which compose the various layers) are like a pile of building blocks stacked one upon another. Because of this structure, groups of related protocols are often called stacks or protocol stacks.

Data is passed down the stack from one layer to the next, until it is transmitted over the network by the network protocols. The four layers in this reference model are crafted to distinguish between the different ways that the data is handled as it passes down the protocol stack from the application layer to the underlying physical network.

At the remote end, the data is passed up the stack to the receiving application. The individual layers do not need to know how the layers above or below them function; they only need to know how to pass data to them.

Each layer in the stack adds control information (such as destination address, routing controls, and checksum) to ensure proper delivery. This control information is called a header and/or a trailer because it is placed in front of or behind the data to be transmitted. Each layer treats all of the information that it receives from the layer above it as data or, and it places its own header and/or trailer around that information.

These wrapped messages are then passed into the layer below along with additional control information, some of which may be forwarded or derived from the higher layer. By the time a message exits the system on a physical link (such as a wire), the original message is enveloped in multiple, nested wrappers; one for each layer of protocol through which the data passed. When a protocol uses headers or trailers to package the data from another protocol, the process is called encapsulation.

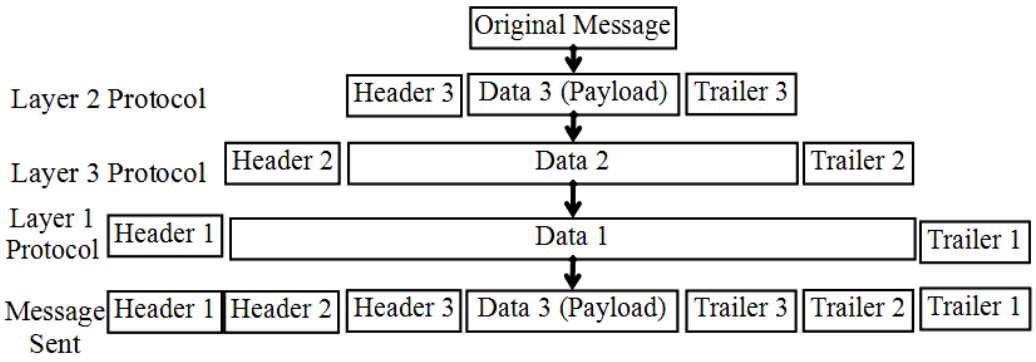

Figure 3.1. Encapsulation of data for network delivery in TCP/IP model.

Each information consist of header and data that passes through from an upper layer to lower is called payload of lower layer. For example, when sending a mail via SMTP protocol, the output of mail application (such as Microsoft Outlook) is payload of TCP layer. With TCP header, the mail itself is the payload of IP layer and so on. This process is illustrated in following figure. As u can see from the figure below each protocol layer adds

a header and trailer to the data coming from upper layer. This data, which contains original message and headers/trailers of upper layers, is called payload of that protocol layer.

3.3.1. Layers in the TCP/IP model

Lower layers serve upper layers for transmitting data over internet. Layers tasks are isolated from each other. That means layers upper layers don't need to know the details of duty of lower layers. This abstraction allows upper layers to provide services that the lower layers cannot, or choose not, to provide. For example, IP is not designed to be reliable and is a best effort delivery protocol. This means that all transport layers must choose whether or not to provide reliability and to what degree. UDP provides data integrity (via a checksum) but does not guarantee delivery; TCP provides both data integrity and delivery guarantee (by retransmitting until the receiver receives the packet).

3.3.1.1. Application layer. The application layer is used by most programs for network communication. Data is passed from the program in an application-specific format, and then encapsulated into a transport layer protocol. Data sent over the network is passed into the application layer where it is encapsulated into the application layer protocol. From there, the data is passed down into the lower layer protocol of the transport layer.

The two most common lower layer protocols are TCP and UDP. Common services have specific ports assigned to them (HTTP has port 80; FTP has port 21; etc.) while clients use ephemeral ports. Routers and switches do not utilize this layer but bandwidth throttling applications do.

3.3.1.2. Transport layer. The transport layer's responsibilities include end-to-end message transfer capabilities independent of the underlying network, along with error control, fragmentation and flow control.

End to end message transmission or connecting applications at the transport layer can be categorized as either:

- Connection-oriented: TCP

- Connectionless: UDP

The transport layer provides this service of connecting applications together through the use of ports. Since IP provides only a best effort delivery, the transport layer is the first layer to address reliability.

TCP is a connection-oriented protocol that addresses numerous reliability issues to provide a reliable byte stream:

- data arrives in-order
- data has minimal error (i.e. correctness)
- duplicate data is discarded
- lost/discarded packets are resent
- includes traffic congestion control

The dynamic routing protocols which technically fit at this layer in the TCP/IP Protocol Suite (since they run over IP) are generally considered to be part of the Network layer; an example is OSPF (IP protocol number 89).

UDP is a connectionless datagram protocol. Like IP, it is a best effort or "unreliable" protocol. Reliability is addressed through error detection using a weak checksum algorithm. UDP is typically used for applications such as streaming media (audio and video, etc) where on-time arrival is more important than reliability, or for simple query/response applications like DNS lookups, where the overhead of setting up a reliable connection is disproportionately large.

Both TCP and UDP are used to carry a number of higher-level applications. The applications at any given network address are distinguished by their TCP or UDP port. By convention certain well known ports are associated with specific applications.

3.3.1.3. Internet or internetworking layer. As originally defined, the Network layer solves the problem of getting packets across a single network. Examples of such protocols are X.25, and the ARPANET's Host/IMP Protocol.

With the advent of the concept of internetworking, additional functionality was added to this layer, namely getting data from the source network to the destination network. This generally involves routing the packet across a network of networks, known as an internetwork or (lower-case) internet.

In the Internet protocol suite, IP performs the basic task of getting packets of data from source to destination. IP can carry data for a number of different upper layer protocols; these protocols are each identified by a unique protocol number. All routing protocols, such as OSPF, and RIP are also really part of the network layer, although they might seem to belong higher in the stack.

3.3.1.4. Network access layer. The Network access layer, which is the method used to move packets from the network layer on two different hosts, is not really part of the Internet protocol suite, because IP can run over a variety of different link layers. The processes of transmitting packets on a given link layer and receiving packets from a given link layer can be controlled both in the software device driver for the network card, as well as on firmware or specialist chipsets. These will perform data link functions such as adding a packet header to prepare it for transmission, and then actually transmit the frame over a physical medium.

This layer is also responsible for encoding and transmission of data over network communications media. It operates with data in the form of bits that are sent from the Physical layer of the sending (source) device and received at the Physical layer of the destination device.

3.3.1.5. TCP packet structure. TCP is transport layer protocol explained in previous section. As can be seen from the figure below, a TCP packet consists of TCP header and data coming from application layer. Applications can have large variety of programs such

as a web browser, telnet client program or an e-mail client program. These applications are distinguished in transport layer with sockets (ports).

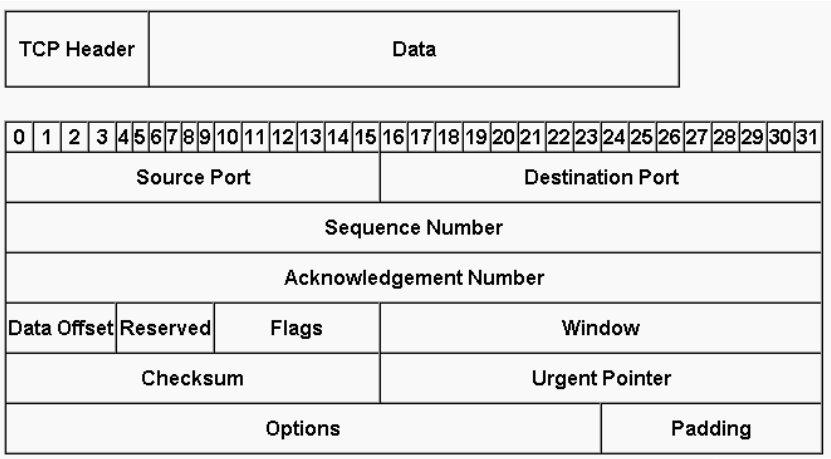

Figure 3.2. TCP protocol header structure. In the upper figure payload is shown as “Data”.

Transport layer protocols are responsible for fragmentation in both sides. Also as TCP is a connection oriented protocol there are flags about connection status of both sides such as FIN, RST. The descriptions of some of the fields of TCP header is as follows:

Source Port: The (software) port number of the source host.

Destination Port: The (software) port number of the destination host.

Sequence Number: The sequence number of the first data octet in this segment. (Not used when the SYN flag is set.)

Acknowledgement Number: The sequence number of the next data octet to be sent by the source. This field is used when the ACK flag is set.

Data Offset: The number of 32 bit words in the TCP Header.

Flags:

- URG: The Urgent Pointer field is significant

- ACK: The Acknowledgement field is significant
- EOL: End of Letter
- RST: Reset the connection
- SYN: Synchronize sequence numbers
- FIN: No more data from the sender
- Window: The maximum number of octets which the sender will accept.
- Checksum: This checksum covers both the header and data portion of the TCP packet.
- Urgent Pointer: Points to the first urgent data byte in the packet.

Most network IDS methods use flags in TCP header to detect intrusions such as DDoS attacks.

3.4. PAYL model

To understand the work done in this thesis first PAYL model proposed in [7] should be fully understood.

A brief description of this model is given in section 2.9. In this section a detailed study of PAYL model will be given. The PAYL model is proposed in [7]. Ke Wang et al. presented a payload-based anomaly detector, PAYL, for intrusion detection. PAYL models the normal application payload of network traffic in a fully automatic, unsupervised and very efficient fashion. Initially during a training phase a profile byte frequency distribution and their standard deviation of the application payload flowing to a single host and port is computed. Then Mahalanobis distance during the detection phase is used to calculate the similarity of new data against the pre-computed profile. The detector compares this measure against a threshold and generates an alert when the distance of the new input exceeds this threshold. In the paper, effectiveness of the method on the 1999 DARPA IDS dataset and a live collected dataset on the Columbia CS department network, demonstrated. Nearly 100% accuracy is achieved with 0.1% false positive rate for port 80 traffic, but method is not effective for port 22, 21 and 25.

3.4.1. Payload Modeling and Anomaly Detection

In this section an introduction to payload based modeling will be given. Starting point of this thesis PAYL method will be explained briefly.

3.4.1.1. Payload based anomaly detection. There are many design choices in modeling payload in network flows. The primary design criteria and operating objectives of any anomaly detection system entails:

- automatic "hands-free" deployment requiring little or no human intervention,
- generality for broad application to any service or system,
- incremental update to accommodate changing or drifting environments,
- accuracy in detecting truly anomalous events, here anomalous payload, with low (or controllable) false positive rates,
- resistance to mimicry attack and
- efficiency to operate in high bandwidth environments with little or no impact on throughput or latency.

These are difficult objectives to meet concurrently, yet they do suggest an approach that may balance these competing criteria for payload anomaly detection. Ke Wang et al. considers that "language-independent" statistical modeling of sampled data streams best exemplified by well known n-gram analysis. Many have explored the use of n-grams in a variety of tasks. The method is well understood, efficient and effective. The simplest model one can compose is the 1-gram model. A 1-gram model is certainly efficient (requiring a linear time scan of the data stream and an update of a small 256-element histogram).

Unlike the network packet headers, payload doesn't have a fixed format, small set of keywords or expected tokens, or a limited range of values. Any character or byte value may appear at any position of the datagram stream. To model the payload, it is needed to divide the stream into smaller clusters or groups according to some criteria to associate similar streams for modeling. The port number and the length are two obvious choices. We

may also condition the models on the direction of the stream, thus producing separate models for the inbound traffic and outbound responses.

Usually the standard network services have a fixed pre-assigned port number: 20 for FTP data transmission, 21 for FTP commands, 22 for SSH, 23 for Telnet, 25 for SMTP, 80 for Web, etc. Each such application has its own special protocol and thus has its own payload type. Each site running these services would have its own "typical payload" flowing over these services. Payload to port 22 should be encrypted and appear as uniform distribution of byte values, while the payload to port 21 should be primarily printable characters entered by a user and a keyboard.

Within one port, the payload length also varies over a large range. The most common TCP packets have payload lengths from 0 to 1460. Different length ranges have different types of payload. The larger payloads are more likely to have nonprintable characters indicative of media formats and binary representations (pictures, video clips or executable files etc.). Thus, a payload model for each different length range for each port and service and for each direction of payload flow is computed.

This produces a far more accurate characterization of the normal payload than would otherwise be possible by computing a single model for all traffic going to the host. However, many centroids might be computed for each possible length payload creating a detector with large resource consumption.

For a payload, the average frequency of each ASCII character 0-255 is computed. Some stable character frequencies and some very variant character frequencies can result in the same average frequency, but they should be characterized very differently in the model. Thus, in addition to the mean value, the variance and standard deviation of each frequency are computed as another characterizing feature. So for the payload of a fixed length of some port, each character's relative frequency is treated as a variable and its mean and standard deviation is computed as the payload model.

