Proceedings of the 2007 IEEE Symposium on Computational
Intelligence in Security and Defense Applications (CISDA 2007)

Anomaly Detection for Application Level Network Attacks Using Payload Keywords

Like Zhang, Gregory B. White
Department of Computer Science
University of Texas at San Antonio
San Antonio, Texas 78249 USA

Abstract*-Network anomaly intrusion detection is designed to provide in-depth defense against zero-day attacks. However, attacks often occur at the application level, which means they are payload associated. Since traditional anomaly detection works by monitoring packet headers it provides little support for defending against such activities. In this paper, we will explore how the packet payload can be used for identifying application level attacks. First we will discuss the current status of network anomaly detection, and emphasize the importance of payload based detection research using existing problems. Then we provide a brief introduction to several related approaches on this topic. Based on the discussion, an efficient method to detect payload related attacks will then be proposed. The method is divided into a training phase and a detection phase. In the training phase, we will perform Principal Component Analysis (PCA) on several important packet fields to reduce the data dimension, and then construct the most appropriate profile based on the PCA results. In the detection phase, an anomaly score will be assigned to each incoming packet based on the profile. We then present the experiment based on the DARPA'99 dataset with details to explain our approach. Comparison with other similar mechanisms demonstrates the advantage of the proposed method at identifying payload related attacks.

1. INTRODUCTION

Network intrusion detection systems (NIDS) can be categorized as either signature based or anomaly based. The signature based approach, which is usually based on signatures of already known attacks or vulnerabilities, is widely deployed in various industry security products. While anomaly detection, which attempts to identify attacks based on profiles of normal network activities, is still in many respects in the research stage. In other words, signature based approaches match the packet with specific identifiers, while anomaly based approaches exclude all packets not fitting with the current activity profile. The popularity of signature-based IDS is due to the fact that it works well once the specific pattern of attacks can be identified. In addition, it is comparatively much easier to implement. However, the significant disadvantage is that it cannot identify new attacks, and performs poorly when faced with mutations of known attacks.

Anomaly detection, which experienced early attention in IDS research, has recently caught people's attention again. In theory, anomaly based detection should be able to identify any attack, including "zero-day" attacks, which includes unknown or novel malicious events, as mentioned in [1]. The signature-based approach can do nothing about this since no signature or fingerprint is known at the time when a new attack is released. Usually, people need to have some time to identify an attack or virus after it is found for the first time in order to be able to add the signature to the database. During this window of time, machines can be compromised. Since anomaly detection holds the promise of working in such conditions, the ability to defend against unknown attacks is demanding increased research in this area.

Various approaches have been proposed. Earlier research using data mining for network anomaly detection is described in [2] [3]. This approach used RIPPER rule learning algorithm to construct the profile of normal network conditions. In [4] [5] [6], an efficient anomaly score scheme was proposed to detect new network activities based on either packet fields or bytes. Other approaches applied different machine learning algorithms to network packet header fields and tried to detect abnormal instances based on the constructed models. A detailed comparison of the anomaly detection results from some popular adopted machine learning algorithms is provided in [7].

Athough anomaly detection has great potential and seems to have a more promising future, it is extremely difficult to achieve the goal. Since network traffic is extremely complicated, and new applications are emerging everyday, researchers have not been able to find a reliable method to construct a model of "normality". At this moment, how to efficiently identify new attacks is still unclear. The dilemma between detection rate and false alarms is the major problem for researchers. According to [8], which is based on experiments in 1999, the best tested system was able to detect only half of the attacks. In the comparison carried out by University of Minnesota in 2003 [7], most of the mechanisms could detect only about half or less than half of the attacks at the false positive rate of 0.02%. To increase the detection rate higher, for example 80%, the false positive rate will increase dramatically to around 1%, which means thousands of false alarms per day.

When we examined these experiments, we found that most of the failures happen to the application level attacks. For example, most of the anomaly detection mechanisms could easily detect network level attacks such as arp poison, SYN flood, teardrop, and others. For such attacks, the detection rate could even reach 100% with a very low false alarm rate. However, the problem happens when application level attacks are involved. U2R (User to Root) and R2L (Remote to Local) attacks are very popular in the DARPA'99 dataset, and most of the tested IDS systems performed very poorly in identifying such attacks (less than half or none at all), as described in [7].

The reason behind this is actually fairly simple. All of the anomaly detection schemes only consider the packet header fields, such as the ip address, port number, flags, etc., so they work well when the attack involves only the related fields. However, once the payload is involved, these methods have no way to identify them. For example, a popular overflow attack is to send some fields with extremely long arguments (e.g. ps and sendmail). Since the header fields are still valid, these header-based NIDS will consider the packets normal.

