A Parameterless Learning Algorithm for Behavior-Based Detection

Wang, Can
Department of Informatics, Kyushu University | Institute of Systems, Information Technologies and Nanotechnologies

Feng, Yaokai
Department of Advanced Information Technology, Kyushu University | Institute of Systems, Information Technologies and Nanotechnologies

Kawamoto, Junpei
Department of Informatics, Kyushu University | Institute of Systems, Information Technologies and Nanotechnologie

Hori, Yoshiaki
Organization for General Education, Saga University | Institute of Systems, Information Technologies and Nanotechnologies

他

https://hdl.handle.net/2324/1498300

A Parameterless Learning Algorithm
for Behavior-based Detection

Can Wang

Department of Informatics
Kyushu University
Institute of Systems, Information Technologies and
Nanotechnologies
Fukuoka, Japan
wangcan@itslab.inf.kyushu-u.ac.jp

Yaokai Feng

Department of Advanced Information Technology
Kyushu University
Institute of Systems, Information Technologies and
Nanotechnologies
Fukuoka, Japan
fengyk@ait.kyushu-u.ac.jp

Junpei Kawamoto

Department of Informatics
Kyushu University
Institute of Systems, Information
Technologies and Nanotechnologies
Fukuoka, Japan
kawamoto@inf.kyushu-u.ac.jp

Yoshiaki Hori

Organization for General Education
Saga University
Institute of Systems, Information
Technologies and Nanotechnologies
Saga, Japan
horiyo@cc.saga-u.ac.jp

Kouichi Sakurai

Department of Informatics
Kyushu University
Institute of Systems, Information
Technologies and Nanotechnologies
Fukuoka, Japan
sakurai@csce.kyushu-u.ac.jp

Abstract—The frequency and the extent of damages caused by network attacks have been actually increasing greatly in recent years, although many approaches to avoiding and detecting attacks have been proposed in the community of network security. Thus, how to fast detect actual or potential attacks has become an urgent issue. Among the detection strategies, behavior-based ones, which use normal access patterns learned from reference data (e.g., history traffic) to detect new attacks, have attracted attention from many researchers. In each of all such strategies, a learning algorithm is necessary and plays a key role. Obviously, whether the learning algorithm can extract the normal behavior modes properly or not directly influence the detection result. However, some parameters have to determine in advance in the existing learning algorithms, which is not easy, even not feasible, in many actual applications. For example, even in the newest learning algorithm, which called FHST learning algorithm in this study, two parameters are used and they are difficult to be determined in advance. In this study, we propose a parameterless learning algorithm for the first time, in which no parameters are used. The efficiency of our proposal is verified by experiment. Although the proposed learning algorithm in this study is designed for detecting port scans, it is obviously able to be used to other behavior-based detections.

Keywords— attack detection, behavior-based detection, learning algorithm, port scan, network security

I. INTRODUCTION

Many users of the Internet have realized the importance of preventing their computers from being attacked. According to the Ministry of Internal Affairs and Communications of Japan, about 79.5 % of the population of Japan were using the internet in 2012, of which 72.2% are anxious for preventing their computers from network attacks. Such users include many companies, universities, governments and other organizations[1]. Although many approaches to avoiding and detecting internet attacks have been and are being proposed in the community of network security, the frequency and the extent of damages caused by network attacks have been actually increasing greatly in recent years. Thus, how to fast and efficiently detect actual and potential keep-changing attacks is still a big challenge.
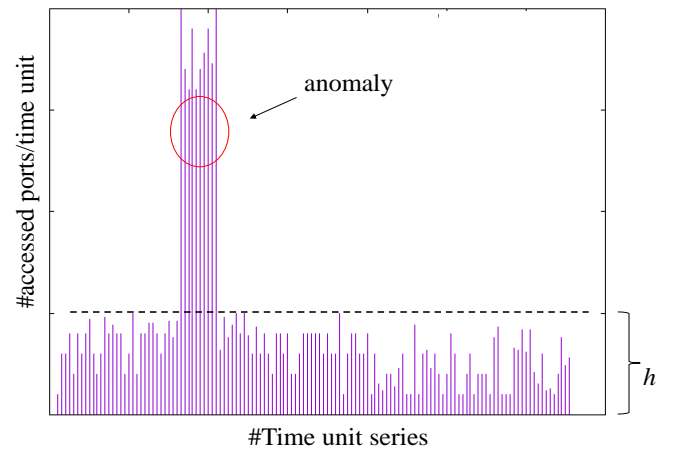

Fig. 1. Example of behavior-based method.