Figure 3.3 provides an example showing how the payload byte distributions vary from port to port, and from source and destination flows. Each plot represents the

characteristic profile for that port and flow direction (inbound/outbound). Notice also that the distributions for ports 22 (inbound and outbound) show no discernible pattern, and hence the statistical distribution for such encrypted channels would entail a more uniform frequency distribution across all of the 256 byte values, each with low variance. Hence, encrypted channels are fairly easy to spot. Notice that this figure is actually generated from a dataset with only the first 96 bytes of payload in each packet, and there is already a very clear pattern with the truncated payload.

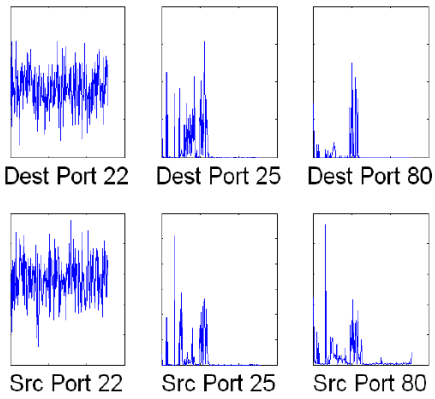

Figure 3.3. Example byte distributions for different ports. For each plot, the X-axis is the ASCII byte 0-255, and the Y-axis is the average byte frequency [7]

Figure 3.4. displays the variability of the frequency distributions among different length payloads. The two plots characterize two different distributions from the incoming traffic to the same web server, port 80 for two different lengths, here payloads of 200 bytes, the other 1,460 bytes. Clearly, a single monolithic model for both length categories will not represent the distributions accurately.

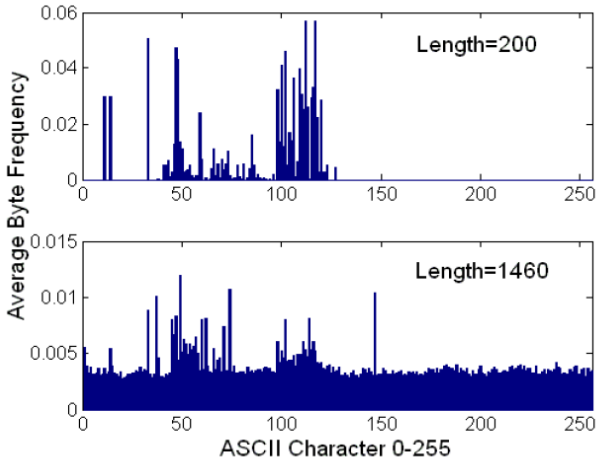

Figure 3.4. Example byte distribution for different payload lengths for port 80 on the same host server [7]

3.4.1.2. PAYL method. PAYL operates as follows. Initially many example payloads during a training phase are observed and the mean and variance of the byte value distribution are computed, producing model, for specific payload length I and port j: Mij. During detection, each incoming payload is scanned and its byte value distribution is computed. This new payload distribution is then compared against model Mij; if the distribution of the new payload is significantly different from the norm, the detector flags the packet as anomalous and generates an alert.

The formula for the Mahalanobis distance is:

$$d(x, \bar{y}) = (x - \bar{y})^T C^{-1} (x - \bar{y}) \tag{3.1}$$

To speed up the computation, simplified Mahalanobis distance is used:

$$d(x, \bar{y}) = \sum_{i=0}^{n-1} (| x_i - \bar{y}_i | / \bar{\sigma}_i) \tag{3.2}$$

The variance is replaced by the standard deviation, n is fixed to 256 under the 1-gram model. In case of $\bar{\sigma}_i$ equals zero the distance will become infinite. To avoid this situation, give a smoothing factor to the standard deviation similar to the prior observation:

$$d(x, \bar{y}) = \sum_{i=0}^{n-1} (| x_i - \bar{y}_i | / \bar{\sigma}_i + \alpha) \tag{3.3}$$

The smoothing factor α reflects the statistical confidence of the sampled training data. Larger the value of α gives less confidence.

An incremental online version may "age out" old data from the model keeping a more accurate view of the most recent payloads flowing to or from a service.

To compute the incremental version of the Mahalanobis distance, first the mean and the standard deviation of each ASCII character is computed, for each new sample observed. For the mean frequency of a character, compute:

$$\bar{x}(w) = \frac{\sum_i 1(x_i \in w)}{n} \tag{3.4}$$

Update te mean:

$$\bar{x} = \frac{\bar{x}xN + x_{N+1}}{N+1} = \bar{x} + \frac{x_{N+1} - \bar{x}}{N+1} \tag{3.5}$$

Compute variance as:

$$Var(X) = E(X - EX)^2 = E(X^2) - (EX)^2 \tag{3.6}$$

And update the variance similar to x

After computing byte frequency models for each port, and payload length, a large model vector M_{ij} is obtained. For lower memory consumption and simplicity model size should be reduced. So model size is reduced by clustering.

Clustering is done as follows:

- For a specific byte frequency model Mij compute Manhattan distance between Mij and models that has lengths (i-1) (i+1).
- If Manhattan distance between models are lower than 0.5 merge them.

3.5. POSEIDON: a 2-tier Anomaly-based Network Intrusion Detection System

In [22] POSEIDON, an anomaly-based network intrusion detection system is proposed. POSEIDON is payload-based, and has a two-tier architecture: the first stage consists of a Self-Organizing Map, while the second one is a modified PAYL system. Benchmarks of this method on the 1999 DARPA data set show a higher detection rate and lower number of false positives than that of PAYL method.

POSEIDON (Payl Over Som for Intrusion DetectiON) is a two-tier network intrusion detection architecture. The first tier consists of a self-organizingmap (SOM), and is used exclusively to classify payload data; the second tier consists of a slight modification of the well-known PAYL system [7] (see Figure 3.5.).

POSEIDON is payload-based: it uses only destination address and service port numbers to build a profile for each port monitored, and it does not consider other header features. They have extensively benchmarked POSEIDON w.r.t. PAYL (also by replicating the PAYL experiments) and PHAD using the 1999 DARPA benchmark [6]. PAYL and PHAD are the reference anomaly based systems based on payload analysis.

On this data set, their experiments show:

- a higher detection rate and lower number of false positives than PAYL and PHAD.
- a reduction of the number of profiles used w.r.t. PAYL. (see Table 5.3 and Table 5.4)

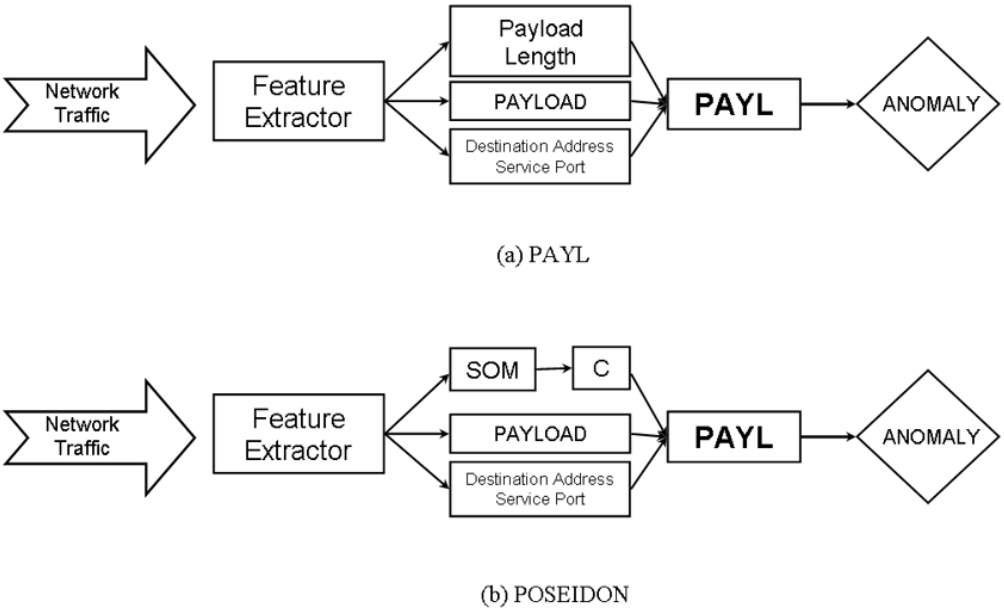

Figure 3.5. PAYL and POSEIDON architectures

For the classification phase, a self-organizing map in general can yield to a high quality classification, i.e. clusters with a high intra-cluster similarity and high inter-cluster dissimilarity, without having to take into account the length of the packet. This can be used to build good profiles. At the same time, a SOM is not as effective when it comes to the detection phase, i.e. to finding whether a given packet is anomalous with respect to the cluster it has been classified in.

In a SOM, the detection phase is accomplished by comparing the current packet quantization error with matching cluster quantization error: this method can be heavily influenced by payload byte order, because it is based on a distance function.

By combining a SOM with the n-gram algorithm they obtained an architecture that combines the advantages of the SOM (the realization of clusters with high intra-cluster similarity) with those of PAYL (the ability to detect when a packet is anomalous w.r.t. a given cluster).

POSEIDON, like most network intrusion detection systems, is packet-oriented. This architecture presents two main advantages: firstly, POSEIDON can identify and block an attack while it is taking place (intrusion prevention). Secondly, connection-based systems are computationally more expensive, in particular they require a huge amount of memory resources to keep all the segments to analyse. This makes connection-based system more suitable for off-line analysis. On the other hand, connectionbased systems support a finer-grained analysis.

Their starting point is the PAYL architecture. The algorithm receives as input a packet and classifies the packet, without prejudice for any of its properties, such as length, destination port or application data semantics. The idea is that the classifier keeps as much information as possible

3.6. Detecting Anomalies in Network Traffic Using Maximum Entropy Estimation

The maximum entropy estimation algorithm proposed in [22] is taken as one of the reference models for this thesis work. For maximum entropy estimation the algorithm is used in [22]. In this section a brief description for maximum entropy estimated network anomaly detection method will be given.

Yu Gu et al. developed behavior-based anomaly detection method that detects network anomalies by comparing the current network traffic against a baseline distribution. By computing a measure related to the relative entropy of the network traffic under observation with respect to the baseline distribution, the anomalies that change the traffic either abruptly or slowly are distinguished. In addition, this method provides information revealing the type of the anomaly detected. It requires a constant memory and a computation time proportional to the traffic rate.

In the proposed method the network traffic is divided into a set of packet classes. The work focuses on anomalies concerning TCP and UDP packets. The selection of the packet classes is out of concern.

Let Ω be the set of determined packet classes. Given a sequence of packets S = {x1, x2 …xn} as the training data, the empirical distribution $\tilde{P}(w)$ over in this training data is:

$$\tilde{P}(w) = \frac{\sum_i 1(x_i \in w)}{n} \tag{3.7}$$

where $1(x)$ is an indicator function that takes value 1 if x is true and 0 otherwise.

Suppose a set of feature functions $F = \{f_i\}$ are given, f_i be an indicator function f_i $\Omega \rightarrow \{0,1\}$. By using Maximum Entropy estimation, a density model P is searched that satisfies $E_P(f_i) = E_{\tilde{P}}(f_i)$ for all fi $\in$ F and has maximum entropy. In [28], it has been proved that under such constraints, the Maximum Entropy estimate is guaranteed to be (a) unique, and (b) the same as the maximum likelihood estimate using the generalized Gibbs distribution, having the following log-linear form:

$$P(w) = \frac{1}{Z}\exp(\sum_i \lambda_i f_i(w)) \tag{3.8}$$

For each feature f_i, a parameter $\lambda_i \in \Lambda$ determines its weight in the model; Λ is the set of parameters for the feature functions. Z is a normalization constant that ensures that the sum of the probabilities over is 1. The difference between two given distributions P and $\tilde{P}$ is commonly determined using the relative entropy or Kullback-Leibler (KL) divergence:

$$D_{\tilde{P}\|P}(w) = \tilde{P}(w)\log\frac{\tilde{P}(w)}{P(w)} \tag{3.9}$$

Maximizing the likelihood of the distribution in the form of (3.4.2) with respect to $\tilde{P}$ is equivalent to minimizing the K-L divergence of $\tilde{P}$ with respect to P.

3.6.1. Feature selection

The feature selection step is a greedy algorithm which chooses the best feature function that minimizes the difference between the model distribution and the empirical distribution from a set of candidate feature functions.

Let Ω be the set of all packet classes, $\tilde{P}$ the empirical distribution of the training data over Ω , and F a set of candidate feature functions. The initial model distribution over Ω, $P_0(w) = \frac{1}{z}$ and Z=|Ω|, which is a uniform distribution over Ω

Now let P_i be a model with *i* feature functions selected

$$P_i(w) = \frac{1}{Z}\exp(\sum_{j=1}^{i}\lambda_i f_i(w)) \tag{3.10}$$

And we want to select the $(i+1)^{st}$ feature function. Let *g* be a feature function in $F=\{f_1,\ldots f_i\}$ to be selected into the model and λg be its weight, then let,

$$P_{i,\lambda_g,g}(w) = \frac{1}{Z'}\exp(\sum_{i}\lambda_i f_i(w)\exp(\lambda_g g)) \tag{3.11}$$

And let

$$\begin{aligned} G_{P_i}(\lambda_g, g) &= D(\tilde{P}\|P_i) - D(\tilde{P}\|P_{i,\lambda_g,g}) \\ &= \lambda_g E_{\tilde{P}}(g) - \log E_{P_i}(\exp(\lambda_g g)) \end{aligned}$$

Where $E_{P_i}(g)$ is the expected value of g with respect to the distribution of P. $G_{P_i}(\lambda_g, g)$ is a concave function with respect to λ_g and

$$G_{P_i}(g) = \sup_{\lambda_g} G_{P_i}(\lambda_g, g) \tag{3.12}$$

is the maximum decrease of the K-L divergence that can be attained by adding g into the model. The feature function g with the largest gain $G_{P_i}(g)$ is selected as the $(i+1)^{st}$ feature function to the model.

