Fall 2010

A Critical Analysis of Payload Anomaly-Based Intrusion Detection Systems

Anthony F. Mercurio
Regis University

Regis University
College for Professional Studies Graduate Programs
Final Project/Thesis

Disclaimer

A CRITICAL ANALYSIS OF PAYLOAD ANOMALY-BASED INTRUSION DETECTION SYSTEMS

A THESIS

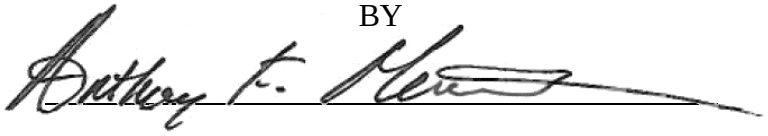
SUBMITTED ON 30 OF OCTOBER, 2010

TO THE DEPARTMENT OF INFORMATION SYSTEMS, OF THE SCHOOL OF COMPUTER & INFORMATION SCIENCES

OF REGIS UNIVERSITY

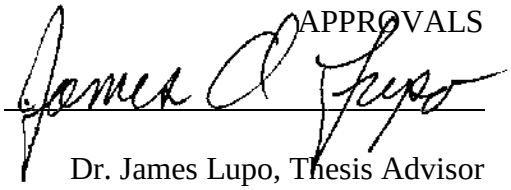
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS OF MASTER OF SCIENCE IN INFORMATION ASSURANCE

BY

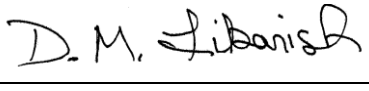


Anthony F. Mercurio

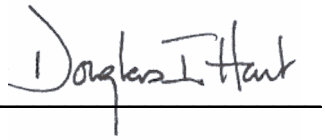
APPROVALS



Dr. James Lupo, Thesis Advisor



Daniel M. Likarish



Douglas I. Hart

Abstract

Examining payload content is an important aspect of network security, particularly in today's volatile computing environment. An Intrusion Detection System (IDS) that simply analyzes packet header information cannot adequately secure a network from malicious attacks. The alternative is to perform deep-packet analysis using n-gram language parsing and neural network technology. *Self Organizing Map* (SOM), *PAYL over Self-Organizing Maps for Intrusion Detection* (POSEIDON), *Anomalous Payload-based Network Intrusion Detection* (PAYL), and *Anagram* are next-generation unsupervised payload anomaly-based IDSs. This study examines the efficacy of each system using the design-science research methodology. A collection of quantitative data and qualitative features exposes their strengths and weaknesses.

Acknowledgements

I sincerely appreciate everyone who has helped, guided, and supported me in the accomplishment of this paper. I owe a very special thanks to my wife and children whose tremendous patience has allowed me to pursue my education throughout the years. In addition, I would like to thank Professor Dan Lakarish and Dr. James Lupo for guiding me in the right direction; their relentless help and patience is greatly appreciated. Furthermore, I am indebted to my loving parents who immigrated to this country and allowed me to fulfill my dreams and aspirations. Finally and most importantly, I thank the Lord for the strength and willingness to continue in times of difficulty.

Table of Contents

List of Figures

List of Tables

Chapter 1 – Introduction

The primary function of the IDS is to protect the Confidentiality, Integrity, and Availability (CIA) of information and information systems. It is an integral part of well-managed comprehensive security enclave. The IDS is a vital tool as the degradation or non-availability of network resources could be detrimental to business, particularly as network applications and protocols become vulnerable to attacks. While the IDS cannot provide total security, it is a means to deter malicious attacks from propagating freely throughout networks.

In general, IDSs are separated into two broad categories. Anomaly-based systems compare attack-free data to network traffic where anomalous events are identified as deviations from the norm, while misuse-based systems match signatures or unique character strings to known attacks. This study focuses on anomaly detection, where specially designed systems analyze packet data content for anomalous or suspicious activity. The purpose of this thesis is to determine whether payload anomaly-based IDSs are effective at detecting malicious attacks.

Problem Statement

According to Lee et al., (2003) most IDSs monitor threats at the lowers layers of the TCP/IP protocol stack, thereby reducing the ability to detect higher-level threats. Network packet payload analysis is a foreseeable solution since application attacks are embedded in the payload versus header portion of the Internet Protocol (IP) packet. However, the ability to detect payload embedded attacks remains a continuous challenge due to the complexity of high-dimensional data and dynamic structuring of protocols. SOM, POSEIDON, PAYL, and Anagram are examples of the latest advances in intrusion detection technology. These systems should be evaluated to determine their completeness and accuracy in which they detect threats.

Significance of Study

The primary purpose of this research is to determine the effectiveness of payload anomaly-based IDSs and their respective classification and analysis techniques. The knowledge gained allows practitioners to execute better-informed decisions. For instance, management-oriented audiences may need to determine whether an investment is worthwhile, whereas technology-oriented audiences may need to determine the practicality of a new system or design. Exposing strengths and weaknesses to define what a system can or cannot accomplish is highly important. New knowledge of performance may provide a better understanding of how systems interface within the real-world.

Scope

This study begins with a broad overview and continues on to explain the details of payload content analysis, design, and evaluation. It is based primarily on the collection of quantitative data to include the function and capability of SOM, POSEIDON, PAYL, and Anagram. The objective is to evaluate the efficacy of these systems and to acquire further knowledge of payload anomaly detection. However, this study will not address management requirements, operation and maintenance, cost, and training as they fall outside the scope of the objectives.

Objectives

Objective I. Explore the methods and steps needed to analyze data payload content.

Objective II. Explore the methods to distinguish normal from unauthorized activity.

Objective III. Explore the methods of correlating alerts.

Objective IV. Explore and explain the effectiveness of packet payload classification and analysis techniques.

Outline

This chapter includes the introduction, significance of the study, problem statement, scope, and objectives. Chapter 2 presents background information and general concepts related to payload intrusion detection. Chapter 3 reviews relevant literature. Chapter 4 explains the research process. Chapter 5 is the taxonomy of payload anomaly-based IDSs. Chapter 6 is a brief overview of the alert correlation techniques. Chapter 7 describes the different phases of payload processing, along with general design requirements. Chapter 8 provides the evaluation while Chapter 9 captures the results. Finally, Chapter 10 provides the conclusion and final recommendation.

Chapter 2 – Background

In general, a Network Intrusion Detection System (NIDS) uses either an anomaly-based or signature-based approach. The signature-based system requires prior knowledge of an attack. Signatures are manually prepared by an administrator or analyst and input into the system as a reference of future attacks. A comparison is then accomplished with incoming traffic to detect intrusions. In contrary, the anomaly-based system detects intrusions by comparing normal attack-free traffic to incoming traffic. This reveals patterns that deviate from normal activity. There are significant differences between the two design philosophies (Hwang, Liu, & Chen, 2004).

Signature-based systems are based on the misuse model. They are unable to detect attacks without a signature match. In general, misuse detection produces fewer false alarms, but cannot detect unknown attacks. This approach requires substantial knowledge and experience with manually inputting signatures into the IDS. Thus, regular updates are needed to prevent attacks from reoccurring. Furthermore, signature matching has a limited capability to detect attacks from multiple connections such as the example with fragmented worms. Sophisticated attackers may exploit this vulnerability by using multiple points of entry to penetrate the network. SNORT and BRO are examples of misuse IDSs (Hwang, et al., 2004).

In contrary, anomaly-based systems are based on behavioral modeling. The primary advantage of these systems is their ability to detect unknown attacks. This approach uses statistical properties and mathematics to detect new attacks. For example, algorithms determine whether data is normal or anomalous. The term *normal* means the activity derived from its origin (Eskin, et al., n.d.). Therefore, a deviation from normal activity is categorized as anomalous or unknown. Unlike misuse IDSs, anomaly-based IDSs detect attacks from multiple

connections and perform with *unlabeled* (raw) data. SOM, POSEIDON, PAYL, and Anagram are examples of anomaly-based NIDS (Hwang, et al., 2004).

Importance of Payload Intrusion Detection

One may argue that organizations rely more heavily on payload anomaly-based intrusion detection for protection. Kiani, Clark, and Mohay (2008) posited that "75% of cyber attacks occur at the application layer, [while] 97% were vulnerable to web attacks" (p.47). Symantec corporations reported that 69% of vulnerabilities were caused by web services (Bolzoni, Crispo, & Etalle, 2007). The impact or loss of services can be detrimental to e-business, and potentially the economy. As previously mentioned, the IDS cannot provide complete security, but it does offer the capability to identify multiple levels of attacks. Cheema, Akram, and Iqbal (2009) demonstrate the effectiveness of analyzing payload content versus analyzing strictly IP header information.

The experiment was setup to compare six anomaly detectors, some that analyze header information and others that consider payload content. The 1999 DARPA Intrusion Detection dataset, Air Force Research Laboratory, n-gram word sequence, and public domain content were included in the test. The results showed major differences in the range of attacks detected strictly when analyzing payload content. The data set had a total of 201 instances (samples) with 58 different attacks. A total of 107 instances and 33 types of payload-based attacks were discovered. Thus, over 50% of the attacks were discovered using payload analysis as a criterion.

Ultimately, payload-based intrusion detection should overcome the limitations of signature-based systems. A major hindrance for systems such as BRO or SNORT is their lack of true performance. Before the IDS correctly identify a protocol violation, it must distinguish what protocol is actually in use. Another problem is dealing with unstructured protocols such as

TELNET, where any byte pattern may appear valid. While registering a port could fix the problem, hackers are still able to conceal traffic using non-standard ports. A NIDS cannot assume protocols map to a specific port (Dreger, Feldmann, Main, Paxson, & Sommer, n.d.). Thus, using port criteria to detect network attacks may willingly expose a network to vulnerabilities. According to Bolzoni, Crispo, and Etalle (2007), the inability to detect new attacks and frequent updating as the environment changes are other drawbacks associated with signature-based IDSs.

Threat Overview

According to Pfleeger & Pfleeger (2007), a threat is a "is a set of circumstances that has the potential to cause loss or harm" (p. 7). Wang and Stolfo (n.d.), group threats into five main categories (p.9):

- Scans and Probes: Surveillance and probing (e.g., port sweep)
- Denial of Service (DoS): An attempt to shutdown or deny services with false requests (e.g., SYN flood, and ping of death)
- Remote to Local (R2L): An unauthorized attack from a remote machine (e.g., password guessing, and buffer overflows)
- User to Root (U2R): An unauthorized attack from local super user (e.g., buffer overflow attack)
- Data: Examples include file manipulation and policy violations

As previously noted, network attacks may disrupt protocols and their ability to provide services. Several of the most common protocols vulnerable to attacks include Simple Mail Transfer Protocol (SMTP) to communicate e-mail; Hypertext Transfer Protocol (HTTP) to communicate web pages; File Transfer Protocol (FTP) to send and receive files; Terminal Emulation Protocol

(TELNET) to perform remote operations via host terminal; and Simple Network Monitoring Protocol (SNMP) to control network devices (Pfleeger & Pfleeger, 2007).

General Concepts

The general concepts discussed in this study are related to SOM, POSEIDON, PAYL, and Anagram. The SOM may function as individual anomaly detector or integrated to form a hybrid solution. POSEIDON uses an artificial neural network (SOM) and PAYL to execute the intrusion detection. Anagram is an alternative anomaly detector that employs advanced techniques to store signatures and to perform data analysis. Finally, PAYL is a system that performs anomaly detection using 1-gram data analysis. This section provides a brief introduction and explanation of core concepts discussed throughout the paper.

N-gram analysis is the primary means IDSs use to examine payload content. It is a *language parser*, a method to predict the next sequence in a data set. Demashek (1995) originally coined the term to define an order of items where *n* could be a symbol, letter, or word. N-grams come in different sizes such as a unigram, bigram, trigram, and so forth. The n-gram represents a string of characters using statistical properties, to detect anomalous byte sequences. Therefore, a string or *signature* is a unique pattern that identifies the similarity between an originating packet and malicious content (Parekh, et al., 2006).

Depren, Topallar, Anarim, and Ciliz (2005) argued that unsupervised anomaly payload-based systems are needed due to the limitations of rule-based and statistical-based IDSs. Typically, rule-based systems search for abnormal behavior that violate set rules, while statistical-based systems indentify normal behavior using data mining techniques. Unfortunately, these examples require manual updating from network administrators, known as *supervised learning*. However, the ideal approach is to employ *unsupervised learning*, which

does not require human interaction; systems are initially setup and run autonomously (Lichodzijewski, Heywood, & Heywood, 2002). Anomaly-based IDSs are examples of unsupervised learning techniques.

In 1982, Kuevo Kohonen intrigued the community with the first representation of an unsupervised Artificial Neural Network (ANN). Kohonen (1988) produced a low-dimensional map of high-dimensional data. Payload data points where mapped into a graphical format. Neural networks fall under the category of machine-learning systems, also known as self-learning systems. They have the "ability to change its execution strategy as it acquires new information" (Garcia-Teodoro, 2008, p.21). Hence, they can readily adapt to change. The advantage of Kohonen's SOM is the ability to add new inputs into patterns it has already discovered.

The SOM according to Ramadas, Ostermann, and Tjaden (2003) converts "statistical relationships between data points in a high-dimensional space into geometrical relationships between points in a two-dimensional map" (p.37). These visual maps represent data points known as *neurons* (see Figure 3). The interconnection of neurons is *non-linear,* which means it uses a non-sequential ordering. During the learning phase neurons compete to be the winner. The competition is based on weighted factors or the strength of a connection from the input (i.e. network traffic) to the neuron. The *input space* builds the neighborhood of neurons using *vector quantization,* a method to map a range of values. Vector quantization is applied in many applications such as data clustering. However, its original function was to compress data (Kohonen, 1988).

Bolzoni, Zambon, Etalle, and Hartel (2006) use the advantage of SOM and PAYL to present POSEIDON, an unsupervised hybrid IDS. The system substitutes payload length and

frequency distribution with an artificial neural network known as a SOM. The concept is based on cognitive learning, a competitive process the human brain uses to learn. Neurons form the basic component or structure of the neural network. In the POSEIDON architecture, SOMs are used during preprocessing. Their function is to map high-dimensional data points onto a single or multi-dimensional grid. The number of neurons, radius, and training samples can greatly affect the topology or mapping of neurons.

Wang and Stolfo (2004) use the combination of type, length, and distribution to detect anomalous events using PAYL. They developed and successfully tested the system using *Byte Frequency Distribution* (BFD) and 1-gram payload modeling. The former is a requirement to model normal data, whereas the latter is a requirement to detect irregularities within text or ASCII characters. The BFD is the total number of n-gram occurrences, values that are identified in a sampling of payload data. PAYL uses the BFD and standard deviation to compute an anomaly score, which is a measurement that defines the similarity between attacks. Therefore, distance metrics determine similarities between payloads, while n-gram analysis detects anomalies.

The anomaly detector Anagram employs higher order binary n-gram modeling. The technique has several advancements over PAYL's 1-gram modeling. Anagram uses n-grams extracted from payloads to create unique signatures. They are generated using a sliding window of variable length *n*. For example, if *n*=3 and the string represented the letter set {a b c d e f}, the outcome would be a variable of abc, bcd, cde, or def (Ingham & Inoue, n.d.). The major difference between binary n-gram analysis and 1-gram analysis is the latter has limitations and can be easily replicated using different forms of mimicry tactics. This will be discussed latter in further detail.

Correlating alerts is another important aspect of intrusion detection. PAYL primarily uses *String Equality* (SE), *Longest Common Substring* (LCS), and *Longest Common Subsequence* (LCSeq). These techniques correlate attacks using ingress/egress signature matching. For example, fragmented worm attacks are identified by comparing strings across multiple sites or networks. The SE, LCS, and LCSeq are predominantly associated with PAYL. POSEIDON uses ATLANTIDES and PANACEA, designed specifically to correlate alerts and to classify attacks. RIPPER and SVM are examples of correlation and classification techniques (see Figure 1).

ATLANTIDES and PANACEA are additional systems that interface with POSEIDON.. Similar to PAYL, ATLANTIDES correlates alerts using ingress/egress technique. However, a major difference is the system correlates attacks based on user requests that employ higher-level applications. The system is engineered to reduce false positives. In contrary, PANACEA correlates alerts using *Repeated Incremental Pruning to Produce Error Reduction* (RIPPER) and *Support Vector Machine* (SVM). RIPPER uses "IF-THEN" rules to predict a class; SVM is a technique to classify input features.

Finally, Anagram employs *Bloom Filters*. In 1970, Burton Bloom devised a method to test the probability of whether an element (number, letter, or object) pertains to a data set {1, 2, 3…*n*}. These elements can be added, but taken away from the dataset. As the name implies it is a method to filter data. This approach is used not only with intrusion detection, but also with user inquiries (e.g., database requests). The Bloom Filter uses a one-way hashing function that eliminates false negatives. In mathematics, hashing represents data as a single integer, which is then mapped to an index of an *array*. An array can be thought of as a table of indexes that

correspond to a unique value. A hash function identifies a unique *key* such as a social security number or name to that value (Broder & Mitzenmacher, 2004).

Figure 1 represents the systems and techniques discussed in this paper. The hierarchical diagram begins with the main IDSs and ends with their respective alert correlation techniques. Notice the distinction between 1-gram modeling and binary-gram modeling. PAYL uses 1-gram modeling based on frequency distributions while the remaining IDSs use binary n-gram modeling. This distinction between these techniques will be further discussed. It is important to note that only SOM and POSEIDON use neural technology. ATLANTIDES and PANACEA are categorized as correlation tools, not IDSs and can be integrated with POSEIDON to extend its intrusion detection capability.

