Layered higher order n-grams for hardening payload based anomaly intrusion detection

Neminath Hubballi, Santosh Biswas, Sukumar Nandi

Department of Computer Science and Engineering

Indian Institute of Technology, Guwahati, India

Email:{neminath,vsiyengar,santosh_biswas,sukumar}@iitg.ernet.in

***Abstract*—Application based intrusion detection involves analysis of network packet payload data. Recently statistical methods for analyzing the payload are being used. Since behavior of every application is not same a different model is necessary for each application. Studies have revealed that higher order n-grams are good for capturing the network profile. In this paper we introduce a concept of layered version of n-gram for payload based anomaly network intrusion detection. Each layer works as an independent anomaly detection system. A packet is declared as normal after passing through all the layers. A packet is declared as anomalous if at any layer it is declared as anomalous and we stop further processing the packet. We create a set of bins and equally distribute the distinct n-grams to each bin. Each such n-gram is a 2 tuple where the first element is byte values of the n-gram and second is the frequency of gram in the entire training data. We assign an anomaly score to each bin based on the frequency of the individual gram in the bin and is termed as coverage of the bin. We evaluate the proposed scheme on normal traffic of DARPA 99 dataset mixed with a set of attacks. Experimental results shows the efficacy of the method with a false alarm rate as low as 0.001%.**

I. Introduction

In the world of internet and networking the amount of computer security breaches taking place are ever increasing. There are handful number of security components ranging from simple firewalls to intrusion detection systems(IDS) to intrusion prevention systems (IPS). Traditionally Intrusion detection systems are of two main types as signature based and anomaly based. The former method uses a set of signature database and raises an alarm whenever it sees traffic that satisfy the signature rules or constraints. SNORT [1] and BRO[9] are examples of signature based IDS. As they use database of known attack signatures, they fail to detect new or novel attacks. Zero day/novel attacks and worms which exploit a known vulnerability can cause serious security threat, thus detecting these attacks is important from the network security perspective and is a challenging task. Anomaly detection technique learns the normal(benign) behavior and trigger an alarm whenever it sees traffic that do not confirm to the learnt benign behavior. If the true normal behavior is captured and modeled, anomaly IDS is capable of detecting new or zero day attacks. The underlying conjecture here is every true zero day attack must differ from the normal behavior. But the important issue with anomaly IDS is to get a clean data (which is attack free) and ensuring the collected data actually represents the true benign behavior of the network. Machine learning techniques have been used extensively as anomaly detection systems in the literature [2]. Since it is difficult to collect a labeled dataset which is representative of the true benign network behavior, many approaches in the literature use unsupervised learning techniques. Unsupervised anomaly detection systems are based on the assumption that every zero day attack must contain something new which is not seen in normal traffic.

Further anomaly detection systems are divided into two groups based on the feature set they use: (i)Works on features generated from the packet headers and (ii) Works on the payload data of the packet. Former performs well on the attacks which violate network protocol standards and denial of service attacks as there is some visible change in the traffic structure. This technique will not suffice for detecting the application level attacks. This is because in the application based attacks content is hidden deep inside the packet and often packets do not violate the protocol behavior during operation. Although anomaly based IDS are capable of detecting new or novel attacks, they suffer from huge false alarms. Accuracy of detection of an attack by the anomaly IDS depends on the accurate modeling of benign network behavior. Modeling the protocol behavior from packet header features is relatively easy. However analysis of the payload data of packet is computationally expensive. This is because payload is huge and there is no defined structure for the payload data. In addition, the payload of different applications differ. For example, what content HTTP application normally has is not same as telnet application. In order to learn the benign behavior of these applications a separate model needs to be learned for each application. In the present work we use the n-gram based analysis of HTTP application payload for detecting the attacks. We propose a layered higher order n-gram analysis approach of payload for detecting the attacks. Each of these layers have higher order n-grams stored in an efficient tree structure termed as $condensed\ tree$ which constitutes the normal profile of the network.

Rest of the paper is organized as follows. In section II we review the existing literature for payload based anomaly detection. In section 3 we discuss the proposed mechanism

and in section 4 we give the experimental results. Finally the paper is concluded in the section 5.