The algorithm followed in this thesis is based on the algorithm in [28]. Details algorithm will be explained in methodology chapter.

3.6.2. Parameter estimation

After a new feature function is added to the log-linear model, the weights of all feature functions are updated. Given a set of training data and a set of selected feature functions $\{f_i\}$, the set of parameters is then estimated. Maximum Entropy estimation locates a set of parameters $\Lambda=\{\lambda_i\}$ in (3.4.3) for $\{f_i\}$ that minimizes the K-L divergence of $\tilde{P}$ with respect to P:

$$\Lambda = \arg\min_{\Lambda} \sum_{w \in \Omega} \tilde{P}(w) \log \frac{\tilde{P}(w)}{P(w)} \tag{3.13}$$

There are a number of numerical methods that can be used to find solution to this equation. In this thesis, the L-BFGS method is used. The details of L-BFGS method will not be explained in this thesis.

4. METHODOLOGY

The principle of maximum entropy (ME) is widely used as a statistical inference tool in many fields such as computer vision, econometrics, and natural language processing. It is a technique that can be used to estimate input probabilities more generally. The result is a probability distribution that is consistent with known constraints expressed in terms of averages, or expected values, of one or more quantities, but is otherwise as unbiased as possible.

The purpose of this work is to improve intrusion detection capability (C_{ID}) of PAYL model, explained in section 2, with estimating byte frequency distributions (empirical distribution here) by using maximum entropy approach. The methodology in this work includes a combination of maximum entropy model and PAYL model. This can be summarized as the application of maximum entropy estimation on payloads of TCP packets.

This chapter contains details about:

- Modeling
- Alarm Generation
- Detection

4.1. Modeling

Modeling process has two phases. Initially empirical distribution models of payloads in dataset according to their lengths and TCP ports, are calculated. These models are saved in a matrix. Length corresponds to number of bytes in payloads.

After calculation of empirical distributions, a matrix that has a size (maximum payload length * 256) is obtained. Since model space will be too large, after calculation of empirical distributions a clustering algorithm is used. There are two reasons for this

clustering operation; to decrease model space and to create more accurate models. When there is not enough sample data in training data for some ports and payload lengths, clustering of neighboring empirical distributions is helpful to create more accurate models.

The block diagram of modeling process can be seen from following figure:

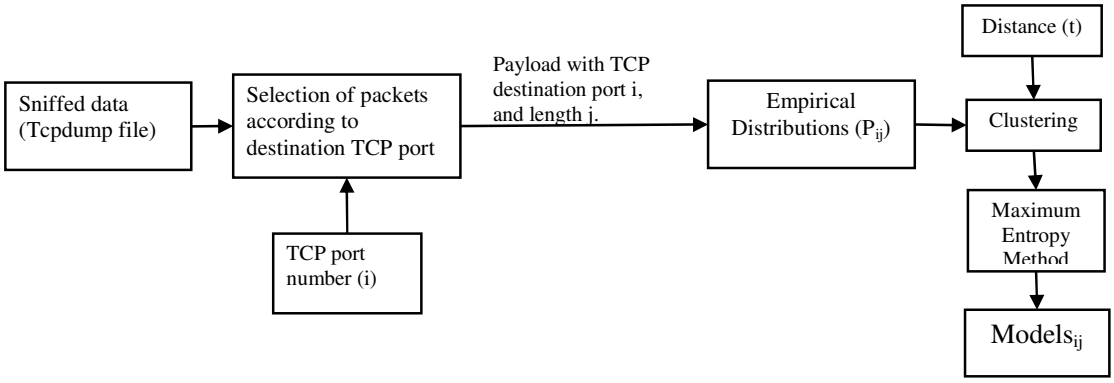

Figure 4.1. Block diagram of modeling process. Above steps are repeated for TCP port numbers: 25, 22, 21 and 80.

4.1.1. Dataset

The 1999 DARPA IDS data set was collected at MIT Lincoln Laboratories for evaluation of intrusion detection systems. All the network traffic including the entire payload of each packet was recorded in tcpdump format and provided for evaluation. In addition, there are also audit logs, daily file system dumps, and BSM (Solaris system call) logs.

The data 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. In figure 4.2 average connections per day can be seen. As seen from the figure most used TCP protocols are http, smtp ftp-data and telnet. These TCP protocols work on TCP port numbers 80, 25, 21 and 23.

Majority of attacks in dataset are toward these four TCP protocols (See Table A.1). In Me-PAYL implementation these four TCP protocols are examined.

This dataset has been used in many research efforts and results of tests against this data have been reported in many publications. Although there are problems due to the nature of the simulation environment that created the data, it still remains a useful set of data to compare techniques. The top results were reported by [6].

In this work only the inside network traffic data is used. This data was captured between the router and the victims. Because most public applications on the Internet use TCP (web, email, telnet, and ftp), and to reduce the complexity of the experiment, only the TCP traffic to the ports 21,22,25 and 80 of all hosts are examined. However in PAYL experiment only traffic of the hosts 172.016.xxx.xxx which contains most of the victims, and ports 0-1023 which covers the majority of the network services is examined. Examining only traffic of hosts in network 172.016.xxx.xxx can reduce background traffic and decrease false positive rate. These differences should be taken into account while comparing two techniques.

4.1.2. Data units

Because payload lengths can vary with large numbers, it is meaningful to take part of large payloads. For example; when payload length is larger than 1000 bytes, then take first 1000 bytes of that payloads for modeling.

In PAYL model proposed in [7] five different data unit were used in computation of empirical distributions of payloads. These are:

- Per Packet Model: which uses the whole payload of each network packet
- First 100 Packet Model: which uses the first 100 bytes of each network packet
- Tail 100 Packet Model: which uses the last 100 bytes of each network packet
- Per Conn Model: which uses the whole payload of each connection
- Truncated Conn Model: which uses the first 1000 bytes of each connection?

From obtained results in paper it is said that:

1. The payload-based model is very good at detecting the attacks to port 21 and port 80.
2. For port 23 and port 25 the payload-based model is not as good as the models for port 21 and 80.
3. For these two ports, the packet-based models are better than the connection-based models.

This is likely due to the fact that the actual exploit is buried within the larger context of the entire connection data, and its particular anomalous character distribution is swamped by the statistics of the other data portions of the connection. Per packet model detects this anomalous payload more easily.

For the DARPA 99 data, experiments can be done using each packet as the data unit and each connection as the data unit. In this thesis both packet based and connection based analysis are performed. For packet based analysis whole payloads of packets are used. However in connection based analysis first 2000 bytes of payloads are used for modeling and detection. Lengths of payloads of TCP connection can vary between 0 and 65535 bytes.

4.1.3. Selection of payloads from Tcpdump file

Sniffed data is in format of Tcpdump file. To analyze payloads of packets initially this information from Tcpdump file must be extracted from file. For this purpose a program which is coded in c++ is used. This program selects payloads of packets which have TCP destination port equal to input TCP port from Tcpdump file. The resulting payload is a vector of numbers that correspond to decimal equivalent of hex values of bytes.

4.1.4. Empirical distribution

In constituting empirical model, for every payload of packet that has TCP port i and

length j in training data, an empirical distribution model is calculated. For every incoming packet in training data, after payload of packet has been extracted, a histogram of payload is calculated with (4.1.1.).

$$\bar{x}(w) = \frac{\sum_i 1(x_i \in w)}{n} \tag{4.1.1}$$

Where *w* represents byte value ranging from 0 to 255.

After calculation of histogram of incoming packet, empirical distribution model is updated with:

$$\tilde{P}_m(w) = \frac{(\tilde{P}_{m-1}(w).(m-1) + \bar{x}_m(w))}{m} \tag{4.1.2}$$

Where $\tilde{P}_m(w)$ is empirical distribution trained with *m* samples, $\bar{x}_m(w)$ is the empirical distribution of m^{th} payload calculated with equation (4.1.1). $\tilde{P}_{m-1}(w)$ is the empirical distribution that is trained with *m-1* sample. $\tilde{P}_m(w)$ is calculated for every TCP port *i* and payload length *j*.

In following figures some examples of empirical distributions for ports 21, 23, 25 and 80 are given. For every port and every used data unit a random empirical distribution model is chosen in figures. Notice that empirical distribution of different payload lengths also differ from each other.

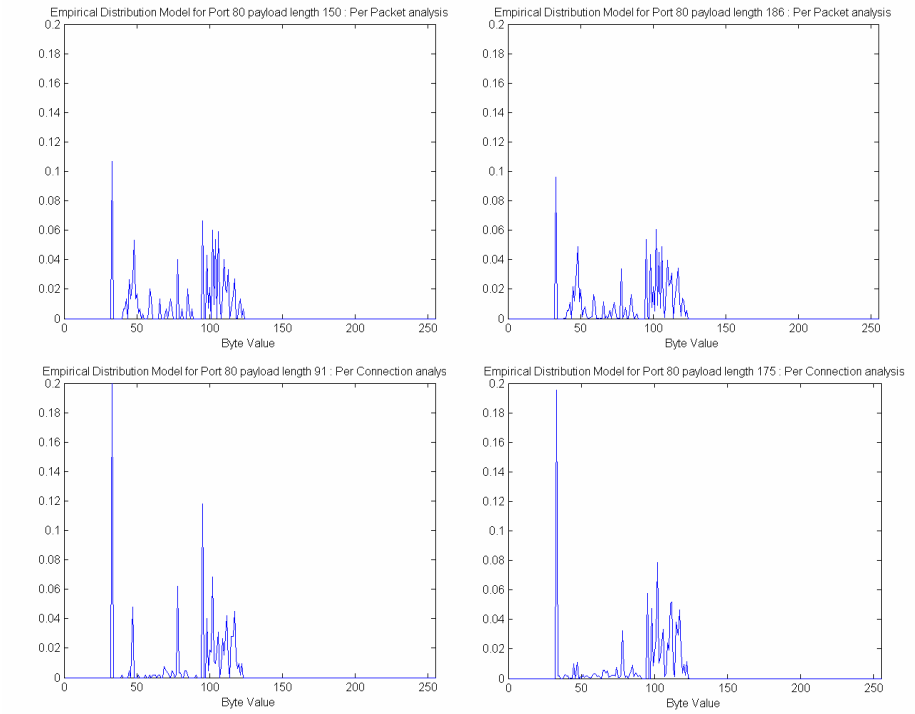

Figure 4.2. Empirical distribution for TCP port 80 and some payload lengths.

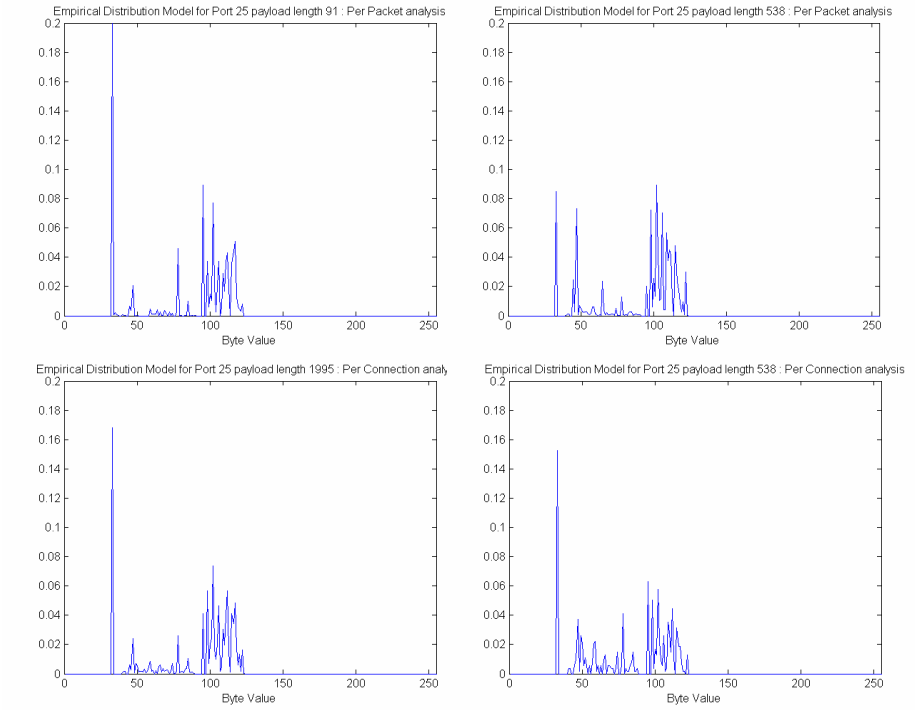

Figure 4.3. Empirical distribution for TCP port 23 and some payload lengths.