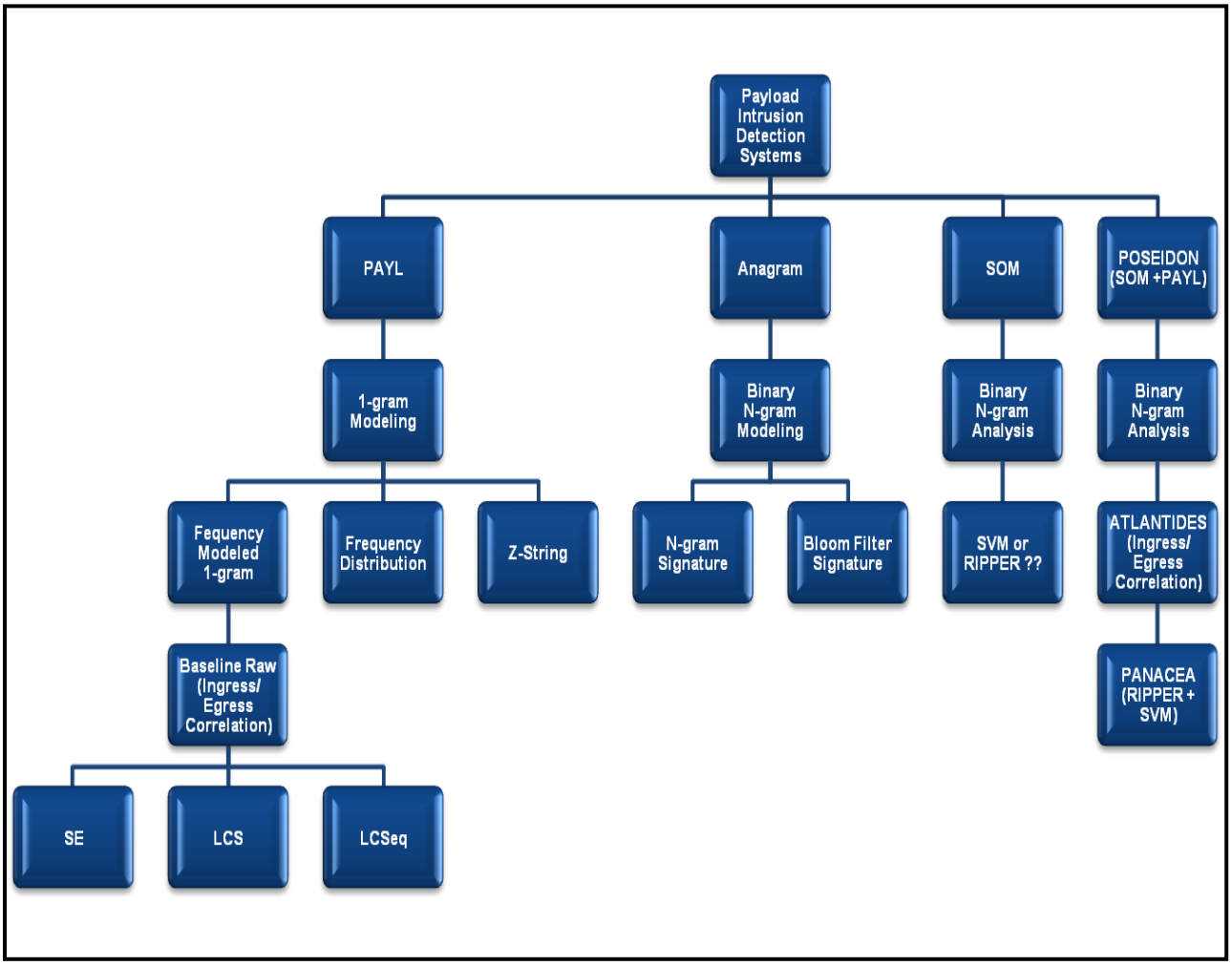


Figure 1. The hierarchical view of core concepts. The figure shows the hierarchical relationship between the IDSs (SOM, POSEIDON, PAYL, and Anagram) and their respective techniques. PAYL employs 1-gram modeling and correlation techniques such as frequency distribution, Z-string, SE, LCS, and LCSeq. The other IDSs employ binary n-gram analysis. Anagram correlates attacks using n-gram signatures and Bloom Filter signatures. POSEIDON employs add-on components known as ATLANTIDES and PANACEA to correlate and classify alerts or attacks. SOM and POSEIDON are unique systems that add neural network features during anomaly detection.

Chapter 3 – Review of Literature and Research

The fundamental process of anomaly-based intrusion detection includes data collection, preprocessing, analysis, clustering, and detection. Perhaps the most subjective part of intrusion detection is selecting the necessary attributes during preprocessing as it may affect both performance and security. For example, selecting too many attributes may degrade performance, while choosing too few attributes may weaken security, allowing malicious attacks to propagate freely. Perona, et al., (2008) identified several features administrators need to consider prior to processing: automation, generality, computational efficiency, and accuracy. The following chapter contains the literature review based on a collection of scholarly documents from academic databases.

Payload Data Processing

Kevin et al., (1990) applied the SOM to distinguish between normal and abnormal characteristics of data. However, the concept of using multiple variations of SOMs was attributed by Rhodes et al., (2000) where maps are used to process TCP, UDP, and IMCP protocol traffic. Litchodzijewski et al., (2002b) confirmed six features were sufficient to process data using a two-layer SOM hierarchy. Kayacik et al., (2006) used the same six features with their version of SOM architecture. They also experimented with a hierarchical SOM-based system to demonstrate preprocessing was effective. Bolzoni et al., (2006) tested and successfully employed SOMs during preprocessing. The technique compares the neuron weight array and extracts the winning neuron for PAYL processing. Zanero (2007) argued that SOM can function on two levels, by examining each packet payload and then compressing the information into a byte. It is then passed on to an unsupervised algorithm. While Perona et al., (2008) provided

general characteristics to consider during payload processing, Zanero (n.d.) focused more on specific properties. Listed below are the primary tasks the SOM uses during preprocessing:

1. Preserve as much information between the similarities of packets as the objective of clustering is to place similar objects together to detect anomalies.
2. Separate packets based on the protocol. Tan and Collie (1997) argued protocols can be detected automatically using SOM.
3. Classify packets according normal or malformed payload. Zanero (n.d.), Bolzoni, et al., (2006), and Kohonen (1988) argued SOMs are able to categorize packets more rapidly and efficiently than other clustering techniques.

SOM can achieve the requirements listed above using clustering techniques. The process includes attributes for classification and detection of anomalous attacks. SOM classifies packets with similar lengths to limit the number of models generated. In contrary, training for PAYL is considerably different.

Labib and Vemuri (n.d.) extracted specific traffic features using only a portion of the IP address during the classification phase. Five distinct features are used for preprocessing, two numbers for sender/receiver and one for the protocol. The test was conducted in real-time and proved that irregular neurons (nodes) are detectable during a possible DoS attack. They proposed a structure consisting of unit vectors, source/destination IP address, and protocol in use. However, the experiment did not include explicit, but rather implicit time requirements defined by Lichodzijewski (2002b).

Del Pino (n.d.) claimed SOMs can process data faster than other learning techniques while preserving topographic data such as the relationship between sender/receiver and protocol. These features are critical when distinguishing between normal and intrusive behaviors. Powers

and He (2008) explained how neurons compete when responding to a stimulus. The process begins with a *connection vector*, which consists of an incoming connection and its associated features. The vectors are then flagged and placed onto the SOM. However, the connections are not labeled as an attack during computation (weight vectors). The neurons must compete according to the connection vector most similar to the weight vector.

According to Powers and He (2008), unsupervised learning means training data is not labeled. Eskin et al., (n.d.) argued unlabeled data offers several advantages such as finding hidden attacks and easy data sampling. For example, SOMs use the connection vector and do not have *a priori* knowledge of an attack. This is perhaps the most critical aspect of unsupervised learning, the ability to discover hidden attacks. In addition, they extracted cluster information through labeling (not to be confused with labeling of raw input data). POSEIDON uses a similar approach, grouping packets into clusters for PAYL processing. However, Xiao and Han (2006) insisted SOMs have several drawbacks such as the need for class label and slow convergence. The answer to the problem is the Evolving Self-Organizing Map (ESOM), where a modification to the prototype nodes within a neighborhood allows for better computation and pattern learning.

Anomaly Detectors

Zanero (n.d.) insisted SOMs have several advantages with detecting payload patterns. Test results showed how the technique successfully learned reoccurring patterns by compressing them into a single byte. The method was also used to cluster payload data. However, *computational complexity* may be a problem with unsupervised learning when too many features are considered during the learning phase. Zanero (n.d.), argued there are existing clustering algorithms that function faster than SOMs such as *k-means*, which is a classical algorithm that

assigns a score by calculating mean distance between clusters (Laskov, Dussel, Schafer, & Rieck, n.d.). An in-depth explanation of k-means clustering is beyond the scope of this paper. However, K-means is not more efficient than SOM during recognition. According to Bolzoni et al., (2006) SOMs are able to cope with high-dimensional data as opposed to K-means. Overall, SOM is a more robust algorithm and outperforms other classification methodologies.

Wang and Stolfo (2006) engineered PAYL as independent language parser using n-gram extracted from packets. The n-gram is the sequence of values in a packet payload. A sliding window is transposed over the entire payload (1-byte) and frequency is calculated. Damashek (1995) used the n-gram analysis to categorize text. PAYL "detects anomalies by combining an n-gram analysis algorithm with a classification method based on clustering of packet payload data length" (Wang & Stolfo, 2006, p.4). N-gram clustering has also been successfully used by Forrester and Hofmeyr (2002).

While PAYL has many capabilities, it does have several shortcomings. According to Thorat, Khandelwal, Brushadeshwar, and Kisore (n.d.), the inspection phase does not consider the entire payload for anomaly detection. This presents a major problem in high-speed and high bandwidth networks. Their solution is to use content-based payload partitioning. This may be a valuable feature to consider with packet processing and could potentially reduce the number of false positives. However, the IDS that inspect the entire payload may run out of memory during a possible DoS attack (Lee, Solo, & Mok, n.d.). The solution was to combine multiple detection models where one device monitors packet-head information and the other payload. PAYL uses the Mahalanobis distance to measure the difference between the model and test pattern. While the method is effective at displaying abnormal byte distributions it does not prevent certain attacks (e.g., CPU instructions) that mimic distribution.

Anagram is an anomaly-based system that employs binary n-gram analysis. Wang, Parekh, and Stolfo (n.d.), tested the capabilities of Anagram by comparing frequency-based and binary-based approaches. The binary based approach yielded significantly better results. In addition, the Bloom Filters offer significant advantages, particularly with representing data without a fix number of n-grams. Thus, Anagram methodology can represent a model with very few bytes. This greatly reduces memory requirements.

Lastly, POSEIDON is an anomaly-based system that employs PAYL and SOM. While the developers claim the system has a high detection rate, there are a few areas that require further explaining. Vliet (n.d.) questioned the ability of POSEIDON to work in a real-environment given small network changes. The concern is whether the model could yield the same promising results as defined in the 1999 DARPA intrusion detection data set. Test results from *Turnover Poseidon* proved the system could compensate for small network changes and yield better results than the original POSEIDON design. Bolzoni et al., (2006) uses a modified version of PAYL with the addition of SOM, both classification methods are trained separately, which could present difficulties with accuracy. In addition, the test results appear biased without additional attacks added to the test data set.

Alert Correlation Techniques

PAYL monitors output alerts during the analysis phase. The unit of analysis is the alert generated and association with a violation (Gu, et al., 2006). The major alert correlation techniques are raw packet correlation, frequency-based alert correlation, and n-gram alert correlation. The base-line contains raw packets for correlation and is comprised of SE, LCS, and LCSeq. The SE is considered more restrictive than LCS and LCSeq. These correlation

examples can be used as signatures and imported into other IDSs (Matrawy & Abdelaziz, 2008). These techniques SE, LCS, and LCSeq determine whether two payload alerts match each other.

Wang et al., (n.d.) tested PAYL over three real-world datasets. Variations of worms were placed randomly within the test data. The results showed PAYL can detect worms that fragment their content into small packets. This is an evasion technique commonly used during worm attacks. The experiment included testing over three different sites yielding a 0.1% false positive rate. In addition, the system was capable of detecting worms during the first dissemination using ingress/egress traffic correlation.

PAYL uses the 1-gram alert correlation based on frequency distribution and Z-string. These methodologies offer a rapid and efficient way to correlate data content. According to Thorat et al., (n.d.), a 1-gram model is the most convenient way to model the payload. PAYL uses this technique during the detecting phase. Dharmapurikar, Perekh, Wang, and Stolfo (2006) claimed the Mahalanobis distance (or Manhattan) is calculated "between the distance of the candidate packets and the frequency model" (p.2). The greater the distance from the model the more likely the activity is suspicious and thus may contain malicious content. Z-strings can speed the distance computation because they rapidly identify characters that are infrequently found in normal traffic. Another benefit of Z-strings is to distribute the signature to another site to avoid further infection (Dharmapurikar, et al., 2006).

Binary modeled n-gram alert correlation allows for greater privacy and increased ability to correlate attacks. Anagram uses an n-gram binary modeling approach that offers significant enhancements over frequency modeled 1-gram used with PAYL (Wang, Perekh, & Stolfo, 2006). Higher accuracy, computational efficiency, model space, quick correlation, and robust signatures are a few advantages of using Anagram. Wang et al., (2004) defended Anagram's

ability to avoid mimicry attacks by randomizing packets, making it progressively difficult for attackers to simulate normal traffic. This can be achieved using dummy bits or padding. This makes it difficult for attackers to craft an entire packet as normal content.

Dharmapurikar, Krishnamurthy, Sproull, and Lockwood (n.d.) argued over the benefits of using Anagram Bloom Filters to match signatures in streaming data. The Bloom Filters query strings within a database to determine its membership; the answer to the query can never be a false negative. Wang, Parekh, and Stolfo (n.d.) argued that Bloom Filters are more efficient than PAYL's frequency distribution modeling with regards to memory and computation. The Bloom Filters allow a "mixture of different size n-grams extracted from packet payloads" (Wang, et al., n.d., p. 2). However, Perdisci et al., (2008) argued that Bloom Filters would not work in high-bandwidth or high data rate networks. Anagram does not consider frequency distribution. In the contrary, it stores n-grams within the Bloom Filters. During detection phase, a score is assigned based on the number of unobserved and malicious n-grams.

Bolzoni, Etalle, and Hartel (n.d.) successfully tested SVM and RIPPER, demonstrating the ability to classify alerts with high accuracy. Chen, Hsu, and Shen (2004) describe how SVM is able to classify data based on the use of support vectors. It is used to separate two classes. If by case the classes cannot be separated then the input data is mapped into a high-dimensional feature space. The authors also claim that SVM is better at classifying attacks than neural-based techniques. Bolzoni, et al., (n.d.), confirmed SVMs work remarkably well with classifying alerts. According Pietraszek (n.d.), RIPPER is highly accurate and works efficiently with malicious data content. The technique is concise and spontaneous because it employs rule learning.

Attacks

An attack is generally considered an attempt to intentionally compromise the CIA of information or information system (Whitman & Mattord, 2005). There are many classes of attacks. The primary types related to network attacks are: read manipulate, spoof, flood, redirect, and combination. Table 1 provides a breakdown of the different types of attacks. The emphasis is to highlight how an attack is associated with the method of exploitation.

Table 1

Examples of Main Attack Vectors

Main Attack Vectors	Examples
Read	Sniffer, Direct Access
Manipulate	Buffer Overflow, Web, Mimicry, SQL Injection
Spoof	Web Redirect
Flood	DDoS, DoS, Mac
Redirect	ARP
Combination	MITM, Virus, Worm, Trojan

Buffer overflow. A buffer overflow injects "an instruction sequence into the victim application and transfers the control of the application to the injected code" (Hsu, Guo, & Chiueh, 2006, p.2). They are the most common and widely spread method of attack as they account for 50% of known vulnerabilities. Typically, the type of request from the user or feature of a network service rarely used is an indicator of a buffer attack (Krugel, et al., 2002). Regardless, buffer attacks are a serious threat, especially to misuse IDSs. For example, BRO has a no packet filter drop application, where packets are dropped if too many arrive at once. However, the IDS will drop packets if the buffer is full, potentially allowing attacks to pass through (Paxson, 1999).

Mimicry. Wagner and Dean (2001) acknowledged how mimicry attacks are serious threat to anomaly detectors. Wang, Perekh, and Stolfo (2006) recognized PAYL's limited ability to avoid this type of attack. PAYL uses 1-gram byte sequences modeling which could be circumvented by modifying the data grams to the type of traffic, thus appearing as normal activity. As previously discussed, Anagram can avoid mimicry attacks by randomizing data grams. POSEIDON has the capability of avoiding mimicry attacks by means of SOM. For example, extra bytes are flagged as abnormal traffic during the analysis phase.

HTTP. There are different methods to detect anomalies within HTTP traffic. Bolzoni and Etalle (n.d.), use a raw-data methodology to detect HTTP anomalies. In addition, they integrated POSEIDON to distinguish anomalies from irregular text. In contrary, Ingham and Inoue (n.d), demonstrate using n-grams as a substring and employing a sliding window to detect anomalies. PAYL uses payload length and 1-gram modeling. Unfortunately, using payload length as a method to distinguish normal from anomalous payloads is flawed (Estevez-Tapiador et al., 2004).

Worms. Worms are a serious threat to organizations, particularly *Zero-day* worms. According to Verwoerd and Hunt (2001), the Code Red worm in 2001 infected 359,000 machines in approximately 14 hours. Wang, Cretu, and Stolfo (n.d.), studied the effects of worms and their ability to launch attacks and propagate quickly, leaving minimal time for detection. These attacks use payload content to carry out malicious activities. Testing is conducted on three real-world datasets containing worms. PAYL detected the attack within the first attempt; an important ability that cannot be provided solely by rule-based IDS such as Snort or Bro (Smith, Matrawry, Chow, & Abdelaziz, 2008). According Wang and Stolfo (n.d.), PAYL can detect and deter worms from spreading by using ingress/egress alert correlation techniques.

Automatic worm signature generation is a benefit of using this technique. Test result indicated that LCS and LCSeq were both able to detect Code Red and Code Red II executables. Z-String can also be used in a distributed form by correlating a worm attack among different sites.

SQL injections. A SQL injection attack occurs when a system is unable to clear harmful characters, thus exposing sensitive information without authorization. The attack targets CPU instructions. Bolzoni and Etalle (n.d.), demonstrate how payload-based systems are able to detect this type of attack. PAYL does not provide any information about the detector's ability to avoid SQL injection attacks. However, there is evidence of POSEIDON being able to detect this type of attack.

Evaluating IDSs

March and Smith (1995) argued that design-science methodology should meet two stipulations—to build and evaluate. Building refers to an artifact that is capable of functioning, whereas evaluate refers to establishing criteria to determine whether the product meets the proposed specifications. Essentially, the evaluation should determine how well the artifact works using non-mathematical representations of the artifact such as natural science or behavioral science research methodology. The mathematical representation of the artifact should reference to the design-science research methodology. According to Delone and McLean (1992), the effectiveness of an information system is measured by the output and its level of influence.

The metrics used during a study should "define what a research area is trying to accomplish" (March & Smith, 1995, p.261). Grover, Jeong, and Segars (1996), argued that the evaluation of an information system should include an *evaluative referent* which describes "the

relative standard that is used as a basis for assessing performance" (p.180). This goal-centered approach is a method of evaluating a system based on the objectives that are met or achieved.

Gu, et al., (2006) argued the common metrics for evaluating an IDS are "false positive rate, which is the probability that the IDS outputs an alarm when there is no intrusion and true positive rate, which is the probability that the IDS outputs an alarm when there is an intrusion" (p.1). According to Ingham and Inoue (n.d.), there are two reasons to evaluate the IDS. The first is to verify whether an algorithm is effective at detecting attacks and secondly to compare and select the better of two or more algorithms to implement.

