Intrusion Detection of DoS/DDoS and Probing Attacks for Web Services

Jun Zheng and Ming-zeng Hu

The Research Center of Computer Network and Information Security Technique,
P.O.Box 320, Harbin Institute of Technology, Harbin, China, 150001
{zhengjun, mzhu}@hit.edu.cn

Abstract. The (Distributed) Denial of Service (DoS/DDoS) attacks have become the main devastating threats to web services, and generally, the Probing attacks are the prior steps of DoS/DDoS attacks. To achieve the aim of the information assurance, an intrusion detection mechanism based on the Vector Quantization (VQ) technique is proposed for countering DoS/DDoS and Probing attacks in this paper. The normal network traffic usage profile can be modeled and represented by the codebook of VQ from which the abnormal behavior deviation of TCP traffic can be measured quantitatively well. In data processing, according to the characters of DoS/DDoS and Probing attacks, we implement the novel feature extraction of TCP flow state. We apply the detection mechanism to DARPA Intrusion Detection Evaluation Data Set. It is shown that the network attacks are detected with more efficiency and relatively low false alarms.

1 Introduction

With the ever-fast development of the Internet, all kinds of network service are rising today. For example, the Web service becomes more and more popular and general in the Internet. Because of the extensive public available, the Web service also becomes the main object of malicious attacks, especially when the e-business flourishes. To strengthen the security of Web severs and connected database, in addition to security techniques, such as access control policy, authentication and encryption, Intrusion Detection System (IDS) becomes an important and traditional security-barrier against network-based attacks.

Among the malicious attacks, maybe DDoS/DoS and Probing attacks are the main threat. The DDoS/DoS attacks destroy the service availability directly, whenas Probing attacks usually are used to explore the detail information of severs, for example, the network Worms usually utilize probing attacks to scan the certain scope network using some given scanning policy. Both of two attacks are the TCP attacks mainly. According to the statistical data from Moore [1], the majority of DoS/DDoS attack which is main threat to the whole Internet is deployed by using TCP as 90~94%. In this paper, we concentrate on the TCP attacks via extracting the TCP header information.

There are two general approaches in Intrusion Detection: Misuse Intrusion Detection (MID) and Anomaly Intrusion Detection (AID). Similar to virus detection, misuse detection is based on the pattern matching to hunt for the signatures extracted

W. Fan, Z. Wu, and J. Yang (Eds.): WAIM 2005, LNCS 3739, pp. 333 – 344, 2005.

from the known attacks. However, AID constructing the normal usage behavior profile, named historical or long-term behavior profile. And the analysis model looks for deviations of the short-term behavior profile from the normal. The deviations can be treated as the baselines of estimating the attack activities from normal behaviors. The Fig.1 describes the relation and contrast of main logical procedures in MID and AID. The two approaches are contrary as well as unitive some time.

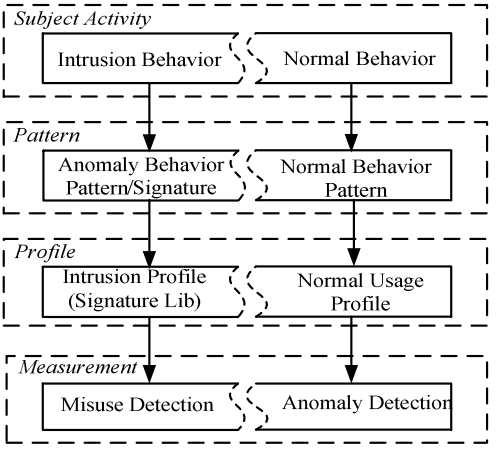


Fig. 1. Logic procedures in Intrusion Detection

In this paper, we propose a new AID mechanism using the Vector Quantization (VQ) [2]. The normal network traffic usage profile can be modeled in the codebook of VQ from which the abnormal behavior deviation can be measured quantitatively well. Furthermore, in data processing, according to the characters of DoS/DDoS and Probing attacks, we implement novel feature extraction of TCP flow state to achieve the maximal abnormal attack characteristics. The evaluation bed used in this paper is the 1999 DARPA Intrusion Detection Evaluation Data Sets.