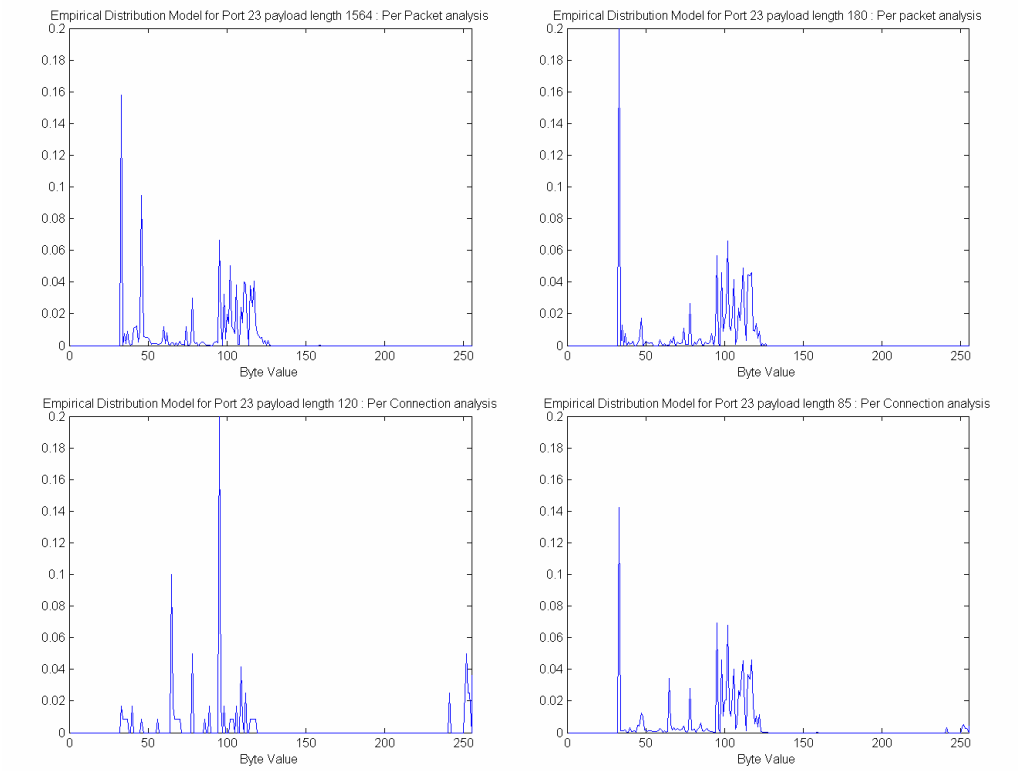

Figure 4.4. Empirical distribution for TCP port 23 and random payload lengths.

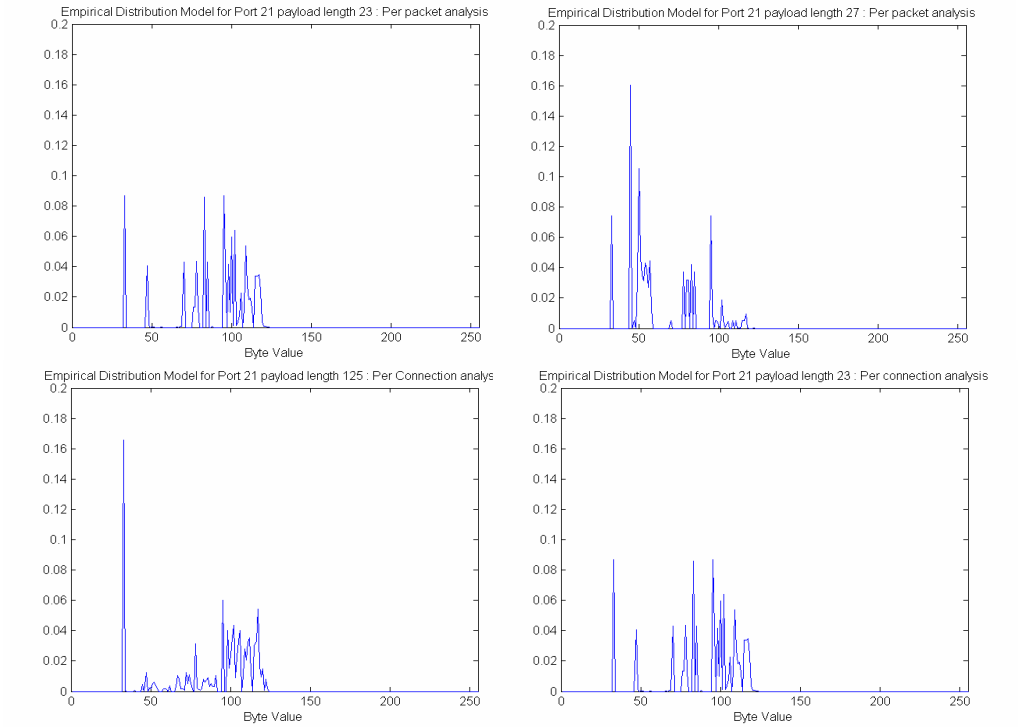

Figure 4.5. Empirical distribution for TCP port 23 and random payload lengths.

4.1.5. Clustering

After calculation of empirical distributions for every TCP port i and payload length j. of training data, resulting model matrix will have a size of maximum payload length times 256. Such fine-grained modeling might introduce several problems. First, the media files that may be measured in gigabytes and many length bins may be defined causing a large number of centroids to be computed.)

Further, the empirical distribution for payloads of length bin i can be very similar to that of payloads of length bins *i-1* and *i+1*; after all they vary by one byte. Storing a model for each length may therefore be obviously redundant and wasteful. Also every empirical model will be fed into maximum entropy modeling large model size will result a computational burden.

Another problem is that for some length bins, there may not be enough training samples. Sparseness implies the data will generate an empirical distribution that will be an inaccurate estimate of the true distribution leading to a faulty detector.

The other solution is to "borrow" data from neighboring bins to increase the number of samples; i.e. we use data from neighboring bins used to compute other "similar" models.

Initially in clustering two neighboring models are compared using the simple Manhattan distance to measure the similarity of their empirical distributions. If their distance is smaller than some threshold t (see Figure 4.1), those two models are merged using the equation in (4.1.2). These comparing and merging operations are repeated until no more neighboring models can be merged. This clustering model is the same that is used in PAYL model in [7].

Now for a payload in test data with length i sent to port j, the model Mij, or the model it was merged with, is used. But there is still the possibility that the length of the payload of test data is outside the range of all the computed models. For such test data, the model whose length range is nearest to that of the test data is used.

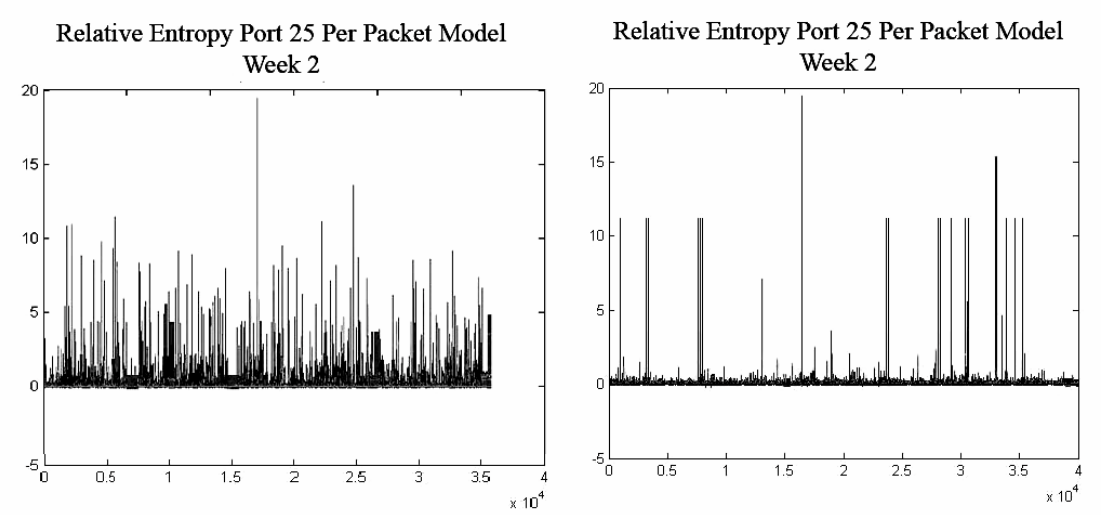

Figure 4.6. Resulting relative entropy for port 25 calculated from 2nd week sniffed data with clustering (right) and without clustering (left). From figures it can be seen that clustering improves detection by providing more accurate models.

4.1.6. Generating Maximum Entropy Models

The Principle of Maximum Entropy is based on the premise that when estimating the probability distribution, you should select that distribution which leaves you the largest remaining uncertainty (i.e., the maximum entropy) consistent with your constraints. That way you have not introduced any additional assumptions or biases into your calculations. One appealing property of the ME model is its flexibility to incorporate additional pragmatic features. In other words Principle of Maximum Entropy is a technique that can be used to estimate input probabilities more generally. The result is a probability distribution that is consistent with known constraints expressed in terms of averages, or expected values, of one or more quantities, but is otherwise as unbiased as possible.

Suppose the distribution model is required to satisfy a set of constraints reflecting properties of the training data, the Maximum Entropy estimation then follows the principle that a good model of the data should be one that satisfies these constraints only and makes no other assumptions. Here, making no other assumptions implies requiring maximum

uniformity of the model. In other words, data is represented with the most uniform model of all the models that satisfy the constraints.

Let $\Omega = \{w_1, w_2, \ldots, w_{255}\}$ be the set of classes corresponding each byte value from 1 to 255. A mathematical measurement of the uniformity of a distribution P is its entropy

$$-\sum_{w \in \Omega} P(w) \log P(w) \tag{4.1.3}$$

This entropy is bounded below by 0 corresponding to the case that there is no uncertainty, and below by the log of cardinality of Ω corresponding uniformity.

Given a sequence of bytes in payloads $S = \{x_1, ..., x_n\}$ as the training data, an empirical distribution $\tilde{P}(w)$ is calculated for every payload length i and destination port j as described in section 4.1.1.

Suppose we are given a set of feature functions $F = \{f_i\}$, f_i is a feature function, which is an indicator function $f_i : \Omega \to \{0,1\}$. By using the Maximum Entropy estimation, we are looking for a density model P that satisfies $E_P(f_i) = E_{\tilde{P}}(f_i)$ for all $f_i \in F$ and has maximum entropy. In [28], it has been proved that under the constraints $E_P(f_i) = E_{\tilde{P}}(f_i)$ for all $f_i \in F$, the Maximum Entropy estimate is guaranteed to be (a) unique, and (b) the same as the maximum likelihood estimate using the generalized Gibbs distribution, which is in a log-linear form of

$$P(w) = \frac{1}{Z} \exp\left(\sum_i \lambda_i f_i(w)\right) \tag{4.1.4}$$

For each feature f_i, a parameter $\lambda_i \in \Lambda$ determines its weight in the model. Λ is the set of parameters for the feature functions in the model. Z is a normalization constant that ensures that the sum of the probabilities over Ω is 1. Maximizing the likelihood of the

distribution in the form of (above equation) with respect to $\tilde{P}$ is equivalent to minimizing the Kullback-Leibler divergence of $\tilde{P}$ with respect to P

$$P = \arg\min_{P} D(\tilde{P} // P) \tag{4.1.5}$$

As

$$\prod_{w \in \Omega} P(w)^{\sum 1(x_i \in w)} \propto \exp(-D(\tilde{P} // P)) \tag{4.1.6}$$

Feature functions in the log-linear model (above equation) represent certain properties of the training data in the learned model. For the sake of efficiency, feature functions are often selected to express the most important characteristics of the training data in the learned log-linear model, and in return, the log-linear model expresses the empirical distribution with the fewest feature functions and parameters. For example, numerical characters, and some special characters that are used in html coding or in command line in telnet protocol should be threaded as important classes in byte values.

Thus, the Maximum Entropy estimation procedure consists of two parts: feature selection and parameter estimation. The feature selection part selects the most important features of the log-linear model from candidate features initially determined. The parameter estimation part assigns a proper weight to each of the feature functions. These two parts are performed iteratively such that the set of feature functions increases one at a time and the weights for the feature functions in the log-linear model are adjusted as each new feature function is added. As the feature functions in the model are added, the model approaches to the empirical distribution; and the parameter estimation ensures that the model satisfies the Maximum Entropy principle.

4.1.6.1. Feature selection. The feature selection step is a greedy algorithm which chooses the best feature function from a set of candidate feature functions that minimizes the difference between the model distribution and the empirical distribution.

Let Ω be the set of all of byte classes, $\tilde{P}$ the empirical distribution of Ω in the training data, and F a set of candidate feature functions. At the initial state, no information is known about the training data. So the initial model distribution over Ω is

$$P_0(w) = \frac{1}{Z}, \qquad Z = |\Omega|, \tag{4.1.7}$$

which is a uniform distribution over Ω. The subscript 0 indicates that there is no feature in the model.

