Packet Header Anomaly Detection Using Bayesian Topic Models

Xuefei Cao,[*] Bo Chen, Lanjun Dang, Hui Li

January 15, 2016

Abstract

A method of network intrusion detection is proposed based on Bayesian topic models. The method employs tcpdump packets and extracts multiple features from the packet headers. A topic model is trained using the normal traffic in order to learn feature patterns of the normal traffic. Then the test traffic is analyzed against the learned normal feature patterns to measure the extent to which the test traffic resembles the learned feature patterns. Since the feature patterns are learned using only the normal traffic, the test traffic is likely to be normal if its feature pattern resembles the learned feature patterns. An attack alarm is raised when the test traffic's resemblance to the learned feature patterns is lower than a threshold. Experiment shows that our method is efficient in attack detection. It answers the open question how to detect network intrusions using topic models.

1 Introduction

Network intrusion detection has been an important issue nowadays since our daily work depends heavily on networks. Network intrusions fall into four categories including Denial of Service, Remote to Local, User to Root, and Probe. There are two methods to detect intrusions in general, i.e., signature-based methods and anomaly-based methods. The signature-based method predefines the patterns of intrusions and matches the network traffic against the patterns to raise detection alarms. While this method has low false alarm rate, it gives less than satisfactory results in detecting new types of attacks beyond the predefined patterns. Anomaly-based method establishes the normal usage patterns for network traffic and any behavior deviating from the normal patterns is deemed anomalous. Compared with signature-based method, anomaly-based method has the advantage of detecting zero-day exposure attacks, and witnesses a fast development.

LDA (Latent Dirichlet Allocation) [1] is a Bayesian topic model for text categorization. It is used to extract the representations of a collection of documents. Since LDA is capable of generalizing the structure representations, and network intrusions should be different in structure from normal traffic, LDA

[*] Xuefei Cao is with Xidian University, e-mail: xfcao@xidian.edu.cn

model could be used to learn the structures of normal network traffic and to detect the network intrusions.

In this paper, we study the problem of LDA-based network intrusion detection. We employ a comprehensive set of network features, and come up with a new way of applying LDA model to network traffic. We also testify our method on widely tested network traffic data, and the experiment shows the effectiveness of our method.

The remaining of this paper is arranged as follows. Section 2 discusses the related work and our work's relationship with existing work. Section 3 introduces the Latent Dirichlet Allocation. Our scheme is proposed in Section 4. Section 5 gives the performance study while Section 6 concludes the paper.

2 Related Works

There have been continuous efforts to apply LDA model to the analysis of network traffic and cyber data. Cramer et al. [2] study the patterns of the network traffic in a corporation environment using LDA. It discovers the pattern differences in network usage between daytime and nighttime. [4] proposes several constructions of anomaly detectors in LDA's framework, and notices several abnormalities in a laboratory network. [16] applies LDA to model network traffic profiles, and shows the pattern differences when the profiles are represented by LDA topics. These papers succeed in detecting the anomalies using LDA, but there is still gap between bringing up pattern anomalies and detecting intrusions because anomalies do not necessarily relate to attacks. Also, there lacks the report of these the methods' application to intrusion detections. The current results are just initial explorations in the line, it is still an open question that how to use LDA to detect network intrusions [2].

According to the type of network traffic, network intrusion detection systems could be divided into three categories. One type of traffic data in use is system calls. Forrest et al. [5] first use traces of system calls for anomaly detection in UNIX process. For a given host, they train an n-gram model ($n = 3$ to 6) over the normal system calls, and look for trace differences in the test data. Liao et al. [9] improve the method by introducing text categorization techniques. They describe normal program behavior by counting the frequency of system calls using k-Nearest Neighbor classifiers. Then text categorization technique is adopted to convert each process into a vector and the similarity is calculated between processes. The k neighbors with the nearest similarity are chosen to determine whether the process is normal or not.