Behavior-based methods have been discussed first in the work [8]. Fig. 1 shows an example of behavior-based methods. In this example, the horizontal axis represents the time while the vertical axis denotes, for example, the number of accessed ports in each time unit. h can be regarded as the normal range of the number of ports accessed in a single time unit. However, the traffic in the red circle is seemingly an anomaly. If we can know the normal range h (called normal behavior mode in this study) from the history traffic data over a relatively long time period, then detection of anomalies becomes easy with the help of this normal behavior mode. Normal behavior modes (the range h in Fig. 1) can be obtained if we have sufficient clean learning data. However, this is often not feasible for many real situations. Thus, a learning algorithm is necessary that can extract the normal behavior mode from a training dataset possibly containing anomaly data. The learning algorithm plays a core role in behavior-based detection methods. This is because that, after a proper normal mode has been extracted, the anomaly detection means only a comparison between the count result in the current time unit and the normal behavior mode. According to our investigations, the work [9] proposed the newest learning algorithm which is called FHST learning algorithm in this study. That algorithm will be introduced briefly in Section III.

Behavior-based methods have many advantages including the extracted normal behavior modes can reflect the actual features of the specific networks and the normal behavior modes can be upgraded automatically and repeatedly to follow the real situations. In addition, multiple normal modes can be extracted even for the same network, responding to different situations (e.g., weekdays and weekends).

Because of these advantages, behavior-based detection methods have attracted attention from many researchers [9]. In each of all such strategies, a learning algorithm that extracts the normal behavior modes from reference data is necessary and plays a key role. However, some parameters have to be determined in advance in the existing learning algorithms, which is not easy, even not feasible, in many actual applications. Even in the newest learning algorithm proposed in the work [9], which is called FHST algorithm in this study, the two parameters are difficult to be determined in advance.

In this study, we propose a parameterless learning algorithm for the first time. The explanation and experiment result indicate that our proposal can extract a reasonable normal behavior mode from the reference traffic data (history data in this study), although it does not use any parameters. We want to note that, although the proposed learning algorithm is designed for port scan detection in this study, obviously, it is also able to be used to other behavior-based detections. In other words, in this study, port scan attacks are detected as an example using the proposed learning algorithm. For detection of a different kind of attacks, the input of this algorithm (a frequency distribution histogram) may be built in a different way.

A port scan can be defined as an action that sends client requests to a range of server port that addresses on a host or multiple hosts, with the goal of finding an active port and exploiting a known vulnerability of that service [9]. That is, attackers usually conduct port scans to collect vulnerabilities of

Table 1. Behavior-based Detection of Port Scan.

1. Collecting the training data
2. Counting the number of accessed ports in each time unit (see Fig. 1).
3. Drawing the frequency distribution of the number of accessed different ports
4. Extracting the normal behavior mode from the frequency distribution obtained in Step 3, using a learning algorithm
5. Anomaly detection

the targets before starting an actual attacks. By scanning the ports on the target, attackers can determine the type of OS and the application software running on the targets, and examining whether vulnerable ports exist or not. If a security hole is found, actual attacks will be conducted. Thus it is very important for system administrators and other network defenders to detect port scans as possible preliminaries to a more serious attack [3, 4, 5, 6].

Generally, there are the following four types of port scans [9]. 1) Vertical Scan: To scan one or multiple ports of a host from a single source IP address. 2) Horizontal Scan: To scan one vulnerable port of multiple hosts from a single source IP address. 3) Distributed Vertical Scan: To scan one or multiple ports from multiple source IP addresses. 4) Distributed Horizontal Scan: To scan one port of multiple hosts from multiple source IP addresses.

The last two kinds of scan attacks are related to collaborative attacks, which is referred to as next generation cyber-attacks [5]. In order to introduce our novel learning algorithm, the 3rd kind of port scan is taken as example in this study. Thus, henceforth in this study, the term port scan refers to the distributed vertical port scan.

There have been many researches on how to detect port scans [3, 4, 5, 6]. Almost all of them uses threshold values to decide abnormalities. That is, it will be treated as a port scan and an alarm will be generated if the number of ports accessed in a time unit exceeds the given threshold. However, the thresholds are often difficult to determine in advance in real applications and the proper thresholds may change in different situations even for the same network. In the work [9], a learning algorithm is proposed to extract a threshold automatically from history traffic data. The problem is that, in that learning algorithm, two parameters are used and they are not easy to determine. In this study, a parameterless learning algorithm is proposed, which can extract a proper normal behavior mode without using any parameters. The normal behavior mode can be used as a threshold to detect actual attacks.

This paper is organized as follows. After explaining how to realize behavior-based detection of port scans in Section II, the FHST learning algorithm proposed in the work [9] is briefly introduced in Section III. Section IV is our main contribution - a parameterless learning algorithm. Then, in Section V we present our experimental results. Finally, we state our conclusion in Section VI.