The rest of paper is organized as follows: Section 2 explains the background and related work of AID, Section 3 proposes the new framework to construct the usage behavior profile and detect anomalies, and Section 4 illustrates the new method of data processing to intrusion detection. Section 5 describes the details of our experiments. Finally, Section 6 gives conclusions.

2 Background and Related Work

With the knowledge of normal behavior to detect novel intrusions, AID searches for anomalous behavior or deviations from the established baseline. If the deviations exceed the specified threshold of baseline, the activities will be flagged as anomalous. It is no doubt that the most important thing is to establish the profile characteristics of normal activity. Generally, the AID is looked as the binary classification: the given data sample is judged as normal activity or intrusion activity according to the AID model.

The most classical AID model comes from the earlier landmark paper of Dorothy Denning -An Intrusion Detection Model. [3] Denning describes building an "activity profile" of normal usage over an interval of time and proposed to employs statistics to construct a point of reference for normal behavior. Denning's work inspires the ***statistical methodology*** in Anomaly/Intrusion Detection fields.

However, as the attacks and intrusions become more complex, more and more ***machine learning model and data mining model*** are proposed to solve the problems of network security, including general Clustering [4], Outlier Detection, Support Vector Machine [5], Hidden Markov Models [6] and Neural Network [7] and so on. However, because of the complexities of algorithms, one of main universal shortcomings of these methods is that these methods are not enough efficient to detect by the real time style. Without maintaining the profiles of normal behavior, Clustering and Outlier Detection [4] seem to be promising recently but these methods depend mightily on the precondition that abnormal data must take the little percentage among all the audit data for the certain temporal space. These methods can not get higher detection rate in case of DDoS or probing attacks because of the huge mount of activities and the higher proportion attack traffic during some certain time interval. Other approaches try to construct a certain profile according to different data objects in either host computer scope or network scope just as Table 1. SVM can achieve the higher classification rate but its algorithm complexity to describe the normal usage profile embarrasses the efficiency and usability in actual application. The same problems arise to HMM and Neural Network, especially in case of the huge mount of network traffic audit data in the network-based intrusion detection.

Table 1. Profile classification

Data Object	Profile	Scope
System Audit Log (e.g. BSM)	System Profile	Host Based
OS System Call	System Call Profile	
User Command Sequence	User Command Profile	
Network Traffic (Packet/Connection/Flow)	Network Traffic Profile	Network Based

In our methodology, we concentrate on AID and propose a novel approach to establish structural profiles of normal network traffic usage using VQ, and detect anomalies according to the deviations. Hereon, a codebook of VQ serves as the structural depiction of normal network traffic usage profile just deployed in section 3. However, our main aim is to implement a lightweight AID system based on the Quantization Vector to corporate with other Misuse Intrusion Detection systems (e.g. SNORT [8]).

3 Vector Quantization for Intrusion Detection

Vector Quantization (VQ) [2] is an efficient coding technique, especially in data compressions of image and speech signal based on ***the similarity measures between***

feature vectors. Through the compression approach of VQ, it is easy to transmit and to construct the index structure for the multimedia data files with high-dimensions. Traditionally, the initialization of Vector Quantization can be looked as the technology of the input space partition according to the similarities of input feature vectors. In this way, Vector Quantization is an approach to data clustering of a data set.

- ***Definition1.*** Vector Quantization can be defined as a mapping function from Euclidean space R^k into a certain finite subset C:

$$Q: R^k \to C, \quad C=\left\{y_0, y_1, \cdots, y_{n-1} \middle| y_i \in R^k\right\}. \tag{1}$$

  The representative vector y_i in C is called ***codewords***, and *C* is called ***codebook***.

- ***Definition2.*** Given he input vector $x=(x_1, x_2, ..., x_k)$ and the codeword $y_i=(y_{i1}, y_{i2}, ..., y_{ik})$, every input vector will be assigned with a codeword in which the distortion between this codeword and the input vector is smallest among all codewords. The distortion is defined as ***Quantization Errors D*:**

$$D=\|x-y_i\| =[\sum_{j=1}^{k}(x_j-y_j)^2]^{1/2}. \tag{2}$$

The potential of VQ application to the usage profile analysis of network traffic is that VQ can compartmentalize the traffic feature vector space to groups comparing the similarities of feature vectors. All the profiles can be distilled and recapitulated into the codebook. Consequently, through the codebook organization of VQ, we can get the usage profiles to character the network traffic.