Another form of audit data is TCP/IP connection descriptions which include the summarization of high level interactions between hosts such as session duration, type of service, number of failed login attempts, status of guest log-in and so on. Many systems first reconstruct raw network data into connections and extract connection features before carrying out detection techniques. MADAMID [7], Bro [14], EMERALD [15] are systems of this kind. Stolfo et al. produced a standard TCP/IP connection dataset in 1999 [19]. Various machine learning techniques have been applied to the dataset and shown their effectiveness, for example, Naive Bayesian[24, 25], nearest neigbor[8, 6, 3], neural networks[20, 10], fuzzy-logic[22] and so on.

The third intrusion detection system works at packets level. Since attacks exploit software bugs, for it to succeed, certain packet feature fields usually use less testified feature values. The most common paradigm based on TCP/IP packets is the firewall model, which blocks traffic to certain hosts or ports. Recent researches use more compact set of packet feature fields to gain improved detection rate. Wang and Stolfo [23] use packet payload. PHAD [12] models 33 packet features, and detects attacks based on the sum of abnormal score over the features. NETAD [11] filters out the most significant packets such as the beginning and ending packets of a connection, and models the protocol behavior using attributes from the first 48 bytes of the packets. Shon et al. [18] apply genetic algorithm to tcpdump packets. [17] uses the same set of features as PHAD [12], and models network behaviors according to protocols used by the traffic.

3 Topic Models

Latent Dirichlet Allocation is a statistical text model which assumes that a small number of distribution over words, called topics, generate the observed documents. Each document exhibits the topics set with different proportion [13]. Formally, the vocabulary is a vector V, the d-th document $\boldsymbol{w}^{(d)}$ is an unordered word count defined over V of length $\|V\|$. The corpus is $\mathcal{W} = \{\boldsymbol{w}^{(1)}, \boldsymbol{w}^{(2)}, \cdots, \boldsymbol{w}^{(D)}\}$. $\boldsymbol{\theta}_d$ is the topic distribution for the d-th document, and it follows a Dirichlet distribution with prior parameter α with dimension K. A set of K topics $\Phi = \{\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \cdots, \boldsymbol{\phi}_K\}$ are used, ϕ_i denotes the word distribution for the i-th topic and the length is $\|V\|$. Each word of a document, w_n^d has a topic label $z_n^d \in \{1, \cdots, K\}$ which is drawn from $\boldsymbol{\theta}_d$ multinomially. The generative process of LDA is shown graphically in Fig. 1.

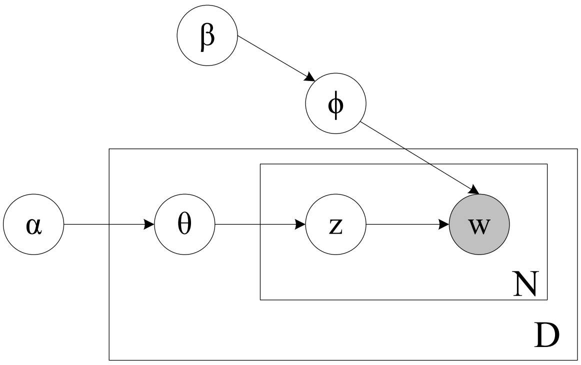

Figure 1: Generative process of LDA model

In the figure, α and β are hyperparameters for $\boldsymbol{\theta}$ and $\boldsymbol{\Phi}$, respectively. The distributions of variables are as follows:

$$\begin{array}{rcl}\boldsymbol{\theta_d} & \sim & \text{Dir}(\alpha) \\ \boldsymbol{z_n^d} & \sim & \text{Multinomial}(\boldsymbol{\theta_d}) \\ \boldsymbol{\phi_k} & \sim & \text{Dir}(\beta) \\ \boldsymbol{w_n^d} & \sim & \text{Multinomial}(\boldsymbol{\phi_{z_n^d}})\end{array}$$

w is an observable variable, $\boldsymbol{\theta}$ and $\mathbf{\Phi}$ are latent variables to be inferred. Two effective methods to do the inference are variational Bayes and Gibbs Sampling. Throughout this paper, we use Gibbs Sampling because Gibbs Sampling is a typical method for the inference of hierarchical Dirichlet structures and is able to calculate the exact conditional posterior. Variational Bayes might be faster but the performance is slightly worse than Gibbs. Given the observed data $\mathcal{W}$, Gibbs sampling resamples $\mathcal{Z}$ according to its conditional posterior and then estimates $\boldsymbol{\theta}$ and $\mathbf{\Phi}$ by examining the distribution of $\mathcal{Z}$. The detailed parameter inference is listed as below.