II. BEHAVIOR-BASED DETECTION OF PORT SCAN

The proposed process in this paper for behavior-based detection of port scan is shown in Table 1. After training data has been collected in Step 1, the number of accessed ports in each time unit is counted. Then, a frequency distribution of the different accessed ports is built in Step 3. In Step 4, using a learning algorithm, the normal behavior mode is extracted from the frequency distribution built in Step 3. At last, as mentioned above, anomaly detection can be easily conducted after the normal behavior mode has been exacted properly in Step 4.

Since Steps 1,2 are straightforward and step 5 is also simple as mentioned above and Step 4 will be explained in details in Section IV as our main contribution. Thus, only Steps 3 is explained here.

For extracting the normal behavior mode, a frequency distribution of the number of accessed different ports in one time unit (see Fig. 2 for an example) is created as follows. The
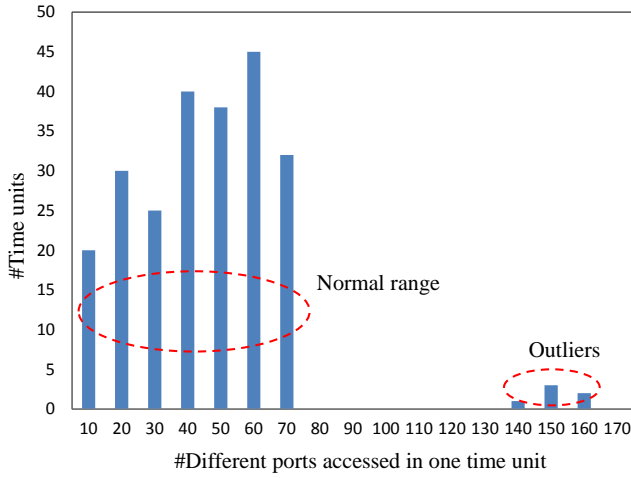

Fig. 2. Frequency Distribution of the number of different accessed ports in one time unit.
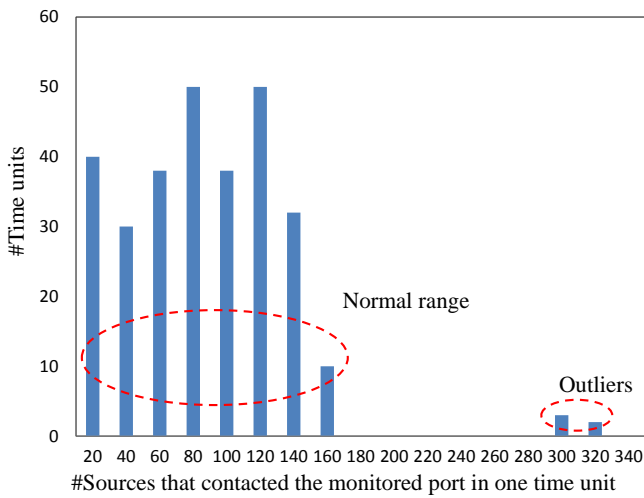

Fig.3. The frequency distribution and the behavior mode in work [9].

total range of the number of accessed ports is divided into bins (the bin-width is a parameter), which is shown in the x-axis. Note that, as an example, the right-most bin in Fig. 2 corresponds to ">170". Then, the number of time units are counted for each bin and shown as the vertical axis. For example, if the vertical axis value is 49 corresponding to the bin whose range is from 40 to 50 at the horizontal axis, then it means that, during the whole learning time period, there are 49 time units in each of which the number of accessed different ports is located in between 40 and 50.

III. THE FHST LEARNING ALGORITHM

As mentioned above, the work [9] proposed the newest learning algorithm, which is called by FHST learning algorithm in this study, for extracting the normal behavior mode from a frequency distribution (an example is shown in Fig.3) of the number of source IPs that connected the monitored port is shown in Fig. 3. Note that, the work [9] is for detecting different attacks from this study. Thus, Fig. 3 is built

Table 2: The FHST Learning Algorithm (From [9]).

Input: Frequency Distribution of the number of sources that accessed one port in one time unit	
Output: Normal behavior mode	

Steps	Descriptions
1	- The bins are checked one by one starting from the rightmost bin in the frequency distribution (see Fig. 3). - The checked bins are placed in Ω - Let d_n be the distance from the bin that was just checked, to the next bin. If there is no next bin, use the distance from the current bin to the y-axis as d_n

2	Check the next bin if it exists.
	 If $((d_n > \alpha^{1)}))$ and (the area$^{2)}$) in Ω is less than $\beta\%^{3)}$ of the total area)) then
	      the bins in Ω are regarded as outliers and are discarded
	      go to step 1 // to find other outliers \\
	 Else
	      put the current bin in Ω
	 go to step 2 // not finished

1) Here α is a threshold.