Unfortunately, most of today's attacks target the vulnerabilities of specific systems or applications as mentioned in [9], or happen on the application level with multiple steps, which was described in [8] for the DARPA'99 experiment. From the point of view of the network layer, these attacks contain no malicious activity, and they don't always generate abnormal network traffic. The only way to defend against them is to explore and analyze the packet payload. In the signature based NIDS, this is done by finding fingerprints of a specific attack, and performing pattern matching on the incoming packet payload. These "fingerprints", however, can only be developed manually or semi-automatically, and are only identifiers for previously known attacks.

The method proposed in this paper focuses on performing anomaly detection on the application level, or payload-related attacks. We also study how to extract the most useful information from the packet to obtain better results. The method is based on the in-depth study of related recent research, which will be described in the next section. And it is developed based on the assumption of being independent of any specific application protocol. Generally, the payload of the packet will be analyzed and a "keyword" will be extracted, as well as its corresponding value. This data will be used with other information from the header fields for network profile construction. In the detection mode, any abnormal packet, which does not conform to the profile, will be assigned an anomaly score. Once the anomaly score reaches the threshold, an alarm will be generated. Details of the implementation will be introduced in section 3.

The rest of the paper is organized as follows. Section 2 is a brief review of recent related research. Since our proposed method learned a lot from them, we will introduce some of the important concepts to help better explain our approach. The proposed method is introduced in section 3. In section 4, experiment results will be discussed with details, and we will demonstrate how the proposed method offers great advantages to detect application level attacks.

2. RELATED WORKS

Payload-based intrusion detection has really only caught people's attention in recent years, and most NIDS research still focuses on packet headers only. The following research provides useful background information for better understanding our proposed method.

2.1 HTTP anomaly detection

In [10][11][12], researchers at the University of Santa Barbara proposed a method to detect web-based attacks. More specifically, it was aimed at the HTTP protocol on the application level. It was different from other IDS techniques which identify attacks based on different packet fields such as source IP, destination IP, destination port, etc., this method extracts important fields from the HTTP request, and constructs a statistical model based on these fields. In their earlier research in [9], three properties were used: the request type, the request length, and the payload distribution. Later, more properties were added such as query parameters, and query length. This method claims to have 0.06% or less false positives when testing on Google and campus networks. However, since it only works for HTTP attacks, it cannot be directly compared to other methods which use the DARPA'99 dataset as the benchmark.

2.2 ALAD

ALAD (Application Level Anomaly Detection) is proposed in [13]. This is the first attempt to use "keywords" in the payload for anomaly detection, and it is used with other header fields to identify attacks. For any packet, the first word of each line will be extracted as a keyword. A packet could thus have multiple keywords. Several pairs of attributes are then created for modeling. Besides pairs like "source ip | destination ip" or "destination ip | destination port", the keyword is used in the pair "keyword | destination port". For each pair, a statistical profile is constructed in the training phase. In the detection phase, packets containing new fields or keywords will get an anomaly score. The anomaly score is based on the assumption that if an experiment is performed n times with r different results, then the probability of the next new result is r/n. Once the anomaly score reaches the threshold, an alarm is generated.
The idea of using keywords is reasonable, and it was able to detect some attacks which a network layer IDS is incapable of. However, the mechanism shows a lot of problems in experiments, which will be addressed in section 4.

2.3 PAYL (Payload-based Anomaly Detection)

Columbia University was the first to apply data mining technique in NIDS, and they have done intense related research. In [14], they proposed an anomaly detection approach based on payload byte distribution. The profile of byte frequency distribution and standard deviation of the payload were built during the training phase. Then in the detecting phase, the Mahalanobis distance was used to measure the difference between the incoming data and the profile. This method works fairly well at identifying new application level attacks, such as detecting malicious executable files or Internet worms [15], however, the problem of false alarms rate still exists. In addition, the overall detection rate does not increase much when testing with DARPA'99 dataset under the low false positive rate condition. The researcher claims it could be improved by cooperating with a signature-based approach, but this would only apply to known attacks.

The above projects have their commonness and uniqueness. They all pay great attention to the packet payload, and attempt to gather as much information as possible for detection, but each one uses a different approach. The first one focuses only on HTTP traffic, and uses a lot of HTTP related data. It is not protocol-independent, and cannot be applied to other applications. The second one, ALAD, uses a more reasonable "keyword" approach, but the implementing mechanism performs poorly in experiments. PAYL uses payload byte distribution, and it is able to detect some novel application attacks such as worms, but performing detection only on byte distribution is obviously not robust enough, so it does not sufficient as a standalone method.