II. Review of literature

One of the first efforts towards use of payload information for anomaly IDS was reported in [13]. It extracts 256 features from the payload of every packet corresponding to 256 possible ASCII values of a byte. Then a simple model of normal system behavior is derived with average and standard deviation. A feature vector is declared as anomalous if mahalnobis distance[13] of feature vector exceeds a predermined threshold δ. Although a more generic n-gram based feature vector with 256^n possible values is given in both [13], [14] the exponential growth of the feature vector size limits the utility of the method due to curse of dimensionality problem. In addition, the method also considers payload length and models different payload lengths by a different model even to a single application. Further [16] pointed out, still there is a false alarms issue with this method.

Analysis of payload based Application Level network Anomaly Detection(ALAD) is another payload based IDS where the first word of each line (in the payload) is treated as a keyword. These keywords are associated with the header fields(port number, ip address etc) of the corresponding packet. Pairs thus generated will be stored and a statistical profile is generated. Since the training pairs are generated form normal traffic the model detects any new pairs of keywords and header fields. In the testing phase each incoming new pair is compared with every keyword header field pair and if no match is found an anomaly score is attached and is added to the total anomaly score of the packet. In [16] it was shown that on DARPA 99 data set ALAD detects only 17 of the total 34 payload based attacks bringing its detection rate to 50%.

The paper [15] proposes a two tier architecture for anomaly intrusion detection. In the first step the entire payload data is reduced to a sizeable quantity with sampling technique and in the next step clustering algorithms are applied and evaluated. In this experiment different clustering techniques such as k-means algorithm, principle direction and partition self organizing map have been tried out.

In [4] a very simple model of binarized versions of n-grams stored in highly space efficient bloom filters is proposed. The method worked remarkably well on HTTP data. Random mixture of n-grams are used for calculating the anomaly score of a packet which is the ratio of new n-grams seen in the packet (which are not there in training data) to the overall n-grams in the packet.

The method of [11] uses multiple one class support vector machines to classify the payload data as normal or attack. The paper proposed a new technique called $2-v$ grams, which pairs every two bytes that are separated by v bytes in the payload. All these pairs generate the feature vector of the payload of a packet. As the number of pairs is 65532 for a packet, a reduction in the number of features is achieved with minimal compromise in accuracy. This reduction in feature space is done using an iterative clustering algorithm. With the reduced feature vectors, k number of one-class support vector machines are trained. During the testing phase, the final decision (attack or normal) is made by a probabilistic algorithm which considers the outputs of all the support vector machines. Although the technique gives a very high accuracy, the complexity of scheme is still a big concern.

III. Layered n-grams

N-gram analysis is a well known concept and has been used successfully in many applications [13]. In this section we describe the proposed layering scheme of higher order n-grams for payload based anomaly detection. An n-gram is a subsequence of n consecutive items in a given sequence of items. For example consider the sequence a, b, c, d, e, f then the subsequences of 2 are (a, b), (b, c), (c, d),(d, e) and (e, f). In our case the bytes of network packet is the sequence. The value of n is the number of elements paired together in a defined order. Typically the value of n ranges from 1 to 8. Since each byte can take one of the 256 possible values, the number of 1 grams possible are 256. In general, the size of n-gram is 256^n. Lower order n-grams such as 1-gram and 2-grams even though detect attacks but detection rate are low [8]. Higher order n-grams are computationally expensive to generate and mentain since the size of feature vector grows exponentially as n increases. The number of n-grams generated from a packet is very low compared to the feature size. For example with $n = 3$ cardinality of the feature vector is 16777216 while a packet typically can contain 1500 bytes of data; thus the feature vector generated appears too sparse. To reduce the complexity, the scheme $anagram$ proposed in [4] stores the distinct n-grams in the entire training dataset using space efficient bloom filters.