Chapter 4 – Methodology

This study is based primarily on the behavioral science and design science research methodologies. As depicted in Figure 2, the behavioral science research methodology explains how the artifact functions while design-science research explains how well the artifact functions. The integration of the two bridges the gap from theory to practice. The reference "artifact" is an analogous to the IDS (e.g., SOM, POSEIDON, PAYL, and Anagram). Notice the research objectives from Chapter 1 were carefully selected to define exactly what material is to be analyzed to reach a final ultimatum. The effectiveness of the IDSs is explicitly addressed in the evaluation scenarios detailed in Chapter 8. Defining how well the artifact works will depend on the interpretation and results from the data collection.

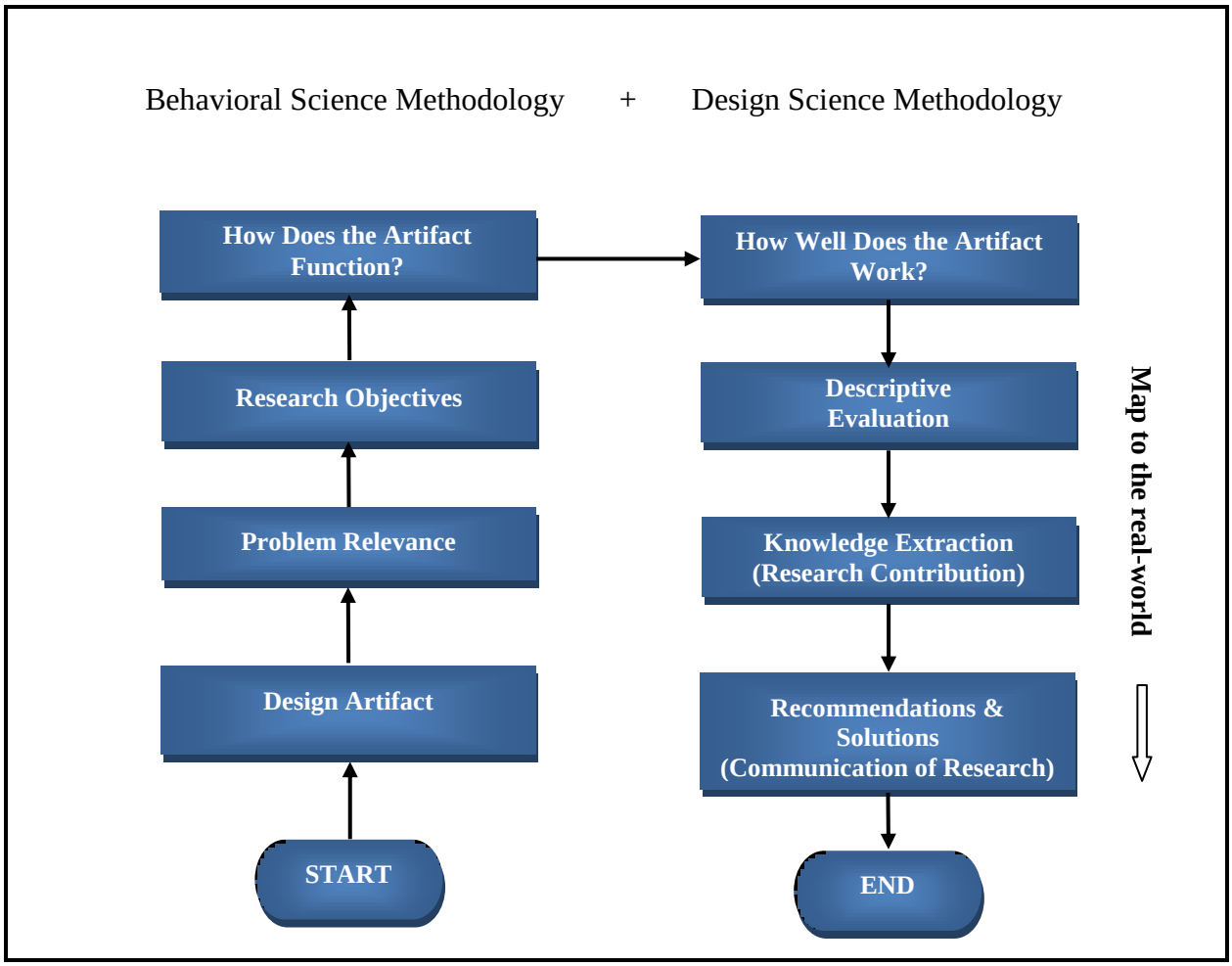


Figure 2. The research process flowchart. The illustration shows research process using the design science and behavioral science research methodologies.

Procedures

The research process began with selecting the IDSs to research. Rosemann and Vessey (2008) argued the role of applicability is an important aspect of design research. The artifact should meet three requirements: importance, accessibility, and applicability. Importance should enforce practical needs and address real-world problems. In this regard, payload-based attacks are a significant threat to business applications. Accessibility focuses on results rather than the research process. Addressing the practicality of payload anomaly detection is imperative. For example, it may be impractical to perform payload anomaly detection over a Wide Area Network (WAN). Lastly, applicability means research should provide guidance. The results from this study should provide value to the research community.

The next step in the research process was to establish the problem relevance. There is an enduring need to advance research in this field of information assurance, particularly with intrusion detection. Research should expand the knowledge base to raise awareness and promote innovation; otherwise there is risk of losing ground in the struggle against malicious attacks. While the majority of anomaly detectors in this study have not been tested in the real-world, the data collection provides examples of attacks likely to be encountered by the IDS. Determining the effectiveness of IDSs is a relevant problem for practitioners.

Perhaps the most difficult aspect of evaluating the effectiveness of IDSs is the lack of public data to analyze and compare systems. This can significantly impact an assessment. The sensible approach was to build well-defined scenarios based on the design-science research

methodology. Performance is documented by the collection of quantitative data. The research contribution is therefore knowledge of how the systems react to network attacks.

Analysis

A significant part of the research process was to gather quantitative data to form well-defined scenarios as detailed in Chapter 8. The objective of the analysis is to expose strengths and weaknesses of IDSs and their associated techniques. For example, when comparing PAYL's frequency-based approach with Anagram's binary-based approach, the later approach produced a significantly lower false positive rate. This new knowledge would be beneficial for an organization wishing to minimize false positives.

In this chapter the behavioral science and design science research methodology describe the methods to analyze payload content using modern IDSs. The behavioral science research methodology explained how an artifact functions. This is way to validate whether the artifact performs as originally designed or intended. The design science research approach evaluates the artifact for utility, to expose weaknesses and to seek improvements. The combination of the two research methodologies may also bridge the gap between functionality and performance.

Chapter 5 – Taxonomy of Payload Anomaly-based Intrusion Detection Systems

Ideally, the complete security package should include the IDS to protect information and network resources from malicious threats. While several versions of payload anomaly-based IDSs are available, many struggle to perform as expected while others can hardly cope with the complexity of high-dimensional data. This chapter focuses on the taxonomy of IDSs proven successful in detecting malicious attacks. Notice the intent was not to provide an exhaustive list of payload anomaly-based IDSs, rather a select few successfully tested and results well-documented within the literature.

SOM

Labib and Vemuri (n.d.) use the advantages of SOMs to process data in real-time over high-speed data rates to provide topological mappings of normal and intrusive behaviors. The mappings show a visual representation of the data collection (Giradin, 1999). As previously noted from above, Kohonen introduced the SOM algorithm to include competitive learning between neurons, which are then mapped within a one-dimensional or two-dimensional topology. A SOM is able to classify data and distinguish similarities based on the distance between neurons. Rhodes, Mahaffey, and Cannady (2000) used multiple maps to detect intrusions. Similarly, Giradin (1999) provided a visual representation of network events using a map image. A SOM contains a neighborhood and winning neuron as illustrated in Figure 3. The winning neuron is surrounded by other neurons, which forms the neighborhood. Each neuron has a *weighted vector* or value and is adjusted by the input value.

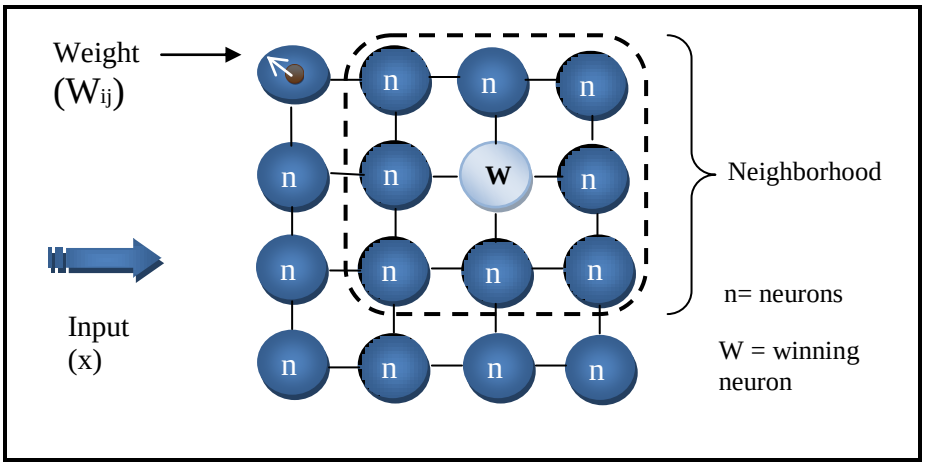


Figure 3. An example of a SOM. The figure represents a SOM with winning neuron and surrounding neighborhood. The input *x* is any network input or raw data. The Weight W_{ij} is the measurement distance (using Euclidean formula) from the input data *x* to the individual neuron. Each neuron has a weight. The weight closest to the input is identified the BMU. The neighborhood is self-constructed surrounding the winning neuron, identified as *W* (Rhodes, et al., 2000).

POSEIDON

Knowing the differences between connection-oriented and packet-oriented intrusion provides further understanding of how POSEIDON works. Connection-oriented systems use connections and statistical information to determine whether anomalies occur. In contrary, packet-oriented systems analyze the entire or portion of the payload for malicious attacks. While connection-oriented systems provide a finer-grained analysis, they have several drawbacks. They are known to be computational expensive and require excessive memory. Furthermore, they are generally more suitable for off-line analysis. POSEIDON's design can overcome these limitations to provide higher-level detection and reduction in the quantity of models generated (Bolzoni, Zambon, Etalle, & Hartel, 2006).

POSEIDON's internal structure includes PAYL and SOM. It employs a modified version of the original PAYL, along with the SOM to replace payload length as a criterion to cluster packets. The differences between POSEIDON and PAYL are illustrated in Figure 4 and Figure 5. The goal of the SOM is to preserve as much information about the payload as possible to cluster similar items using PAYL. The three phases of SOM processing include initialization, training, and classification (Bolzoni, et al., 2006):

- *Initialization.* The IDS technician must confirm the number of nodes, learning rate, and radius during the initialization. These must be fixed parameters. The number of nodes will determine the level of classification. For example, a significant amount of nodes may produce too sparse classification while a small network too coarse of classification (e.g., classify data within the same neuron).
- *Training*. The training phase uses iterations using weight array and distance function (Euclidean or Manhattan). The total samples equals to the number of interactions. SOM and PAYL are trained separately. The SOM must learn the clustering of packets while PAYL is used to classify an attack based on the distance of the cluster.
- *Classification*. "Input data is compares to all the weight arrays and the most similar neuron determines the classification of the sample. The winning neuron is then returned" (p.4).

The learning process includes the following steps (Girardin, 1999, p.5; Kayacik, n.d., p.8):

1. Initialize the weight factors by assigning random values w_{ij}.
2. Present an input pattern x.
3. Determine the winning neuron by calculating the distance between input vector x and weight vector w; the winner is the one identified with the shortest distance.

4. Update all neurons using weight vectors; modify the neurons surrounding the winner within the neighborhood. This is used to determine which neuron to modify.
5. Repeat steps until for all input data.

After the steps from above have been completed, the neuron that represents the smallest distance from the input vector *x* and each neuron is determined the *Best Matching Unit* (BMU). The BMU indentifies the position of the neighborhood and acts as a class label (e.g., normal, DoS, probe, U2R, or R2L). Figure 4 represents the POSEIDON architecture from an internal view. A SOM extracts the winning neuron and is used as a clustering technique.

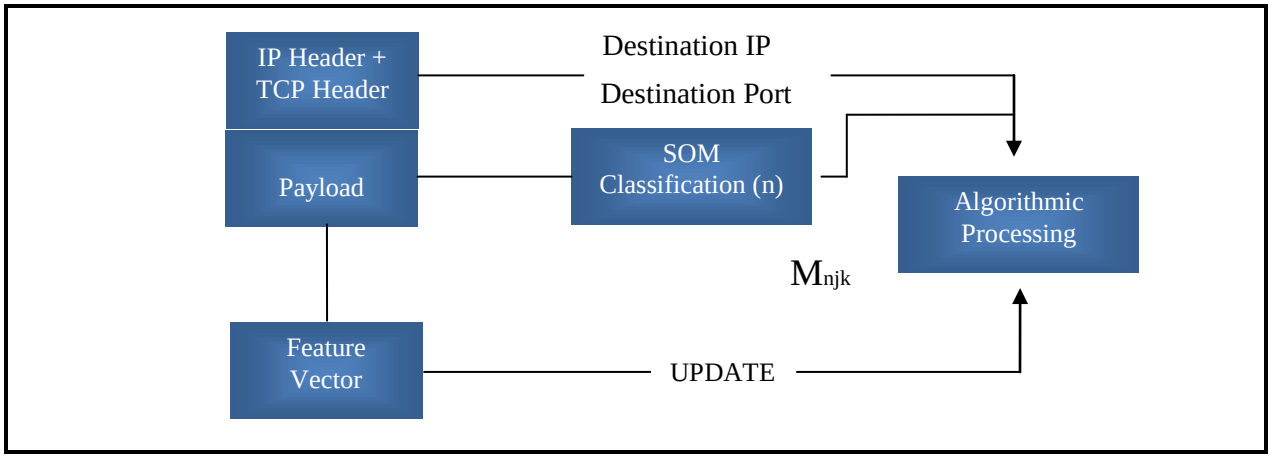


Figure 4. POSEIDON's internal function. The model includes the SOM used during preprocessing where a value is extracted for PAYL modeling. Notice the payload length is not considered as opposed to the PAYL model. POSEIDON uses M_{njk}, instead of $M_{ljk.}$ where, M= model, j = destination address; k = port; and n = neural network or SOM classification. The IDS uses header information and SOM classification to update the feature vector. A value is extracted and updated for algorithmic processing. The illustration has been partially modified to eliminate the need to explain higher-level mathematics, which is beyond the scope of this paper (Bolzoni & Etalle, n.d.).

PAYL

The PAYL IDS employs 1-gram binary analysis and clustering of packets based on payload data length. The technique considers each input packet with destination port, payload length byte, frequency, and standard deviation. During the detection phase, the packets are processed and compared to the values of the newly created models, also known as *centroid* models. A comparison is made using the *Mahalanobis Distance*, which in mathematics is the distance measured between two samples (e.g., data or model) to determine their similarity. A major deviation from the norm creates an alert. In effort to reduce the number of models, PAYL uses a clustering technique. The distance between each model is calculated using *Manhattan Distance*, a mathematical concept that measures points along a grid. A threshold *t* is assigned and models are merged accordingly (Wang & Stolfo, 2004).

PAYL uses a form of n-gram analysis where n=1 or the number of adjacent bytes within the payload. It is the average number of ASCII characters (0-255). A sliding window is transposed over the entire payload and the numbers of n-gram occurrences are counted. The *feature vector*, also used with POSEIDON is calculated by "dividing number of occurrences of each n-gram by the total number of n-grams" (Wang & Stolfo, n.d., p.4). The standard deviation is also calculated as the ASCII characters (0-255) are treated as variables. Figure 5 represents the internal function of PAYL, which is similar to POSEIDON without the SOM used during preprocessing.

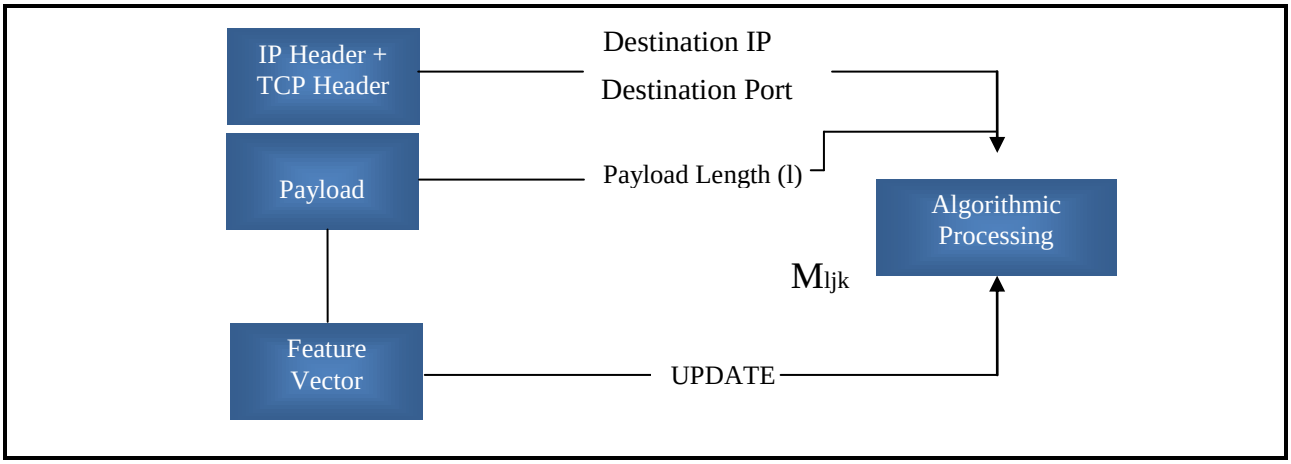


Figure 5. The PAYL's internal function. The figure represents PAYL without the use of the SOM. The feature vector is a mathematical formula used in pattern recognition or machine learning that considers numerical features of objects and their vectors. The vector in mathematics represents a straight line with a starting point and sense of direction or termination point. The illustration has been partially modified to eliminate the need to explain higher-level mathematics, which is beyond the scope of this paper (Bolzoni & Etalle, n.d.).

Anagram

Anagram is a content anomaly-based detector that employs binary n-gram analysis over PAYL's 1-gram analysis. A major change is the higher-order model to test network traffic. Perhaps the most ingenious feature of Anagram is the use of Bloom Filters. The filters separate data, placing n-grams into two distinct filters labeled b1 and b2. Incoming traffic is analyzed for n-grams; one filter contains normal traffic while the other filter contains infected or bad traffic. Anagram stores normal n-grams into filter b_1 and n-grams from known attacks into filter b_2. The detection phase includes a comparison of n-grams. A payload is categorized as anomalous if a major deviation exists in the percentage of n-grams in either type filter. In addition, packets are flagged anomalous when there are too many n-grams or too little in normal traffic (Smith,

Matrawy, Chow, & Abdelaziz, 2008). Following the training phase, a score is assigned according to the number of n-grams *not* identified. This approach can analyze data without an impact to network throughput. N-gram analysis is particularly useful in indentifying new byte sequences, those not previously identified.