Suppose g is a feature function $g \in F$. We define 1-parameter family of distributions $\{P_{\lambda_g g} : \lambda_g \in R\}$ as follows:

$$P_{0,\lambda_g,g}(w) = \frac{1}{Z'} \exp(\lambda_g g(w)) \tag{4.1.8}$$

The family of distribution is referred to as the indication of P_0 by g. We also define

$$G_{P_0}(\lambda_g, g) = D(\tilde{P} // P_0) - D(\tilde{P} // P_{0,\lambda_g,g}) \tag{4.1.9}$$

Here, $G_{P_0}(\lambda_g, g)$ is the difference of the two Kullback-Leibler divergences related to P_0 and $P_{0,\lambda_g,g}$. It is the improvement that future g brings to the model with its weight λ_g. We define $G_{P_0}(g)$ to be the greatest improvement by adding g to P_0:

$$G_{P_0}(g) = \sup_{\lambda_g} G_{P_o}(\lambda_g, g) \tag{4.1.10}$$

g

And $G_{P_0}(g)$ is referred to as the gain of the candidate feature g. In order to minimize the difference between the model distribution and the empirical distribution, the feature function in F with the maximum gain is chosen. We name the selected feature function f_1

$$f_1 = \arg\max_g G_{P_0}(g) \tag{4.1.11}$$

And obtain the new model with one feature function selected as

$$P_1(w) = \frac{1}{Z}\exp(\lambda_1, f_1(w)) \tag{4.1.12}$$

Now let P_i be a model with i feature functions selected.

$$P_i(w) = \frac{1}{Z}\exp(\sum_{j=1}^{i} \lambda_j f_j(w)) \tag{4.1.13}$$

And we want to select the $i+1^{st}$ feature function. Similar to the above procedure, let g a function in $F \setminus \{f_1,...f_i\}$ that has not yet been selected. We obtain

$$P_{i,\lambda_g,g}(w) = \frac{1}{Z'}\exp(\sum_i \lambda_i f_i(w))\exp(\lambda_g g) \tag{4.1.14}$$

Adding a new feature function would require all the parameters to be adjusted, which makes it computationally expensive to compute the exact form of the model with g added for all g in the candidate feature function set. As a compromise, the feature selection is based upon an approximation. The improvement of adding a new feature g is approximated by adjusting only the weight of the parameter of g and remaining parameters unchanged. Considering large number of candidate feature functions, this approximation makes feature selection practical. With this approximation, we have

$$
\begin{aligned}
& G_{P_i}(\lambda_g, g) \\
&= D(\tilde{P} \,//\, P_i) - D(\tilde{P} \,//\, P_{i,\lambda_g,g}) \\
&= \sum_w \tilde{P}(w) \log \frac{P_{i,\lambda_g,g}}{P_i(w)} \\
&= \lambda_g E_{\tilde{P}}(g) - \log E_{P_i}(\exp(\lambda_g g))
\end{aligned}
$$

Which is a concave function with respect to λ_g, and

$$G_{P_i}(g) = \sup_{\lambda_g} G_{P_i}(\lambda_g, g) \tag{4.1.15}$$

The feature function g with the largest gain $G_{P_i}(\lambda_g, g)$ is selected as the $i+1^{st}$ feature function to the model.

For a candidate feature function g in the form of an indicator function, $G_P(\lambda_g, g)$ is maximized by

$$\hat{\lambda}_g = \arg\max_{\lambda_g} G_P(\lambda_g, g) = \log\left\{\frac{E_{\tilde{P}}(g)(1-E_P(g))}{E_P(g)(1-E_{\tilde{P}}(g))}\right\} \tag{4.1.16}$$

In which $E_{\tilde{P}}(g)$ is the expected value of g with respect to the distribution of $\tilde{P}$ and $E_P(g)$ is the expected value of g with respect to the distribution of P. At this value of λ_g,

$$G_P(g) = G_P(\hat{\lambda}_g, g) = D(B_{\tilde{P}} \,//\, B_P) \tag{4.1.17}$$

Where $B_{\tilde{P}}$ and B_P are random variables of Bernoulli distributions given by

$$P(B_{\tilde{P}} = 1) = E_{\tilde{P}}(g) \qquad P(B_{\tilde{P}} = 0) = 1 - E_{\tilde{P}}(g) \tag{4.1.18}$$

$$P(B_P = 1) = E_P(g) \qquad P(B_{\tilde{P}} = 0) = 1 - E_P(g) \tag{4.1.19}$$

After a new feature is added to the log-linear model, the parameter estimation procedure will update the weight of all feature functions in the model to minimize the Kullback-Leibler divergence between the model distribution and the empirical distribution.

4.1.6.2. Parameter Estimation. Given a set of training data and a set of selected feature functions $\{f_i\}$, the set of parameters is then estimated. The Maximum Entropy estimation locates a set of parameters $\Lambda = \{\lambda_i\}$ in (4.1.4) for $\{f_i\}$ that minimizes the Kullback-Leibler divergence of $\tilde{P}$ with respect to P:

$$\Lambda = \arg\min_{\Lambda} \sum_{w \in \Omega} \tilde{P}(w) \log \frac{\tilde{P}(w)}{P(w)} \tag{4.1.20}$$

Computing these parameters in the Maximum Entropy model directly is not easy. There are a number of numerical methods that can be exploited, including iterative scaling methods like Generalized Iterative Scaling and Improved Iterative Scaling, as well as first order methods such as steepest ascent, conjugate gradient, and second order methods like quasi-Newton or limited memory variable metric method (LBFGS). In this thesis work LBFGS algorithm is used., By using as inputs the empirical distribution, the set of features $\{f_i\}$, and the values of the features under each packet class, the Maximum Entropy estimator can compute the set of parameters $\{f_i\}$ that minimizes the Kullback-Leibler divergence of the empirical distribution and the distribution defined by the log linear model.

The model is built by iterating (4.1.5) and (4.1.20) until some stopping criterion is met. This stopping criterion can be either that the K-L divergence of *P* with respect to $\tilde{P}$ is less than some threshold value, or that the gain of adding a new feature function is too small to improve the model.

The feature functions are selected from a set of candidate feature functions. Since the domain in our work consists of byte values, feature functions are selected according to usage of characters in TCP protocols of our interest.

1) Initial data:

   (0) A set of training data with emprical distribution $\tilde{P}$

   (1) A set of candidate feature functions F

   (2) An initial density model P_0, $P_0(w) = \frac{1}{Z}$, $Z = |\Omega|$

2) Iterated steps

   (0) Set $n = 0$

   (1) Feature Selection

   (2) For each feature function $g \in F, g \notin \{f_i\}$, compute the gain $G_{P_n}(g)$

   (3) Let f_{n+1} be the feature function with the largest gain

3) Parameter Estimation

   (0) Update all the parameters corresponding to the selected features and set P_{n+1} to be the updated model

   (1) Check the iteration stopping criterion

4) If the iteration stopping criterion is not met, set $n = n+1$, go to (1). Otherwise, return the learned model P_{n+1}

Figure 4.7. Model induction Algorithm

In the above model induction algorithm, step (1) performs the feature selection procedure, and step (2) performs the parameter estimation procedure. As more and more feature functions are added to the model, the induction algorithm is able to reduce the Kullback-Leibler divergence between the empirical distribution $\tilde{P}$ and the model distribution P.

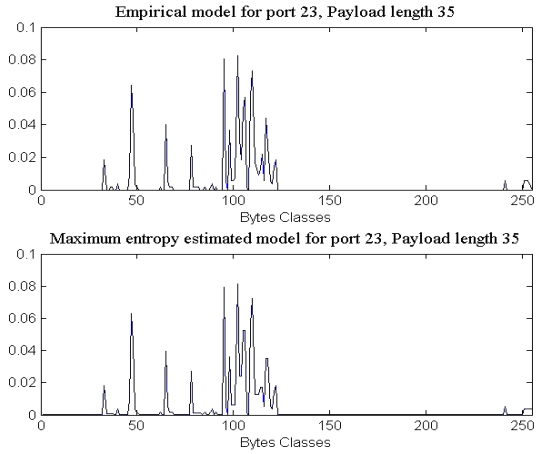

Figure 4.8. An example for empirical distribution and resulting maximum entropy model.

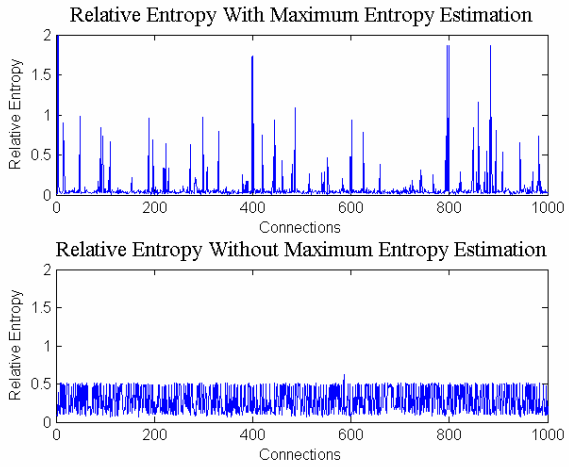

Figure 4.9. Relative entropy calculated between empirical distributions of payloads of test data and maximum entropy estimated model (above). Relative entropy calculated between empirical distributions of payloads of test data and empirical distributions of training data.

From Figure 4.7 empirical distribution for port 23 per connection analysis and payload length 35 and maximum entropy estimated models can be seen. There seem to be no distinctive difference between empirical distribution and maximum entropy estimation of it. But when looking Figure 4.8 the difference of these models arises more clearly. Resulting relative entropy with maximum entropy estimated model is less noisy and attack

points can be seen more clearly. However when clustered empirical distributions are used as models attack instances can not be distinguished clearly.

From figure 4.7 and 4.8 the improvement of using maximum entropy model can be seen clearly. Also from these figures the reason for higher detection rates with false alarm rates can be understood better.

4.2. Alarm Generation

For alarm generation from test data like modeling process initially packets of our concern are selected from test Tcpdump file. After a packet is recovered from Tcpdump data an empirical distribution of packet is calculated with (4.1.1).

After calculation empirical distribution of test data unit, relative entropy for every test payload between empirical distribution of test data and corresponding maximum entropy model is calculated.

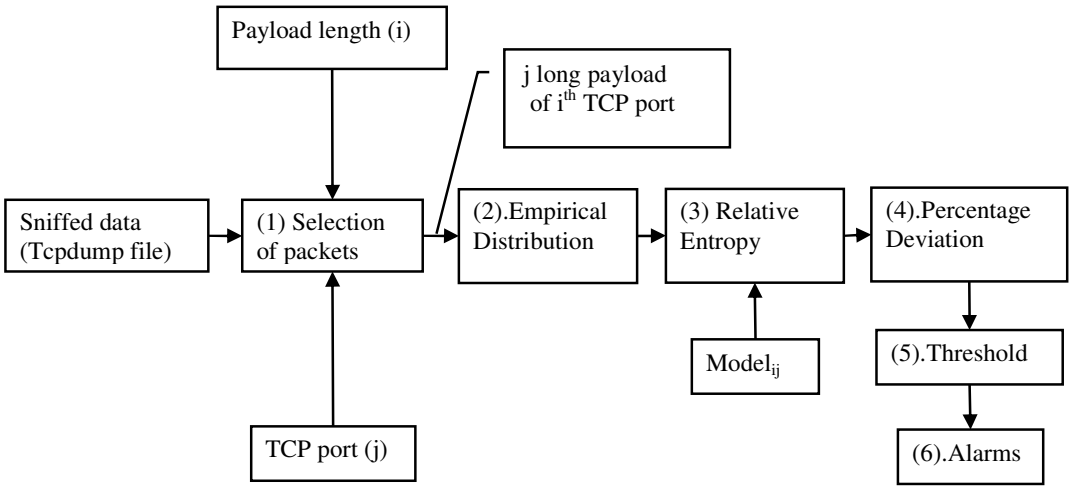

Figure 4.10. Block diagram of detection process

4.2.1. Relative Entropy

Relative entropy shows the difference between the distribution of the packet class in the current network traffic and the baseline distribution. If this difference is too large, it indicates that a portion of appearance of some byte classes in payloads of training data changes significantly in payloads of test data. This serves as an indication of anomalies in payloads of TCP protocols. In other words, TCP protocol of our concern is used in an anomalous way, that doesn't appear in normal payloads of training data.

Relative entropy is defined as:

$$D_{\tilde{P}//P}(w) = \tilde{P}(w)\log\frac{\tilde{P}(w)}{P(w)} \tag{4.2.1}$$

Here, P is the baseline distribution obtained from Maximum Entropy estimation. This gives us a quantitative value that describes the distortion for each byte class w from that of the baseline distribution. If $D_{\tilde{P}//P}(w)$ is large, the empirical distribution of test data payload deviates a lot from that in the baseline distribution, which is used as an indication of anomalies.

Because it is not practical to save relative entropy of all 256 classes of bytes, it is chosen to save maximum value of relative entropy of all byte classes. So, all decision is done according to that maximum value. From following figures relative entropy for all ports and all data units can be seen.

From figures below it can be seen that in connection based analysis

4.2.2. Percentage Deviation

Percentage deviation algorithm is based on median calculation of input vector of $\underline{x}$,

$$PD_X = (x - medain(\underline{x})) * 100 \tag{4.2.2}$$

In probability theory and statistics, a median is a number dividing the higher half of a sample, a population, or a probability distribution from the lower half. The median of a finite list of numbers can be found by arranging all the observations from lowest value to highest value and picking the middle one [aabf].