3. AN EFFICIENT APPROACH

The proposed method uses a keyword approach, as does ALAD, but does so in a different manner. First, the ALAD extracts the first word of each line in the payload as the keyword, while our method only processes the first line to extract the first word and associated parameters. Our experiments show this is more useful than processing all lines. Second, ALAD matches the keyword with the destination port number to identify potential attacks, but it actually does not help much since most attacks will not violate this property. For example, web attacks could always happen on port 80, and the keywords (first word in each line) are always the same (e.g. GET, POST, ACCEPT, etc.). Our method extracts extra information after the keyword, which is usually the value of the keyword, and we also consider the overall payload properties. This information from the payload itself proves very important in detecting application level attacks, and provides much higher accuracy for detection. In addition, ALAD, and most other approaches, arbitrarily select some packet fields for profile developing, while our approach performs Principal Component Analysis (PCA) first to reduce the data dimension in order to find the fields with the most variance.

The method is divided into 2 phases, as shown in figures 1 and 2.

3.1 Training Phase

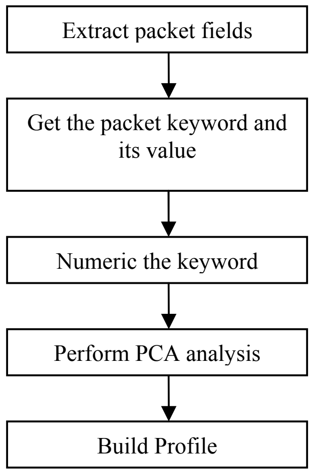


Figure 1 Training Phase

The purpose of training phase is to construct the profile for the normal network traffic, so it should be fed with clean data without any attack. The profile contains all of the acceptable values for different fields in packets. However, isolated field do not help much in detecting malicious activities. For example, suppose we know the acceptable destination IP range is 172.016.x.x, and the port is from 0~1024. If we do not associate the information, they will not help much. An attack could happen on the IP 172.016.1.1 at port 21 (FTP), but FTP might not be an open service on this IP. We need to find some way to construct the profile based on relationships among various packet fields, rather than on standalone ones.

A simple but effective solution is to create pairs among the fields. For example, we can build a profile for "destination IP : destination port", so we know which port is allowed on a specific IP. And we can also use multiple fields such as "destination IP: destination port: payload length". Then we come to an important question: which fields should be selected?

PCA is a popular technique in image processing, patter recognition and data analysis. It is used for data dimension reduction and multivariate analysis. Simply stated, it could simplify a dataset by using linear transformation to transform the original data set into a new coordinate system. The greatest variance of the original data exists on the first coordinate in the new system, the second greatest variance is on the second coordinate, and so on. This transform is done by finding out the eigenvectors. A detailed discussion can be found in [16].

Since the PCA technique can be used to reduce the data dimension and find out the most variance by coordinate, it could be used in NIDS to find out which field makes the greatest contribution in the data variance. Such methods have been studied in [17][18][19], here we also use a similar approach. The following discussion is based on the DARPA'99 dataset, and we use the first day of week 3 (clean data without attacks) for PCA analysis

.

Step 1: Extract packet fields

First we filter out the TCP packets, because all payload related attacks in DARPA'99 are TCP traffic. We can use all the fields for later processing, but it is not necessary. For example, the TCP flags can be used to identify some attacks, but it is not a critical field to differentiate network packets. The purpose in extracting the packet fields is to find out the most significant variance by PCA so the fields must be those which vary a lot in the traffic. With this requirement, 9 properties are selected for extraction: Header Length, IP Version, Packet Length, Source IP, Destination IP, Source Port, Destination Port, Payload Size, and Payload. This provides the following data shown in table 1:

Header Length	20
IP Version	4
Packet length	65
Src. IP (inet_addr)	2655724996
Dest. IP(inet_addr)	1769017516
Source Port:	1024
Dest. Port	25
Payload Length	25
Payload	EHLO jupiter.cherry.org\r\n

Table 1 Sample Data: 9 fields extracted for processing

Step 2: Packet Keyword and the value

After extracting the fields, we need to process the payload to obtain the keyword and corresponding value. The keyword is defined as the first word in the first line, and the value is all the data at the rest of the line. For the example in table 1, the keyword is "EHLO", and its value is "jupiter.cherry.org". The payload of a packet could contain multiple lines, but usually the first line is the most important and defines the packet action so only the first line is used in our algorithm.