The IDSs in this study differ significantly from each other, but use similar approaches to analyze payload content. For example, n-gram analysis is applied in the same manner over a payload, but results differ when *n* is greater than 1. The difference is apparent when comparing frequency-based modeling and binary n-gram modeling (see Table 9). These are the primary techniques to identify anomalous events or sequences. In this chapter, one may observe how neural technology plays an important role with intrusion detection. As an individual system, SOMs can detect a variety of attacks, but with integration of PAYL can increase the ability to detect threats. POSEIDON uses SOM as a preprocessing tool to extract the winning neuron and cluster packets with the assistance of PAYL. Finally, Anagram is anomaly intrusion detection system that employs binary n-gram analysis to discover anomalous byte sequences within payload data. It is a fairly simple method to determine whether unusual n-grams are present using a scoring system. Using Bloom Filters is an efficient way to separate normal from irregular or suspicious n-grams.

Chapter 6 – Alert Correlation Techniques

The basic function of alert correlation techniques is to match similarities between threats, which are then indentified by their unique signature string. Alert correlation techniques are categorized as raw payload correlation (baseline), frequency-modeled 1-gram alert correlation, and binary-modeled n-gram alert correlation with addition of SVM and RIPPER as independent methodologies to classify attacks. The baseline raw methodology correlates ingress and egress traffic by detecting similarities between strings (or signatures). It is the simplest form to correlate signatures. The 1-gram alert correlation technique employed by PAYL provides adequate knowledge of the packet payload (Parekh, et al., 2006). The binary n-gram analysis employed by Anagram captures sequences of characters to identify anomalous n-grams. Finally, RIPPER and SVM are alert correlation tools that employ unique techniques to classify alerts.

Raw Correlation

The raw correlation techniques are applicable mainly to PAYL. The term *raw* refers to a collection of metadata (e.g., packet length or payload length). Parekh, Wang, and Stolfo (2006) categorized the raw correlation alerts as SE, LCS, and LCSeq. The SE identifies a signature match between two packets using ingress/egress correlation technique. It is the strictest of the techniques and reduces false positives considerably. Unfortunately, a major fallacy with SE is the failure to detect fragmented worms. LCS on the other hand can better cope with the problem of worm fragmentation. However, LCS is not as precise as SE and requires higher computational overhead. Lastly, the LCSeq can detect polymorphic worms, but has the tendency to produce higher number of false positives (Parekh et al., 2006).

1-Gram Frequency Modeling

The 1-gram frequency modeling is an alternative correlation technique. PAYL uses frequency distribution, which are values rather than sequential information about the payload. The Manhattan Distance is the space between the frequency distributions to determine similarities among packets. The Z-string is similar to frequency distribution modeling. It uses a rank structure; the concept behind the "Zipf String" is to classify frequency distributions from most to least suspicious packet (Wang & Stolfo, 2006).

Binary N-Gram Modeling

Binary-based alert correlation is considerably different than 1-gram alert correlation. For example, n-gram binary modeling works better at modeling data sequences. The binary n-gram signature creates a list of suspicious packets using n-gram analysis by capturing malicious byte sequences. The n-gram signature publishes the signature within the Bloom Filter. It is important to note the Bloom Filter n-gram signature correlation is not associated with the Anagram Bloom Filter model. A significant drawback of n-gram analysis is lack of data privacy such as exposing a password in the clear (Parekh et al., 2006).

Bolzoni, Crispo, and Etalle (2007) devised a new way of correlating alerts based on client to server requests over HTTP traffic. Contrary to raw correlation and 1-gram frequency modeling described above, ATLANTIDES correlates higher-level application attacks. It could be integrated with a signature-based system or anomaly-based system and operate without human assistance. One of the advantages of the system is the ability to interface with POSEIDON. It can also reduce the number of false positives generated by approximately 50%.

SVM and RIPPER

PANACEA uses SVM and RIPPER as the primary means to classify alerts. SVM uses a hyper-plane to separate training data from its origin (Eskin, et al., n.d.). According to Bolzoni, Etalle and Hartel (n.d.), the SVM algorithm has been modified to classify *non-linear data*; a type of data structuring that is not aligned sequentially. The *support vectors* are a subset of training data measured between Class A and Class B. Figure 6 is an example of a hyper-plane and is the simplest model to represent how the SVM separates data. The gray circles and squares are support vectors. The data points nearest to the support vectors define the margin. SVM is quite complex and uses high-level mathematics.

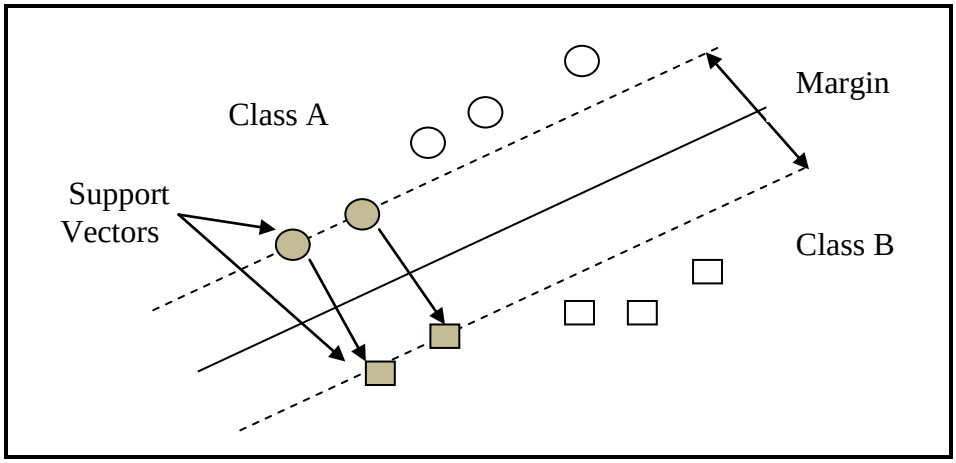


Figure 6. Illustration of SVM hyper-plane. The figure illustrates how support vectors determine the margin between Class A and Class B (Chen, Hsu, & Shen, 2004).

RIPPER was adapted as a self-learning tool based on the concept of data mining, which is the process of extracting models from data found in data repositories. PANACEA uses RIPPER's rule induction algorithm to classify items into categories (Axelsson, 2000a). For example, "IF-THEN" rules are used to create conditions or rule sets. This technique correlates alerts to the type of attack.

The alert correlation techniques discussed are divided among 1-gram frequency modeling and binary-n-gram modeling techniques. The raw correlation techniques are the simplest to implement and represent information in its original form. It is intolerant to variations to include fragmentation and polymorphism. However, LCS is not susceptible to fragmentation, hence is not as strict as SE. LCSeq is considerably different than the previous correlation techniques because it can handle insertion and reordering of data. Z-string is an alert correlation technique based on payload byte distribution. The Bloom Filter is used to preserve privacy across using n-gram analysis. Lastly, SVM and RIPPER are complex alert correlation techniques that classify attacks into respective groups. The ability to classify alerts is a major advantage, especially when it can assist analysts with correlating attacks.

Chapter 7 – Payload Processing and Design

Unsupervised payload-based anomaly detection is a structured process. The IDS begins with a model which represents normal network activity and is built from learned behavior, known as training data. During the training phase, a clean data set is necessary to separate normal attack-free traffic from noisy or malicious traffic. To speed the process data objects may be clustered into groups with similar characteristics. A deviation from a model characterizes an anomaly or anomalous behavior and is calculated using a mathematical distance function or algorithm. The greater the distance the more likely an event is conspicuous. This chapter explains the different stages of payload processing and includes a brief explanation of how to implement and test the IDS.

Payload Processing

Achieving a higher detection rate requires processing the entire payload to preserve as many characteristics as possible (Zanero & Serazzi, 2004). However, the amount of data to analyze depends on the accuracy of the IDS and the algorithm it employs. With unsupervised learning any raw data may undergo processing, with limitations based strictly upon the operating system and available memory. Figure 7 represents the different phases of payload processing. The data features consist of extracting header features from TCP/IP dump or sniffer packets and converting the results into a binary form. Preprocessing is necessary to limit the amount of data to analyze. Input patterns may undergo preprocessing prior to the analysis phase, and then clustering of data to follow (see Figure 7). The detection phase is the last part of the process, to identify abnormal network activity.

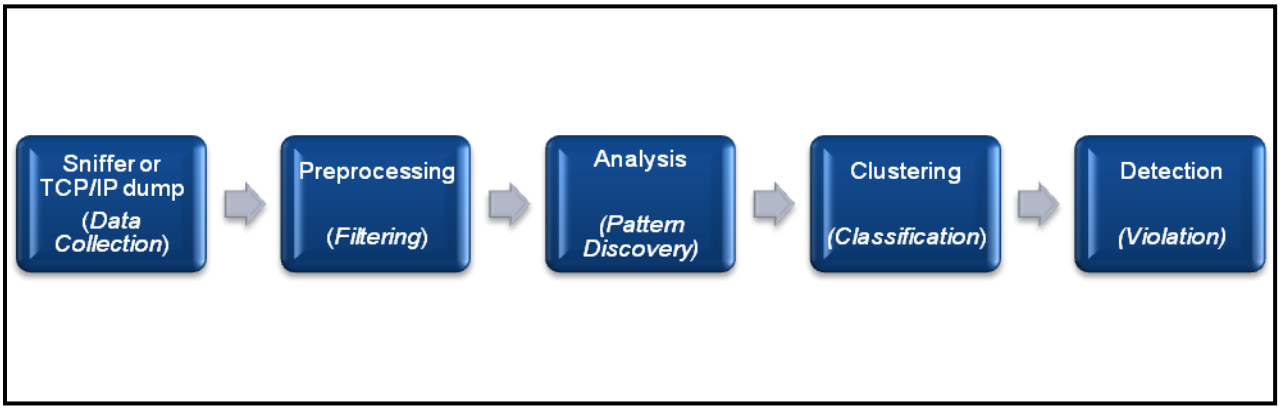


Figure 7. The phases of payload modeling. The figure illustrates the different steps needed during payload processing.

Data collection. Data collection is raw data from a network input. For example, a TCP dump file provides a list of connections. A connection would be defined as a TCP packet that starts and ends within a specified timeframe and between the source and destination address using a well-defined protocol. Data processing is executed as continuous or in batches. According to Axelsson (2000a), these two approaches are represented explicitly and implicitly, the first assigns a time stamp while the second uses a First in First Out input connected to a neural network.

Preprocessing. According to Vessanto, Himberg, Alhoniemi, and Parhankangas (1999), preprocessing limits the amount of overall processing. It provides a suitable representation of data prior to the analysis phase. For example, basic TCP/IP features (e.g., protocol type, service type, or status flag) that use alphanumeric codes must be translated to numerical values. During preprocessing, the amount of information extracted from the data content may impact performance and security (Kayacik, et al., n.d.). For example, the more attributes selected the higher the detection rate because there are more characteristics that correlate to the threat.

However, this may impact performance as a system will require more time to process the information gathered.

Analysis. This is the method to analyze data. It may include techniques or processing engines to determine where and how to separate data input prior to clustering. Anomaly-based systems rely on mathematical algorithms and distance formulas to model data. For example, Bolzoni and Etalle (n.d.) argued HTTP requests can be separated into regular and irregular parameters or text. The former includes well-formatted data such as numbers or dates, whereas the latter includes unsuitable raw data such as images or binary data. An HTTP request defines a structure with a syntax and request parameter as illustrated in Figure 8. Violations during the analysis phase are discovered using 1-gram modeling or binary modeling analysis techniques.

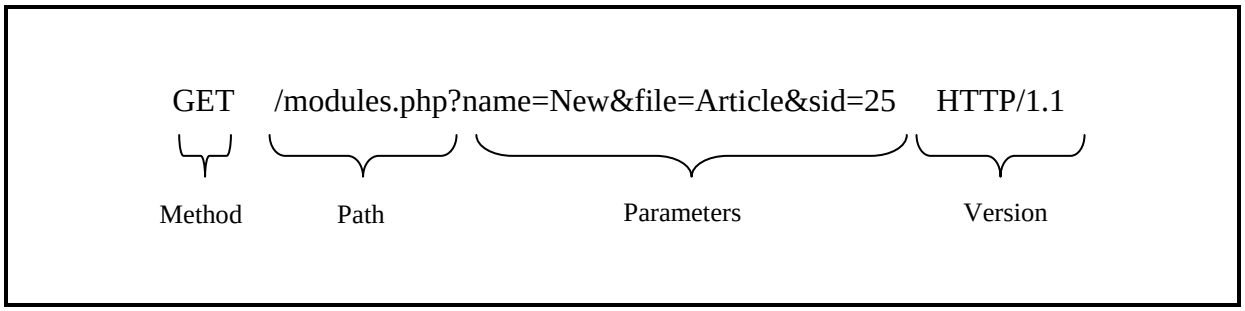


Figure 8. Sample HTTP request (Bolzoni & Etalle, n.d.).

Clustering. The purpose of clustering data is to place similar items or objects into well-defined groups. Clustering techniques are classified as supervised or unsupervised. An example of a supervised algorithm is *k-Nearest Neighbor*. An alternative algorithm known as *k-Means* clustering falls within the category of unsupervised learning. According to Zanero (2007), a SOM compresses information into a single byte and groups the information (objects) with similar attributes. Therefore, the cluster contains characteristics of the packet. POSEIDON uses the SOM to extract attributes and PAYL to build models of each cluster. Figure 9 represents a

SOM with PAYL modeling. The neurons within each SOM are considered a cluster. Another purpose of clustering is to ensure models do not become excessively large.

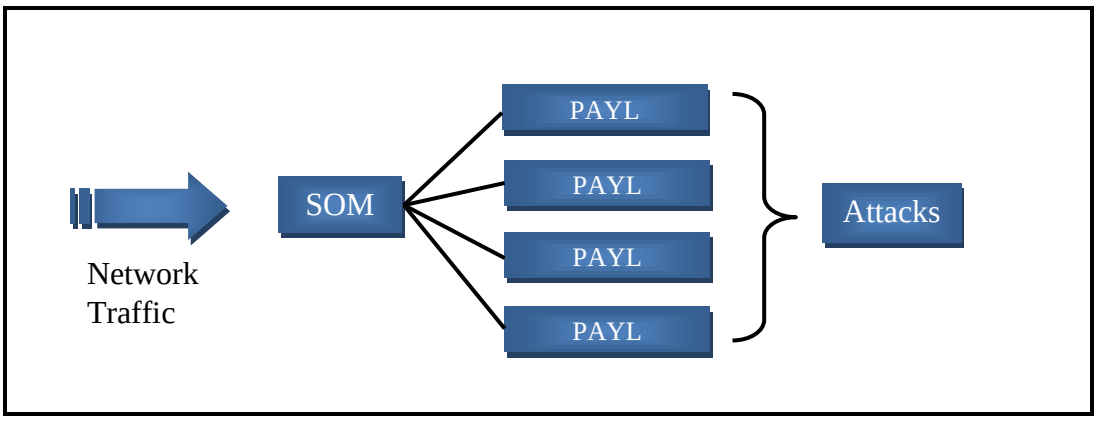


Figure 9. POSEIDON clustering technique using PAYL and SOM. The figure is a simple model illustrating how packet payload attributes is extracted using a SOM. PAYL groups similar objects together, an essential step during payload processing and intrusion detection.

Detection. Ideally, the purpose of the detection phase is to identify a true violation or deviation from normal network behavior. The following summarizes how the IDSs in this study identify anomalous activity. The SOM processes data using hierarchical layers and identifies abnormal behavior based on plotted data points. POSEIDON uses the SOM in a different manner. A violation is present when the mapping of neurons differs from a normal population. PAYL measures frequency distribution from normal traffic to determine whether a violation or attack has occurred. Finally, Anagram uses n-gram analysis a higher form of modeling with use of Bloom Filters. A score is assigned based on the deviation from the trained data (Wang, et al., n.d.).

Overview and Setup

Implementing the IDS will require preplanning to determine what a product should accomplish and what requirements should be met. For example, if the objective or policy is to enforce user privacy then the requirement would be to ensure the CIA of information and information systems. Obviously, an IDS or technique that exposes user privacy would violate the policy and not meet the defined objective. In general, there are two main goals IDSs should adhere to (Bace & Mell, 2001):

- *Accountability.* This is the ability to track and indentify who or what is responsible for causing the unauthorized activity. While it is imperative to stop anomalous threats, knowing where the threat originates is equally desirable. In many cases accountability may be difficult to achieve, but should not be ignored.
- *Response*. This is a reaction to an event and includes the appropriate actions to stop or deter the threat from continuing to do harm. An example would be blocking a TCP/IP port or modify an access list on a firewall. If accountability cannot be achieved, a response will suffice as the goal is to recognize the attack is present and should be blocked.

Selecting the appropriate IDS will depend on the goals and objectives of the organization while considering the strengths and limitations of in-place systems. Organizational policies may sway which IDS to choose, but ultimately the selection should depend on the best product to ensure the CIA of information and information systems. It is important to realize that the effectiveness of the IDS is dependent upon good training data (Kayacik, et al., n.d.). A model should represent the behavior of a system without the presence of attacks.

Training

Training is required to learn the network and evaluate whether systems truly identify known and unknown attacks. Researchers and developers typically use a benchmark to evaluate IDSs such as DARPA98, Kddcup 99, or Lincoln MIT Laboratory datasets. Anomaly-based IDSs will use normal and anomalous data during training. Generating random anomalies allows the IDS to detect previously unknown attacks. The basic features extracted are dependent upon the level of knowledge needed to detect an attack (Kayacik, et al., n.d.). The following are brief examples of training processes.

- *Example 1-SOM training.* The training phase for SOM includes the use of multiple iterations using weight array and mathematical distance function Euclidean or Manhattan. The total samples equal the number of interactions. Essentially, the neurons model the input space, which is broadly defined as metadata (e.g., network connections, event logs, or system call traces; Eskin et al., n.d.). The neuron that responds the best is chosen as the winner. The surrounding neurons define the neighborhood (Ramadas, Ostermann, & Tjaden, 2003). A separate SOM can be trained to detect a specific protocol.
- *Example 2- PAYL training***.** PAYL extracts the payload, then performs a HEX to ASCII conversion, defines the port and length and then progresses into the packet capture phase. During this phase the frequency and relative frequency of each byte is calculated along with n-gram analysis (variance and mean). A model is produced and compared to the new payload distribution. The Mahalanobis is used to compare new payload with a computed or *centroid model* (see Appendix for illustration). PAYL also uses a clustering technique with payloads of the same length to build a single

representation of each training model. Clustering offers several benefits such as reducing the number of false positives and improved accuracy (Wang & Stolfo, 2004).