Unlike anagram which did not store the frequency information of the n-grams, we do a rating of packet based on frequency of the n-grams that are seen in the training data. A list of distinct n-grams along with frequency of individual n-gram in the entire training information is stored. As testing packet is given as input, the proposed approach generates n-grams of a certain length and compare them with the already stored n-grams in the database. A score called as $anomaly\ score$ is assigned to the n-gram based on frequency of gram in the database(entire training data). To achieve this, the training data consisting of n-grams is divided into bins of frequency counts and an anomaly score is assigned to this bin. Most frequently occurring n-grams are grouped into one bin and given a score of 0. Next frequently occurring bin is given a score slightly higher and so on. The n-grams which do not find a match in the database are given a very high score. A true zero day attack must contain some n-grams which are not seen in the entire training data and hence are detected.

We mentain a layer of n-grams by storing distinct n-grams seen in the training data along with their frequency counts. In the testing phase, first lower order n-grams are generated and tested against the stored profile. Based on the bin to which the gram falls, an $anomaly\ score$ is added to a running sum. If the $anomaly\ score$ of the n-grams present in the packet is more than a preset threshold τ we stop further analysis of the packet and declare it as anomalous. If a packet is declared as normal by a layer beneath, it is passed to the next higher layer for further analysis. $Anomaly\ score$ of the packet is calculated by equation 1 and compared to the threshold. The

packet is declared as normal only after passing through all the layers maintained.

$$Score_i = \frac{\sum Score[Bin[gram]]}{T_{total}} \quad (1)$$

where $Score$ is the running sum of $anomaly\ score$, Bin is set of frequency bins generated, $gram$ is the n-gram generated from a packet in a particular layer and T_{total} is the total number of n-grams frequency. This layered scheme has the following advantages compared to Anagram.

- As Anagram uses a mixture of n-grams all the higher order n-grams need to be generated. As we use the layered approach a packet that contain enough abnormal content is detected at the lower n-grams.
- As it is difficult to have a clean training data an accidental presence of a attack packet may add n-grams to normal database; this leads to false negatives during testing phase since anagram is a binary based approach. As the attack packets are very low they fall in the lower frequency bin and hence are assigned higher $anomaly\ score$; thus our model detects such attacks too.
- Compared to random mixture of the n-grams layered approach gives a systematic way to generate the n-grams.
- Preserves the frequency of n-grams as in PAYL.
- Excessive training in case of anagram increases the false negatives due to collisions in the bloom filter. But our algorithm uses an efficient tree structure for storage(discussed bellow) and can accommodate any amount of training data.

We use frequency bins to accumulate the grams with near by frequencies into one group and one anomaly score is assigned to the bin. Now the grams which fall into a bin will receive the same anomaly score although their frequencies are slightly different. This reduces the different number of anomaly scores we need to keep. We use the Sturges' rule to determine the number of bins required which is given by equation 2. Then the number of elements which fall into individual bin is given by equation 3.

$$k = \lceil 1 + \log(Distinct\ grams) \rceil \quad (2)$$

$$w = \frac{Distinct\ grams}{k} \quad (3)$$

where $Distinct\ grams$ is the different n-grams in the entire training data, k is the number of bins and w is the number of n-grams falling into individual bin.

This approach distributes total number of distinct n-grams for a particular value of n into k bins each having w grams. We introduced a criteria for assigning a score for the bin based on the coverage of the bin. Coverage is defined as *Number of frequencies that fall into the bin compared to the total frequency* This score is given by equation 4.

$$Score_{bin_i} = \left| \log \left(\frac{Frequency\ of\ grams\ in\ the\ bin}{Total\ number\ of\ grams} \right) \right| \quad (4)$$

The above equation ensures that high frequency n-grams and corresponding bins are scored lower compared to the infrequent n-grams.

A. Complexity and simplicity tradeoff

Generating higher order n-grams is computationally very expensive task. However these higher order n-grams are good for detecting the rare events and hence attacks. On the other hand if lower order n-grams are considered, the model is simple and easy to handle since processing of packet can be done easily. As payload analysis is a deep packet inspection technique it needs considerable amount of processing and memory. One way to handle this massive processing is to do the computation in the hardware. Hardware based processing can increase the speed by many fold.