2) The area denotes the number of time units.

3) β is another threshold, used to avoid the case where most of the bins are regarded as outliers.

Note that: d_n is large enough (e.g., larger than 2α) then β is ignored, which means that if a bin cluster is far enough from the y-axis, it will be regarded as an anomaly even if this cluster has a large enough area.

in a different way from Fig.2. The difference is in the definition of the horizontal axis. In Fig. 3, the meaning of the horizontal axis is "the number of sources that contact the goal in one time unit (see Fig. 3), however in Fig. 2 the horizontal axis means "the number of different ports accessed in one time unit". Anyway, the purpose of the learning algorithm in the work [9] is the same as that in this study in the meaning that they both try to find the normal range from a frequency distribution.

The learning algorithm in the work [9] is as follows. In the example shown in Fig. 3, the small bins located at the right far from the largest group tend to be regarded as outliers, which obviously should be discarded. After all the outliers have been removed, the range of the remaining bins on the x-axis is regarded as the normal range of the number of sources in one time unit. That is, the learning algorithm can obtain the normal mode from the frequency distribution (built from history traffic data) of the number of different sources that contact the monitored port in one time unit. During the detection, if the number of sources in the current time unit exceeds this normal mode, an alert is given. The learning algorithm proposed in work [9] is shown in Table 2.

From Table 2, we know that this learning algorithm checks all the bin from right to left and the two parameters (thresholds) α and β are used to determine if a bin or bin-group is outlier. α is a threshold on the area (total value of a bin-group) and β is a threshold on distance from the current bin to its left-neighboring bin. That is, if the current bin or bin-group has small enough area and big enough distance from its left neighbor bin, then the current bin or bin-group will be regarded as outlier. Obviously, these two thresholds are very important since they decide the result of learning phase.

However, how to properly determine these two parameters is obviously a difficult issue in many actual applications. In this study, we propose a parameterless learning algorithm, in which no parameters are used.

IV. OUR PARAMETERLESS LEARNING ALGORITHM

4.1 Algorithm

As mentioned above, the purpose of the learning algorithm is in this study to extract the normal behavior mode from the frequency distribution of the number of accessed different ports, an example is shown in Fig. 2. In this example, the right-most three bins tend to be removed as outliers. After that the normal range can be found. Our proposed parameterless learning algorithm is shown in Table 3.

Note that, in the algorithm shown in Table 3, we assume that all bins in the frequency distribution are divided into some groups by zero-bins. If only one bin-group exists then the end position of this bin group is reported as learning result (see Step 1), which means no bins can be removed. Area of a bin-group in this study means the summation of the values (y-axis) of all the bins in this bin-group.

Table 3. Our Proposed Learning algorithm

Steps	Descriptions
Input: Frequency Distribution of the number of accessed different ports in one time unit	
Output: Normal behavior mode	
Initializing	**Left_Pointer**: pointing to endpoint of the left-most 1st bin-group **Right_Pointer**: pointing to the end point of the right-most bin-group **Span**: the difference between the right-most and the left-most bins **Total_area**: summation of all bins **Dist_right**: the distance between the Left_Pointer and its right-neighboring bin-group **Dist_left**: the distance between Right_Pointer and its left-neighboring bin-group **Area_right**: the area of the Left_Pointer's right-neighboring bin-group **Area_left**: the area of Right_Pointer's left-neighboring bin-group
Step 1 Special case	If the frequency distribution contains only one bin group then, the end position of this bin group is reported as learning result
Step 2 Move Right_Pointer to left	While (Dist_left/Span > Area_left/Total_area) Move Right_Pointer to the end of the left-neighboring bin-group
Step 3 Move Left_Pointer to right	While (Area_right/Total_area $\geq$ Dist_right/Span) Move Left_Pointer to the start of the next right- hand zero-bin-group

Step 4
 	
 Finish or cir-culate
	If (Left_Pointer=Right_point)
	
	 learning result=Left_Pointer
	
	 Else
	
	 1) Combine the left-neighboring and the right-neighboring bin-groups of Right_Pointer
	
	 2) Let Right_Pointer point to the end of the new bin-group
	
	 3) Go to Step 1.

