Port Scan Behavior Diagnosis by Clustering

Lanjia Wang[1], Haixin Duan[2], and Xing Li[1]

[1] Department of Electronic Engineering, Tsinghua University, Beijing, 100084, P.R. China
wanglanjia@ns.6test.edu.cn
xing@cernet.edu.cn

[2] Network Research Center, Tsinghua University, Beijing, 100084, P.R. China
dhx@cernet.edu.cn

Abstract. Detecting and identifying port scans is important for tracking malicious activities at early stage. The previous work mainly focuses on detecting individual scanners, while cares little about their common scan patterns that may imply important security threats against network. In this paper we propose a *scan vector model*, in which a scanner is represented by a vector that combines different scan features online, such as target ports and scan rate. A center-based clustering algorithm is then used to partition the scan vectors into groups, and provide a condense view of the major scan patterns by a succinct summary of the groups. The experiment on traffic data gathered from two subnets in our campus network shows that our method can accurately identify the major scan patterns without being biased by heavy hitters, meanwhile, possessing simplicity and low computation cost.

Keywords: port scan detection, network security, clustering.

1 Introduction

Port scan, which aims to gather information about hosts in networks, is a fundamental step in today's Internet attacks as well as worm propagation. Therefore, detecting and identifying scans is useful for tracking these malicious activities at early stage to minimize damage.

It is a challenging task to detect scans, however. Scans are broadly categorized into four well known types [13]: *vertical scans*, *horizontal scans*, *coordinated scans* and *stealth scans*. In each type, advanced scan techniques can be used to evade detection.

Another problem is that even if we just focus on the common scans (such as TCP SYN scan), it is still difficult to confirm the malice of all the sources or give a comprehensive explanation of the cause of these scans. This situation is mainly due to the activity of the worms and viruses, which result in the obfuscation of network operators when they have to deal with a flood of logs of scans.

Much work has been done on scan detection. One simple class is what Snort [10] and Bro [8] follow, detecting N connections within a time interval T. The second class of techniques is statistics-based method [6,12]. Many other approaches

S. Qing et al. (Eds.): ICICS 2005, LNCS 3783, pp. 243–255, 2005.

are built upon the observed fact that scanners are more likely to make failed connections [3,8,9]. Another research presented in [4] considers scalable detection. In addition, some work on malicious activity detection, especially worms [1,7,11,15], is also related to scan detection.

Some of the above approaches show good performance in their special scenarios. However, there are still certain weaknesses. The first is about detection accuracy. Simple threshold based methods [4,8,9,10] often generate many false alarms and focus on heavy hitters, while methods with good performance are usually complex, sensitive to parameters or confined to certain context [3,4,6,12]. In addition, few approaches consider the second problem discussed above, namely, how to deal with a flood of scan logs.

In this paper, we pay more attention on the further analysis of detected scans. Our basic idea is that many scanners behave alike, since they scan for the same or similar causes, such as worm or virus infection. Our approach aims to provide a global view of important scans and their implication by diagnosing scan behavior.

As some previous work did, we flag the hosts that have made failed connections as *suspect scanners*. To characterize scan behavior, we build a *scan vector model* in which a suspect scanner is represented by a vector. Then we cluster the scan vectors into *scan groups*. The scanners in one group have similar scan behavior, reflecting certain *scan pattern*. The groups succinctly summarize different scan patterns that imply security threats against network.

Our method works online. It processes packets sequentially and reports major scan patterns periodically. We evaluate it on eight days traffic data gathered at the ingress of two subnets in our campus network. This evaluation shows that our method can: (i) identify and summarize major scan patterns in network without being biased by heavy hitters, (ii) effectively limit false alarms, (iii) and at the same time be easily implemented due to its simple model and low computation cost.

The paper is organized as follows. In Sect. 2, we show how our clustering based method diagnoses port scan behavior. Then evaluation on real traffic data is presented in Sect. 3. We discuss future work and conclude the whole paper in Sect. 4.