$$\begin{aligned} p(z_n^d = j| \sim) &\propto \theta_{dj} \cdot \phi_{jw_n^d} \\ p(\boldsymbol{\theta}_d| \sim) &\propto \frac{\Gamma(\Sigma_{i=1}^{|\alpha|}\alpha_i)}{\prod_{i=1}^{|\alpha|}\Gamma(\alpha_i)} \cdot \prod_{i=1}^{K} \theta_{di}^{n(d,i)+\alpha_i-1} \\ p(\boldsymbol{\phi}_k| \sim) &\propto \frac{\Gamma(\Sigma_{i=1}^{|\beta|}\beta_i)}{\prod_{i=1}^{|\beta|}\Gamma(\beta_i)} \cdot \prod_{i=1}^{V} \phi_{ki}^{n(\cdot,k,i)+\beta_i-1} \end{aligned}$$

, where $n(d,i)$ is the number of words in document d assigned to topic i, $n(\cdot,k,i)$ is the number of word k assigned to topic i in the corpus, and $\Gamma(n) = (n-1)!$.

4 Packet header anomaly detection using LDA

To apply LDA to network traffic data, the first step is to transfer the network traffic, either in the form of tcpdump packets or TCP/IP connections, into documents which are expressed by words. In LDA, a document is treated with the notion of *bag-of-words*, i.e., the order of word tokens is neglected, and a document is expressed using a predefined vocabulary and the times each word in the vocabulary appearing in the document. Tcpdump traffic enables us to draw an analogy between network traffic and documents easily. The tcpdump traffic within a time slot, say five minutes, can be viewed as a document, and the document can be converted into words by means of the packets' network features. To be more specific, tcpdump traffic is composed of packets which contains many feature fields, such as the IP address, MAC address, packet length and so on. The features are repeatable across the session. Then the repetition of features in a session resembles the repetition of words in a document. Therefore, the vocabulary is defined as all possible network features appearing in the traffic. Each packet is defined by a set of N distinct features, and then converted to a set of N words. Table 1 is an analogy between text categorization and network traffic when applying LDA.

In our scheme, we extract $N = 16$ features from a packet. These features are proved to be effective in the detection of intrusions [12]. Table 2 lists all the features used.

Another question is how to detect attacks using LDA given the traffic transformed. Our method is to use LDA to generalize the structure of normal traffic, and to employ the document likelihood as a detector. Attack-free traffic is used to train the LDA model, thus the resulted topic-word distribution Φ in fact describes the structure for normal traffic. Likelihood is employed to define the extent to which a test document deviates from the normal structure. If a test documents's likelihood is lower than the threshold, the document is regarded as abnormal. Since different hosts may have different behavior structures, our

Term	Text Categorization	Network Traffic
D	total number of documents	total number of five-minute tcpdump sessions
$\boldsymbol{w^{(d)}}$	the d-th document	the d-th five-minute tcpdum session
$\boldsymbol{V}$	the collection of unique words in corpus	the collection of unique features in traffic
w_n^d	the n-th word in the d-th document	the n-th feature in the d-th session (extracted from the $\lceil \frac{n}{N} \rceil$-th packet in the session)
$n(i, j)$	the number of i-th word assigned to the j-th topic	the number of i-th feature assigned to the j-th topic

Table 1: Analogy between text categorization and network traffic

Packet Layer	Features Extracted
Ethernet Layer	higher 3 bytes of MAC source, higher 3 bytes of MAC destination
IP Layer	total length, type of service, fragment flags, time to live, IP source, IP destination
Transport/ Control Layer	TCP flag, TCP checksum, TCP URG pointer, TCP option UDP checksum, service port, ICMP checksum
Others	packet size

Table 2: Feature List

scheme is host-based. Each host has its own threshold, which is the minimum of likelihood of the training documents.