Step 3: Number the keywords

PCA is used to obtain the eigenvectors of a matrix, so it cannot work with characters or strings. In order to associate the keyword with the other fields, we need to map the keywords into numbers. In this case, we simply save each unique keyword into an array so we can use the sequence number instead of the keyword itself, as in table 2:

ID	Keyword
0	Unknown (non ASCII string)
1	GET
2	EHLO
…	

Table 2 Keyword mapping table

After the mapping, we obtain a matrix as in table 3

20	4	40	2655724996	1769017516	1024	25	0	0
20	4	126	1769017516	2655724996	25	1024	86	3
20	4	65	2655724996	1769017516	1024	25	25	2
…								

Table 3 Packet Matrix

Step 4: PCA Analysis

The following is the result after applying PCA on the matrix in table 4:

-0.00	-0.00	0.00	0.00	0.00	-0.00	-0.00	0.00	-0.97	Header_Len
-0.00	0.00	-0.00	-0.00	0.00	0.00	0.00	-0.00	-0.20	IP_Version
-0.00	0.00	-0.02	-0.00	-0.70	-0.10	-0.44	-0.53	0.00	Packet_Len
0.68	0.72	-0.00	0.00	0.00	0.00	0.00	-0.00	0.00	Src_IP
0.72	-0.68	-0.00	-0.00	-0.00	-0.00	0.00	-0.00	0.00	Dst_IP
-0.00	0.00	-0.71	0.69	-0.00	-0.00	-0.00	0.00	-0.00	Src_Port
-0.00	0.00	-0.69	-0.71	0.00	-0.00	-0.00	0.00	-0.00	Dst_Port
-0.00	0.00	-0.00	-0.00	-0.70	0.10	0.44	0.53	-0.00	Payload_Size
0.00	0.00	0.00	0.00	0.00	-0.96	0.25	-0.02	0.00	Keyword

Table 4 PCA Results

Each column in table 4 stands for an eigenvector, and each row stands for a field in the original data. The first eigenvector demonstrated that the most significant variance in the original data is the source IP and destination IP, so does the second eigenvector. However, PCA is only a tool without any knowledge of intrusion detection. A significant variance in IP addresses does not indicate any malicious attack. In fact, if we take the IP address factor into the profile and use it in experiments, the false alarms will be extremely high. Although it might "detect" some attacks because most attackers are from IP addresses which do not exist in the training data, it does not make any sense unless the protected network is not open for public access.

The third and fourth eigenvectors indicate the source port and destination port are significant variances. And the packet length and payload size stand out in the fifth, seventh and eighth eigenvectors. Keyword is the most significant one in the six eigenvector. Header length and IP version (which is always version 4) are only significant in the last eigenvector, which is the weakest one, so we do not take them into consideration.

Step 5: Build Profile

After the PCA process, we found the following properties are important for identification: source port, destination port, packet length, payload size, and keyword. Since packet length is the payload size plus IP header length, and we consider only payload related attacks, packet length is removed from consideration.

There is one property that is not processed by PCA. The keyword value is a key factor to identify application level attacks since most malicious activities involve invalid parameters or extremely long arguments. How to take the most advantage of the payload is well worth studying, but here we only use its length associated with the keyword because most attacks can be identified by inappropriate payload length or parameter length.

We create several property pairs based on the above discussion. For each pair, we collect all possible combinations in the training data, and save them in a hash table, as in table 5:

Properties	Values
DstPort:Keyword	80:GET, 80:POST, 25:EHLO, 1024:220, 25:RCPT, …
Keyword:Value_length	GET:125, 354:50, Received:60, …
Keyword:Payload_lenth	GET:1020, EHLO:65, …

Table 5 Properties used for profile

3.2 Detection Phase

The detection phase is demonstrated in fig. 2.

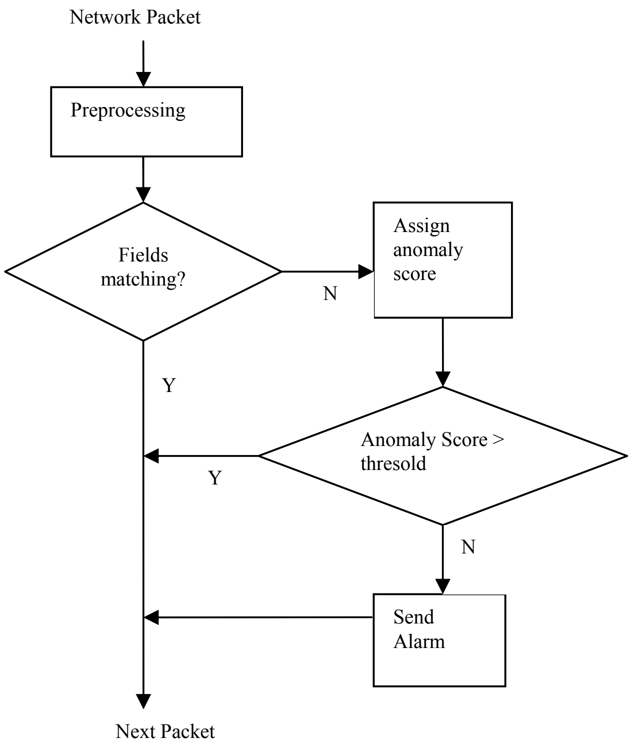


Figure 2 Detection Phase

Step 1: Preprocessing

The preprocessing is accomplished to extract the necessary fields for profile matching.