After calculation of maximum of relative entropy for all classes percentage deviation is applied to output of 3rd step in figure 2. This helps us increasing detection rate while decreasing false alarm rate.

Also to apply thresholding more easily resulting percentage deviation is normalized between 0 and 100.

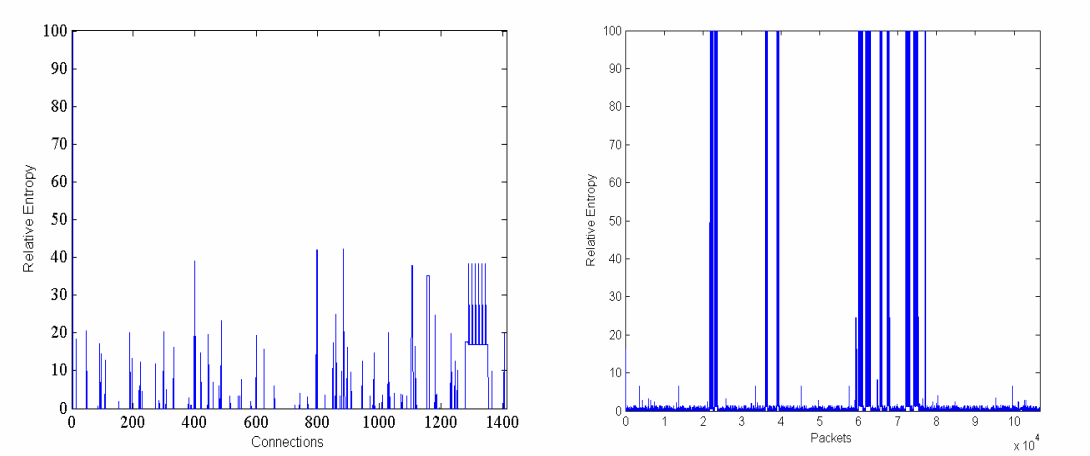

Figure 4.11. Percentage deviation of relative entropies for TCP port 25 . These are results for packet based analysis (left) and connection based analysis (right) for 4th and 5th weeks of test data.

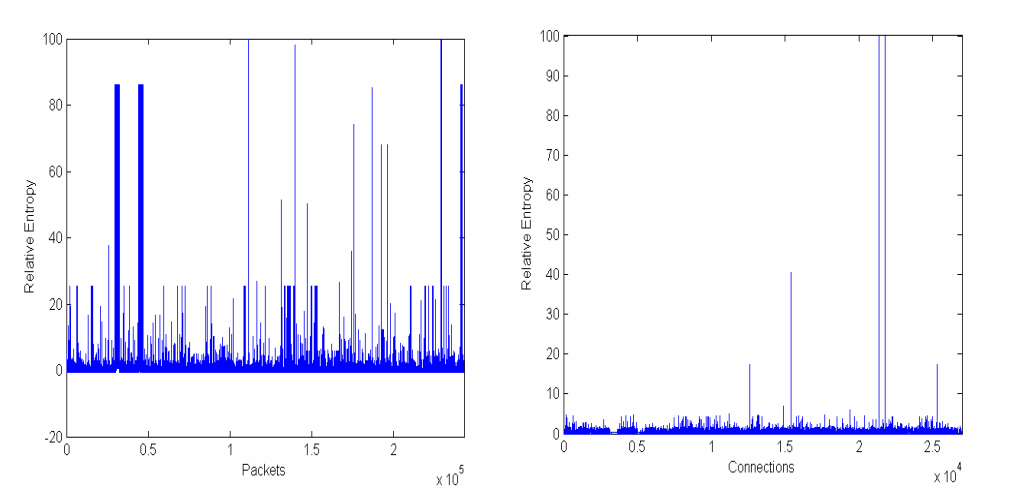

Figure 4.12. Percentage deviation of resulting relative entropies for TCP port 21 . These are results for packet based analysis (left) and connection based analysis (right) for 4th and 5th weeks of test data.

4.3. Detection

After calculation of percentage deviation alarms are generated with thresholding. If resulting percentage deviation is higher than threshold alarm is generated. When an alarm is generated in time instance of any attack it is accepted to be true positive, else it is accepted to be false positive.

Threshold is selected manually. For different threshold values different FP and FN rates are obtained. Threshold that gives maximum C_{ID} value is selected.

5. RESULTS AND PERFORMANCE ANALYSIS

5.1. Introduction

As mentioned before our intrusion detection method is tested with DARPA 1999 dataset. Also calculation method of (FP), false negative (FN) and base rate (B) with will be given.

5.2. Results

5.2.1. Calculation of False Positive, False Negative and Base Rate

Explanations about terms FP, FN and base rate was given previously in section 3.2. Following equations can be used to calculate these ratios.

$$false \quad positive \quad rate = \frac{number \quad of \quad false\, alarms}{number \quad of\, clean \quad samples} \tag{5.1}$$

$$false \quad negative \quad rate = \frac{number \quad of \quad missed\, attack \quad samples}{number \quad of\, attack \quad samples} \tag{5.2}$$

$$Base \quad Rate \quad = \frac{number \quad of \quad attack\, samples}{number \quad of\, all \quad samples} \tag{5.3}$$

$$True \quad Positive \quad Rate = 1 - false \quad negative \quad rate \tag{5.4}$$

$$True \quad Negative \quad Rate = 1 - false \quad positive \quad rate \tag{5.5}$$

To calculate these rates the number of samples that contains attacks should be known.

In DARPA 1999 dataset the information given about attack instances are:

- Name: name of the attack
- Category: in which category the attack can be placed.
- Start Time: start date and time of the attack
- Duration: duration of attack in seconds, minutes and hours.
- Attacker: Attackers IP address/addresses
- Victim: Victims IP address/addresses
- Username: Username used in attack, if used.
- Ports: If attacker used well known ports in attacker/victim machine, these ports are given.

From given properties of attacks in DARPA 1999 data set it's hard find out number of packets or connections generated by attacks for calculation of false positive and false negative rates exactly. Generally, if an intrusion detection system produces an alarm, at the time attack takes place, this alarm is accepted as true positive. If generated alarm is out of all attack times it is accepted as false positive or in other words false alarm. According to these given information about attacks, to find attack rate time information can be used.

Time intervals for all attacks are known. Packets that fall into time intervals where attacks takes place are assumed to be data samples of attacks. Other packets that fall out of attack intervals are known as clean attack samples. So with number of attack and clean samples base rate in Equation (5.3) can be calculated.

Alarm generation and detection phases explained in section 4.2. But it is useful to remind alarm generation process. Relative entropy is calculated between maximum entropy model and empirical distribution of test data. The resulting of this relative entropy calculation is a vector with 256 length. If maximum of this vector is above a threshold an alarm is generated. Time of generated alarm is known. If an alarm is generated in time that an attack takes place it is accepted to be true positive, and the attack is accepted to be detected. After finding detected attacks, missed attacks are determined.

To find false positive and false negative rate number of samples that fall into truly detected attacks and missed attacks must be known. To find out these values a method similar with finding number of all attack samples is followed. As truly detected attacks and time intervals that these attacks takes place are known, if we find packet that fall into these attack intervals, number of samples of truly detected attacks can be assumed. Number of samples of missed attacks can be found with the same way. With these informations false positive and fals negative rates can be easily calculated with equations 5.1 and 5.2. true positive and true negative rates can be calculated with equations 5.4 and 5.5.

5.2.2. C_{ID}, PPV, NPV

In the following table CID and all other performance metrics can be found. It should be pointed that calculation of these ratios are independent from each other. That means from

Table 5.1. CID False Positive False and False Negative Rates. Notice that these values are shown in percentage.

Model	Base Rate (%)	FP (%)	TP (%)	TN (%)	FN (%)	PPV (%)	NPV (%)	Cid (%)
TCP port 80 per connection	6,98	0,47	99,77	99,53	0,23	94,05	99,98	92,78
TCP port 80 Per Packet	63,58	0,00	94,29	100,00	5,71	100,00	90,93	81,43
TCP port 25 per connection	9,56	15,20	99,96	84,80	0,04	41,01	100,00	49,85
TCP port 25 per packet	8,07	0,36	100,00	99,64	0,00	96,07	100,00	95,03
TCP port 21 per connection	7,57	3,52	96,48	99,07	0,93	69,74	99,92	73,26
TCP port 21 per packet	2,42	9,74	96,98	90,26	3,02	19,80	99,92	43,05
TCP port 23 per connection	59,67	8,26	97,55	91,74	2,45	94,59	96,20	71,57
TCP port 23 per packet	11,68	1,64	100,00	98,36	0,00	88,98	100,00	87,36

From table above it can be seen that C_{ID} value for TCP port 21 with per packet analysis is relatively low with compared to other TCP protocols. This is because of high false alarm rate in FTP protocol per packet analysis. This high false alarm rate is not only encountered in Me-PAYL but also it is the condition in POSEIDON and PAYL benchmarks. In [8] it is said that all these packets that causes false positives are sent by the same source host, which is sending FTP commands in a way that is typical of the Telnet protocol (one character per packet, with the TCP flag PUSH set). These packets are marked as an attack because the training model does not contain this kind of traffic over the FTP control channel port, although it is normal traffic.

This high false alarm rate does not occur in per connection analysis. This is because all these packet that causes false alarms are sent form same source host and from few connections. Many packets can flow in one TCP connection. This is the reason of lower false positives in connection based analysis for only FTP port and this dataset. From Table 5.5 it can be seen that for all other TCP ports packet based analysis create lower false positive rates.

During experiments with PAYL and POSEIDON they found the same behavior. For this reason they decided to present benchmarks results of PAYL and POSEIDON also without taking into account these false positives that is nearly 3000 false alarms. The reader must consider this difference when comparing results.

In next section ROC curves and detections of three methods are compared. Because they ignore 3000 false alarms for port 21 Roc curves of POSEIDON and PAYL for port 21 can not be compared with Me-PAYL.

5.2.3. ROC Curves

As can be seen from following figures for different TCP protocols packet based analysis and connection based analysis gives different results. It will be useful to mention that calculation method of false positive and false negative rates are done with approximation of packets of attacks (See Section 5.2.1). Note that the x-axis and y-axis present false positive rate and detection rate respectively in figures.

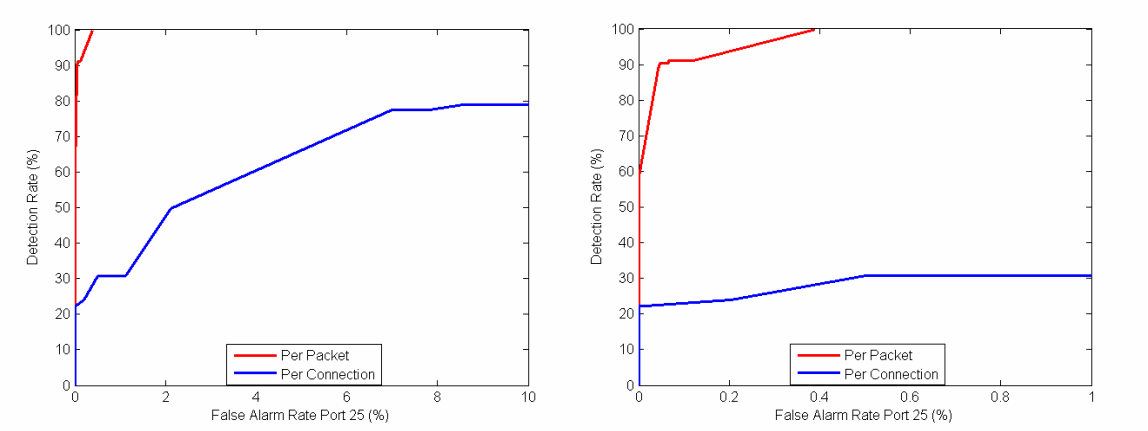

Figure 5.1. ROC curves for TCP port 25. As can be seen from figure (left), for port 25 packet based analysis gives better results than connection based analysis. Zoomed version of this figure can be seen at right.

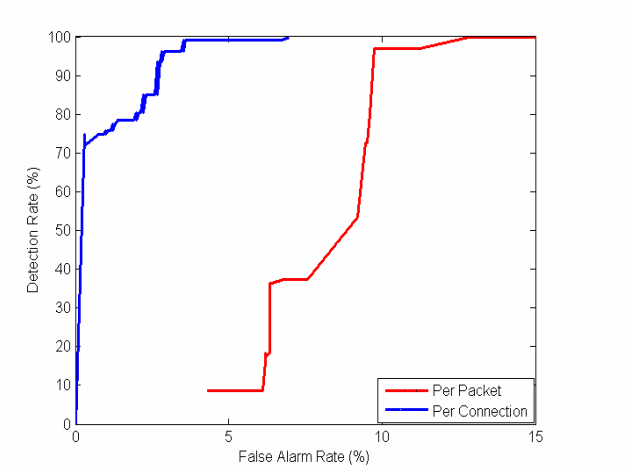

Figure 5.2. ROC curves for TCP port 21. For port 21 connection based analysis gives better results than packet based analysis.