4.2 Rough Idea

In this algorithm, two pointers of the Right_Pointer and the Left_Pointer are used, which initially point, respectively, at the end positions of the right-most bin-group and the left-most bin-group of the frequency distribution (see Fig. 2 for an example). The Left_Pointer moves to right by one step (jump one bin-group) if a certain condition is satisfied. The Right_Pointer moves to left by one step (jump one bin group) if a certain Condition is satisfied. The bins jumped by the Left_Pointer are regarded as normal ones. And the bins jumped by the Right_Pointer are regarded as abnormal ones. The movement conditions of the Left_Pointer and the Right_Pointer will be discussed later.

The above movement of the two pointers are repeated. When the two pointers meets each other, the position of the two pointers is reported as the final result. If both of the two pointers cannot move further and there are still some bins between them, then the two left-neighboring bin-groups of the Right_Pointer are merged into a new bin-group. After that, the algorithm will be repeated again. The convergence of this algorithm can be guaranteed, which will be discussed later.

Note that, if the frequency distribution contains only one bin group then, the end position of this bin group is reported as learning result. This means that no outliers exist if all the bins are distributed continuously. See Step 1 in this algorithm.

In this algorithm, bin-group is the minimum unit by which the Left_Pointer and the Right_Pointer are moved. In other words, the Left_Pointer may only point at the end point of some bin-group when it moves from left to right. And the Right_Pointer also only point at the end point of some bin-group when it moves from right to left.

4.3 Movement Conditions of Right_Pointer and Left_Pointer

In Step 2, the movement condition of the Right_Pointer is "Dist_left/Span > Area_left/Total_area", which means that, when we try to decide whether or not the left-neighboring bin-group of the Right_Pointer should be regarded as outliers, the ratio of the area of this bin-group to the total bins, which means how heavy it is, and the ratio of the distance of this group from its left-neighboring bin-group to the whole span, which means how far this bin-group separates from its left-neighboring bin, are checked and compared with each other. That is, the smaller its area is and the farther it is from its left-neighboring bin, the more the bin-group tends to be regarded as outliers and be skipped.

The movement condition of the Left_Pointer is "Area_right/Total_area ≥ Dist_right/Span" (see Step 3 of this algorithm), which means that, when we try to decide whether or not the right-neighboring bin-group of the Left_Pointer should be included in the normal mode, the ratio of the area of this group to the total bins, which means how heavy it is, and the ratio of the distance of this bin-group from its right-neighboring bin-group to the whole span, which means how far it separates from its right-neighboring bin, are checked and compared with each other. That is, the bigger its area is and the nearer it is from its right-neighboring bin-group, the more it tends to be included in the normal mode (normal range).

4.4 Finish or Repeat

As mentioned above, the movements of the two pointers are repeated until they cannot move further. At that time, if the two pointers points at the same place, that place is reported as the final learning result. If there still exist some bins between the two pointers, the final result has not been decided yet. In that case, the two left-neighboring bin-groups of the Right_Pointer are combined into a new bigger bin group. Then, the whole process will repeated again. The convergence of this algorithm can be guaranteed, which will be discussed later.

4.5 Convergence

As mentioned in Step 4, the algorithm is possibly repeated. Thus, the convergence of this algorithm should be guaranteed. If the final result couldn't be obtained at the current round, the bin groups will be merged as mentioned in Step 4. In each round, two bin groups are merged. Thus, as the worst case, there are only one bin-group left between the two pointers as the merge operation is conducted continuously. In that case, the Dist_left=the Dist_right and Area_left = Area_right. Thus, either of the Right_Pointer and the Left_Pointer must move further. This is because that one of the two movement conditions must be met. After that movement, the two pointers will point at the same location, where will be the final learning result.

V. EXPERIMENT

5.1 Parameters for Building Frequency Distribution

Two parameters, time unit and bin-width, are necessary when building the frequency distribution. The former one can be determined according to the scale of the real traffic and how quickly we want to find the attacks. The bin-width can be decided according to how accurately we want to learn the normal mode, which actually has not large influence on the final detection result, according to our investigation, if the number of accessed different ports in one time unit changes a lot during attacks. Note that, these two parameters do not belong to the learning algorithm because the frequency distribution has to be built before the learning algorithm is used. That is, the frequency distribution built using the above two parameters is the input of the learning algorithm.

Our experiment is conducted with the time units of 10, 30 and 60 minutes, respectively, and with the bin-widths of 250, 500, 750, 1000, 2000, 3000. However, only the results with the bin-width of 500, 1000 are presented here because we found that the bin-width does not influence much the learning result according to our many investigations our many experiment results.

5.2 Experiment data

The data used in this study are collected from a darknet in June, 2011, provided by National Institute of Information and Communications Technology (NICT), Japan. The traffic data in June are used as training data. That is, all the frequency distributions of the number of accessed different ports are built using the traffic data of June, 2011.