Step 2: Profile Matching

Matching the profile simply means comparing the property pairs in the profile. For example, table 5 defines three pairs: port->keyword, keyword->value_length, and keyword->payload_length. We then need to check if the incoming packet matches the profile. If the incoming packet is on the destination port 80, keyword is “GET”, and the value length is 30, then we search the profile hash table to see if we can find pairs of “port 80:GET” and “keyword GET:30”.

Step 3: Anomaly Score

Our anomaly score mechanism is very efficient. For each profile (port:keyword or keyword:length), we assign a different anomaly score. In the experiment, once an anomaly score is assigned, an alarm will be generated. Similar methods, such as ALAD, uses a scheme based on temporal probability approach, which strongly relies on the arbitrarily selected threshold and can only detect the attack if it lasts for a while, though it does not provide any advantages. We still have a threshold which is set to 0 (once anomaly score > 0, send alarm), but it is only set for future research. At this moment, the selection of threshold does not affect the result.

4. EXPERIMENT

Our experiment is based on the DARPA’99 intrusion detection dataset. In order to compare with other approaches, especially ALAD and PAYL, which are typical payload-based mechanisms. We use the same data set: week 3 (attack free) of inside network traffic for training, week 4 and 5 for testing. The truth table is from the website of MIT Lincoln Lab [20], which indicates there are a total of 201 instances of 58 different attacks, and 177 are visible in the inside tcpdump data.

Because we only focus on payload-related TCP based attacks, not all types of attacks in DARPA’99 need to be considered. Based on the truth table and attack description on the MIT Lincoln Lab website, we developed the payload-related attacks summary in table 6. There are a total of 33 types of attacks, most of which are U2R and R2U, and a total of 107 instances Because the original DARPA’99 dataset contains many other packets and attacks not related to our consideration, we only consider the detection rate and false alarm rate according to the data in table 6.

Attack	Instances #
Apache 2	3
Back	4
CrashIIS	8
Mailbomb	4
Teardrop	3
Casesen	3
Eject	2
Ffbconfig	2
Fdformat	3
Loadmodule	3
Perl	4
Ps	4
Sechole	3
Xterm	3
Yaga	4
Framespoofer	1
Ftpwrite	2
Guest	3
Httptunnel	3
Imap	2
Named	3
Ncftp	5
Netbus	3
Netcat	4
Phf	4
Sendmail	2
Sshtrojan	3
Xlock	3
Xsnoop	3
Ntinfoscan	3
Satan	2
Guesstelnet	4
Guessftp	2
Guesspop	1
Anypw	1
Total	107

Table 6 Payload-related Attacks in DARPA’99

4.1 Properties Selection

In section 3.1, we have mentioned there are 5 best properties for detection: source port, destination port, payload size, keyword, keyword value length. The combination of these properties, however, is still too much. If we assign anomaly scores to all of them, the false alarm rate will be unacceptable, even if it is possible to detect all attacks. Table 7 shows the results by using different property combinations for detection of week 4 and 5.

Notice here we only care about the detection accuracy for the payload-related attacks such as crashIIS, ps, perl, and ncftp, as in table 6. All properties not directly related to the payload will not be taken into consideration even if they could increase the overall performance. In method 1, we use 4 combinations, but found the dstport:keyword and srcprot:keyword combinations work poorly in payload-related attacks, although it could help identifying some DOS and probe attacks. The false positive rate in method 1 is unacceptable. The result in method 1 could be fine-tuned by incorporating other header field information, but it only helps for the network level detection and has nothing to do with our payload based concern.

Method 2~4 performs fairly well in identifying U2R and R2L attacks, as well as some payload related DOS and probe instances. Among these three, method 3 stands out for low false positive rates and acceptable detection rates. Details for the attacks detected by method 2-4 are listed in table 8. We see that method 3 did not miss many attacks even if removing one condition. It detects 31 instances in all 107 payload related attacks at the false positive rate of 0.012. There have not been other similar experiments focusing on payload based attacks only. In [21], header based methods have been applied for U2R and R2L attacks, which are mostly payload based, and the detection rate is generally less than 20% at the false rate of 0.1. In the following subsections, we compare our methods with two other payload based approaches.