It has to be noted that in the proposed approach the individual n-grams need to be stored and searched for a match for every gram generated. This adds to the running cost. Here we provide a software based approach of hashing to reduce the cost of searching and also the memory requirement of storage. The storage structure is adopted from [3]. The idea is motivated by the antimonotonicity property of the n-grams i.e., there can not be an n-gram of higher order with higher frequency than that of its constituent lower order n-grams. For example consider the sequence of numbers 1,2,5,2,3,4,1,4,2,1,2,5,2,3,4,1,4,2. There are 5 distinct one grams namely {1}, {2},{3},{4},{5} with frequencies 4,6,2,4 and 2. We represent the n-gram by a tuple where the first element is the actual n-gram and second element is the frequency count of the gram. The above 1-grams are represented as {(1),4}, {(2),6},{(3),2},{(4),4},{(5),5}. Similarly there are 9 different two grams {(1,2),2},{(2,5),2},{(5,2),2},{(2,3),2},{(3,4),2}, {(4,1),2},{(1,4),2},{(4,2),2},{(2,1),1}. For example the tuple {(1,2),2} the first element (1,2) is 2 gram and it occurs 2 time in the above sequence. These tuples are stored in a tree structure called as $condensed\ tree$. We mentain a list of buckets for the distinct one grams seen in the training dataset. Each node has a pair of elements first is the actual n-gram and second represents its frequency of occurrence in the training data. The tree construction starts at 1-gram and goes successively for 2,3 and so on. For example to store the two grams we begin at 1-gram and have the required number of buckets. The first byte value of 2-gram is hashed to a bucket of one gram and the second byte value is attached at a second level bucket. Each of the initial one gram buckets plays the role of head or root node in a tree. Entire bucket structure looks alike a tree as shown in figure 3. In the testing phase for any higher order n-gram the search begins at the one gram and by a hashing of each of subsequent bytes of the n-gram. For example, let 3-gram {(2,3,4),2} is to be searched. First we have to hash the first byte 2 and go to second bucket. Then the second byte 3 is hashed to the first child of above bucket and then finally to its child. This structure avoids the need of storing all levels of n-grams separately and hence save the space. In the diagram we have shown the details upto 3 grams, but can be extended to any level required.

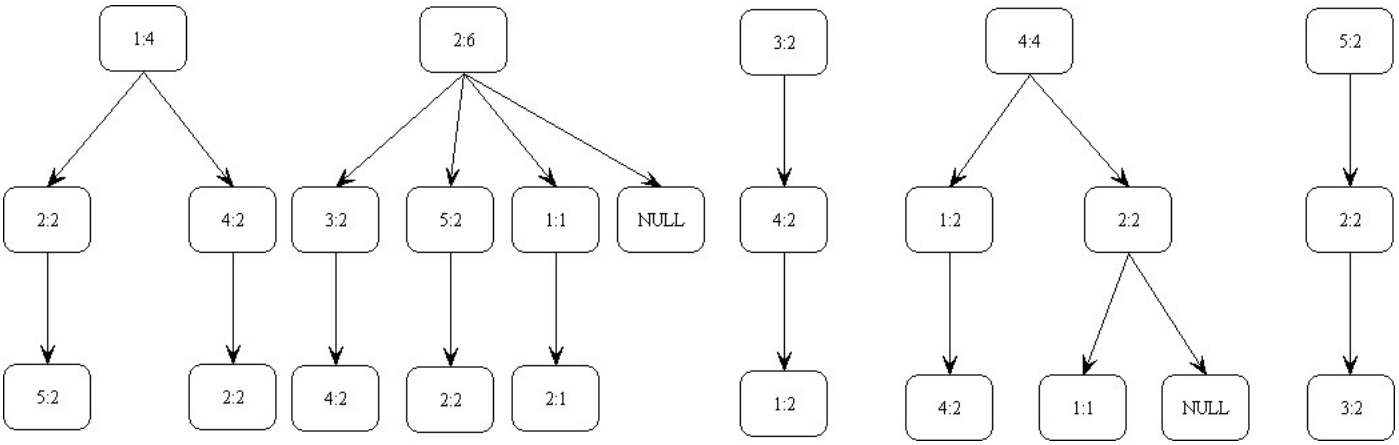

Fig. 1. N-gram bucket structure

IV. Experimental results

In this section we report the experimentation done to evaluate the scheme. We used the traffic to port number 80 which contains HTTP data. Specifically we report results on DARPA 99 dataset. For the attacks we used a collection of attack packets given by the authors of [11].