2 Diagnosis Method for Port Scan Behavior

Our method contains two parts: a *scan vector model* generating vector sequence, and an extended *center-based clustering* algorithm, which comprises primary *vector clustering* and some *optimizing strategies*. Fig. 1 shows this framework, which finally outputs a summary of major scan patterns.

2.1 Scan Vector Model

Method in this paper focuses on TCP SYN scans towards protected network and is based on failed connections. A *failed connection* in our method is defined as a unique destination $\langle IP, port \rangle$ pair that a source has never successfully connected

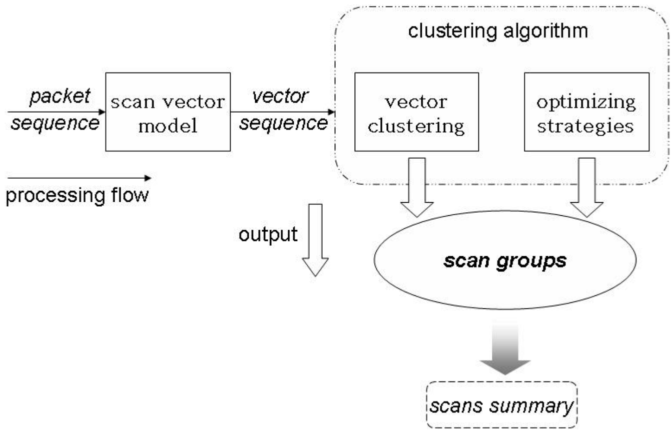


Fig. 1. Framework of Scan Behavior Diagnosis Method

("successful" means that the SYN-ACK packet is observed within a reasonable period T_a) in a *measurement interval* (a preset time period I_m).

We model a suspect scanner's behavior during any numbers of measurement intervals as a vector (or a point) $X = (x_1, x_2, \ldots, x_J) = (x_1, x_2, x_3, y_0, \ldots, y_I)$ ($J = I+4$) in a J-dimensional space. Each vector element x_j is the value of a feature of scan behavior as follows:

x_1	*Address-related feature.* It represents the distance from the scanner's address to the protected network address. If they are in the same /16, $x_1 = 2$; $x_1 = 1$ for the same /8; otherwise $x_1 = 0$.
x_2	*Rate-related feature.* It is the average number of failed connections the scanner initiates in a measurement interval. So, $x_2 > 0$.
x_3	*Targets-related feature.* It is the average number of the scanner's destination hosts in a measurement interval. Also, $x_3 > 0$.
y_i	*Ports-related feature.* It indicates whether port i (i is a real port number, therefore $I = 65535$ generally) has been scanned by the scanner. If so, $y_i = 1$, otherwise $y_i = 0$.

Once a source's first failed connection is observed, its first measurement interval starts and its vector X will be updated over time. At the end of the mth ($m = 1, 2, \ldots$) interval, we particularly denote $X = X^m = (x_1^m, x_2^m, \ldots, x_J^m)$, which can be derived from its last vector X^{m-1} and increment vector $\Delta X^m = (\Delta x_1^m, \Delta x_2^m, \ldots, \Delta x_J^m)$ observed in the mth interval. According to above definitions, ΔX^m gives the source's target ports, the number of failed connections and destinations during its mth interval. Obviously, $X^1 = \Delta X^1$, and we compute X^m ($m > 1$) from X^{m-1} and ΔX^m as:

$$\begin{cases} x_1^m = x_1^{m-1} \\ x_2^m = \alpha x_2^{m-1} + (1-\alpha)\Delta x_2^m \\ x_3^m = \alpha x_3^{m-1} + (1-\alpha)\Delta x_3^m \\ x_j^m = max(x_j^{m-1}, \Delta x_j^m) \qquad j = 4, 5, \ldots, J \end{cases} . \tag{1}$$

If we denote operation (1) as $\oplus$, we can rewrite the whole process as:

$$\begin{cases} X^1 = \Delta X^1 \\ X^m = X^{m-1} \oplus \Delta X^m \qquad m > 1 \end{cases} .$$

Note that in (1), x_2^m and x_3^m are not precise average of history values. In our experiment we found that the result is not sensitive to $\alpha \in (0, 1)$ and approximate average values can work well, so we simply set $\alpha = 1/2$.