- *Example 3-Anagram training*. Contrary to PAYL, Anagram requires more training as higher order n-grams need longer periods of training to build high-quality models. According to Wang et al., (n.d.), it is necessary to calculate the rate at which the technique is able to observe new n-grams within normal data flow. For example, when new n-grams decrease to a minimum the model is likely to be more stable. The rate can then be applied to detect attacks. However, there is a peak threshold where the false positive rate increases over time. This suggests the binary-based approach cannot cope well with noisy or infected training data. Consequently, Anagram must function as a semi-supervised anomaly detector where incoming data is filtered through Snort to eliminate previously known attacks. If there is a match the packet is dropped (Wang, Parekh, & Stolfo, 2006).

Testing

Testing is a critical process necessary to evaluate the capability of the IDS and to identify intrusions. Bolzoni and Etalle (n.d.) determined that completeness and accuracy are the main criteria to evaluate a system based on the model employed and associated qualities:

- Completeness is defined as $\frac{\text{Number of True Positives}}{\text{Number of True Positives} + \text{Number of False Negatives}}$
- Accuracy is defined as _____

Accuracy measures how precise the IDS can detect attacks while completeness measures the percentage of attacks detected, known as the detection rate (Lee, et al., n.d.). The following definitions describe the four metrics commonly used to evaluate the IDS:

- Definition 1. True positive is an actual alert triggered from a successful attack.
- Definition 2. False positive is a false alarm, an anomalous event of no significant threat unless proven otherwise.
- Definition 3. True negative produces no alarm; an intrusion has not occurred and activity appears to be normal.
- Definition 4. False negative is not detected because the event simulates normal activity.

The threshold also has considerable weight when setting up and testing the IDS. According to Bolzoni & Etalle (n.d.), a lower threshold yields more alarms, significantly raising the false positive rate. In contrary, a higher threshold yields lower alarms and thus would lower the false positives. While setting the threshold is entirely subjective, ultimately it should be set to capture all attacks. Setting the appropriate threshold is based on the algorithm or groups of algorithms employed by the IDS.

Implementation

One of the first steps when implementing the IDS is to determine where the product will be deployed. The *control strategy* will determine whether monitoring, detection, and reporting occur from a centralized or distributed location. The difference being a central control strategy would monitor the network from a single location, while a distributed control strategy would channel information to other systems (Bace & Mell, 2001). According to Axelsson (2000a), NIDSs can be setup in real-time or in batch mode or interval-based. In real-time, information is feed continuously from a source or network traffic; whereas the interval-based IDS observes specific points and uses a concept known as store and forward. The IDS can be setup external to the firewall, on a major backbone, or on a critical subnet (Bace & Mell, 2001).

Secondly, one must choose the method to analyze data, which depends on the input source. According to Bolzoni et al., (n.d.), the *context* is knowledge of the techniques attackers use to execute an attack and how this may influence the source to be analyzed. For example, it is logical to monitor HTTP traffic to detect SQL injections or data flow to detect DoS attacks. However, it is important to understand that the input information is often insufficient to detect sophisticated attacks. Regardless, a universal algorithm to detect any type of attack does not exist. Thus, intrusion detection will employ a number of techniques to detect a combination of attacks. While there are no guidelines suggesting how to build a payload anomaly-based IDS, there are three general phases that summarize how to perform anomaly intrusion detection (Bolzoni et al., n.d.):

- Phase I. Acquire input resource knowledge.
- Phase II. Select a model based on the input data; this will be based on the type of threat and protocol that one wished to filter.
- Phase III. Use different methods to analyze the data for intrusions.

Ideally, the three phases offer context knowledge, data abstraction, and diversification analysis. These are the critical phases needed to build the IDS while dealing with the complexity associated with payload content analysis.

It is evident payload processing can be quite complex. It involves a series of phases to preprocess the data, analyze it, cluster it, and then on to perform the intrusion detection. Preprocessing is the method to filter data and extract the necessary information to form the knowledge discovery. Once this is achieved the analysis phase begins, techniques are employed to examine payload content for n-grams; the clustering phase will then ensure objects are placed

into similar groups. The intrusion detection is last part of the phase. The second part of this chapter addressed the implementation of IDSs.

Implementing the IDS requires a thorough background and experience with anomaly-based systems. The first step is to determine where to position the IDS; this requires planning. In addition, one should determine whether the system will be setup using a centralized or decentralized configuration. However, training is a prerequisite as the comparison between clean data and attack (noisy) data must take place prior to deploying the system. The IDS must first learn the network. The administrator will then need to set a threshold to determine the number of false positives to generate. Naturally, one should understand how the threshold affects the detection rate. For example, a higher threshold may produce excessive false positives.

Chapter 8 – Evaluation

The primary purpose to test the effectiveness of IDSs is to determine the accuracy rate and detection rate (Fessi, et al., 2005). The accuracy rate is calculated based on the number of false positives generated, while the detection rate is based on how well the IDS detect attacks. As previously discussed, the typical metrics for evaluating systems are: False positive rate, true positive rate, false negative rate, and true negative rate. Whether the anomaly detector meets or exceeds a standard depends on the criteria set by the system or administrator. Factors that may impact accuracy are algorithms, correlation techniques, speed, and the amount of data to analyze. This chapter includes scenario-based examples demonstrating the effectiveness of SOM, PAYL, Anagram, and POSEIDON in detecting malicious attacks.

SOM

Kayacik, et al., (n.d.) tested a hierarchical SOM model using the International Knowledge Discovery and Data Mining Tools Competition (KDD-99) test bed. The objective was to stress the SOM detection capability using common metrics. They soon discovered SOMs are adaptable systems capable of filtering data at excessive speed while maintaining high accuracy rate. This was realized using well-defined models during the training phase. They also discovered that 6-basic features were necessary for payload processing and to build the knowledge domain. These are the necessary attributes to perform the intrusion detection.

Table 2 illustrates the differences between a 2-layer and 3-layer SOM hierarchy. The false positive rate is significantly higher with a 2-layer SOM due to the number of features considered during the analysis. The reason is the each hierarchical layer is more selective, which impacts the detection rate. The test also proved the 3-layer architecture significantly reduces the

false positive rate. Ultimately, the test confirmed SOM's high detection rate and capability of functioning under stressed conditions.

Table 2

Test Results Showing Differences between 2-Layer and 3-Layer SOM Hierarchy

Hierarchy	Partition	False Positive Rate (%)	Detection Rate (%)
2-Layer SOM	10% KDD	10.6	99.8
	Normal	15.7	99.9
	50/50	8.25	99.8
3-Layer SOM	10% KDD	1.75	99.7
	50/50	2.6	99.1

Note. The data collected represents a comparison between 2-layer and 3-layer SOM hierarchy. The additional 10% KDD accounts for 14 supplementary attacks added to the training data to stress the SOM. The 50/50 is a balance of the number of attacks and exemplars. The data collected is from Kayacik, et al. (n.d.).

POSEIDON

POSEIDON has the advantage of using SOMs to classify or cluster similar packets during preprocessing and PAYL to detect the actual attack. A visual representation provides the administrator with a clearer perspective of where attacks occur. Bolzoni & Etalle (n.d.), argued the optimal method to detect anomalies is to separate traffic according to regular-text and irregular-text patterns. Table 3 represents a comparison of Anagram and POSEIDON and their ability to detect regular HTTP requests with raw-data parameters removed. It appears both methodologies can detect the same type of attacks, but POSEIDON clearly has a lower false positive rate.

Table 3

Comparison between Anagram and POSEIDON

#Training Samples	Anagram	POSEIDON
5000	20/20* 144783**	20/20 1461
10000	20/20 133023	20/20 1387
20000	20/20 121484	20/20 1306
50000	20/20 100705	20/20 1251

Note. The table shows a comparison between Anagram and POSEIDON. Evidently, Anagram shows a significantly higher number of false positives. The data collected is from Bolzoni and Etalle (n.d.).

* Attacks

**Number of False Positives

Bolzoni and Etalle (n.d.) also compared the detection capabilities of PAYL and POSEIDON using TCP/IP packets. The test included similar requirements and conditions provided by the originators of PAYL. A successful attack is determined only when correctly identified as malicious, while a packet incorrectly identified is considered a false positive. Table 4 represents a comparison of the PAYL and POSEIDON models. The results are based on completeness, the combination of detection rate and accuracy rate. Clearly, POSEIDON outperforms PAYL in terms of detection capability and accuracy rate.

Table 4

Comparison between PAYL and POSEIDON

Model Type		PAYL	POSEIDON
Number of Models		4065	1622
Applications			
HTTP	DR	89.00%	100.00%
	FP	0.17%	0.0016%
FTP	DR	95.50%	100.00%
	FP	1.23%	0.93%
TELNET	DR	54.17%	95.12%
	FP	4.71%	6.72%
SMTP	DR	78.57%	100.00%
	FP	3.08%	3.69%

Note. DR= detection rate, FP= false positive rate. The table shows a comparison between POSEDON and PAYL demonstrating differences in detection rate and false positives generated. The values in this table are from Bolzoni and Etalle (n.d.).

In Table 5, POSEIDON achieved a perfect detection rate, but the system also generated a relatively high number of false positives. Bolzoni et al., (n.d.) argued these numbers can be reduced using ATLANTIDES, which is simply an ingress/egress correlation technique adopted specifically for client to server requests. ATLANTIDES eliminates the number of false positives generated from false alerts. The data in Table 5 shows that while the detection rate remains the same, the number of false positives is reduced by approximately half.

Table 5

Comparison between POSEIDON and ATLANTIDES

Protocol		POSEIDON	POSEIDON + ATLANTIDES
HTTP	DR	100%	100%
	FP	1683 (2.83%)	774 (1.30)

Note. DR= detection rate, FP= false positives. This comparison demonstrates how the addition of ATLANTIDES greatly reduces the number of false positives. The values in this table are from Bolzoni, Crispo, and Etalle (n.d).

According to Bolzoni et al., (n.d.), current anomaly-based systems are unable to group alerts into specific categories based on the type of attack. Their solution is to employ algorithms that extract information and classify attacks accordingly. PANACEA meets these requirements by employing *Alert Information Extractor* (AIE) and *Attack Classification Engine* (ACE). The AIE performs the processing and extracts information, which is then passed along to the ACE for classification. SVM and RIPPER are the attack classification engines. The two techniques are able to classify alerts from high-dimensional data. SVM "outperforms competitors in 50% of tests and ranks in the top 3 in 90% of them" (Bolzoni, Etalle, & Hartel, n.d., p.8).

When comparing SVM and RIPPER, both appear successful with categorizing different types of attack classes. Notice in Table 6 there is minor differences in the overall percentage of attacks correctly classified. According to Bolzoni, Etalle, and Hartel (n.d.), the classification of attacks will exceed the 75% detection rate in every example. However, SVM is superior to RIPPER when high diversity among classes is present and level of accuracy. In addition, it is noted that RIPPER will outperform SVM when the class has sufficient training samples.

Table 6

Comparison between SVM and RIPPER Classification Techniques

Attack Class	SVM	RIPPER
Path Traversal	98.60%	99.10%
Cross-site Scripting	97.50%	98.40%
SQL Injection	97.60%	96.20%
Buffer Overflow	37.50%	37.50%
Percentage of total attacks correctly classified	98.00%	97.70%

Note. The data shows SVM and RIPPER achieve high detection rates, demonstrating both systems are highly effective. The values in this table are from Bolzoni, et al. (n.d.).

PAYL

Wang and Stolfo (n.d.) tested the capabilities of PAYL using 1999 DARPA IDS dataset and Columbia University network. The experiment included scanning incoming traffic on ports 21, 23, 25, and 80. The system used 5 different criteria (e.g., number of bytes analyzed) during the process analysis. The "per packet model" uses the entire payload of each packet for threats. Table 7 represents the test results and approximate values. In only one particular instance, PAYL was able to achieve 100% detection rate. The detection rate for port 80 was nearly at 100%, which demonstrates PAYL's ability to cope with attacks embedded within web traffic. The overall detection rate when restricting false positive rate to less than 1% is 58.8%. In analyzing the data collected from Table 7, the detection capability is fairly high; more than 40% of the attacks may go undetected.

Table 7

PAYL Detection Capability Using Port Criteria

Per Packet Model (Entire Payload)		
Port	Detection Rate (%)	False Positive Rate (%)
21	95	1.1
25	79	3.0
23	51	4.0
80	89	0.2
Overall Detection Rate < 1%	57/97	
False Positive Rate	58.8%	

Note. Port 21= FTP, Port 25= SMTP, Port 23= TELNET, and Port 80= HTTP. The values in this table are from Wang and Stolfo (n.d.).

Smith, Matrawy, Chow, and Abdelaziz (2008) argued the optimal way of detecting worm distribution is to scan incoming and outgoing traffic. Wang, Cretu, and Stolfo (n.d.) placed this theory into practice by testing PAYL for worm detection. The experiment included an anonymous business and Columbia University network with variations of Code Red I, Code Red II, WebDAV, and other worms that exploit Windows media services. The worm set was placed randomly in the data. An example of Code Red packet is illustrated in Figure 10.

```
GET./default.ida?
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXX%u9090%u6858%ucb3%u7801%%u9090%u9090
%u8190%u00c3%u0003%u8b00%u531b%u53ff%u0078
%u0000%u0
```

Figure 10. Illustration of Code Red II Packet (Stolfo, Parekh, & Locasto, 2007).

Wang, et al., (n.d.) compared three sites labeled EX, W, and W1 using a Receiver Operating Characteristic (ROC) curve, which is a method to compare and evaluate properties of IDSs. PAYL was able to detect 4 worms among 40 packets. More than half of the worms were detected and classified as anomalous. The false positive rate for the EX dataset was 0.1%. The test also included the W32.Blaster worm, which was easily detected. Remarkably PAYL was able to detect fragmented worms using ingress/egress correlation proving worms can be identified and stopped at the first attempt to propagate.

Stolfo, Parekh, and Locasto (2007) continued with their testing using ingress/egress correlation techniques previously identified as SE, LCS, and LCSeq. An alternative experiment was setup to capture all incoming payload packets with unlimited buffer size. The threshold was lowered to reduce false positives and to capture 100% of the attacks deemed present. The adjustment will allow for increased noise to test whether the correlation techniques can separate normal data from malicious content.

According to Stolfo et al., (2007) LCSeq has the slowest correlation rate (speed), but is considered appropriate as a baseline for raw payload correlation. Ultimately, all of the techniques correlated the fragmented attacks. However, there were limitations with detecting polymorphic worms. Table 8 summarizes the test results using the different correlation techniques. Notice that LCS and LCSeq detected all worm propagations with no false alerts. SE failed to detect the propagation because it must correlate the entire packet, which is not feasible with fragmented worm packets.

Table 8

String Correlation Techniques

String Type	Detect Propagate	False Alerts
SE	No	No
LCS	Yes	No
LCSeq	Yes	No

Note. The table shows the distinction between the three types of ingress/egress correlation techniques employed by PAYL. The values in this table are from Wang, Cretu, and Stolfo (n.d.).

The experiment also included Manhattan Distance and LCSeq of Z-string to evaluate the similarity between strings. Interesting is that LCSeq detects polymorphic worms that attempt to change the padding, the false bits inserted within the payload using cross-site collaboration. This method of correlating attacks could resolve the problem with worms hibernating and which no longer produce a record within a buffer. Wang et al., (n.d.) determined the best option would be to examine different domains or sites that demonstrated common characteristics of an attack. In theory, cross examination will determine whether payload models are different across numerous sites.

A major concern during testing was lack of privacy when sharing information among sites. For example, a false positive could reveal actual payload content. This is an important issue to consider when distinguishing among different analysis techniques. For example, a 1-gram frequency distribution is capable of preserving the privacy of payload content. The Z-string correlation technique is another option to correlate suspicious payloads. False positives in this case would not reveal payload content.

Anagram

Anagram has several advantages over PAYL such as compactness, resiliency, and security. The Bloom Filter is a key component and is described as a one-way data structure where data is inserted, but not extracted. Data insertion can be verified if presented a second time to the Bloom Filter. While the Bloom Filter is relatively small in size (e.g., 10k bits) it can verify thousand of entries with great accuracy. However, if the filters are full it will generate excessive false positives when entries map to the same location. Lastly, Bloom Filters have added security because it employs a one-way hashing algorithm. Reverse engineering is virtually impossible. This means content could be filtered without releasing sensitive information to the public (Stolfo et al., 2007).

Wang et al., (n.d.) evaluated and compared the binary-based approach of Anagram with frequency-based approach of PAYL. The former approach yielded better results than the latter approach with less false positives. During the experiment Anagram had a 0.01% false positive rate. Anagram models a combination of n-grams (e.g., 2-gram, and 3-gram, etc). However, larger n-grams will require further training. The length of training will depend on whether the false positive rate increases significantly. Bloom Filters also conserve memory overhead due to improved data structure. However, large filters can waste memory. While the binary approach is fast and does not require excessive memory, it is intolerant to noisy data. In addition, manual cleanup is impractical for large training data (Wang et al., n.d.). Another advantage of Anagram is its ability to produce signatures. Table 9 illustrates the difference between frequency-based and binary-based approaches when achieving 100% detection rate.

Table 9

Comparison between Frequency-Based and Binary-Based Approaches

Methodology	3-gram	4-gram	5-gram	6-gram	7-gram	8-gram
Freq-based	30.40205	0.18559	0.08858	0.13427	0.30792	0.41477
Binary-based	0.02109	0.01406	0.00634	0.00703	0.00914	0.00914

Note. The table is a comparison between frequency- based and binary-based approaches. The higher the n-grams the less accuracy is achieved. The values in this table are from Wang, et al. (n.d.).

Unlike PAYL which relies on frequency distribution to detect variations within packets, Anagram successfully avoids mimicry attacks using *randomized modeling*, a method to scramble a portion of the payload that is extracted and modeled. With this approach hackers are unlikely to mimic byte sequences using a padding technique or fillers which contain false data. Wang et al., (n.d.) tested Anagram using a polymorphic worm with different padding lengths. Table 10 shows the padding length of different packets represents by a tuple (*x, y*), where *x* is the number of bytes and *y* is the variant sequence. In the worst case scenario (e.g., 1460, 100), the false positive rate reached approximately 0.1%. It is important to note that the longer the packet length the greater the false positive rate.

Table 10

Padding Length Based on Mimicry Attacks

Version (x, y)	418.10	418.100	730.100	730.100	1460.10	1460.100
Padding Length	125	149	437	461	1167	1191

Note. The values in this table are from Wang, et al. (n.d.).

Based on Anagram's improved performance and ability to cope with embedded payload attacks, it raises the standard of intrusion detection. A primary reason is that hackers must be more creative with innovating new attacks. Furthermore, the anomaly-based system creates unique signatures to identify attacks across multiple sites. These signatures can be imported into a misuse IDS. Overall, Anagram offers several advantages that include improved accuracy, computational efficiency, fast correlation, and alert signature generation.