The details of the dataset used for experimentation are discussed bellow.

- **DARPA:** HTTP requests from the first week of DARPA 99 [10] tcpdump files are separated. Although this dataset has been criticized by [6], [7] for the simulation environment used for the collection of packets to the best of our knowledge this is the only publicly available dataset with complete payload and packets available. In our experiments on DARPA 99 dataset we used the first week normal traffic for n-gram generation. There are 5 days of traffic in this week. We randomly divided the dataset into two groups for each day. First group consist of 75% of the packets and second consist of 25% of the packets. All the 5 days 75% traffic is used as training data and remaining 25% data is used as part of testing data.
- **Attacks:** We experiment with several non-polymorphic and simple shellcode based attacks and morphed attacks.
  - Generic attacks: This set include the attack dataset shared by authors of [5] and some additions made by authors of [11].
  - Shell-code attacks: Contains 11 shell-code attacks.
  - CLET attacks: Consist of 96 polymorphic attacks generated by the CLET[12] engine.

The details about the characteristics of the datasets used in the experimentation are given in the Table I and Table II.

Shellcode attacks carry the shellcode in the payload to overflow the buffer of the victim machine. The morphed attacks are crafted attacks which are designed to evade the IDS, especially signature based IDS engines.

Higher order n-grams capture the normal profile precisely. As n increases, number of distinct grams generated also increases as the events comprising n bytes together also decreases. Figures 2 show the number of distinct n-grams seen in DARPA 99 day 1 normal traffic. Figure 3 shows the number of distinct n-grams seen for 75% of the 5 days DARPA 99 normal traffic. It is to be noted that different n-grams found increases almost linearly as n increases.

TABLE I
DARPA dataset characteristics

Day	Number of packets	Training	Testing
1	161602	121201	40401
2	196605	147454	49151
3	189362	142021	47341
4	268250	201187	67063
5	150847	113135	37712
Total	966666	724998	241668

TABLE II
Attack dataset characteristics

Type	Attack	Number of packets
Generic	66	205
shell-code	11	93
CLET	96	792

The 25% of normal testing traffic from DARPA dataset is mixed with the attacks n-grams (given by the authors of [11]) and evaluated. $Recall$ and $Precision$ are two important parameters to evaluate the efficiency of the IDS and are given by equations 5 and 6 respectively.

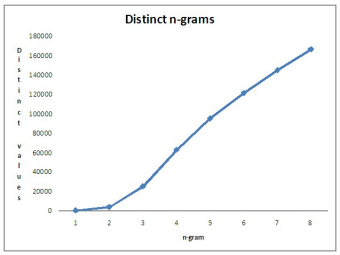

Fig. 2. Number of distinct n-grams for DARPA day 1 traffic

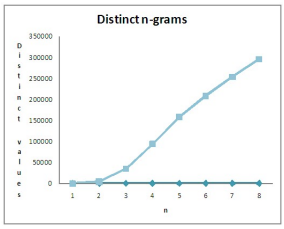

Fig. 3. Number of distinct n-grams for DARPA training traffic

TABLE III
RECALL FOR GENERIC ATTACK

Layer	Recall
3	100%
4	100%
5	100%
6	100%
7	100%
8	100%

$$Recall = \frac{TP}{TP + FN}. \quad (5)$$

$$Precision = \frac{TP}{TP + FP}. \quad (6)$$

$Recall$ obtained for various layers is given in tables 3,4,5. $Recall$ and $Precision$ of the scheme is reported for the individual layers and also cumulatively where i includes the attacks detected at layer 1 to i. Table III gives the $Recall$ for the generic attacks, Table. IV for the shellcode attacks and Table.V for the morphed attacks for individual layer. Table. VI gives the $Precision$ of the scheme with individual layers on the DARPA 99 dataset mixed with generic attacks.