It is well-known that a common difficulty in network security researches is the fact that real traffic data in many companies or other organizations are often not available for researchers. Fortunately, it has been confirmed by many studies [12,13,14,15,16] that global trends of network threats can be observed by monitoring darknets. A darknet is a set of unused IP addresses [17]. Obviously, there are no actual services (web, mail, etc.) in darknets since these addresses have not been distributed to any legal users. Thus, except missconfigures in the sources, all the traffics coming to darknets are regarded as anomalies [13].

5.3 Experiment result

Fig. 4 is the frequency distribution of the number of accessed different ports, where the time unit is 10 minutes and the bin-width is 500. In this case, the learning result of our proposed learning algorithm is 44. Thus, the normal mode is 44*500 = 22000.

Fig. 5 is the frequency distribution, where the time unit is 10 minutes and the bin-width is 1000. In this case, the learning result of our proposed learning algorithm is 22. Thus, the normal mode is 22*1000 = 22000.

Fig. 6 is the frequency distribution, where the time unit is 30 minutes and the bin-width is 500. In this case, the learning result of our proposed learning algorithm is 81. Thus, the normal mode is 81*500 =40500.

Fig. 7 is the frequency distribution, where the time unit is 30 minutes and the bin-width is 1000. In this case, the learning result of our proposed learning algorithm is 45. Thus, the normal mode is 45*1000 =45000.

Fig. 5．Frequency distribution: time unites=10 minutes, bin width of bin=1000.

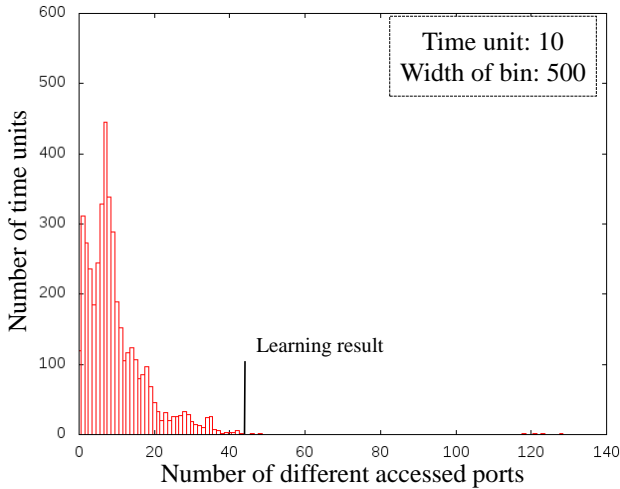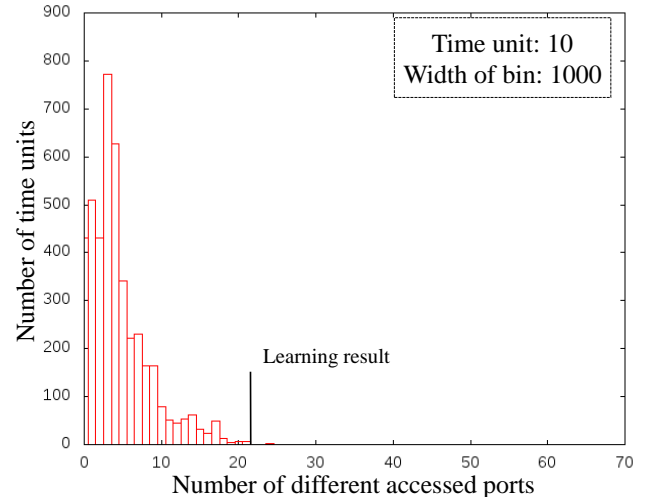Fig. 6．Frequency distribution: time unites=30 minutes, width of bin=500.
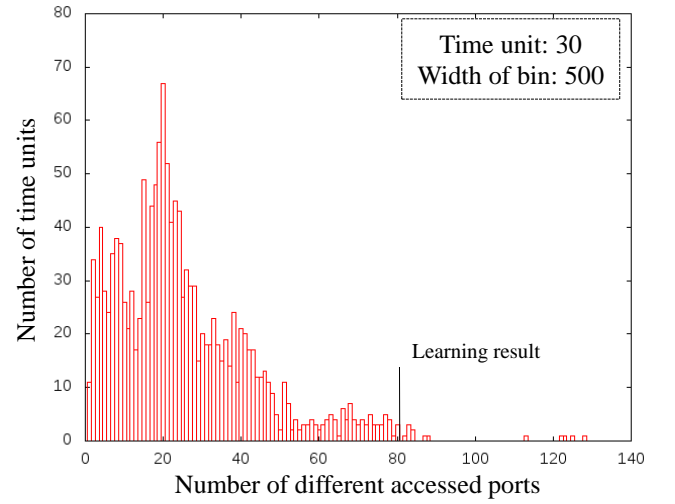