Currently there is no standard framework to determine or measure the effectiveness of payload anomaly-based IDSs. And theory alone cannot predict performance. As a result, it is necessary to transition theory into practice and to determine whether IDSs perform as originally intended. Testing is required to fully comprehend and justify the effectiveness of IDSs. This is achievable with a data set consisting of multiple attacks randomly injected and not manipulated to alter test results. Testing and evaluating a product in a lab environment should provide enough the knowledge to anticipate what is expected in the real-world.

Chapter 9 – Results

The SOM as a standalone IDS proved effective at detecting malicious content. Kayicik et al., (n.d.), proved the system could detect DoS, Probe, U2R, and R2L attacks. However, the detection rates varied considerably among the attack exemplars. In addition, results varied when transitioning from a 2-layer to 3-layer hierarchy. For example, as the number of hierarchies increased the false positive rate declined significantly. Kayicik et al., (n.d.), also noted that the detection rates varied by the number of features selected during preprocessing. Regardless, the SOM detected a wide range of attacks despite Bolzoni et al.'s, (n.d.) argument that the system could not handle the payload distribution of current applications. The statement is still debatable as there is no knowledge proving otherwise.

POSEIDON's test results confirmed the integration of a SOM with PAYL significantly improved the detection rate. The SOM set up as a preprocessing tool extracts packet characteristics, while the integration of PAYL determines whether an attack is present. Combining the advantages of both systems exceeded PAYL's detection capability as a standalone system. All in all, POSEIDON proved highly versatile with the capability to detect multiple attacks, some which targeted CPU instructions and protocol violations. The addition of ATLANTIDES and PANACEA further increased its attack capability. ATLANTIDES reduced HTTP false positives by nearly 50%, while PANACEA categorized attacks with an approximate 97% accuracy rate in all categories.

While test results confirmed POSEIDON's exemplary means to detect network attacks, several issues remain in question. It is uncertain whether a large network of users would hinder the analyst's ability to identify threats within a one-dimensional SOM. In addition, there is no evidence suggesting how POSEIDON will react to small network changes or whether the

detection rate and mapping over time would be negatively impacted. Another concern is POSEIDON's dependency on additional systems to perform data clustering and intrusion detection. A hybrid solution with the integration of SOM and PAYL may create further complexity as both systems must be trained separately or collectively. Furthermore, POSEIDON generated a relatively high number of false positives in several of its tests. While higher numbers of false positives is expected of anomaly-based systems, they should be minimized. Bolzoni et al., (2007) argued ALANTIDES could help resolve this problem, but limited details explain how the system would interface with POSEIDON. Lastly, the system is not engineered to detect DDoS or DoS attacks. Perhaps the misuse-based IDS may help resolve this issue with manually created signatures.

While PAYL has its limitations, it does offer several distinct capabilities. The IDS can rapidly identify anomalies using 1-gram analysis and ingress/egress correlation, regardless of whether the attack is fragmented, as the case with worm attacks. PAYL functions as a distributed detection system, sharing information between sites. It can also preserve user privacy with less invasive techniques than Anagram. However, the system has a few drawbacks. While the originators of the system claim it can detect diverse attacks, it is designed mostly to identify and correlate worm attacks. In addition, PAYL's use of byte distributions to detect anomalous activity allows attackers to execute mimicry attacks, simulating normal network activity. Furthermore, PAYL uses payload length to create models, which can be a number from 0 to 1460 on an Ethernet-based LAN (Wang & Stolfo, 2004). The number of models could increase exponentially with relative ease.

Anagram is a unique anomaly detector that offers several advantages over its predecessor PAYL. The use of Bloom Filters is a major success as it provides enhanced signature matching and reduction of false negatives. The system also generates signatures, which could be imported into the misuse-based IDS such as SNORT or BRO (Parekh et al., 2006). Furthermore, Anagram provides improved security using byte sequence analysis, which indentifies any illegal attempt to manipulate payload content. Zero-day exploits are detectable with this capability. The use of higher order n-grams to detect anomalous payload attacks is another major advantage, considering the limitations of lower order n-grams, where n =1.

Despite the advantages noted from above, Anagram has several drawbacks. The IDS must be calibrated when dealing with noisy data. As a result, the system must perform in a semi-supervised mode as opposed to unsupervised mode as originally designed. Regardless, manual clean-up is impractical with excessive training data. Perhaps a more concerning issue is dealing with a violation of user privacy. Anagram's deep analysis and inspection of packet information may expose sensitive data to the public. Lastly, the problem with Bloom Filters becoming saturated will cause excessive false positives. It appears there is no solution to resolve the current problem (Wang, Parekh, & Stolfo, n.d.).

Chapter 10 – Conclusions

In this analysis, a multifaceted approach was necessary to determine the effectiveness of IDSs and their capability to identify malicious attacks. Referring back to Chapter 8, the plan was to highlight the accuracy and detection capability based on the collection of quantitative data. However, to focus simply on numeric data would undermine the behavioral aspects of each system. As a result, qualitative features were integrated to enhance the knowledge base and bridge the gap between management-oriented and technical-oriented audiences. The behavioral science and design science research methodologies proved effective in meeting these goals.

If one should argue, validating whether IDSs are effective at detecting all attacks is highly improbable. However, testing provides a good indication of performance, the level in which IDSs are able to detect specific attacks. This is feasible provided the training data set is current and free from manipulation. Optimally, testing should include randomly injected attacks. The IDSs in this study are all capable of detecting payload embedded attacks, but there are limitations. Based on the data collection, POSEIDON raises the standard with respect to identification, analysis, and overall detection. It offers exceptional performance with the capability to detect lower-level and higher-level protocol attacks.

Recommendation

By far, POSEIDON is the most versatile system with the greatest capability to detect a variety of attacks. It is the most effective system based on completeness and accuracy, a criteria recognized by Bolzoni et al., (n.d.). Perhaps a key feature is POSEIDON's ability to perform byte-level analysis using SOMs. The use of neural technology provides a significant advantage when payload content analysis is a requirement. POSEIDON's ability to interface with other systems provides the scalability needed to improve the detection rate while maintaining a

relatively high accuracy rate. This has been confirmed in numerous test results. A final recommendation would be to interface POSEIDON with a signature-based system to form a hybrid solution. Ideally, the system should be setup within a small-sized to medium-sized network.

The next evolution of payload anomaly-based IDSs must be highly versatile, capable of analyzing high-dimensional data content to deter malicious attacks from propagating. At a minimum, these systems should be automated, should be resilient to attacks, and should be able to achieve high accuracy rate. Continuous research and innovation will be necessary to achieve these objectives. Knowledge and innovation will play a greater role as the level of sophistication and spread of malicious attacks continue. In the future, the ability to produce a highly dynamic and efficient system to detect both known and unknown attacks will require a single architecture that includes anomaly-based and misuse-based systems.

References

Amini, M., Jalili, R., & Shahriari, H.R. (2006). RT-UNNID: A practical solution to real-time network-based intrusion detection using unsupervised neural networks. *Computers & Security, 25*, 459-468. Retrieved May 10, 20010 from Elsevier digital database.

Axelsson, S. (2000a). Intrusion detection systems: A survey and taxonomy. Chalmers University of Technology, Goteborg, Sweden. Retrieved March 23, 2010, from Google Scholar.

Axelsson (2000b). The base rate fallacy and the difficulty of intrusion detection. *ACM Transactions on Information and System Security, 3*(3), 186-205. Retrieved March 23, 2010, from Google Scholar.

Aydin, M.A., Zaim, A.H., & Ceylan, K.G. (2009). A hybrid intrusion detection system design for computer network security. *Computer and Electrical Engineering, 35*(3), 517-526. Retrieved March 29, 2010, from Elsevier database.

Bace, R., & Mell, P. (2001). NIST Special Publication on Intrusion Detection Systems. Retrieved September 4, 2010, from http://csrc.nist.gov/publications/PubsSPs.html

Bolzoni, D., Zambron, E., Etalle, S., & Hartel, P. (2006). POSEIDON: A 2-Tier anomaly based intrusion detection system. Retrieved April 7, 2010, from Elsevier database.

Bolzoni, D., & Etalle, S. (n.d.). Approaches in anomaly-based intrusion detection systems. *University of Twente, Enschede; Netherlands*. Retrieved March 23, 2010, from Google Scholar.

Bolzoni, D., & Etalle, S. (n.d.). Boosting web intrusion detection systems by inferring positive signatures. *University of Twente, Enschede; Netherlands.* Retrieved May 7, 2010, from Google Scholar.

Bolzoni, D., Crispo, B., & Etalle, S. (2007, November). ALTANTIDES: An architecture for alert verification in network intrusion detection systems. *LISA '07*, 141-152. Retrieved June 11, 2010, from Google Scholar.

Bolzoni, D., Etalle, S., & Hartel, P.H. (n.d.). PANACEA: Automating attack classification for anomaly-based network intrusion detection. Retrieved June 11, 2010, from Google Scholar.

Broder, A., & Mitzenmacher, M. (2004). Network applications of bloom filters: A survey. *Internet Mathematics, 1*(4), 485-509. Retrieved June 11, 2010, from Google Scholar.

Cassady, J. (n.d.). Artificial neural networks for misuse detection. Retrieved May 7, 2010, from Google Scholar.

Cheema, F.M., Akram, A., & Iqbal, Z. (2009, July). Comparative evaluation of header vs. Payload based network anomaly detectors. *Proceedings of the World Congress on Engineering, 1*, 1-5. Retrieved April 10, 2010, from Google scholar.

Chen, W., Hsu, S., & Shen, H. (2004). Application of SVM and ANN for intrusion detection. *Computer and Operations Research, 32*, 2617-2634. Retrieved May 30, 2010, from Elsevier database.

Chen, C., Chen, Y., & Lin, H. (2009). An efficient network intrusion detection. *Computer Communications, 41*(1). Retrieved March 30, 2010, from Elsevier database.

Cohen, W. (1995). Fast effective rule induction. *Proceedings of the 12th International Conference on Machine Learning*. Retrieved March 30, 2010 from Google Scholar.

Convery, S. (2004). *Network security architecture*. Cisco Press: Indianapolis, IN.

Delone, W.H., & McLean, E.R. (2001). Information systems success: The quest for the dependent variable. *Information Systems Research, 3*(1), 60-93. Retrieved March 28, 2010, from Elsevier database.

Depren, O., Topallar, M., Anarim, E., & Ciliz, K. (2005). An intelligent intrusion detection for anomaly and misuse detection in computer systems. *Expert Systems with Applications, 29*(4), 713-722. Retrieved March 28, 2010, from Elsevier database.

Dharmapurikar, S., Krishnamurthy, P., Sproull, T., & Lockwood, J. (n.d.). Deep packet inspection using parallel bloom filters. *Washington University in St. Louis*. Retrieved May 7, 2010, from Google Scholar.

Dreger, H., Feldmann, A., Main, M., Paxson, V., & Sommer R. (n.d.). Dynamic application-layer protocol analysis for network intrusion detection. Retrieved April 12, 2010, from Google scholar.

Eskin, E., Arnold, A., Prerau, M., Portnoy, L., & Stolfo, S. (n.d.). A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data. *Columbia University*, 1-4. Retrieved March 23, 2010, from Google Scholar.

Estevez, J. M., Garcia, P., & Diaz-Verdejo, J. E. (2004a). Anomaly detection methods in wired networks: a survey and taxonomy. *Computer Communications, 27*(16), 1569-1584. Retrieved March 29, from Elsevier database.

Estevez, J. M., Garcia, P., & Diaz-Verdejo, J.E. (2004b). Measuring normality in HTTP traffic for anomaly-based intrusion detection. *Computer Networks, 45*(2), 175-193. Retrieved March 29, from Elsevier database.

Fessi, B.A., Hamdi, M., Benabdallah, S., & Boudriga, N. (2005). A decisional framework system for computer network intrusion detection. *European Journal of Operational Research, 177*, 1824-1838.

Forrest, S., & Hofmeyr, S.A. (2002). A sense of self for Unix processes. *In S&P '96:*

Proc. 17th IEEE Symposium on Security and Privacy, 120-128. Retrieved March 23, 2010, from IEEE digital database.

Garcia-Teodoro, P., Diaz-Verdejo., J., Macia-Fernandez, G., & Vazquez, E. (2008). Anomaly-based network intrusion detection: techniques, systems and challenges. *Computer and Security, 28*, (1-2), 18-28. Retrieved March 29, 2010, from Elsevier database.

Girardin L. (1999). An eye on network intruder-administrator shootouts. *Proceedings of the Workshop on Intrusion Detection and Network Monitoring,* Santa Clara, CA, USA, April 9-12, 1999. Retrieved May 22, 2010 from Google Scholar.

Grover, V., Jeong, S. R., & Segars, A.H. (1996). Information systems effectiveness: The construct space and patterns of application. *Information & Management, 31*, 177-191. Retrieved March 29, 2010, from Elsevier database.

Gu, G., Fogla, P., Dagon, D., & Wenke, L. (2006). Measuring intrusion detection capability: An information-theoretic approach. *ASIACCS '06*. Retrieved May 4, 2010, from ACM digital database.

Hevner, A. R., March, S.T., Park, J., & Ram, S. (2004). Design-science in information systems research. *MIS Quarterly, 28*(1), 75-105. Retrieved May 4, 2010, from Elsevier digital database.

Hsu, F., Guo, F., & Chiueh, T. (2006, December). Scalable network-based buffer overflow attack detection. *ANCS06'*, 1-7. Retrieved April 26, 2010, from ACM digital database.

Hwang, K., Liu, H., & Chen, Y. (2004). Cooperative anomaly and intrusion detection alert correlation in networked computing systems. *IEEE Transaction on Dependable and Secure Computing*. Retrieved March 20, 2010, from IEEE digital database.

Ingham, K.L., & Inoue, H. (n.d.). Comparing anomaly detection techniques for HTTP. Retrieved April 26, 2010, from Google Scholar.

Kayacik, G.H., Zincir-Heywood, N.A., & Heywoord, M.I. (2007). A hierarchical SOM-based intrusion detection system. *Artificial Intelligence, 20*, 439-451. Retrieved May 10, 20010 from Elsevier digital database.

Kevin, L.F, Rhonda, R.H., & Jonathan, H.R. (1990). A neural network approach towards intrusion detection. *Proceedings of the 13th National Computer Security Conference*, 125–34. Retrieved May 10, 20010 from Google Scholar.

Kiani, M., Clark, A., & Mohay, G. (2008). Evaluation of anomaly based character distribution models in the detection of SQL injection attacks. *The 3rd Int'l Conference on Availability, Reliability, and Security,* 47-55. Retrieved March 20, 2010, from IEEE digital database.

Kohonen, T. (1988). An introduction to neural computing. *Neural Networks, 1*(1), 3-16. Retrieved May 10, 20010 from Elsevier digital database.

Krugel, C., Toth, T., & Kirda, E. (2002). Service specific anomaly detection for network intrusion detection systems. *SAC '02.* Retrieved March 22, 2010 from ACM digital database.

Labib, K., & Vemuri, R. (n.d.). NSOM: A real-time network-based intrusion detection system using self-organizing maps. Retrieved May 10, 20010 from Google Scholar.

Lakov, P., Dussel, P., Schafer, C., & Rieck, K. (n.d.). Learning intrusion detection: supervised or unsupervised? Retrieved March 22, 2010 from Google Scholar.

Lee, W., Stolfo, S. J., & Mok, K. (2000, December). Adaptive intrusion detection: A data mining approach. *Artificial Intelligence Review, 14*(6), 533-567. Retrieved March 22, 2010 from Google Scholar.

Lee, W., Stolfo, S. J., Chan, P. K., Eskin, E., Fan, E., W., & Miller, M. et al. (2003). Real time data mining-based intrusion detection. Retrieved March 28, 2010 from Google Scholar.

Lichodzijewski, P., Heywood, A. N., & Heywood, M.I. (2002a). Dynamic intrusion detection using self-organizing maps. *The 14th Annual Canadian IT Security Symposium.* Retrieved March 28, 2010 from Google scholar.

Lichodzijewski, P., Zincir-Heywood, A. N., & Heywood, M.I. (2002b). Host-based intrusion detection using self-organizing maps. *Proceedings of the 2002 IEEE World Congress on Computational Intelligence*. Retrieved May 11, 2010 from Google scholar.

Liu, Z., Florez, G., & Bridges, S.M. (n.d.). A comparison of input representations in neural networks: A case study in intrusion detection. Retrieved May 11, 2010 from Google scholar.

March, S. T., & Smith, G.F. (1995). Design and natural science research on information technology. *Decision Support Systems*, 15, 251-266. Retrieved May 12, 2010, from Elsevier database.

Maselli, G., Deri, L., & Suin, S. (n.d.). Design and implementation of an anomaly detection system: An empirical approach. Retrieved March 22, 2010, from Google Scholar.

Parekh, J. J., Wang, K., & Stolfo, S. J. (2006, September). Privacy-preserving payload-based correlation for accurate malicious traffic detection. *SIGCOMM '06 Workshops*. Retrieved May 6, 2010, from ACM digital database.

Paxson, V. (1999). BRO: A system for detecting network intruders in real-time. *Computer Networks, 31*(23-24), 2435-2463. Retrieved April 22, 2010, from Google Scholar.

Paxson, V., Rothfuss, J., & Tierney, B. (2004). *Bro quick start guide*. Retrieved April 22, 2010, from Google Scholar.

Perdisci, R., Ariu, D., Fogla, P., Giacinto, G., & Lee, W. (2009). McPAD: A multiple classifier system for accurate payload-based anomaly detection. *Computer Networks, 53,*864-881. Retrieved May 26, 2010, from Elsevier digital database.

Perona, I., Gurrutxaga, I., Arbelaitz, O., Martin, J. I., Muguerza, J., & Perez, J. (2008). Service-independent payload analysis to improve intrusion detection in network traffic. Australian Computer Society. Retrieved May, 18, 2010, from Google Scholar.

Pfleeger, C. P., & Pfleeger, S. L. (2007). *Security in Computing*. Boston, MA: Prentice Hall.

Pietraszek, T., & Tanner, A. (2005). Data mining and machine learning—Towards reducing false positives in intrusion detection. *Information and Security Report, 10*(3), 1-18. Retrieved May 12, 2010, from Elsevier digital database.

Powers, S. T., & He, J. (2008). A hybrid artificial immune system and self-organizing map for intrusion detection. *Information Sciences*, 178, 3024-3042. Retrieved May 10, 2010, from Elsevier digital database.

Rajan, S. S., & Cherukuri, V. K. (n.d.). An overview of intrusion detection systems. Retrieved May, 22, 2010, from Google Scholar.

Ramadas, M., Ostermann, S., & Tjaden, B. (2003). Detecting anomalous network traffic with self-organizing maps. *In G. Vigna, E. Jonsson, and C. Kruegal (Eds.): RAID*, 2820, 36-54. Retrieved April 27, 2010, from Google scholar.