The vocabulary list is also host-based and should contain all possible feature values seen by the host, thus we collect the features appearing in the host's training data. However, the training data is attack-free, but attacks usually employ features that seldom used by normal traffic, thus we assign an extra value for each feature field so as to deal with the features that doesn't appear in the training data. The features which don't appear in the training data but appear in the test data will be assigned to this extra value of the field. Let F_i $i \in \{1, 2, \cdots, 16\}$ be the number of anomalies collected in the training data's i-th field, then the length of the vocabulary is $\|V\| = F_1 + F_2 + \cdots + F_{16} + 16$.

Based on the above ideas, our method comprises three steps. First, vocabulary list of a host is built based on the host's attack-free tcpdump packets data during a long enough time. Each packets is denoted by 16 features, and the anomalies of each feature field are collected. The vocabulary is the collection of all the anomalies in the 16 feature fields plus 1 extra word for each feature filed. Then, both training traffic and test traffic are separated into five minute sessions and then each session is turned into a document by counting how many times each feature appearing in the session. Finally, LDA model is trained using the training documents. Likelihood of each training document is computed using Φ and θ_d according to Eq.1. The minimal likelihood of training documents is used as the threshold. In the test phase, we compute the likelihood of each test document, and the document whose likelihood is lower than the threshold is labeled as attack.

$$lik_d = \prod_{n=1}^{Nd} \Sigma_{j=1}^{K} p(w_n^d \| z_n^d = j, \phi_j) \cdot p(z_n^d = j \| \theta_d) = \prod_{n=1}^{Nd} \Sigma_{j=1}^{K} \phi_{j w_n^d} \theta_{dj} \tag{1}$$

5 Experimental Results

The network traffic used in this session is the MIT Lincoln Laboratory DARPA dataset which is the most popular experimental dataset for network intrusion detection systems [21]. Most NIDS would test their efficiency against the dataset, and thus the effectiveness of NIDS could be compared over the same baseline. The DARPA99 dataset provides the simulated tcpdump data of a military base. The audit logs contain three weeks of training data and two weeks of test data. The first week and the third week training data are attack-free while the two weeks of test data contain 201 attack instances of 55 types covering all the four attack categories. In the training phase, we use the third week's 7 days tcpdump traffic (Mar 15 - Mar 19 with two extra days) which are are collected in the internal network (inside.tcpdump). Two weeks of test data (Mar 29 - Apr 2 and Apr 5 - Apr 9) are used to test the efficiency of our method.

Since our model is host-based, we first process the training and test traffic by selecting packets to and from a single host and put them together. There are 15 hosts of concern in the training and test dataset, thus these datasets are separated into 15 sub-datasets respectively. Figure 2 illustrates the separation of training data, each column in the figure is a sub-dataset.

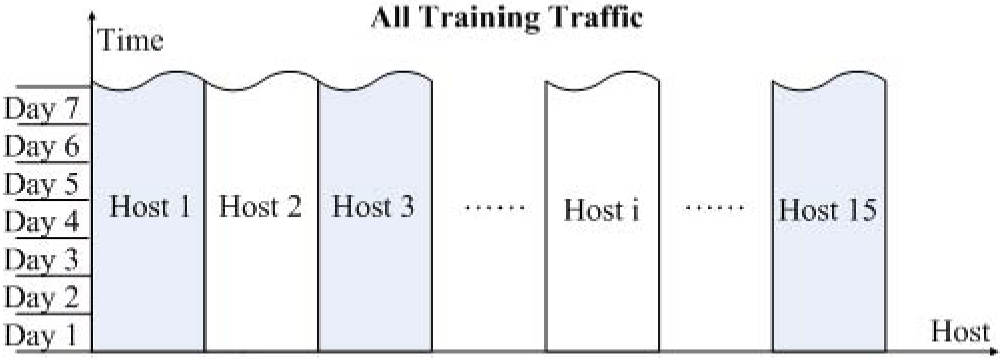

Figure 2: Example of Data Separation for Training Traffic

The vocabulary list is first generated based on the attack-free training data. For each host, the range of feature anomalies is determined by first extracting 16 features from every packet and then filtering the unique values for each field. Then a vocabulary list is generated for each host by adding one extra word for each feature field according to the description in Section 4.