We notice from the Tables III, IV and V that $Recall$ for these attacks is 100% for all layers. This result leaves us with little scope for the benefit achieved by the layered approach. Ideally if all of these attacks are detected by a lower layer

TABLE IV
RECALL FOR SHELLCODE ATTACKS

Layer	Recall
3	100%
4	100%
5	100%
6	100%
7	100%
8	100%

TABLE V
RECALL FOR MORPHED SHELLCODE(CLET) ATTACKS

Layer	Recall
3	100%
4	100%
5	100%
6	100%
7	100%
8	100%

TABLE VI
PRECIOSN FOR DARPA 99 TEST DATA WITH GENERIC ATTACKS

Layer	Accuracy
3	100.000%
4	99.999%
5	99.993%
6	99.017%
7	99.012%
8	99.001%

why would one go for a higher layer n-gram generation which is costly to generate and maintain. The point here is, in our experiments as we used DARPA 99 dataset which is almost clean and artificial, we could recognize some of the bytes in the attack data which are not there in the training data(DARPA 99). In a realistic network data this will not be the case. Our feeling is in a real network data this layered approach actually add value.

To demonstrate the effect of method discussed we conduct an experiment where we add some packets of attack to the training data and test wether still we are able to detect these attacks. Precisely we conduct an analysis of, "to what extent the trained model can survive the uncleanliness of the training data". This is an important characterization since it is almost always difficult to get a clean dataset which is free of attacks. Although one can filter out known attacks from the training data and use it, but this do not filter the zero day attacks present in the dataset. Thus it is important to build a model to survive such a residual attack bytes leftover. This is where the approach of higher order n-grams and layering approach plays a role and is different from the Anagram's binary approach.

In this experiment we deliberately add some some attack packets to the training data. Table 7 shows the $Recall$ obtained for the generic attack set by this addition to the training dataset. Each column in the table reports the number of attack packets that are added to the DARPA clean dataset and the corresponding $Recall$. The observations made out of our experiments are

1) As more attack packets are added, the $Recall$ also falls.
2) Higher order n-grams are more resistant to the addition than lower order n-grams.
3) Beyond 50 packets addition the damage done for the $Recall$ is huge.

Point 2 is important and fully justifies why the higher order n-grams are a better choice for detection. This is because the lower order n-grams have less number of unseen grams in the attack packets compared to the corresponding higher order n-grams. As the attack packets are added they quickly mask these unseen bytes and hence affect the $Recall$. Table.VIII shows the cumulative $Recall$ for the layers and we notice as we move to the higher layer the $Recall$ increases considerably. From the table it is evident upto 100 packets addition the $Recall$ is still well over 80%.

V. CONCLUSION AND DISCUSSION

We proposed a layered, systematic, higher order and frequency usage n-gram model for payload analysis of HTTP

TABLE VII
RECALL FOR GENERIC ATTACKS WHEN THE TRAINING DATA IS NOT CLEAN

Layer	5 packets	15 packets	25 packets	40 packets	50 packets	100 packets	150 packets	205 packets
3	99.51%	64.39%	51.21%	44.87%	41.46%	32.71%	32.73%	31.71%
4	100.00%	55.12%	60.00%	46.83%	41.95%	32.71%	37.10%	31.71%
5	100.00%	66.34%	59.51%	46.83%	43.43%	37.10%	37.10%	31.71%
6	100.00%	66.83%	62.44%	50.24%	44.87%	37.10%	37.10%	32.20%
7	100.00%	76.09%	75.61%	75.61%	65.85%	63.37%	47.32%	38.05%
8	100.00%	76.09%	75.61%	75.12%	64.39%	63.37%	47.32%	40.73%