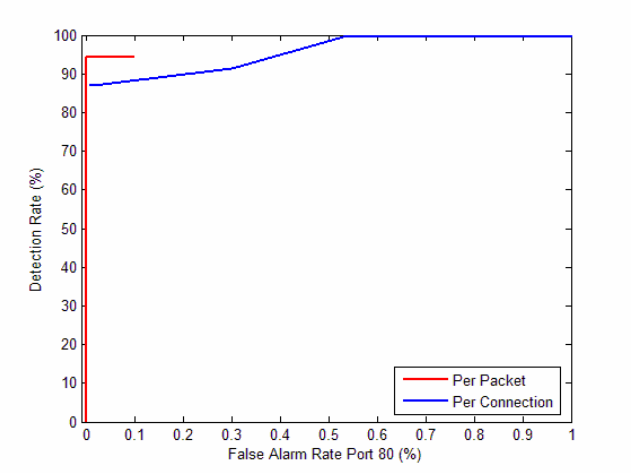

Figure 5.3. ROC curves for TCP port 80. For port 80 connection based analysis gives better results than packet based analysis. It can be seen that packet based analysis gives lower false alarm rates but also lower detection rates.

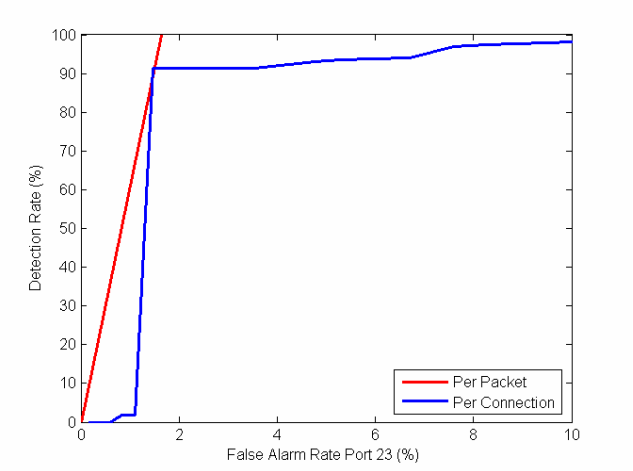

Figure 5.4. ROC curves for TCP port 23. For port 23 packet based analysis gives better results than connection based analysis.

It will be useful to note that the x-axis of ROC curves are not equal to each other. X-axis of curves are adjusted for to show details better.

From above figures we can see that the payload-based model is very good at detecting the attacks to ports 21, 23 25 and 80. For port 21, the attackers often first upload some malicious code onto the victim machine and then login to crash the machine or get

root access, like casesen and sechole. The test data also includes attacks that upload/download illegal copies of software, like warezmaster and warezclient. These attacks were detected easily because of their content which were rarely seen executable code and quite different from the common files going through FTP.

For port 80, the attacks are often malformed HTTP requests and are very different from normal requests. For instance, crashiis sends request "GET ../.."; apache2 sends request with a lot of repeated "User-Agent:sioux\r\n", etc. Using payload to detect these attacks is a more reliable means than detecting anomalous headers simply because their packet headers are all normal to establish a good connection to deliver their poison payload [7]. That's because their content are quite free style and some of the attacks are well hidden. For example, the framespoofer attack is a fake email from the attacker that misdirects the victim to a malicious web site. The website URL looks entirely normal. Malformed email and telnet sessions are successfully detected, like the perl attack which runs some bad perl commands in telnet, and the sendmail attack which is a carefully crafted email message with an inappropriately large MIME header that exploits a buffer overflow error in some versions of the sendmail program.

For ports23 and 25, the packet-based models are better than the connection-based models. This is likely due to the fact that the actual exploit is buried within the larger context of the entire connection data, and its particular anomalous character distribution is swamped by the statistics of the other data portions of the connection. Per packet model detects this anomalous payload more easily.

There are many attacks that involve multiple steps aimed at multiple ports. If we can detect one of the steps at any one port, then the attack can be detected successfully. Thus if we correlate the detector alerts from all the ports and plot the overall performance the results will be better. Above ROC figures only plotted with detections of every TCP ports. In calculation of detection rates of Me-PAYL alarm was not correlated.

To compare results with other payload based methods ROC curves and C_{ID} can not be used. This is because there is no information given in [7] or [8] results in C_{ID}. Instead

FP and TP rates are used to compare methods. But it is still not very fair comparison due to restrictions of other methods xplained in section 5.2.2.

5.3. Comparison With Other Payload Based IDS Methods

In this section comparison of payload based IDS methods with Me-PAYL method will be done. To compare our model with PAYL, we must apply the same restrictions and conditions used by Wang and Stolfo [7]. They focus only on inbound TCP packets, with data payload, directed to hosts 172.016.0.0/16 and ports 1-1024. This reduces background traffic and gives them lower false alarm rates. But in our benchmark all incoming and outgoing packets to all hosts are examined. To examining all packets without regarding their destination hosts is more realistic approach. Reader should take into account this fact while comparing results of methods.

Note that POSEIDON and PAYL algorithms are packet based systems. So they can not be compared with test results of connection based Me-PAYL system. This is the reason for only results of packet based Me-PAYL test for DARPA 1999 dataset is shown in following tables. Results of connection based Me-PAYL benchmark can only be compared with Me-PAYL packet based IDS system.

Table 5.2. Comparison between PHAD, POSEIDON and Me-PAYL detection rates. Detection rates of PHAD and POSEIDON are taken from [8]

Type	Attack	PHAD		POSEIDON		Me-PAYL	
Probe	ntinfoscan	66,67%	(2/3)	100,00%	(3/3)	100,00%	(3/3)
	apache2	100,00%	(3/3)	100,00%	(3/3)	100,00%	(3/3)
	back	0,00%	(0/4)	100,00%	(4/4)	100,00%	(4/4)
Denial of service	crashiis	71,43%	(5/7)	100,00%	(7/7)	71,43%	(5/7)
	phf	66,67%	(2/3)	100,00%	(3/3)	100,00%	(3/3)
Remote to Local	ppmacro	33,34%	(1/3)	100,00%	(3/3)	66,66%	(2/3)
Overall detection rate		65,00%	(13/20)	100,00%	(20/20)	85,00%	(17/20)

Table 5.3. Comparison between PAYL, implementation of PAYL (PAYL exp) in [8] , POSEIDON and Me-PAYL; DR stands for detection rate, while FP is the false positive rate. Detection rates of PAYL and POSEIDON are taken from [8]

		PAYL	POSEIDON	Me-PAYL Per Packet
HTTP	DR	89%	100%	94,29%
	FP	0,17%	0,00%	0,00%
FTP	DR	95,50%	100%	96,98%
	FP	1,23%	11,31%	9,74%
Telnet	DR	54,17%	95,12%	100%
	FP	4,71%	6,72%	1,64%
SMTP	DR	78,57%	100%	100%
	FP	3,08%	3,69%	0,36%
Overall DR with FP<%1		58,8% (57/97)	73,2% (71/97)	62,88% (61/97)

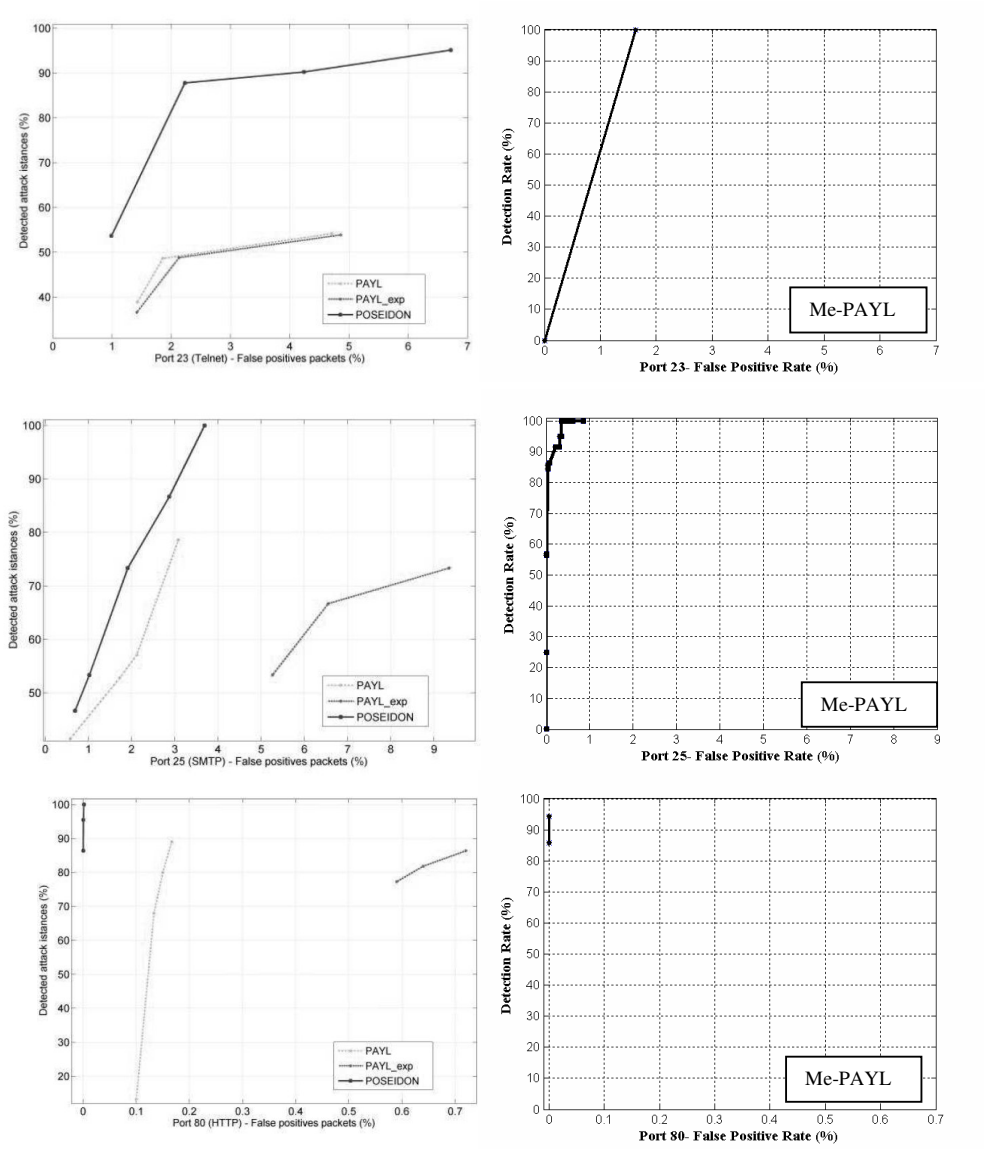

Figure 5.5. Detection rates of payload based IDS models. This figure contains comparative detection rates taken from [8] at right and obtained result from benchmark of Me-PAYL method at left. Because of some restrictions for port 21 in calculation of false alarm rates of POSEIDON, PAYL and PAYL_exp, port 21 does not shown in this figure (see section 5.3)

.

6. CONCLUSION

Payload based anomaly detection is a very effective method for detecting application layer attacks. Application layer is the highest layer in TCP/IP model. Application layer attacks means attacks that produce normal connections in network but have effect on the application layer protocols in victim machine. An example for these attacks is worm. Also payload based methods can detect illegal use of TCP ports. For example a per-to-peer program can communicate with TCP port 80, which is designated for use of web traffic. This approach may also have great value in detecting slow and stealthy worm propagations that may avoid activities of a bursty nature [8]. However in this work we did not tested all these properties because there is no an IDS evaluation dataset. The effectiveness of payload based methods in detecting worms is tested in [23].

The maximum entropy estimation algorithm proposed in [22] is taken as one of the reference models for this thesis work. The Principle of Maximum Entropy is based on the basic assumption that when estimating the probability distribution, you should select that distribution which leaves you the largest remaining uncertainty (i.e., the maximum entropy) consistent with initial constraints. That way any additional assumptions or biases in calculations are not introduced.

In this thesis a new payload based anomaly detection method based on maximum entropy estimation is proposed. The starting point of this method is payload based anomaly detection method (PAYL) proposed in [8]. The efficiency PAYL method is improved by using maximum entropy estimation. And this proposed method Both packet based and connection based method are tested.

Connection based analysis models all data going through one TCP connection and uses again payloads of TCP connections in test data. Packet based analysis uses payloads of packets in both training and test data. For ports23 and 25, the packet-based models are better than the connection-based models. This is likely due to the fact that the actual exploit is buried within the larger context of the entire connection data, and its particular

anomalous character distribution is swamped by the statistics of the other data portions of the connection. Per packet model detects this anomalous payload more easily. For port 80 connection based analysis gives better results. The of reason this is explained in section 5.2.3. For port 21 both packet based and connection based analysis gives same detection rates. But in packet based analysis due to some anomalous packets that do not contain any attack in test data for port 21. These packets do not appear in training data so are detected as attacks. These packets result more false positives in packet based analysis (see section 5.2.3).

Initially in modeling phase payloads of packets/connections are extracted from tcpdump file according to their destination port number. If destination port number of packets/connections is out of interest it is ignored. This is done with an executable program written in C++ language. The outputs of this execution process, payloads, are then fed into empirical distribution modeling process. Empirical distribution of all payloads in training data is calculated and according to their length overall empirical distributions are updated with these new incoming payloads empirical distribution.