Then the traffic can be converted into documents. For each host, the traffic is divided into 300s sessions, i.e, the arriving time of the last packet is at most 300 seconds later than the first packet. Then we count the times each feature value appears in a session, and turn the session into a document. Take the host Pascal (172.16.112.50) as an example. Figure 3 shows the features collected in the training traffic of Pascal. Suppose a session containing 200 packets whose feature distribution is as shown in Figure 4, the resulted document is a 1 × 2788 vector with the {1, 1322, 1323, 1324, 1329, 1330, 1337, 2665, 2667, 2672, 2681, 2685, 2687, 2716, 2718, 2747, 2757, 2760, 2763, 2766, 2770, 2771, 2772} digits setting as {199, 1, 100, 100, 100, 100, 199, 1, 200, 200, 200, 100, 100, 100, 100, 200, 200, 200, 200, 200, 199, 1, 200}, respectively. Note that this session should come from a test session because it contains EtherSize_others, IPLength_others, and ICMPChecksum_others values.

Field name	**Anomaly #**	**Values**
Ether Size	1321	60-184 186-231 ··· 1503-1514
Ether Src Hi	5	x0000c0 x00105a x00107b ···
Ether Dest Hi	6	x0000c0 x00105a x00107b ···
IP Length	1329	NULL 38 40-170 172-217 ··· 1489-1500
IP TOS	4	NULL x00 x08 x10
IP Fragment	3	NULL x0000 x4000
IP TTL	7	NULL 60 63 64 128 254 255
IP Src	29	135.8.60.182 135.13.216.191 172.16.112.10 172.16.112.20 172.16.112.50 ···
IP Dst	33	135.8.60.182 135.13.216.191 172.16.0.1 172.16.112.10 172.16.112.20 172.16.112.50 ···
TCP Flag	9	NULL x02 x04 x10 ···
TCP Checksum	2	NULL xffff
TCP URG	2	NULL 0
TCP Option	2	NULL x020405b4
UDP Checksum	2	NULL xffff
ICMP Checksum	2	NULL xffff
Service Port	16	NULL 20 21 22 23 25 53 79 80 113 123 514 515 6000 6667 8000

Figure 3: Feature anomalies of Pascal

An LDA model is trained for each host. The reason is that different host may be dedicated to different purposes such as mail server, Internet proxy, thus the topic distributions could be totally different among hosts. By using a single LDA model for each host, the detection accuracy could be greatly improved. The number of topics used by LDA is determined according to the size of vocabulary list. We use $T = 150$ for most of hosts whose vocabulary size is around 2000 and $T = 10$ for two hosts with vocabulary size around 100. For all the hosts, $\beta = 0.01$ and $\alpha = 10/T$. LDA model is trained using the training documents.

Field name	Feature Value	Count Number
Ether Size	60	199
	185	1
Ether Src Hi	x0000c0	100
	x00105a	100
Ether Dest Hi	x0000c0	100
	x00105a	100
IP Length	38	199
	171	1
IP TOS	x00	200
IP Fragment	x0000	200
IP TTL	255	200
IP Src	172.16.112.10	100
	172.16.112.50	100
IP Dst	172.16.112.10	100
	172.16.112.50	100
TCP Flag	NULL	200
TCP Checksum	NULL	200
TCP URG	NULL	200
TCP Option	NULL	200
UDP Checksum	NULL	200
ICMP Checksum	0xffff	199
	0xabcd	1
Service Port	NULL	200

Figure 4: Example of session feature distributions

There are at most 1716 documents and at least 829 documents used in training for different hosts. Attacks are detected during the test phase using likelihood according to Equation 1 where Φ is learned in the training phase and θ_d in the test document. For each host, the lowest likelihood of its training documents is used as threshold. The test document whose likelihood is lower than the threshold is labeled as an attack. Given 24037 documents for all the 15 hosts, our method raises 1041 documents as abnormal, of which 730 are true positives and 490 are false positives. Since one instance of attack may contain several documents, there are 94 attacks detected in all. Each host has an average of 3.27 false positive documents per day. Table 3 lists the detection result using our method. We also compare the efficiency of our method against that of

PHAD[12]. The Column 1 is the number of instances detected by our method, Column 2 is by PHAD while Column 3 lists all the attack instances contained in the test dataset.