- ***Definition3.*** The usage profiles *P* of network traffic can be defined:

$$P_i=\left\{x\in R^k : Q(x)=y_i\right\}, i=1,2,...,n. \tag{3}$$

All x with the profiles P_i can be represented by y_i. Here, we can understand that VQ is also can be looked as a clustering processing that input space is grouped into *N* clusters.

3.1 Codebook Design

The first important step to implement Vector Quantization, codebook construction is the training processing by the normal network traffic with a certain algorithm.[2] The LBG algorithm [9] and Competitive Learning [10] are the most widely used approach to design codebook of VQ. Here, we use the Competitive Learning --Kohonen Learning Algorithm (KLA) [11] to train the structural codebook of normal network traffic. The codebook can be treated as the dictionary of normal network traffic behavior where the similar traffic behaviors are clustered to the same Voronoi partition space [12] represented by one codeword, also known as the centroid. After training, the normal traffic behaviors have been compacted to the codebook and the similar behaviors have been clustered together. The usage profiles are embodied in the codebook at last.

Table 2. Parameters in KLA

Parameter	Value
Topology	Hexa
Neighborhood function	Bubble
Dimension (X×Y)	30×30 40×40
Learning rate function	Linear
Training rate of first phase	0.5
Radius in first phase	20
Training length of first phase	50000
Training length of second phase	500000
Training rate of second phase	0.02
Radius in second phase	10

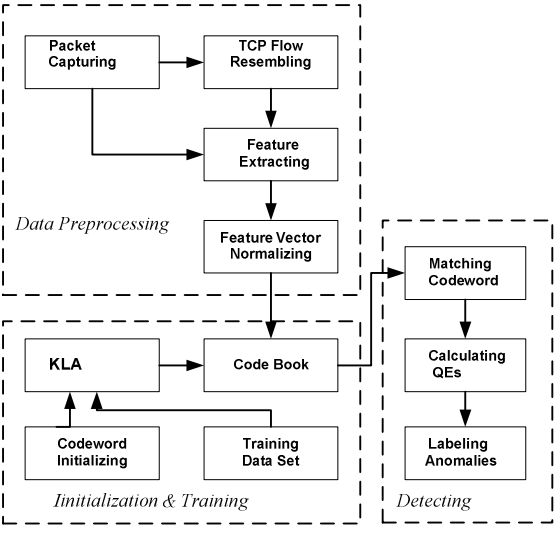


Fig. 2. Vector Quantization framework for intrusion detection

The codewords close to the input vectors by iterations. The codebook construction processing is the coupling processing between the all codewords and the input vector of network traffic.

- ***Definition4***: ***Every TCP Flow is a data point in the n-dimension feature space*** R^n ***and*** R^n ***is*** **Euclidian space.**

$$\text{TCPFlow} = \left\{ X \middle| X \in \mathbf{R^n} \right\}. \tag{4}$$

Every TCP Flow is expressed by the form of feature vector: $X = (x_1, x_2, \cdots, x_n)$.

The following are the main steps involved in KLA:

The input vector: $X = (x_1, x_2, \cdots, x_n)$ The codeword: $W = (w_1, w_2, \cdots, w_n)$

Step1. To initialize every codeword of KLA with random values: $W_j(0)$

Step2. To compute the distance between the input vector X_i and the codeword $W_j(t)$, designate the winner neuron node j^* with the smallest distance. j^* is also called the ***Best Matching Unit (BMU)***.

$$j^* = \arg \min_{1 \le j \le m} \left\| X_i - W_j(t) \right\|. \tag{5}$$

The Euclidean distance is chosen as ***Quantization Errors (QEs)***:

$$D = \left\| X_i - W_j(t) \right\| = \left(\sum_{k=1}^{n} (x_{ik} - w_{jk}(t))^2 \right)^{1/2}. \tag{6}$$

Step3. To update the winner vectors of the winner node and its neighborhood:

$$w_{jk}(t+1) = w_{jk}(t) + \alpha(t)[x_{ik} - w_{jk}(t)] \quad j \in N(t). \tag{7}$$

$N(t)$ is the non-increasing neighborhood function, $\alpha(t)$ is learning rate function $0 < \alpha(t) < 1$.