Rhodes, B. C., Mahaffey, J. A., & Cannady, J. D. (2000). Multiple self-organizing maps for intrusion detection. *Proceedings of the 23rd National Information Systems Security Conference*. Retrieved May 10, 20010 from Google Scholar.

Rosemann, M., & Vessy, I. (2008). Toward improving the relevance of information systems research to practice: The role of applicability checks. *MIS Quarterly, 32*(1), 1-22. Retrieved May 10, 2010, from Elsevier digital database.

Smith, C., Matrawy, A., Chow, S., & Abdelaziz, B. (2008). Computer worms: Architectures, evasion strategies, and detection mechanisms. *Journal of Information Assurance and Security, 4*, 69-83. Retrieved April 13, 2010, from Google scholar.

Stolfo, S., Parekh, J., & Locasto, M. (2007). Developing collaborative profiles of attackers: A longitudinal study. *Columbia University Final Progress Report.* Retrieved May 28, 2010, from Google scholar.

Thorat, S. A., Khandelwal, A. K., Bruhadeshwar, B., & Kishmore, K. (n.d.). Payload content based anomaly detection. Retrieved April 8, 2010, from Google scholar.

Verwoerd, T., & Hunt, R. (2002). Security architecture testing using IDS—a case study. *Computer Communications, 25* (15), 1288-1294. Retrieved March 29, 2010, from Elsevier data base.

Vesanto, J., Himberg, J., Alhoniemi, E., & Parhankangas, J. (1999, November). Self-organizing map in Matlab: the SOM tool box. *Proceedings of the Matlab DSP Conference,* 16-17, 35-40. Retrieved April 8, 2010, from Google scholar.

Vliet, F. (n.d.). Turnover POSEIDON: Incremental learning in clustering methods for anomaly based intrusion detection. Retrieved May 10, 2010, from Google Scholar.

Wang, K., & Stolfo, S. J. (2004). Anomalous payload-based network intrusion detection. *In Jonsson, E., Valdes, A., Almgren, M., eds.: Proceedings 7th Symposium on Recent Advances in Intrusion Detection,* 3223, 203-222. Springer-Verlag.

Wang, K., Cretu, G., & Stolfo, S. J. (n.d.). Anomalous payload-based worm detection and signature generation. Retrieved April 9, 2010, from Google Scholar.

Wang, K., Parekh, J. J., & Stolfo, S J. (2006). Anagram: A content anomaly detector resistant to mimicry attack. *Technical Report CUCS-020-06, Columbia University*. Retrieved April 9, 2010, from Google Scholar.

Whitman, M. E., & Mattord, H. J. (2005). *Principals of Information Security*. Boston, MA: Thompson Course Technology.

Zanero, S., & Savaresi, S. M. (2004). Unsupervised learning techniques for an intrusion detection system. *In SAC '04: Proc. 19th Annual ACM Symposium on Applied Computing*, 412–419. Retrieved March 29, 2010, from ACM digital database.

Zanero, S. (2005). Analyzing TCP traffic patterns using self organizing maps. *Proc. 13th International Conference on Image Analysis and Processing*, 83–90. Springer. Retrieved March 29, 2010, from Google Books.

Zanero, S. (2007). 360° anomaly based unsupervised intrusion detection. Retrieved May 13, 2010, from Google Scholar.

Annotated Bibliography

Amini, M., Jalili, R., & Shahriari, H.R. (2006). RT-UNNID: A practical solution to real-time network-based intrusion detection using unsupervised neural networks. *Computers & Security, 25*, 459-468. Retrieved May 10, 20010 from Elsevier digital database.

The authors explain unsupervised neural-network-based IDSs and include the main components of the RT-UNNID. Incoming packets are separated by protocol and then converted into a binary form to be input into a neural engine. This methodology employs misuse-based and anomaly-based techniques.

Axelsson, S. (2000a). Intrusion detection systems: A survey and taxonomy. Chalmers University of Technology, Goteborg, Sweden. Retrieved March 23, 2010, from Google Scholar.

Axelsson prepares the taxonomy of IDSs and their associated principles and functions. The article provides a breakdown of the anomaly intrusion detection, the difference between self-learning and programmed systems; programmed and state modeling classification. Overall, the article provided a fairly comprehensive classification guide.

Aydin, M.A., Zaim, A.H., & Ceylan, K.G. (2009). A hybrid intrusion detection system design for computer network security. *Computer and Electrical Engineering, 35*(3), 517-526. Retrieved March 29, 2010, from Elsevier database.

The article focused on hybrid intrusion detection systems. Several types of IDSs are introduced, along with packet flow and rule structure of SNORT. The anomaly IDSs includes Packet Header Anomaly Detection (PHAD) and Network Traffic Anomaly Detection (NETAD) and how the detectors are added to SNORT. Results from the experiment conclude that PHAD and NETAD are able to detect more attacks than SNORT.

Bolzoni, D., Zambon, E., Etalle, S., & Hartel, P. (2006). POSEIDON: A 2-Tier anomaly based intrusion detection system. P*roceedings of the Fourth IEEE International Workshop on Information Assurance,* Retrieved April 7, 2010, from Elsevier database.

The authors combine the advantages of SOM and PAYL to create a 2-tier anomaly-based intrusion detection system. POSEIDON is the latest IDS that support the processing and mapping of high-dimensional data and classification. The IDS can achieve high detection rate with low false positives. The IDS was benchmarked using the 1999 DARPA data set. The results show high detection rates when compared to using PAYL. While the article provides convincing evidence that POSEIDON can detect payload anomalies, there is no supporting evidence of how the architecture will work within a large network. Further research will be necessary to understand the capabilities of SOMs.

Bolzoni, D., & Etalle, S. (n.d.). Approaches in anomaly-based intrusion detection systems. *University of Twente, Enschede; Netherlands*. Retrieved March 23, 2010, from Google Scholar.

Bolzoni and Etalle examine the fundamental requirements of an anomaly-based NIDS. The NIDS can be classified according to algorithms in use, whether they analyze a single packet or whole, and the specific type of data such as header-based or payload-based. The article explains the difference between payload-based and header-based systems. In addition, it briefly explains how to set up an anomaly-based system such as POSEIDON. The article provides details of how POSEIDON works and the differences when compared to PAYL.

Bolzoni, D., & Etalle, S. (n.d.). Boosting web intrusion detection systems by inferring positive signatures. *University of Twente, Enschede; Netherlands.* Retrieved May 7, 2010, from Google Scholar.

Bolzoni and Etalle focus on detecting anomalies within HTTP traffic flow. The authors test the capabilities of their proprietary IDS known as POSEIDON. Their model is based on detecting anomalies based on regular and irregular inputs. For example, regular request are well-formatted integers, dates, or session cookies, whereas irregular requests (parameters) contain parts of a blog, email, or images. This is known as raw data and can be detected using n-gram analysis via Anagram. The authors demonstrated how POSEIDON worked remarkably well, achieving 100% rates with regular HTTP requests.

Bolzoni, D., Crispo, B., & Etalle, S. (2007, November). ATLANTIDES: An architecture for alert verification in network intrusion detection systems. *LISA '07*, 141-152. Retrieved June 11, 2010, from Google Scholar.

Bolzoni, Crispo, and Etalle build an architecture that correlates alerts based on ingress/egress methodology, a concept employed by PAYL. The primary purpose of ATLANTIDES is to reduce the number of false positives generated to ensure there is an output response to client requests. For example, the major difference between this methodology and PAYL's ingress/ingress correlation is that ATLANTIDES correlates alerts based on a user request. However, the system is not designed to handle all possible attacks (e.g., DoS attack). ATLANTIDES may integrate with POSEIDON, but there is little evidence suggesting how the two would interface.

Cheema, F.M., Akram, A., & Iqbal, Z. (2009, July). Comparative evaluation of header vs. Payload based network anomaly detectors. *Proceedings of the World Congress on Engineering, 1*, 1-5. Retrieved April 10, 2010, from Google scholar.

The article compares header versus payload based network detection and the limitations of misuse detection. The authors argue that attackers may employ malicious content via packet payload (e.g., worm). The authors compare three header-based anomaly detectors and three header plus payload anomaly detectors. The results from the experiment confirm the header plus payload detectors that gathered more information (attributes) were able to achieve a higher detection rate as opposed to the others. However, detectors that employ a combination of header plus payload may create a greater delay and are less accurate.

Chen, W., Hsu, S., & Shen, H. (2004). Application of SVM and ANN for intrusion detection. *Computer and Operations Research, 32*, 2617-2634. Retrieved May 30, 2010, from Elsevier database.

Chen, Hsu, and Shen focus on the use of neural network technology and support vector machine. The authors explain the function of SVMs. While the majority of the article is fairly technical and includes high level mathematics, it does explain how SVM interfaces with a neural Artificial Neural Network (ANN). Test results confirmed the effectiveness of SVM; it appears to function better than the ANN.

Dharmapurikar, S., Krishnamurthy, P., Sproull, T., & Lockwood, J. (n.d.). Deep packet inspection using parallel Bloom Filters. Washington University in St. Loius. Retrieved May 7, 2010, from Google Scholar.

The article focuses on the use of Bloom Filters from a different perspective than Wang and Stolfo who are the orginators. The main advantage of Bloom Filters is the ability to detect signatures or patterns without affecting network performance. Strings are clustered into groups based on similarities; it is also a method to distinguish whether there are signature matches. Consequently, network signatures can then be imported into the misuse-based system. The Bloom Filters can also perform one-way hashing, which eliminates any false negatives.

Depren, O., Topallar, M., Anarim, E., & Ciliz, K. (2005). An intelligent intrusion detection for anomaly and misuse detection in computer systems. *Expert Systems with Applications, 29*(4), 713-722. Retrieved March 28, 2010, from Elsevier database.

The authors in this paper propose a hybrid intrusion detection system which includes anomaly and misuse detection. According to Depren (2005), "the architecture uses a Self-Organizing Map (SOM) structure to model normal behavior" (abstract section). The authors create an architecture using KDD 99 dataset with hybrid modules and decision support system (DSS) that interfaces with a system administrator. The self organizing maps consist of the following sub-modules: preprocessor, TCP, UDP, and ICMP analyzer, and communication module that interfaces with a DSS (anomaly block diagram). The SOM is based on unsupervised learning. The misuse detection module uses decisions trees for supervised learning.

Dreger, H., Feldmann, A., Main, M., Paxson, V., & Sommer R. (n.d.). Dynamic application-layer protocol analysis for network intrusion detection. Retrieved April 12, 2010, from Google scholar.

The article touched basis on a number of arguments ranging from general to specific problems as they are related to intrusion detection. The authors stress the difficulty of detecting intrusions based on the specific protocol employed. In addition, they compare the difference between signature-based detection and port-based detection. Furthermore, they present a framework based that employs BRO intrusion detection system to analyze packet payloads.

Eskin, E., Arnold, A., Prerau, M., Portnoy, L., & Stolfo, S. (n.d.). A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data. Columbia University, 1-4. Retrieved March 23, 2010, from Google Scholar.

While a portion of the information presented has been reiterated in other pieces of literature, there a few key points worth examining. The authors present a geometric framework for unsupervised anomaly detection. Essentially, it is a method of detecting intrusions with unlabeled data. For example, supervised anomaly detection requires normal data, which can be fit into a model. If the training data contains intrusions there will be no way of detecting instances in future attacks as they will be considered or labeled as normal data. A solution would be to use unsupervised anomaly detection, which does not require a normal training data set.

Estevez, J. M., Garcia, P., & Diaz-Verdejo, J. E. (2004). Anomaly detection methods in wired networks: a survey and taxonomy. *Computer Communications, 27*(16), 1569-1584. Retrieved March 29, from Elsevier database.

The authors cover several topics related to anomaly detection with two major objectives to consider: detect abnormal behaviors and reduce the number of false alarms. The discussion begins with a generic explanation of an anomaly detection system and

respective components. In addition, anomaly detection methods and models (e.g., behavior and specification-based) in networks are further discussed. The authors claim there are three criteria to classify anomaly detection methods in networks, which are network feature analyzed, behavior, and analysis scale. There are also several case studies addressing the issues surrounding anomaly detection.

Estevez, J. M., Garcia, P., & Diaz-Verdejo, J. E. (2004). Measuring normality in HTTP traffic for anomaly-based intrusion detection. *Computer Networks, 45*(2), 175-193. Retrieved March 29, from Elsevier database.

The article is about measuring the normality of HTTP traffic using anomaly intrusion detection. Results from the experiment indicate that certain features (from the HTTP traffic) can be examined to discover anomalous connections or activities. The authors extract knowledge from the application layer protocol to evaluate http traffic using the Markov model. The authors argue that payload length should not (solely) be used to distinguish between normal and anomalous payloads. However, the payload length can be used with other statistics to predict attack(s). The authors also describe an architecture based on protocol-dependent segmentation using misuse and anomaly-based detectors.

Fessi, B.A., Hamdi, M., Benabdallah, S., & Boudriga, N. (2005). A decisional framework system for computer network intrusion detection. *European Journal of Operational Research, 177,* 1824-1838.

The authors developed a decisional framework for NIDS. The core arguments include the functional components and capacity of an IDS, cost incurred, and associated risk, et cetera. The authors mention the anomaly component consists of building a profile of normal behavior and then measuring the deviation from the standard. If there is a

difference than it may classified as anomalous. The major problems with current IDSs are the number of false positives generated and processing capabilities. The authors argued that current IDSs should be modeled to overcome these limitations or problems. The article also describes the efficiency of IDS, which can be measured using conditional probabilities known as true positive rate, false positive rate, false negative rate, and true negative rate.

Garcia-Teodoro, P., Diaz-Verdejo., J., Macia-Fernandez, G., & Vazquez, E. (2008). Anomaly-based network intrusion detection: techniques, systems and challenges. *Computer and Security*, *28*, (1-2), 18-28. Retrieved March 29, 2010, from Elsevier database.

The article is primarily focused on anomaly detection, but covers a few aspects concerning the differences between signature and anomaly detection. According to Garcia-Teodoro et al., a generic NIDS consist of parameterization, training stage, and detection stage. The detection techniques can be divided into three broad categories based on the behavioral mode: statistical, knowledge, and machine learning. The article also includes the subtypes related to each category and associated platforms. Finally, the authors address the most significant challenges in the area of intrusion detection: low detection efficiency, low throughput, and absence of appropriate metrics, to name a few.

Girardin, L. (1999). An eye on network intruder-administrator shootouts. *Proceedings of the Workshop on Intrusion Detection and Network Monitoring*, Santa Clara, CA, USA, April 9-12, 1999. Retrieved May 22, 2010 from Google Scholar.

Girardin (1999) argued that a visual representation of network activity using SOMs provides “new ways to explore, track, and analyze intruders” (p.3). For example, maps

represent profiles of network activity in a two-dimensional grid. Each unit is referred to as a neuron. The SOM algorithm is initiated using weights that are assigned to each neuron. The distance between the input and weight is calculated to determine the closest value. The process continues until a map of the topology represents the input space. While the SOM may function in an unsupervised mode, there are still manual configurations that must be applied (e.g., number of iterations, size of the neighborhood, and units).

Gu, G., Fogla, P., Dagon, D., Wenke, L., & Skoric, B. (2006, March). Measuring intrusion detection capability: An information-theoretic approach. *ASIACCS*. Retrieved May 4, 2010, from ACM digital database.

The authors argue most methods of evaluating an IDS are inconclusive or do not provide enough evidence to justify whether the method of evaluation is truly accurate. They propose a new metric called Intrusion Detection Capability (C_{id}). The new metric considers: TP rate, false positive rate, negative predictive value, and base rate. The authors argue the C_{id} metric could outperform existing metrics. In addition, it can provide better analysis than the ROC graph commonly used to evaluate IDS approaches. The Intrusion Detection Capability can provide better results with PAYL, which relies on a threshold to determine the optimal point of performance. Therefore, the C_{id} metric could affect the detection capability.

Hsu, F., Guo, F., & Chiueh, T. (2006, December). Scalable network-based buffer overflow attack detection. *ANCS06'*, 1-7. Retrieved April 26, 2010, from ACM digital database.

Hsu, Guo, and Chiueh argued that vast amount of vulnerabilities (according to CERT 50%) are caused by buffer overflow attacks. The authors examine various anomaly

detection techniques and their capability in detecting payload attacks (e.g., buffer overflows). There is uncertainty about whether PAYL is able to detect anomaly payload attacks when traffic payload changes.

Hwang, K., Liu, H., & Chen, Y. (2004). Cooperative anomaly and intrusion detection alert correlation in networked computing systems. *IEEE Transaction on Dependable and Secure Computing*. Retrieved March 20, 2010, from IEEE digital database.

The article focuses on anomaly-based intrusion detection, but from a unique perspective. A comparison between signature-based and anomaly-based are examined to highlight and expose their differences. While the paper is primarily based on using a technique that would vaguely be used with anomaly-based systems, there is significance in how the techniques employed are able to scan data for anomalous events.

Ingham, K.L., & Inoue, H. (n.d.). Comparing anomaly detection techniques for HTTP. Retrieved April 26, 2010, from Google Scholar.

The article examines anomaly detection techniques to include character distribution for HTTP traffic. The authors study Wang and Stolfo's methodology of detecting anomalous events, based on the Mahalanobis distance. PAYL uses packet length which is correlated to character frequencies. The authors claim that attackers can easily manipulate packet size. This is a significant disadvantage when using character length to detect anomalies. The article includes several tests, which include n-gram testing. Tests showed that 6-gram testing presents optimal results (e.g., less false positives).

Kayacik, G.H., Zincir-Heywood, N.A., & Heywoord, M.I., (2007). A hierarchical SOM-based intrusion detection system. Artificial Intelligence, 20, 439-451. Retrieved May 10, 20010 from Elsevier digital database.

An intrusion detection system can be represented using a hierarchy of SOMs. The authors test the IDS using KDD benchmark dataset. The goal of the test was to demonstrate the extent in which the SOMs detect attacks. The authors answer to the major concerns such as the ability to partition data and features of the data set that are most significant. Performance testing concluded that a two-layer SOM hierarchy works best. The document includes test data results with various percentages. The results from the experiment indicate that detectors should be built with protocol and service.

Kiani, M., Clark, A., & Mohay, G. (2008). Evaluation of anomaly based character distribution models in the detection of SQL injection attacks. *The 3rd Int'l Conf. on Availability, Reliability, and Security*, 47-55. Retrieved March 20, 2010, from IEEE digital database.

Kiani, Clark, and Mohay explore anomaly-based systems and the difficulties of evaluating IDSs. The authors argued the results from the 1999 DARPA test do not contain enough variety of attacks to thoroughly evaluate the detection capabilities of PAYL, among other systems. The test results did not indicate whether PAYL can detect SQL attacks.