Attack	**Det 1**	**Det 2**	**#**	**Attack**	**Det 1**	**Det 2**	**#**
anypw	1	0	1	netcat	4	1	4
apache2	3	2	3	ntfsdos	1	0	3
arppoison	1	0	4	ntinfoscan	2	1	3
back	3	0	4	perl	1	0	4
casesen	2	1	3	phf	2	0	4
crashiis	1	1	8	pod	4	4	4
dict	1	0	1	portsweep	3	13	15
dosnuke	4	4	4	ppmarcro	0	1	3
eject	2	0	2	processtable	3	1	4
fdformat	1	0	3	ps	2	0	4
ffbconfig	1	0	2	queso	0	3	4
ftpwrite	1	0	2	satan	2	2	2
guessftp	1	1	2	sechole	0	1	3
guesspop	1	0	1	selfping	1	0	3
guesstelnet	2	2	4	sendmail	1	1	2
httptunnel	1	0	3	smurf	5	5	5
imap	2	0	2	snmpget	1	0	4
illegalsniffer	1	1	2	tcpreset	1	0	3
ipsweep	3	4	7	teardrop	3	3	3
mailbomb	3	2	4	udpstorm	2	2	2
mscan	1	1	1	warez	2	1	4
named	3	1	3	xlock	3	1	3
ncftp	1	0	5	xsnoop	2	0	3
neptune	3	1	4	xterm1	1	0	3
netbus	2	3	3	yaga	3	0	4
Total	45	24		**Total**	49	40	

Table 3: Attacks detected

As can be seen from the table, our detection system outperforms PHAD in the detection of many types of attacks. This is because that in our method the behavior rule for the normal traffic, i.e. the topic-word distribution Φ, is generalized automatically by LDA which makes full use of all features and treats every feature as an independent variable. A topic is a representation of all the normal features with different probability. A document, or a session of traffic, should be a combination of normal topics. The likelihood is a measurement of the extent to which a document resembles the normal behavior rule. The lower the likelihood, the more likely the document contains an attack. By the contrast, PHAD built the model by adding up all anomaly values of each feature field, and used the sum to separate attacks from normal traffic. This method depends on a single variable which is too strong and lost the information presented by normal features. Thus our method can detect many types of attacks which is not detected by PHAD. Our method detects fewer probe attacks including `portsweep`, `queso` and `ipsweep` than PHAD does. This is due to the inborn insufficiency of host-based solutions and could be improved by increasing the weight of certain features such as port number and TCP flag, and this will be our next step work.

6 Conclusion

In this paper, we propose an LDA-based network intrusion detection scheme. Our scheme collects the features used by normal traffic and uses LDA to build the behavior rules that normal traffic should follow. Likelihood is employed as a detector to detect the attacks in test phase. The comparison between our scheme and existing scheme proves the possibility and efficiency of our solution. This work answers the question of how to detect network intrusions using LDA model. Our future work is to further increase the detection rate by improving the model and to consider the payload data in order to better utilize the advantage of the language models.

References

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[2] C. Cramer and L. Carin. Bayesian topic models for describing computer network behaviors. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 1888 –1891, may 2011.

[3] Levent Ertoz, Michael Steinbach, and Vipin Kumar. Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data. In Daniel Barbara and Chandrika Kamath, editors, *Proceedings of the Third SIAM International Conference on Data Mining (SDM 2003)*, volume 112 of *Proceedings in Applied Mathematics*. Society for Industrial and Applied Mathematics, 2003.

[4] E.M. Ferragut, D.M. Darmon, C.A. Shue, and S. Kelley. Automatic construction of anomaly detectors from graphical models. In *Computational Intelligence in Cyber Security (CICS), 2011 IEEE Symposium on*, pages 9 –16, april 2011.

[5] Stephanie Forrest, Steven A. Hofmeyr, Anil Somayaji, and Thomas A. Longstaff. A sense of self for unix processes. In *In Proceedings of the 1996 IEEE Symposium on Security and Privacy*, pages 120–128. IEEE Computer Society Press, 1996.