Step4. To repeat Step2 and Step3 until KLA learning stabilizes

In this paper, the parameters of KLA used to train the codebook in this paper can be referred in the following Table2.

3.2 Detection

The Vector Quantization framework for the TCP attack detection is described in the following Fig.2. After training, the codebook has been constructed for the normal network traffic. In other words, Voronoi partition [12] is formed in the normal traffic behavior space. Every codeword of codebook represents the centroid of the more similar group according to the minimum Euclidean distances among the input vectors of network traffic. In detection phase, the input feature vectors will be processed to the codebook, which is known as searching best matching codeword, i.e. the nearest neighbor searching. The anchor point named as the nearest neighbor codeword has been found for each input feature vector by exhaustively searching to compute its minimum Euclidean distance to all of the codewords in the codebook. The ***Quantization Errors*** [2], the Euclidean distance to the nearest neighbor, can be utilized to measure the similarity of short-term behavior in input space and long-term behavior embedded in codebook.

4 Data Processing

Before AID process, it is necessary to do data preprocessing to extract the feature attributes from IP packets, and then, the Date Normalization will be processed to project whole feature attributes to a unit range no matter the continue attributes or quantitative attributes. In the paper, data preprocessing is focused on TCP traffic.

4.1 TCP Flow Feature Attribute

The extraction of feature attributes of network traffic is the foundation of machine learning algorithms in AID. Moreover, excellent detection models or algorithms must be combined with the rational feature vector extraction to improve the attack recognition capability. Traffic features should prefer to differentiate usual traffic profiles from anomaly traffic profiles. The aim of feature extraction is to achieve the maximum difference degree between usual usage behaviors and anomaly behaviors. A feature vector of the TCP traffic flow is shown in Table 3.

Table 3. Feature attributes of TCP Flow feature vector

Feature Attribute	Describe
SrcIP	source IP address
DestIP	destination IP address
SrcPort	source port
DestPort	destination port
PktSize	average packet size in one TCP Flow
SrcBytes	the number of bytes from source
DestBytes	the number of bytes from destination
FlowState	TCP Flow closed state
Fre_SrcIP	frequency of a certain source IP in time-window
Fre_DestIP	frequency of a certain destination IP in time-window

There are nine flags involved in the connection establishment of TCP 3-way handshake protocol and the connection close of TCP 4-way handshake protocol. We devised a 9-bit number to identify the connection state. The flag will be set to 1 if the corresponding flag is observed during the establishment-close process. Otherwise, the corresponding flag will set to 0. A decimal function with the non-superposed value is used to quantitate the whole connection process — $Sum(Flag_0, Flag_{1,\ldots\ldots,} Flag_8)$:

$$\begin{aligned} FlowState &= Sum(Flag_0, Flag_1, \ldots, Flag_8) = \sum_{i=0} Flag_i \cdot 2^i \\ &= RST_{active} \cdot 2^0 + RST_{passive} \cdot 2^1 + SYN \cdot 2^2 + ACK_{syn} \cdot 2^3 + ACK \cdot 2^4 \\ &+ FIN \cdot 2^5 + ACK_{fin} \cdot 2^6 + FIN' \cdot 2^7 + ACK'_{fin} \cdot 2^8 \end{aligned} \tag{8}$$

The TCP connection with the normal close is:

$$Sum = (111111100)_2 = (508)_{10}$$

For example: There are two statuses according to one SYN probing attack

(1) The target port is not open and the target computer responses a $\boldsymbol{RST}_{passive}$:

$$Sum = RST_{passive} \bullet 2^1 + SYN \bullet 2^2 = 1\times 2^1 + 1\times 2^2 = 6$$

(2) The target port is open and the target computer responses an $\boldsymbol{ACK}_{syn}$:

$$Sum = SYN \bullet 2^2 + ACK_{syn} \bullet 2^3 + RST_{active} \bullet 2^0 = 1\times 2^2 + 1\times 2^3 + 1 = 13$$

The number 6 and 13 describe two abnormal TCP connection states in the process of SYN probing attack. Consider the situation that TCP protocol has the re-transmission mechanism and a TCP Flow aggregates some TCP connections so that the certain flag will repeatedly in a TCP Flow, we substituted the 32-bit number (4 bytes) for the 9-bit number, as in Fig 3. So if the occurrence time of one certain flag is less than 15, the sum will not be repeated. The number of occurrence time will take value of 15 if it exceeds 15. ($RST_{passive}$/RST_{active} occurrence time is less than 3).