TABLE VIII
CUMULATIVE RECALL ACROSS THE LAYERS FOR GENERIC ATTACKS WHEN THE TRAINING DATA IS NOT CLEAN

Layer	5 packets	15 packets	25 packets	40 packets	50 packets	100 packets	150 packets	205 packets
3	99.51%	64.39%	51.21%	44.87%	41.46%	32.71%	32.73%	31.71%
4	100.00%	81.46%	62.93%	46.83%	46.83%	36.09%	37.10%	31.71%
5	100.00%	86.82%	67.31%	48.78%	46.83%	41.46%	37.10%	33.17%
6	100.00%	87.31%	75.73%	51.21%	47.56%	42.92%	37.10%	33.07%
7	100.00%	96.58%	95.95%	96.09%	90.73%	85.58%	76.58%	36.09%
8	100.00%	97.07%	96.58%	96.09%	90.73%	85.58%	77.56%	46.09%

data. We rate the packet based on the frequency of n-grams generated from packet seen in training dataset. Obviously most frequently occurring n-grams in training dataset are rated low and vise versa. We evaluated the performance of the method on the DARPA 99 dataset mixed with a range of attacks and reported both recall and precision of the proposed method. In addition we also conducted experiments to study the performance of the method under the unclean environment which we named as stressed analysis. The conclusion drawn out of the stress analysis experiment is, "as the training dataset gets dirty the $Recall$ falls". However higher order n-grams can circumvent this problem to a good extent.

The present scheme detects attacks relying on individual layer decisions. To make the scheme better a probabilistic approach of combining the outputs of all the layers for detection can be done and currently we are working on that.

REFERENCES

[1] *www.snort.org*.

[2] Jung-Min Park Animesh Patcha, *An overview of anomaly detection techniques: Existing solutions and latest technological trends*, Computer Networks **51** (2007,), 34483470.

[3] Jian Pei Jiawei Han and Yiewen Yin, *Mining frequent patterns without candidate generation,*, ACM SIGMOD, (2000,), 1–12.

[4] Janak J. Parekh Ke Wang and Salvatore J. Stolfo, *Anagram: A content anomaly detector resistant to mimicry attack*, Proc. Recent advances in intrusion detection,LNCS **4219** (2007,), 226–248.

[5] H.Inoue K.L. Ingham, *Comparing anomaly detection techniques for intrusion detection for http*, Proc. Recent advances in intrusion detection,LNCS **4637** (2007,), 42–62.

[6] McHugh, *Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection evaluations as permormed by lincoln laboratory*, ACM Transactions on Information and System Security, **3** (2000,), no. 4, 262–294.

[7] P.K. Chan M.V. Mahoney, *An analysis of the 1999 darpa lincoln laboratory intrusion detection data for network anomaly detection,*, Recent adavnces in Intrusion Detection, (2003,), 220–237.

[8] D. Dagon O. M. Kolesnikov and W. Lee, *Advanced polymorphic worms: Evading ids by blending in with normal traffic*, Technical Report GIT-CC-04-13, College of Computing, Georgia Tech, (2004,).

[9] Vern Paxon, *Bro: A system for detecting network intruders in real-time*, Computer networks **31** (1999), 23–24.

[10] D.J. Fried J.Kobra R.Lipmann, J.W. Haines and K.Das, *The 1999 darpa off-line intrusion detection eveluation,*, Computer Networks, **34** (2000,), no. 4, 579–595.

[11] Prahlad Fogla Giorgio Giacinto Wenke Lee Roberto Perdisci, Davide Ariu, *Mcpad: A multiple classifier system for accurate payload-based anomaly detection*, Computer Networks **53** (2009), 864–881.

[12] M.Underdruk T.Detristan, Y.Malcom, *Ploymorphic shelcode engine using spectrum analysis,*, Phrack Issue, **0x3d** (2003,).

[13] Ke Wang and Salvatore J. Stolfo, *Anomalous payload-based network intrusion detection*, Proc. Recent advances in intrusion detection,LNCS **3224** (2004), 203–222.

[14] ———, *Anomalous payload-based worm detection and signature generation*, Recent advances in intrusion detection, 2005, pp. 203–232.

[15] Stefano Zanero and Sergio M. Savaresi, *Unsupervised learning techniques for an intrusion detection system*, Symposium on Applied Computing, ACM, 2004, pp. 412–419.

[16] Like Zhang and Gregory B. White, *Analysis of payload based application level network anomaly detection*, Proceedings of the 40th Hawaii International Conference on System Sciences, IEEE, 2007, p. 99.