	Total Instances detected	Payload related detected	Overall FP rate
Method 1 srcport:keyword dstport:keyword keyword:payload_len keyword:value_len	81	40	0.225
Method 2 keyword:payload_len keyword:value_len	41	34	0.03
Method 3 keyword:payload_len	40	31	0.012
Method 4 keyword:value_len	19	16	0.007

Table 7 Comparison for different property combinations.

	Method 2	Method 3

Types of payload related attacks detected	Ps(2)
	Sshtrojan(2)
	Guesstelnet(2)
	Netbus(2)
	Ntinfoscan(2)
	Teardrop(3)
	Back(2)
	Crashiis(4)
	Netcat(2)
	Yaga(3)
	Casesen
	Eject
	Ftpwrite
	Neptune
	Ffbconfig
	Fdformat
	Phf
	Satan
	Sechole
	Framespoofer
		Ps(2)
		Guesstelnet(2)
		Netbus(2)
		Ntinfoscan(2)
		Teardrop(3)
		Crashiis(5)
		Yaga(3)
		Casesen
		Sshtrojan
		Eject
		Ftpwrite
		Back
		Ffbconfig
		Netcat
		Fdformat
		Phf
		Satan
		Sechole
		Netcat

Table 8 List of detected payload related attacks in method 2 and 3

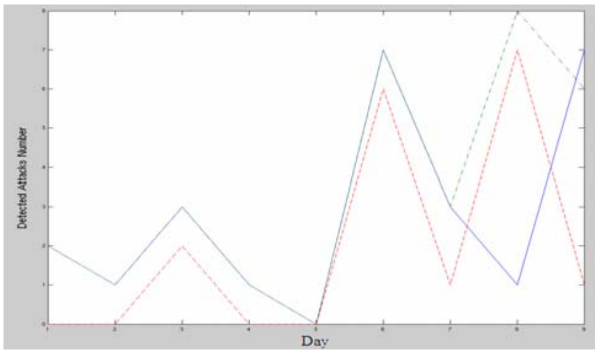
4.2 Comparison with ALAD

ALAD[5] is a similar approach to ours with a focus on application-level attacks. It also uses "keywords", although not the same ones as ours. ALAD uses 6 different property combinations, but contains only one payload-related combination: "keyword: dstport". The other properties selected are all in packet headers. ALAD is in fact an NIDS approach mixed with both header fields and payload. The source code of ALAD can be found at the author's website (http://www.cs.fit.edu/~mmahoney/dist/). Table 9 is the overall results between ALAD and our method.

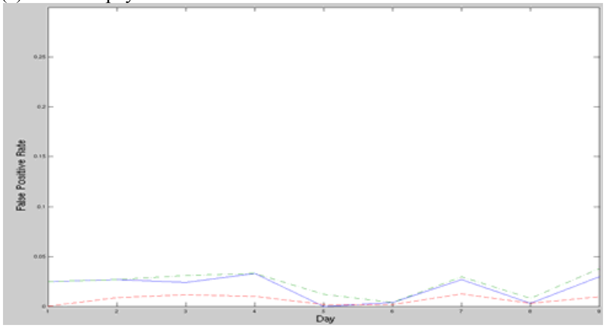
	ALAD	Our method
Total Attacks Detected	23	40
Payload related Attacks	17	31
Overall False Positive Rate	0.004	0.012

Table 9 Comparison with ALAD

From table 9, our method detects more payload related attacks than ALAD but at a higher false rate. However, when we look into the ALAD source code, and remove the keyword property from the profile to make it a header based detection approach, we found it still detects 22 attacks. The only missed one is the "smurf" attack, which is actually a ICMP based DOS attack, and it has nothing to do with the TCP payload. The finding indicates ALAD's keyword implementation does not really rely on the keyword approach to work. What really works in ALAD is still the packet header. Although the overall false positive rate of our method looks higher than ALAD, it is greater because of day 1, 2 and 4. In these days, ALAD cannot detect any real attack, so it generates very few alarms. From fig. 3, which gives detailed comparison between our method and ALAD day by day, we find the false positive rate of our method is close to ALAD at days when payload related attacks are popular (day 6, 8). When the payload related attacks are few, ALAD does not work, but our system still can detect them with acceptable false alarm rates (day 1~4, although we have higher false alarms which varies from 140~210 per day, it detects real attacks while the ALAD detects nothing, and the false alarms rate is acceptable). Figure 3 also shows that we can limit the false alarm rate to 200 per day without much decrease in performance.



(a) Detected payload related Attacks



(b) False Positive Rate

Figure 3 Comparison with ALAD for 9 day in week 4 and 5.

The solid blue line is our method with 200 false alarms per day limitation. The green stands for our method without false alarm limitation. The red dotted line is the result from ALAD.