[6] ShengYi Jiang, Xiaoyu Song, Hui Wang, Jian-Jun Han, and Qing-Hua Li. A clustering-based method for unsupervised intrusion detections. *Pattern Recognition Letters*, 27(7):802 – 810, 2006.

[7] Wenke Lee and Salvatore J. Stolfo. A framework for constructing features and models for intrusion detection systems. *ACM Transactions on Information and System Security*, 3:227–261, 2000.

[8] Yang Li, Binxing Fang, Li Guo, and You Chen. Network anomaly detection based on tcm-knn algorithm. In *Proceedings of the 2nd ACM symposium on*

Information, computer and communications security, ASIACCS '07, pages 13–19, New York, NY, USA, 2007. ACM.

[9] Yihua Liao and V. Rao Vemuri. Using text categorization techniques for intrusion detection. In *Proceedings of the 11th USENIX Security Symposium*, pages 51–59, Berkeley, CA, USA, 2002. USENIX Association.

[10] Guisong Liu, Zhang Yi, and Shangming Yang. A hierarchical intrusion detection model based on the pca neural networks. *Neurocomputing*, 70(79):1561 – 1568, 2007.

[11] Matthew V. Mahoney. Network traffic anomaly detection based on packet bytes. In *Proceedings of the 2003 ACM symposium on Applied computing*, SAC '03, pages 346–350, New York, NY, USA, 2003. ACM.

[12] Matthew V. Mahoney and Philip K. Chan. Learning nonstationary models of normal network traffic for detecting novel attacks. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 376–385, New York, NY, USA, 2002. ACM.

[13] David Mimno and David Blei. Bayesian checking for topic models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 227–237, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[14] Vern Paxson. Bro: A system for detecting network intruders in real-time. In *Computer Networks*, pages 2435–2463, 1999.

[15] Phillip A. Porras and Peter G. Neumann. Emerald: Event monitoring enabling responses to anomalous live disturbances. In *In Proceedings of the 20th National Information Systems Security Conference*, pages 353–365, 1997.

[16] David Gerald Robinson. Statistical language analysis for automatic exfiltration event detection. In *Sandia National Laboratories Report*, april 2010.

[17] Solahuddin B. Shamsuddin and Michael E. Woodward. Modeling protocol based packet header anomaly detector for network and host intrusion detection systems. In *Proceedings of the 6th international conference on Cryptology and network security*, CANS'07, pages 209–227, Berlin, Heidelberg, 2007. Springer-Verlag.

[18] Taeshik Shon, Xeno Kovah, and Jongsub Moon. Applying genetic algorithm for classifying anomalous tcp/ip packets. *Neurocomputing*, 69(1618):2429 – 2433, 2006.

[19] Salvatore J. Stolfo, Wei Fan, Wenke Lee, Andreas Prodromidis, and Philip K. Chan. Cost-based modeling for fraud and intrusion detection: Results from the jam project. In *In Proceedings of the 2000 DARPA Information Survivability Conference and Exposition*, pages 130–144. IEEE Computer Press.

[20] Adel Nadjaran Toosi and Mohsen Kahani. A new approach to intrusion detection based on an evolutionary soft computing model using neuro-fuzzy classifiers. *Comput. Commun.*, 30(10):2201–2212, July 2007.

[21] Chih-Fong Tsai, Yu-Feng Hsu, Chia-Ying Lin, and Wei-Yang Lin. Intrusion detection by machine learning: A review. *Expert Systems with Applications*, 36(10):11994 – 12000, 2009.

[22] Chi-Ho Tsang, Sam Kwong, and Hanli Wang. Genetic-fuzzy rule mining approach and evaluation of feature selection techniques for anomaly intrusion detection. *Pattern Recogn.*, 40(9):2373–2391, September 2007.

[23] Ke Wang and Salvatore J. Stolfo. Anomalous payload-based network intrusion detection. pages 203–222, 2004.

[24] Yun Wang. A multinomial logistic regression modeling approach for anomaly intrusion detection. *Computers & Security*, 24(8):662–674, November 2005.

[25] Yun Wang, Inyoung Kim, Gaston Mbateng, and Shih-Yieh Ho. A latent class modeling approach to detect network intrusion. *Computer Communications*, 30(1):93 – 100, 2006.