When all empirical distributions of all length for a specific TCP port number are calculated a matrix with size 256 times maximum payload length is obtained. To overcome high model space clustering process is performed. Clustering here is simply merging similar models. Merging similar neighboring models (see section 4.1.5) results more accurate models, if there are not enough samples in training data. We can say that clustering is not only decreases model space but also increases efficiency of method with more accurate models.

In PAYL algorithm empirical distributions of payload of packets are used as models and Mahalanobis distance is used for alarm generation. In Me-PAYL method maximum entropy estimated empirical distributions of payloads are used as models. This extra modeling scheme increases computational load in modeling process but also gives us an increase in detection rate and decrease in false alarm rate. In alarm generation phase, using relative entropy instead of Mahalanobis distance gives us lower memory usage because we do not need to save variances of all models.

Me-PAYL method was tested with DARPA 1999 IDS evaluation dataset which is created in MIT Lincoln Laboratories. DAPRA 1999 dataset is the largest dataset with complete payloads [8]. Because of this reason in benchmark of payload based IDS methods this dataset is used.

Another payload based IDS method is POSEIDON proposed in [7]. It is a modified version of PAYL model. In POSEIDON a self organizing map is used for classification of payloads instead of their lengths. This method increases detection rates while decreasing false alarm rates. In section 5.3, for comparison of methods, tables and ROC curves of benchmarks of PAYL, POSEIDON and Me-PAYL methods with DARPA 1999 dataset can be found.

When comparing three methods, reader should take into account of restrictions of other methods while testing and modeling. Initial restriction about POSEIDON and PAYL is that they focus only on inbound TCP packets, with data payload, directed to hosts 172.016.0.0/16 and ports 1-1024. This reduces background traffic and gives them lower false alarm rates. But in our benchmark all incoming and outgoing packets to all hosts are examined. Examining all packets without regarding their destination hosts is more realistic approach. When considering the restrictions in PAYL and POSEIDON benchmarks, the test result of Me-PAYL method are obtained in relatively high background traffic when compared with PAYL and POSEIDON benchmarks. By ignoring packets going from internal hosts through outside hosts (see figure 1.2), they ignore traffic going from inside host through thousands of emulated web servers and hundreds of emulated workstations.

The experimental results indicate that the method is effective at detecting attacks. In the 1999 DARPA IDS dataset, the best trained model for TCP traffic detected 71 attacks out of 97 with every port's false positive rate lower than 1%. For port 80, it achieves almost 100% detection rate with around 0.4% false positive rate. Unlike PAYL method it is not only does it have favorable detection results for port 80 but also it has favorable results for ports 21, 23 and 25. In both connection based and packet based analysis achieves high detection rates. For ports 23 and 25 packet based analysis results lower false positives while for port 21 and 80 connection based analysis results lower false positives.

For port 21 high false positive rate in packet based analysis is due to the dissimilarities some of FTP packets in test data with that in training data.

In our future work, we plan to evaluate the technique in live environments, implement and measure the costs and speed of the method. Also the efficiency of this method will be tested in worm detection and illegal use of TCP ports with other protocols. There are many attacks that involve multiple steps aimed at multiple ports. If we can detect one of the steps at any one port, then the attack can be detected successfully. Thus if we correlate the detector alerts from all the ports and plot the overall performance we can expect better results.

APPENDIX A: MATLAB CODES

In this part explanation of codes that are used to implement both modeling and detection phases. Also executable programs that are used to extract payloads from tcpdump file will be explained. The implementation is evaluated in MATLAB R12 environment. We choose MATLAB as it provides efficient computations on vectors and matrices. Below the MATLAB functions are listed. Explanations of operations of functions are given with comments in codes.

Main codes:

- Main_code.m: Contain all steps of modeling and detection processes.
- Per_conn_emp.m: Calculate overall empirical distributions for connection based and packet based analysis.
- Per_conn_emp_ip.m: Same with Per_conn_emp.m but can be used for input files that contains destination IP of payloads.
- cluster_model.m: Perform clustering operation.
- Max_Ent_Model.m: Calculate maximum entropy estimation of input empirical distribution.
- Max_Ent_Model_avg.m: Same with Max_Ent_Model.m but has higher stopping threshold.
- detect_cluster_ip.m: Calculate relative entropy values between models and payloads of text data.
- perdeviation.m: Percentage Deviation Calculation.
- detection_performance.m: Include all steps or alarm generation detection and calculation of performance metrics.
- Roc_figure.m: Use different threshold values for drawing ROC curves.

Functions:

- byt_fq.m: Calculate empirical distribution of input vector.

- read_conn_ip.m: Read one line of file that contains payload, destination port source port and time information.
- B0_inv.m: Apply inverse of B0 to vector d.
- bfgs_rec.m: Compute $inv(B_k + 1) * d$ using BFGS inverse recursion.
- lbfgs.m: L-BFGS scheme to minimize a log likelihood function.
- line_search.m: Find approximate solution to the line search problem.
- Log_Likelihood_func.m: Calculation of the function that will be minimized with LBFGS algorithm.
- feature_matrix2.m: Form candidate feature matrix.

Data files:

- attacks_21.mat: Attack instances in dataset for TCP port 21
- attacks_23.mat: Attack instances in dataset for TCP port 23
- attacks_25.mat: Attack instances in dataset for TCP port 25
- attacks_80.mat: Attack instances in dataset for TCP port 80

REFERENCES

1. Abraham, A. and C. Grosan, "Evolving Intrusion Detection Systems", *Genetic Systems Programming: Theory and Experiences , volume 13 of Studies in Computational Intelligence*, pages 57-80. Springer, Germany, 2006.

2. Sundaram, A., "An Introduction to Intrusion Detection", *ACM Cross Roads*, Vol. 2, No. 4, April 1996.

3. Lo, J., *Denial of Service or Nuke Attacks,* http://www.irchelp.org/irchelp/security/

4. Bace, R., P. Mell, " Intrusion Detection Systems ", *NIST Special Publication on IDS*, Nov 2001.

5. Kendall, K., "A Database of Computer Attacks for the Evaluation of Intrusion Detection Systems", *S.M. Thesis, MI T Department of Electrical Engineering and Computer Science*, June 1999.

6. Lippmann, R., J. W. Haines, D. J. Fried, J. Korba and K. Das, "The 1 999 DARPA off-line intrusion detection evaluation ", *Computer Networks,* 34(4): 579-595, 2000.

7. Wang, K. and S. J. Stolfo, " Anomalous Payload-Based Network Intrusion Detection." , *E. Jonsson, A. Valdes, and M . Almgren, editors, RAID '04: Proc. 7th Symposium on Recent Advances in Intrusion Detection*, volume 3224 of LNCS, pages 203–222, Springer-Verlag, 2004.

8. Bolzoni, D., S. Etalle, P. H. Hartel and E. Zambon, " Poseidon: a 2-tier anomaly-based network intrusion detection system.", *Proceedings of the 4th IEEE international Workshop on Information Assurance,13-14 April 2006, Egham, Surrey, UK*, pages 144–156, 2006.

9. Chuanyi, J. and M. Thottan,"Anomaly detection in ip Networks", *IEEE Transac-*

tions on Signal Processing, 51(8):2191-2204, 2003.

10. Alhamaty, M., A. Yazdian and F. Al-qadasi , "Intrusion Detection System Based on The Integrity of TCP Packet", *Transactions on Engineering, Computing and Technology*, V11 ISSN 1305-5313, February 2006.

11. Vladimir, G. and J. E. Johnson, "Multidimensional Network Monitoring for Intrusion Detection", *CoRR cs.CR/0206020*, 2002.

12. Zhang, Y. , Z. Ge , M. Roughan and A. Greenberg, " Network Anomography ", *Proceedings of the Internet Measurement Conference (IMC '05)*, Berkeley, CA, USA, October 2005.

13. Chin-Tser, H., S. Thareja and Y. J. Shin, "Wavelet-based Real Time Detection of Network Traffic Anomalies.", *International Journal of Security,* Vol. 6, No. 3, pp. 309-320, 2008.

14. Rawat, S. and C.S. Sastry, " Network intrusion detection using wavelet analysis.", *Lecture Notes in Computer Science*, 3356. Springer, Hyderabad, India. pp. 224-232, January 28, 2005

15. Seong, S. K., A. L. N. Reddy and M. Vannucci, "Detecting traffic anomalies through aggregate analysis of packet header data", *Lecture Notes in Computer Science (LNCS)*, vol. 3042, pp. 1047-1059, Athens, Greece, May 2004.

16. Seong, S. K., A. L. N. Reddy and M. Vannucci, "Detecting Traffic Anomalies using Discrete Wavelet Transform.", Proceedings *of International Conference on Information Networking (ICOIN),* vol. III, pp. 1375-1384, 2004.

17. Seong, S. K. and A. L. N. Reddy , " Image-based Anomaly Detection Technique: Algorithm, Implementation and Effectiveness.", *IEEE Journal on Selected Areas in Communications High-Speed Network Security*, Volume: 24, Oct. 2006.

18. Wang, H., D. Zhang and K. G. Shin, "Change-Point Monitoring for the Detection of DoS Attacks", *IEEE Transactions on Dependable and Secure Computing,* Vol. 1, No. 4, October –December 2004.

19. Labib, K and V. R. Vemuri, "Detecting and Visualizing Denial-of-Service and Network Probe Attacks Using Principal Component Analysis",*Third Conference on Security and Network Architectures*, SAR'04, La Londe, France, June 2004.

20. Qingtao, W. and Z. Shao, "Network Anomaly Detection Using Time Series Analysis", *Proceedings of the Joint International Conference on Autonomic and Autonomous Systems and International Conference on Networking and Services*, (ICAS/ICNS 2005) 0-7695-2450-8/05, IEEE, 2005.

21. Damiano, B. and S. Etalle, "APHRODITE: an Anomaly-based Architecture for False Positive Reduction", *Technical Report TR-CTIT-06-13*, University of Twente, Netherlands, 2006.

22. Yu, G, A. McCallum and D. F. Towsley, "Detecting Anomalies in Network Traffic Using Maximum Entropy Estimation.", *Internet Measurement Conference*, 345-350, 2005.

23. Ke, W. and S. J. Stolfo, " Anomalous Payload-based Worm Detection and Signature Generation. ", *A. Valdes and D. Zamboni, editors, RAID '05: Proc. 8th International Symposium on Recent Advances in Intrusion Detection*, volume 3858 of LNCS, pages 227–246. Springer-Verlag, 2006.

24. Costa, M., J. Crowcroft, M. Castro , A. Rowstron, L. Zhou, L. Zhang and P. B. Vigilante, "End-to-end Containment of Internet worms.", SOSP *'05: Proc. 20th ACM Symposium on Operating Systems Principles*, pages 133–147. ACM Press, 2005.

25. Kruegel, C. , T. Toth and E. Kirda,"Service specific anomaly detection for network intrusion detection.", *SAC '02: Proc. 2002 ACM Symposium on Applied*

Computing, pages 201–208. ACM Press, 2002.

26. Mahoney, M. and P. K. Chan,“ Learning nonstationary models of normal network traffic for detecting novel attacks.”, *KDD ’02: Proc. 8th ACM SIGKDD International Conference on Knowledge Discovery and Data mining*, pages 376–385, ACM Press, 2002.

27. Gu, G., P. Fogla, D. Dagon, W. Lee and B. Skoric, “An Information-Theoretic Measure of Intrusion Detection Capability.", *Technical Report GIT-CC-05-10*, College of Computing, Georgia Tech, 2005.

28. Pietra, S. D., V. D. Pietra and John Lafferty, “Inducing Features of Random Fields. ”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v.19 n.4, p.380-393, April 1997.

29. Mishra, S. K. , “Median as a weighted arithmetic mean of all sample observations.”, *Economics Bulletin*, Volume: 3, 2004.

REFERENCES NOT CITED

Gilbert, A. C., "Multiscale Analysis and Data Networks", *Applied and Computational Harmonic Analysis 10*, 185–202, 2001.

Mahoney, M. V. and P. K. Chan,"An Analysis of the 1999 DARPA/ Lincoln Laboratory Evaluation Data for Network Anomaly Detection", TR CS-2003-02.

Kumar, S., " Classification and Detection of Computer Intrusions. " Ph.D. Dissertation, August 1995.

Peddabachigari, S., A. Abraham, C. Grosan and Johnson Thomas, "Modeling intrusion detection system using hybrid intelligent systems", *Journal of Network and Computer Applications*, v.30 n.1, p.114-132, January 2007.

Haines, J., L. Rossey, R. Lippmann and R. Cunnigham, "Extending the 1999 Evaluation", *The Proceedings of DISCEX 2001*, Anaheim, CA, June 11-12 2001.

Debar, H., M. Dacier, A. Wespi and S. Lampart, "An Experimental Workbench for Intrusion Detection Systems", *Research Report RZ 2998, IBM Research Division, Zurich Research Laboratory, 8803 Ruschlikon, Switzerland*, March 9, 1999.

Ptacek, T. H. and T. N. Newsham, "Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection", *Secure Networks, Inc. Report*, January 1998.