4.3 Comparison with PAYL

In [14], payload based anomaly detection is performed based on byte distribution. Since they claim different port traffic has different byte variability, and their experiment is based on port traffic, we use our method to do the same experiment and compare the results according to [14] in table 10:

Method	False Positive Rate (%)	Detection Rate (%)
PAYL (Per Conn.)	0.1	15
PAYL (Tail 100)		10
Our method		27.7

(a) Port 21

Method	False Positive Rate (%)	Detection Rate (%)
PAYL (Per Conn.)	0.4	12
PAYL (Tail 100)		15
Our method		22.5

(b) Port 23

Method	False Positive Rate (%)	Detection Rate (%)
PAYL (Per Conn.)	0.4	10
PAYL (Tail 100)		15
Our method		27.7

(c) Port 25

Method	False Positive Rate (%)	Detection Rate (%)
PAYL (Per Conn.)	0.3	100
PAYL (Tail 100)		20
Our method		19.6

(d) Port 80

Table 10 Port based comparison with PAYL. The result of PAYL is measured from [14].

Table 10 shows our method is more stable than PAYL. PAYL is based on the byte distribution of payloads, while the distribution model can be constructed either from whole payloads in the connection (per conn.) or from the last 100 bytes on the tail (tail 100). In [14], the author did not provide a conclusion as to which way is better. Although the per conn. approach shows 100% accuracy in one instance of their experiment, it only works for HTTP traffic, and there is no proof that it could reach the same results except on the DARPA'99 dataset. The PAYL varies considerably based on the selection of threshold because it is based on the Mahalanobis distance from a statistical profile. When we set the false positive rate the same as in our method, PAYL does not provide any advantage. In fact, except for a few cases, our method shows much better detection rates.

5. CONCLUSION AND FUTURE WORK

The proposed anomaly detection method focuses on application level attacks, and our experiment shows its advantage at identifying payload related attacks. We use principal component analysis to perform automatic dimension reduction in network packet fields, then select the most appropriate ones for profile construction. We then compared the results of different profile choices in several experiments. We utilize the concept of payload "keyword", which is the most important component in our method, for payload related attack detection. Combining the keyword with other information, such as payload length, our method demonstrates reasonable performance in the experiments.

The major problem is the lack of similar research to compare. Most other experiments, regardless of whether header- or payload-based, focus on the results of all 201 attacks in DARPA'99, which includes a large portion of attacks without payloads or that do not generate any network traffic. The only similar experiment is done in PAYL [14], which was tested on 97 attacks with nonempty payloads in DARPA'99. PAYL is another payload-based anomaly detection mechanism with similar goal to ours, but they consider only "nonempty" payload attacks in their experiments, while payload-related attacks do not always contain nonempty payloads. We define a "payload-related" attack as any malicious network activity that happens on the application level, even if it has an empty payload. Thus we give out 107 attacks from DARPA'99 dataset in table 6. We also perform similar experiments to PAYL to compare the results, which prove our approach is generally more stable than PAYL under low false positive condition with higher detection rates.

Our method is also compared with another similar approach ALAD. Since ALAD has offered its source code, we are able to perform detailed comparison with it. In table 9 and fig. 3, we provide the experiment results of the comparison of both overall and day-by-day experiment results. Although ALAD is another "keyword" based approach, we found its keyword does not make much contribution to the detection result. All the detected attacks in ALAD are in fact identified by the header field properties. Since ALAD does not actually take advantage of payload information, it is not surprising that it detects fewer payload related attacks than ours. The experiment shows our mechanism identifies more payload related attacks (31 vs 17 in ALAD). Our method has more false alarms, but it could be limited to 200 per day without much loss in detection rate.

The experiment shows the advantage by extracting useful information from the packet payload for application level network attack detection, but there is much still to accompish in the future. Generally, the dilemma between the detection rate and false alarms still exists. It is difficult to make significant improvement by just applying different algorithms to the packet itself since people have attempted such approaches. There are several topics worth exploring. First, as we did in this paper, payloads contain much valuable information, and they hold great potential for further study. In our experiment, we already could achieve reasonable performance compared to other approaches by using extracted keywords and the payload length alone. It is necessary to keep focusing on the payload to obtain as much important information for network anomaly detection as possible. In the future, we would like to find a mechanism to analyze the payload automatically, and build a more accurate model. It is also important to start studying "session-based" detection, because many attacks involve multiple steps, such as installing a program first and then executing it at a later time. These attacks are very difficult to detect because each single step is valid when examined alone. The connection-based or packet-based detection mechanism works poorly for such attacks unless one of its packets shows some significant deviation from the profile. Thus a session-based NIDS mechanism is suggested.