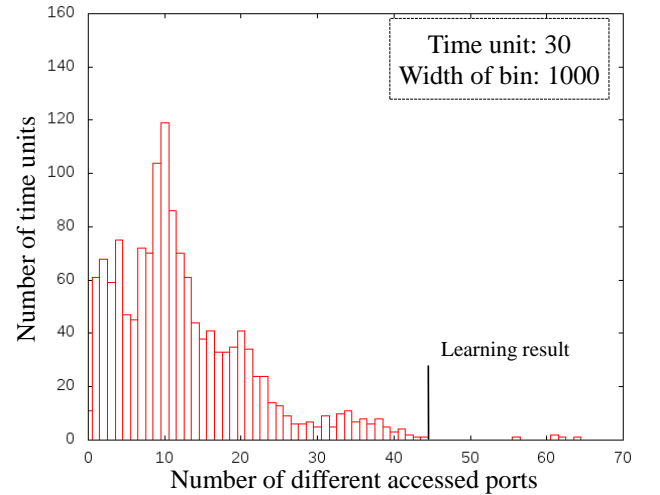Fig. 4．Frequency distribution: time unites=10 minutes, width of bin=500.

Fig. 7．Frequency distribution: time unites=30 minutes, width of bin=1000.

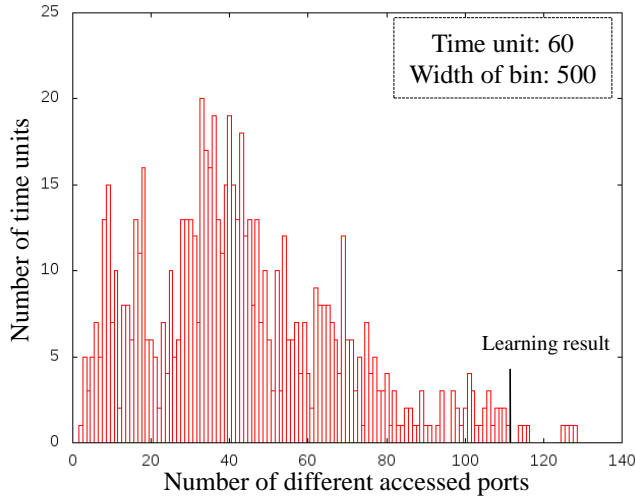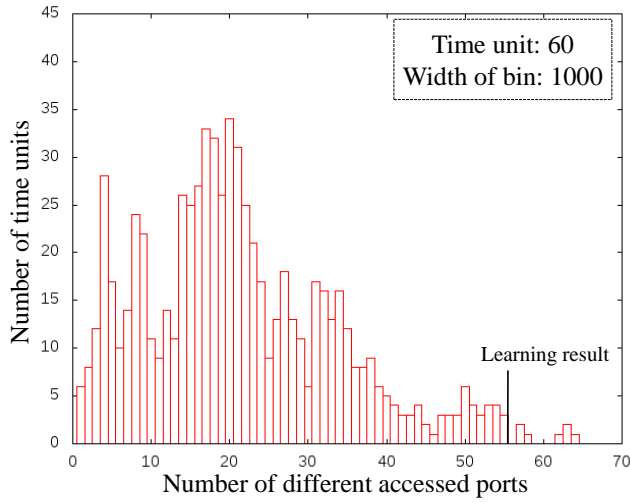Fig. 8．Frequency distribution: time unites=60 minutes, width of bin=500.

Fig. 9．Frequency distribution: time unites=60 minutes, width of bin=1000.

Fig. 8 is the frequency distribution, where the time unit is 60 minutes and the bin-width is 500. In this case, the learning result of our proposed learning algorithm is 112. Thus, the normal mode is 112*500 =56000．

Fig. 9 is the frequency distribution, where the time unit is 60 minutes and the bin-width is 1000. In this case, the learning result of our proposed learning algorithm is 81. Thus, the normal mode is 56*1000 =56000．

5.4 Observations

From the above experiment results shown in Figs 4~9, we can observe that

1) As the time unit increases, the normal mode learned by the learning algorithm also becomes bigger, which is easy to understand. The reason for this is that, the number of accessed different ports in one time unit certainly increases as the time unit increases.

2) For a fixed time unit, the learned normal modes possibly vary a little for different bin-widths. See the learning results in Figs 6 and 7 as examples, although the normal mode does not vary as the bin-widths changes in the other experiments. This is because of the fact that a bigger bin-width may mean a smaller number of zero-bins in the frequency distribution, which will influence the final learning result. Anyway, according to our investigations, the difference is not big, which will not influence the result of detection. This is because that the number of accessed different ports will increase much for real port scan attacks.