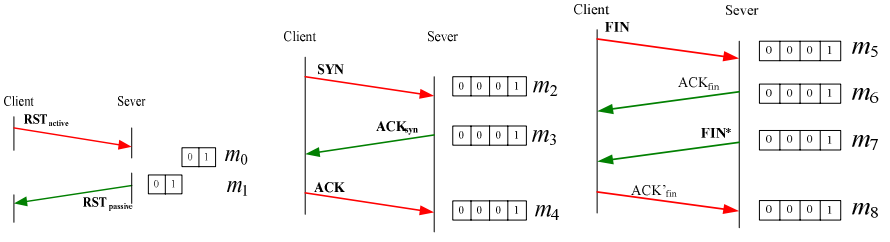


Fig. 3. Quantization of TCP Flow state

The reason that RST takes low-bit and ACK'_{fin} takes high-bit is to maximize the discrepancy between the normal connection closed (ACK_{fin}) and the connection rejected abnormally (RST). The large discrepancy between normal feature attribute and abnormal can help to advance AID accuracy.

The TCP Flow state is the most important feature attribute. Many attacks can result in the abnormal state of connection according to the TCP protocol. Generally, 14 connection-closing states are summarized in AID model based on the data mining [4, 5]. We found that the method of tracking 14 connection states is clumsy relatively in data preprocessing program. What is more important, these 14 states cannot include all the complicated instances of TCP connection state. By state quantization, every state of connection can be mapped to an int data range affording the state synopsis attribute directly leading to the whole improvement of feature vector.

4.2 Time Window

In order to get the correlation and statistical information in a certain time interval (2 seconds in this paper), a time window was designed as in Fig.4. The dashed denotes the flow began out of time window but ended in it while the real line denotes the Whole cirle of flow is in the time window. In the data preprocessing the flows closed in time window are counted no matter when did it began which is convenient to the traffic statistical correlation and the data preprocessing.

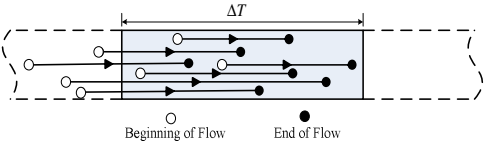


Fig. 4. Time window

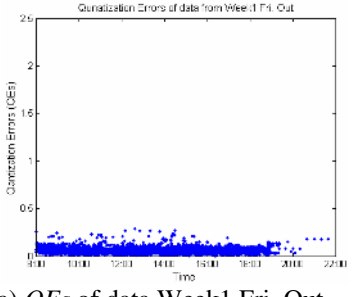
5 Evaluation Method and Result

5.1 Experiment Data Set

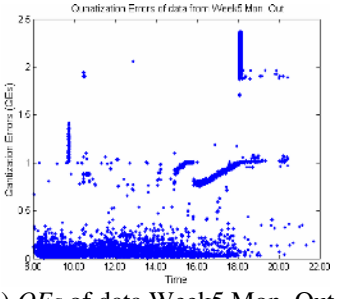
We take a part of the 1999 DARPA Intrusion Detection Evaluation Data Sets of [13, 14] to estimate the structural Vector Quantization for AID off line. The codebook of

Table 4. Attacks in evaluation

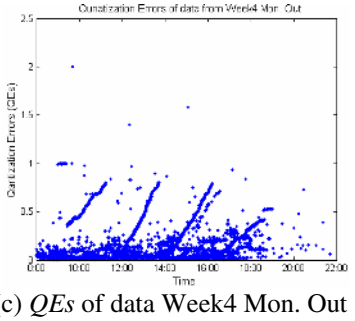
Back (DoS)	3	Ntinfoscan (Probing)	3
Selfping (Probing)	3	Apache2 (DoS)	3
Portsweep (Probing)	12	Queso (Probing)	3
Satan (Probing)	2	Neptune/SYN-flood (DoS)	3
Mscan (Probing)	1	Processtable (DoS)	2
Total	35		