REFERENCES

[1] Levy, E., "Approaching Zero", IEEE Security & Privacy Magazine, vol. 2, issue 4, pp. 65-66, 2004,

[2] Wenke Lee and Sal Stolfo, "Data Mining Approaches for Intrusion Detection", Proceedings of the Seventh USENIX Security Symposium (SECURITY '98), San Antonio, TX, January 1998

[3] Wenke Lee, Salvatore J. Stolfo, Philip K. Chan, Eleazar Eskin, Wei Fan, Matthew Miller, Shlomo Hershkop and Junxin Zhang, "Real Time Data Mining-based Intrusion Detection", Proceedings of DISCEX II, June 2001

[4] Matthew V. Mahoney and Philip K. Chan, "PHAD: Packet Header Anomaly Detection for Identifying Hostile Network Traffic", Florida Institute of Technology technical report CS-2001-04, 2001

[5] Matthew V. Mahoney and Philip K. Chan, "Learning Nonstationary Models of Normal Traffic for Detecting Novel Attacks", Proceedings of the 8th International Conference on Knowledge Discovery and Data Mining, pp. 376-385, 2002

[6] Matthew V. Mahoney, "Network Traffic Anomaly Detection Based on Packet Bytes", Proceedings of the 2003 ACM symposium on Applied Computing, pp. 346-350, 2003

[7] Lazarevic, A., Ertoz, L., Ozgur, A, Srivastava, J., Kumar, V., "A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection", Proceedings of the 3rd SIAM Conference on Data Mining, San Francisco, May, 2003

[8] R. Lippmann, et al., "The 1999 DARPA Off-Line Intrusion Detection Evaluation", Computer Networks, 34(4), pp. 579-595, 2000

[9] H. J. Wang, C. Guo, D. R. Simon, and A. Zugenmaier, "Shield: A Vulnerability-Driven Network Filters for Preventing Known Vulnerability Exploits", ACM SIGCOMM'04, Portland, USA, August, 2004

[10] C. Kruegl, T. Toth, and E. Kirda, "Service Specific Anomaly Detection for Network Intrusion Detection", Proceedings of the 2002 ACM symposium on Applied computing (SAC 2002), pp. 201-208, Madrid, Spain, 2002

[11] C. Kruegl, G. Vigna, "Anomaly Detection of Web-based Attacks", Proceedings of the 10th ACM Conference on Computer and Communication Security (CCS'03), pp. 251-261, Washington, DC, October, 2003

[12] Christopher Kruegel, Giovanni Vigna, and W. Robertson, "A multi-model approach to the detection of web-based attacks", Computer Networks, vol. 48, no. 5, pp. 717-738, August, 2005

[13] Matthew V. Mahoney and Philip K. Chan, "Learning Nonstationary Models of Normal Traffic for Detecting Novel Attacks", Proceedings of the 8th International Conference on Knowledge Discovery and Data Mining, pp. 376-385, 2002

[14] Ke Wang, S. J. Stolfo, "Anomalous Payload-based Network Intrusion Detection", Recent Advances in Intrusion Detection, RAID 2004, Sophia Antipolis, France, September 2004

[15] Ke Wang, Gabriela Cretu, Salvatore J. Stolfo, "Anomalous Payload-based Worm Detection and Signature Generation", Proceedings of the Eighth International Symposium on Recent Advances in Intrusion Detection(RAID 2005)

[16] R. Duba, P. Hart, D. Stork, Pattern Classification, 2nd Edition, Jonh Wiley & Sons, Inc., 2001

[17] M.L. Shyu, Shu Ching Chen, Kanoksri Sarinnapakorn, LiWu Chang, "A Novel Anomaly Detection Scheme Based on Principal Component Classifier," Proceedings of the IEEE Foundations and New Directions of Data Mining Workshop, in conjunction with the Third IEEE International Conference on Data Mining (ICDM'03), pp. 172-179, November 19-22, 2003, Melbourne, Florida, USA.

[18] Khaled Labib and V. Rao Vemuri, "An Application of Principal Component Analysis to the Detection and Visualization of Computer Network Attacks", Annals of Telecommunications, France. Nov/Dec 2005 Issue

[19] Xin Xu, X. Wang, "An Adaptive Network Intrusion Detection Method Based on PCA and Support Vector Machines", Proceedings of the 1st International Conference on Advanced Data Mining and Applications (ADMA'05), pp. 696-703, Wuhan, China, July 22-24, 2005

[20] MIT Lincoln Lab DARPA website, http://www.ll.mit.edu/IST/ideval/data/data_index.html

[21] Wenke Lee, Salvatore J. Stolfo, Kui W. Mok, "A Data Mining Framework for Building Intrusion Detection Models," sp, p. 0120, 1999 IEEE Symposium on Security and Privacy, 1999