3) From the learning results corresponding to Figs 4~9, we can observe that the learning results are reasonable. That is, even determined by ourselves (not by algorithm), these results are also perhaps selected. Again, it does not influence detection result even the learning result actually influences a little.

VI. CONCLUSION AND FUTURE WORK

This paper pointed out that learning algorithm extracting normal behavior modes plays a key role in behavior-based detection methods and the parameters is difficult to determine in advance in the existing learning algorithms. In this study, we proposed a novel learning algorithm, which does not need any parameters. The experiment result shows that our algorithm works well. Although the proposed learning algorithm in this study is explained based on detection of distributed vertical port scans, it is obviously able to be used to other behavior-based detections, for example, the detection in the work [9]. The only difference is the way of building the frequency distribution. That is, the definition of the axes may varies for different applications.In the future, we will verify the performance of this parameterless learning algorithm in situations of other kinds of attacks, including other kinds of port scans. And evaluate our method with other learning algorithm which need parameters.

REFERENCES

[1] http://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h25/, Ministry of Internal Affairs and Communications, Japan.

[2] RFC 2828 Internet Security Glossary

[3] M.H.Bhuyan, D.K.Bhattacharyya and J.K.kalita, "Surveying Port Scans and Their Detection Methodologies," The Computer Journal, 2011, VOL 54, NO.10, pp. 1565-1581.

[4] Y. Chabchoub, C. Fricker, P. Robert, "Improving the detection of online vertical port scan in IP traffic", 7th International Conference on Risks and Security of Internet and Systems, Vol. 10, No. 1-2 , pp.1–6, 2012.

[5] J. Gadge,J and A. A. Patil, "Port scan detection" 16th IEEE International Conference on Networks (ICON), pp. 1-6, 2008.

[6] R Ensafi, JC Park, D Kapur, JR Crandall, "Idle Port Scanning and Non-interference Analysis of Network Protocol Stacks Using Model Checking", USENIX Security'10 Proceedings of the 19th USENIX conference on Security, pp. 17--33, 2010.

[7] S. Xu, "Collaborative Attack vs. Collaborative Defense", 4th International Conferenceon Collaborative Computing (CollaborateCom2008), LNICST 10}, pp. 217—228, 2009.

[8] D. E. Denning, "An Intrusion-Detection Model", IEEE Transactions on Software Engineering - Special issue on computer security and privacy, Vol. 13 No. 2, pp. 222-232, 1987.

[9] Y. Feng, Y. Hori, K. Sakurai, J. Takeuchi, "A Behavior-Based Method for Detecting Distributed Scan Attacks in Darknets", Journal of Information Processing, Vol 21, No. 3, pp. 527-538, 2013.

[10] M.Dabbagh, A.Ghandour, K.Fawaz, W.EL.Hajj and H.Hajj, "Slow Port Scanning Detection", International Conference on Information Assurance and Security (IAS), pp. 228-344, 7th, 2011.

[11] Joanne Treurniet, "A Network Activity Classification Schema and Its Application to Scan Detection", IEEE/ACM TON, VOL 19, NO.5, 2011, pp. 1396-1404.

[12] S. Akimoto, Y. Hori, and K.Sakurai, "Collaborative Behavior Visualization and its Detection by Observing Darknet Traffic", Proc. the 4th International Symposium on Cyberspace Safety and Security (CSS), LNCS 7672, pp. 212-226, 2012.

[13] M. Eto, D. Inoue, J. Song, K.Ohtaka, and K. Nakao, "Nicter: A Large-Scale Network Incident Analysis System", Proc. the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS), pp. 37-45, 2011.

[14] M. Bailey, E. Cooke, F. Jahanian, J. Nazario, and D. Watson, "The Internet Motion Sensor: A distributed blackhole monitoring system", Proc. the 12th ISOC Symposium on Network and Distributed Systems Security (NDSS), pp. 167–179, 2005.

[15] SANS Internet Storm Center. http://isc.sans.org/.

[16] F. Pouget, M. Dacier, and V.H. Pham, "Leurre.com: On the Advantages of Deploying a Large Scale Distributed Honeypot Platform", In E-Crime and Computer Conference (ECCE), 2005.

[17] E. Cooke, M. Bailey, Z.M.Mao, D. Watson, F.Jahanian, and D. McPherson, "Toward Understanding Distributed Blackhole Placement", Proc. ACM CCS workshop on Rapid Malcode, pp. 54-64, ACM Press, 2004.