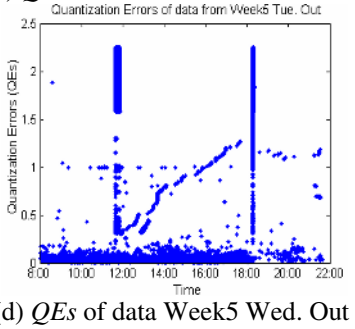
(a) *QEs* of data Week1 Fri. Out



(b) *QEs* of data Week5 Mon. Out



(c) *QEs* of data Week4 Mon. Out



(d) *QEs* of data Week5 Wed. Out

Fig. 5. *QE* distributions of the training data and testing data

Table 5. Attack detection rates for Week4 and Week5 attacks

Date	Attack ID	Attack Name	DR	QE	FPR
Week4 Mon	41.091531	Portsweep	1/3(33.3%)	1.3999	0.31%
	41.111531	Portsweep	5/5(100%)	0.9072~1.2328	
	41.122222	Portsweep	4/5(80%)	1.1001~1.3125	
	41.162715	Portsweep	0/10(0%)	0.6723~0.7213	
Week4 Wed	43.080401	Satan	7/16(43.8%)	0.7164~1.3196	0.11%
	43.164334	Portsweep	3/3 (100%)	1.1281~1.2947	
Week4 Thu	44.080000	Ntinfoscan	9/15 (60%)	0.8793~0.9539	0.14%
Week4 Fri	45.111010	Portsweep	0/8 (0%)	0.7234~0.7379	0.96%
	45.181011	Portsweep	5/5 (100%)	0.9012~1.2318	
Week5 Mon	51.084334	Portsweep	3/3 (100%)	1.0417~1.2256	3.15%
	51.094334	Portsweep	90/106 (84.9%)	0.8039~1.3017	
	51.102700	Apache2	948/1014 (93.5%)	0.8769~0.8972	
	51.140100	Apache2	11/12 (91.7%)	0.8592~0.8874	
	51.180445	Neptune	19731/20480 (96.3%)	1.8038~2.3063	
	51.201715	Selfping	1/1 (100%)	1.0218	
Week5 Tue	52.094514	Selfping	0/1 (0%)	0.7288	3.25%
	52.103409	Back	47/47(100%)	0.8254~2.0578	
	52.113855	Neptune	40950/40960 (99.9%)	1.9088~2.3583	
	52.165435	Queso	7/7 (100%)	1.0017~1.0021	
	52.181637	Neptune	950/1024 (92.7%)	1.9075~2.3572	
Week5 Wed	53.045454	Selfping	5/5 (100%)	0.9059~1.0134	0.37%
	53.102617	Back	40/40 (100%)	0.8906~0.8941	
	53.110516	Queso	4/7 (57.1%)	1.0218~1.0431	
	53.123735	Portsweep	8/13 (61.5%)	1.0047~1.2766	
	53.134015	Queso	4/7 (57.1%)	0.8084~0.8507	
	53.150110	Processt-able	124/375(33.1%)	0.8821~0.8927	
	53.152648	Back	0/40 (0%)	0.5679~0.7221	
	53.171350	Apache2	155/340(45.6%)	0.8832~0.8959	
	53.195130	Portsweep	100/100 (100%)	1.0413~1.6032	
Week5 Thu	54.103459	Portsweep	3/3 (100%)	1.0585~1.3623	0.67%
	54.110416	Ntinfoscan	5/16 (100%)	0.8243~0.8448	
	54.145832	Satan	8817/9120 (96.7%)	0.8023~1.9823	
	54.183002	Ntinfoscan	6/17 (37.3%)	0.8121~0.8528	
	54.195951	Mscan	5724/5724 (100%)	1.0071~1.4995	

(The threshold of *QE* is 0.8)

VQ is designed on the data set attack free in week 3 and week 1. However, consider the fact that the target of this paper is network work attack based on TCP and not general, we test for ten TCP attacks（DoS and Probing mainly）, 35 instance attacks total, as showed in Table 4. We filter out some other attacks out of the test traffic data according to the attack identification [13] of 1999 DARPA deliberately after the feature vectors extracted. A detailed description of these attacks could be found in [13].