Krugel, C., Toth, T., & Kirda, E. (2002). Service specific anomaly detection for network intrusion detection systems. *SAC '02*. Retrieved March 22, 2010 from ACM digital database.

Krugel, Toth, and Kirda explain the advantages of an anomaly detection system capable of analyzing data based on service specific analysis. The model aims at extending the capabilities of traditional IDSs that monitor header information to monitoring payload data. Their methodology includes a packet processing unit and statistical processing unit. The article also briefly mentions the use of anomaly scores, which calculated using mean and standard deviation.

Lakov, P., Dussel, P., Schafer, C., & Rieck, K. (n.d.). Learning intrusion detection: supervised or unsupervised? Retrieved March 22, 2010 from Google Scholar.

The aim of the article was to investigate the differences between supervised and unsupervised techniques and their association with intrusion detection. The authors argued over the decision to use or associate a technique and apply it to the IDS should include an understanding of how to label information. While the article is rather brief it does provide a breakdown and separation of supervised and unsupervised algorithms. The article does not provide a thorough explanation of the algorithms and their capabilities

Labib, K., & Vemuri, R. (n.d.). NSOM: A real-time network-based intrusion detection system using self-organizing maps. Retrieved May 10, 20010 from Google Scholar.

Labib and Vemuri explain the basic concepts related to using SOMs. The technique is efficient because it can classify packets rapidly. It is well-suited for high speed network and provides rapid conversion. However, there are user specifications when using SOMs. For example, the window size may dictate the level or degree of granularity during the analysis phase. There may be risk of losing important information if the window size is too small. On the other hand, if the window is too large then the data becomes too sparse to analyze. These are important factors to consider with intrusion detection. The article briefly describes the SOM structure.

Lee, W., Stolfo, S. J., & Mok, K. (2000, December). Adaptive intrusion detection: A data mining Approach. *Artificial Intelligence Review, 14*(6), 533-567. Retrieved March 22, 2010 from Google Scholar.

Lee and Stolfo examine how data mining can be applied to intrusion detection. Audit data is collected and mined for frequent activity patterns. The patterns are used collaboratively with temporal and statistical features. The authors propose the use of association rules and frequent episodes based on the training data collected. In addition, the authors suggest using meta-learning as a framework for intrusion modeling. The authors support their theories of *Adaptive Intrusion Detection* using real-world audit data.

Lee, W., Stolfo, S.J., Chan, P.K., Eskin, E., Fan, E., W., & Miller, M. et al. (2003). Real time data mining-based intrusion detection. Retrieved March 28, 2010 from Google Scholar.

The authors argue that deployable IDSs must not rely solely on data mining techniques, rather on other factors such as accuracy, efficiency, and usability. Evidence from raw data are known as *features*—used for building IDS models. These features can be extracted, where data mining programs can be applied to compute frequent patterns among data samples. Examples include adaptive learning, unsupervised anomaly detections and combined hybrid intrusion detection. The article briefly addresses the architecture of IDS that employ data mining.

Lichodzijewski, P., Heywood, A.N., & Heywood, M.I. (2002a). Dynamic intrusion detection using self-organizing maps. *The 14th Annual Canadian IT Security Symposium.* Retrieved March 28, 2010 from Google scholar.

The article is mainly an introduction to using SOMs. It explains the methodology of capturing, processing, and preprocessing data. The SOM architecture consists of the first level where data preprocessing and training occurs. The second level SOM consists of clustering data. The article is very brief, but provides preliminary information about using SOMs.

Lichodzijewski, P., Zincir-Heywood, A.N., & Heywood, M.I. (2002b). Host-based intrusion detection using self-organizing maps. *Proceedings of the 2002 IEEE World Congress on Computational Intelligence*. Retrieved May 11, 2010 from Google scholar.

The article explains the methodology of using SOMs in a dynamic environment. The methodology includes data collection, data reduction, data preprocessing, and pattern discovery. The SOM architecture consists of several hierarchical layers. The test results using the DARPA dataset fared well based on the number of patterns. However, it important to note that only 10% of the data was included in the test. For example, 100 patterns yielded a .0020 false positive rate.

Liu, Z., Florez, G., & Bridges, S.M. (n.d.). A comparison of input representations in neural networks: A case study in intrusion detection. Retrieved May 11, 2010 from Google scholar.

Liu et al., conduct a case study on the comparison of input representations in neural networks. A SOM is characterized by neurons that are indicative of the input pattern. The authors conduct an experiment at the University of New Mexico. Training begins by exposing the network to normal and anomalous data. The authors conducted their test using randomly generated anomalies. The article also touched basis on the difference between binary and decimal encoding techniques. The former produces a lower false positive rate than the latter.

Maselli, G., Deri, L., & Suin, S. (n.d.). Design and implementation of an anomaly detection system: An empirical approach. Retrieved March 22, 2010, from Google Scholar.

The empirical study provided by Maselli, Deri, and Suin cover the fundamentals of implementing anomaly detectors on a campus network. The article includes a hybrid approach. Various topics include network traffic monitoring (e.g., static and dynamic), IP

protocol analysis, and implementing network anomaly detectors. While the document provides helpful insight to implementing and IDS, it is poorly structured and lacks quantitative data to support the theory expressed in the abstract.

Parekh, J.J., Wang, K., & Stolfo, S.J. (2006, September). Privacy-preserving payload-based correlation for accurate malicious traffic detection. *SIGCOMM '06 Workshops*. Retrieved May 6, 2010, from ACM digital database.

The article defined the fundamental concepts related to payload anomaly detection with emphasis upon correlation techniques. The main payload anomaly detection methodologies are PAYL and Anagram. The alert correlation include baseline (string equality, longest common substring, and longest common subsequence), frequency alert correlation (frequency distribution, z-string), and binary modeled n-gram correlation (n-gram signature and Bloom Filter n-gram signature). The authors test the capabilities of each correlation technique and present their findings. N-gram modeling offers less privacy than the other models.

Perdisci, R., Ariu, D., Fogla, P., Giacinto, G., & Lee, W. (2009). McPAD: A multiple classifier system for accurate payload-based anomaly detection. *Computer Networks, 53*,864-881. Retrieved May 26, 2010, from Elsevier digital database.

Perdisci et al., compare payload-based anomaly detection systems. While PAYL and Anagram prove to be fairly successful in detecting anomalous attacks, they also produce a relatively high number of false positive (PAYL) and lack of performance (Anagram). PAYL has a high false positive rate because of the algorithms used with frequency distribution. Anagram also has a significant drawback with its use of Bloom Filters and possible difficulty of detecting attacks at high speeds.

Perona, I., Gurrutxaga, I., Arbelaitz, O., Martin, J.I., Muguerza, J., & Perez, J. (2008). Service-independent payload analysis to improve intrusion detection in network traffic. Australian Computer Society. Retrieved May, 18, 2010, from Google Scholar.

Perona et al., argued that payload analysis should fulfill three basic requirements: "to be automatic, to be general, and to be computationally efficient" (2008, p. 4). In addition, they argued that data will require labeling as means to evaluate an IDS. However, the main argument is that payload analysis should be service independent to avoid further complications. The solution is investing research into IDSs that employ unsupervised learning. The authors recommend n-gram analysis, along with considering byte-frequency approach (e.g., similar to PAYL).

Pietraszek, T., & Tanner, A. (2005). Data mining and machine learning—Towards reducing false positives in intrusion detection. *Information and Security Report, 10*(3), 1-18. Retrieved May 12, 2010, from Elsevier digital database.

This article presents a global perspective upon alert management, specifically how an IDS would be set in the environment to detect and classify alerts. The authors indentify and explain the different techniques used with classifying alerts. For example, machine learning approaches include RIPPER and SVM. Pietraszek and Tanner argued that conventional ways of examining alerts is primarily outdated due to limitations such as the need for expert knowledge and lack of automation.

Powers, S.T., & He, J. (2008). A hybrid artificial immune system and self-organizing map for intrusion detection. *Information Sciences*, 178, 3024-3042. Retrieved May 10, 2010, from Elsevier digital database.

Powers and He examine the use of an artificial immune system and SOM due to the limitations of current misuse IDS. The process of creating a SOM is based on connection vectors. It contains characteristics of a connection (e.g., destination port or packets sent). The neuron that is closest to the vector is the one most excited, determined the winner. The winning neuron responds to future events and makes adjustments so that it enhances the response. This is a form of unsupervised learning because the attacks are not previously identified. Attacks can be labeled and grouped together using SOMs.

Rajan, S.S., & Cherukuri, V.K. (n.d.). An overview of intrusion detection systems. Retrieved May, 22, 2010, from Google Scholar.

Rajan and Cherukuri briefly describe the different types of IDSs and provide a brief introduction to the functionalities and metrics used with intrusion detection. The authors provide graphical representations of payload modeling, specifically with PAYL. The three phases of PAYL include training, incremental, detection, and clustering. Intrusion detection will play a crucial role in network security.

Ramadas, M., Ostermann, S., & Tjaden, B. (2003). Detecting anomalous network traffic with self-organizing maps. In G. Vigna, E.Jonsson, and C. Kruegal (Eds.): RAID, 2820, 36-54. Retrieved April 27, 2010, from Google scholar.

Ramadas et al., describe their version of SOM intrusion detection system. Each data point is represented by a neuron, also defined as multidimensional vector. In the learning phase the SOM is modeled based on competitive and cooperative characteristics. The former is the neuron that best responds, defined as the winner. The latter is used to distinguish neighborhood. The authors also express how a SOM should be set up for training. In addition, a SOM model representing web based HTTP traffic was evaluated.

Rhodes, B.C., Mahaffey, J.A., & Cannady, J.D. (2000). Multiple self-organizing maps for intrusion detection. *Proceedings of the 23rd National Information Systems Security Conference*. Retrieved May 10, 20010 from Google Scholar.

The central focus in this article is Kohonen's SOM approach, which automatically detects anomalous attacks using visual maps. SOMs can organize input vectors and arrange them according to their similarities. The authors point out that a Kononen's map has certain limitations with disparate information. The design features are also explained. The experiment using SOMs included C language using Linux machine and how the methodology was used to successfully detect buffer attacks. However, the article did not provide any statistical data that demonstrates SOM performance.

Smith, C., Matrawy, A., Chow, S., & Abdelaziz, B. (2008). Computer worms: Architectures, evasion strategies, and detection mechanisms. *Journal of Information Assurance and Security, 4*, 69-83. Retrieved April 13, 2010, from Google scholar.

The focus of the article is mainly upon detecting worms using payload anomaly detection techniques. The authors describe different attacks patterns and evasion strategies. They argued that PAYL can avoid mimicry attacks, but with limited details. The basic function of PAYL and Anagram are described in detail. A table is included describing the differences between IDSs, honeypots, and firewalls, et cetera.

Stolfo, S., Parekhi, J., & Locasto, M. (2007). Developing collaborative profiles of attackers: A longitudinal study. *University of Columbia Final Progress Report.* Retrieved May 28, 2010, from Google scholar.

The authors conduct a longitudinal study using PAYL and Anagram on a live network. In addition, the study included the use of alert correlation techniques such as string

equality, longest common substring, longest common subsequence, and frequency distribution, Z-string, n-gram signature, and Bloom Filter n-gram signature. Results from experiment showed which correlation technique is stronger. The correlation techniques were able to find Code Red II worms that were segmented across the network. It is possible to generate automatic signatures. Overall, the Bloom Filters provide enhanced capabilities (e.g., compactness, resiliency, and security) over 1-gram and raw distribution analysis.

Thorat, S. A., Khandelwal, A. K., Bruhadeshwar, B., & Kishmore, K. (n.d.). Payload content based anomaly detection. Retrieved April 8, 2010, from Google scholar.

The authors argue that payload content should be examined and included in anomaly detection. The article explains how many systems ignore the payload content and concentrate significantly on solely header information. While there are other systems that do concentrate on the payload many have inherent limitations. Thorat et al., focus on *Content Based Payload Partitioning* (CPP), which is based on their version of a *Payload Content Based Network Anomaly Detection* (PCNAD). The PCNAD system was tested using DARPA IDS data set and proved successful in detecting attacks without considering the entire payload. In addition, it uses CPP to avoid mimicry attacks.

Verwoerd, T., & Hunt, R. (2002). Security architecture testing using IDS—a case study. *Computer Communications, 25* (15), 1288-1294. Retrieved March 29, 2010, from Elsevier data base.

Verwoerd and Hunt discuss the shortcomings of intrusion detection systems. For example, the authors argued the race between attacker capabilities and defense capabilities may result in the latter taking over or surpassing the former. The major

shortcomings identified are typically related to vulnerabilities found in access control, cryptography, IDSs, and firewalls. A case study suggests how to mitigate attacks that affect HTTP traffic. The article also focuses on how a firewall is able to work in collaboration with the IDS and vice versa.

Vesanto, J., Himberg, J., Alhoniemi, E., & Parhankangas, J. (1999, November). Self-organizing map in Matlab: the SOM tool box. *Proceedings of the Matlab DSP Conference*, 16-17, 35-40. Retrieved April 8, 2010, from Google scholar.

Vesanto et al., examine the use of SOMs to visualize data points plotted a 2-dimensional map. It is a topological representation of data samples. The SOM toolbox is software package that researchers use (in a lab environment) to test and demonstrate the capabilities of SOM. The performance test was used to calculate the algorithms based on data size and input dimensions. The article's main purpose is to demonstrate the use of the SOM toolbox.

Vliet, V. (n.d.). Turnover POSEIDON: Incremental learning in clustering methods for anomaly based intrusion detection. Retrieved May 10, 2010, from Google Scholar.

Vliet explained the basic function of POSEIDON with the integration of SOM and PAYL classification techniques. SOM learns and clusters similar data together. PAYL is trained to cluster the SOM models and associated characteristics. Because the original POSEIDON is unable to adapt to network changes, the new improved model can adapt to changes in a real-life network.

Wang, K., & Stolfo, S. J. (n.d.). Anomalous payload-based network intrusion detection. *In Jonsson, E., Valdes, A., Almgren, M., eds.: Proceedings 7th Symposium on Recent Advances in Intrusion Detection, 3223*, 203-222. Springer-Verlag.

Wang and Stolfo tested the capabilities of PAYL, which is primarily based on the 1-gram frequency distribution model (basic model). An n-gram "is the sequence of n adjacent bytes in a payload unit" (Wang & Stolfo, n.d., p. 4). PAYL is able to detect anomalous payload attacks, but it appears the article is mainly focused on worm detection. The model size is reduced using clustering. The article introduces unsupervised learning via Z-string correlation technique, which can prevent worm propagation and worms that morph themselves. According to the authors PAYL uses small amount of memory; however, the model has several drawbacks such as low false positive rate and inability to avoid mimicry attacks. The article provides statistical information and performance indicators of PAYL.

Wang, K., Cretu, G., & Stolfo, S.J. (n.d.). Anomalous payload-based worm detection and signature generation. Retrieved April 9, 2010, from Google Scholar.

Wang, Cretu, and Stolfo tested PAYL, along with ingress/egress traffic correlation techniques. Experimental evidence showed how PAYL and related intrusion detectors can detect new worms. The data collected showed how SE, LCS, and LCSeq can be used to detect worms and to create signatures. There is evidence that even segmented worms among different sites can be correlated. The cross-site collaboration can help detect zero day worms. The article provides substantial evidence that PAYL can be used for worm detection.

Wang, K., Parekh, J.J., & Stolfo, S J. (2006). Anagram: A content anomaly detector resistant to mimicry attack. *Technical Report CUCS-020-06, Columbia University*. Retrieved April 9, 2010, from Google Scholar.

Anagram is a higher order anomaly detection technique that employs Bloom Filters (an array of *m* bits) to conserve space. The model can generate signatures to correlate segmented worm attacks. A significant benefit of Anagram is the ability to avoid mimicry attacks, which is the ability to mimic the characteristics of a distribution byte with padded (or dummy) bits. In addition, it can exceed PAYL with detection and false positive rates. Further, Anagram is easier to train. However, the technique has a significant drawback in that it does not preserve the privacy of information.

Zanero, S. & Serazzi, S.M. (2004). Unsupervised learning techniques for an intrusion detection system. *In SAC '04: Proc. 19th Annual ACM Symposium on Applied Computing*, 412–419. Retrieved March 29, 2010, from ACM digital database.

The authors introduced a two-tier IDS that compensates for the limitations of traditional data mining techniques. Zanero and Serazzi examine the use of unsupervised anomaly detection, which can analyze data within the TCP/IP data flow. It is unrealistic to apply an algorithm against raw data as it would consume massive amounts of resources and would add significant complexity. A solution would be to apply clustering technique to place similar characteristics of objects (data packets) into specified groups. The authors argue that these groups can be created using SOMs.

Zanero, S. (2005). Analyzing TCP traffic patterns using self organizing maps. *Proc. 13th International Conference on Image Analysis and Processing*, 83–90. Springer.

Zanero constructs a SOM model using two-tier architecture. SOM can be used to retain payload information and compress the information into a byte, which are then clustered into groups. Experiments showed that SOM can learn specific patterns and detect anomalous traffic. The first tier retains data characteristics while the second is used to

detect anomalies. Results from the experiment showed a very low false positive rate using a modified SOM algorithm. However, the false positive rate can increase significantly with the number of attributes selected.

Zanero, S. (2007). 360 anomaly-based unsupervised intrusion detection. Retrieved May 13, 2010, from Google Scholar.

The research article examines the use of unsupervised learning for anomaly detection. The SOM architecture compresses data characteristics into a byte of information. What is considered a subset of the information gathered is sent to the second tier algorithm. According to the author, PAYL is the only detection methodology that uses part of the payload for analyzing possible attacks. The SOM architecture used by Zanero can achieve a 58.7% detection rate with .03% false positive rate.

Glossary

ACE	Attack Classification Engine
AIE	Alert Information Extractor
ATLANTIDES	Architecture for Alert Verification in Network Intrusion Detection Systems
ATM	Asynchronous Transfer Mode
BFD	Byte Frequency Distribution
CIA	Confidentiality, Integrity, Availability
DDoS	Distributed Denial of Service
DoS	Denial of Service
FD	Frequency Distribution
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
IDS	Intrusion Detection System
IP	Internet Protocol
IS	Information System
KDD	Knowledge Discovery and Data Mining
LAN	Local Area Network
LCS	Longest Common Substrings
LCSeq	Longest Common Subsequence
NIDS	Network Intrusion Detection System
PAYL	Anomalous Payload-based Network Intrusion Detection
POSEIDON	PAYL over Self-Organizing Maps for Intrusion Detection
R2L	Remote to Local

RIPPER	Repeated Incremental Pruning to Produce Error Reduction
ROC	Receiver Operating Characteristic
SE	String Equality
SOM	Self Organizing Map
SONET	Synchronous Optical Networking
SQL	Standard Query Language
SVM	Support Vector Machine
TCP/IP	Transmission Control Protocol/Internet Protocol
U2R	User to Root
WAN	Wide Area Network