5.2 Quantization Errors and Result

We use ***Quantization Errors*** (*QEs*) to evaluate the on-detecting network traffic vectors. Fig.5 (a-d) presents the *QEs* distributions in outside traffic of 4 days in DARPA data set including one the training data Week1 Fri. and the three-day testing data. The DARPA data of week1 Fri. begins in 8:00 morning and ends about in 22:08 evening (22:04 about in week5 Mon.) after which few data can be observed in those two days. [13]

From the Fig.5 (a), we can observe that *QEs* in training data is well regulated and don't change with the large deviation. However, the strong contrast in Fig.5 (b) is the sharp variation of *QEs* due to the high values of attack traffic *QEs*. Markedly, on 18:04, Neptune attack (Syn-flood) can be viewed with *QEs* values ranging from 1.8038 to 2.3063 in Fig.5 (b). The same case happens in other day data which just can be viewed by Fig.5 (c) and Fig.5 (d).

The overall outcome of evaluation is present in Table 5. The Detection Rate (DR) and False Positive Rate (FPR) are both presented for the whole day circumstances accordingly. Individuals in one attack circumstance are given singly for the better understand. Obviously, every attack with certain ID is composed of the attack multi-flows, from a few to the huge volume such as Neptune because many DoS/DDoS and Probing attacks behave the style of the bursty network traffic. These bursty attacks usually manoeuvre the huge of network traffic to attack computer severs. The single or a few of TCP flows usually are used to explore the service information of some severs, for instance, if the port of 8080 is open in severs. We can conclude that our intrusion detection method exhibit more robust to these bursty attacks with the huge multi-flows and get the higher DR with the lower FPR.

6 Conclusions

The paper proposed an intrusion detection mechanism using the structural Vector Quantization, especially to detect DoS/DDoS and Probing attacks for web services. With the codebook, the normal usage profile of the temporal-spatial scale network traffic could be constructed quantitatively to describe the normal long-term behaviors. The evaluation experiments confirmed that our anomaly intrusion detection framework can achieve the higher detection rate with the lower false detection rate.

References

1. Moore D., Voelker G., and Savage S., :Inferring Internet Denial-of-Service Activity, in Usenix Security Symposium, Washington, D.C., (2001) 401-414
2. Robert Gray and David L. Neuhoff: Quantization. IEEE Transactions on Information Theory, Vol. 44, (1998) 2325-2384

3. D. E. Denning: An Intrusion-detection Model. IEEE Transactions on Software Engineering, Vol. 13(2), (1987) 222-232
4. E. Eskin, A. Arnold, M. Prerau: A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data. Applications of Data Mining in Computer Security, Kluwer, 2002
5. Andrew H. Sung, Srinivas Mukkamala: Identifying Important Features for Intrusion Detection Using Support Vector Machines and Neural Networks. Proceedings of the 2003 Symposium on Applications and the Internet, (2003) 119-123
6. Y. Qiao, X. W. Xin, Y. Bin and S. Ge:Anomaly Intrusion Detection Method Based on HMM, Electronics Letters, 38(13), (2002) 663-664
7. J. M. Bonifaco, E. S. Moreira: An Adaptive Intrusion Detection System Using Neural Network, Research Report, UNESP, Brazil, 1997
8. http://www.snort.org
9. Linde, Y., Buzo, A. and Gray, R. M.: An Algorithm for Vector Quantizer Design. IEEE Transactions on Communications, 28(1), (1980) 84–95
10. Ueda, N. and Nakano, R.: A New Competitive Learning Approach Based on an Equidistortion Principle for Designing Optimal Vector Quantizers. IEEE Transactions on Neural Networks, 7(8), (1994) 1211–1227
11. Kohonen, T.: Self-Organization Maps , 3rd ed., Springer-Verlag, Berlin, 1997
12. Tom Mitchell: Machine Learning, McGraw Hill, New York, 1997
13. http://www.ll.mit.edu/IST/ideval/index.html
14. Richard Lippmann, Joshua W. Haines, David J. Fried, Jonathan Korba, Kumar Das. :The 1999 DARPA Off-Line Intrusion Detection Evaluation, Computer Networks, 34 (4), (2000) 579-595