In this model, we choose feature set from general knowledge and experience about scans. Vector X represents the scanner's scan scheme, strength, desired information, etc. Method built upon current feature set is effective for scan behavior diagnosis in our experiments, while we will try to find whether better choices exist in our future work. We believe our scan vector model is a general framework and can be adopted in many scenarios by choosing appropriate features.

2.2 Basic Concepts Definition

Following two basic concepts are defined to be used in our clustering algorithm.

Group Center. The *center* of a scan group denotes the mean value of vectors in it. Suppose group c has N_c points $\{X_1, X_2, \ldots, X_{N_c}\}$, where $X_n = (x_{n,1}, x_{n,2}, \ldots, x_{n,J})$ $(n = 1, 2, \ldots, N_c)$. The center of group c, $X_c = (x_{c,1}, x_{c,2}, \ldots, x_{c,J})$, is computed as:

$$x_{c,j} = \frac{1}{N_c} \sum_{n=1}^{N_c} x_{n,j} \qquad j = 1, 2, \ldots, J .$$

From this definition, $x_{c,j}$ $(j = 4, 5, \ldots, J)$ indicates the probability of suspect scanners in group c scanning port $j - 4$.

Similarity. Computing the *similarity* $Sim(X_1, X_2)$ between two vectors $X_1 = (x_{1,1}, x_{1,2}, \ldots, x_{1,J})$ and $X_2 = (x_{2,1}, x_{2,2}, \ldots, x_{2,J})$ is a basic step for clustering. We define:

$$Sim(X_1, X_2) = \sum_{j=1}^{J} w_j sim(x_{1,j}, x_{2,j}) \qquad (\sum_{j=1}^{J} w_j = 1, \quad w_j > 0) , \tag{2}$$

where

$$\begin{cases} sim(x_{1,1}, x_{2,1}) = 1 - \frac{|x_{1,1}-x_{2,1}|}{2} & \\ sim(x_{1,j}, x_{2,j}) = \frac{min(x_{1,j},x_{2,j})}{max(x_{1,j},x_{2,j})} & (max(x_{1,j}, x_{2,j}) > 0, \quad j = 2, 3, \ldots, J) \\ sim(x_{1,j}, x_{2,j}) = 0 & (max(x_{1,j}, x_{2,j}) = 0, \quad j = 2, 3, \ldots, J) \end{cases} . \tag{3}$$

The values of w_1, w_2 and w_3 can be chosen arbitrarily, and w_j $(i = 4, 5, \ldots, J)$ is calculated according to the formula:

$$w_j = (1 - w_1 - w_2 - w_3)\frac{max(x_{1,j}, x_{2,j})}{\sum_{k=4}^{J} max(x_{1,k}, x_{2,k})} \qquad j = 4, 5, \ldots, J . \tag{4}$$

This definition means that the similarity between two vectors is the weighed sum of similarities between vector element pairs. Combining (2), (3) and (4) yields:

$$Sim(X_1, X_2) = \sum_{j=1}^{3} w_j sim(x_{1,j}, x_{2,j}) + \frac{w_0 \sum_{j=4}^{J} min(x_{1,j},x_{2,j})}{\sum_{j=4}^{J} max(x_{1,j},x_{2,j})} \tag{5}$$

$$(\textstyle\sum_{j=0}^{3} w_j = 1, \quad w_j > 0) .$$

For arbitrary points X_1 and X_2, we have the similarity $Sim(X_1, X_2) \in (0, 1]$. It quantifies the resemblance between two types of scan behavior. The closer the similarity is to 0, the less similar they are. $Sim(X_1, X_2) = 1$ is equivalent to $X_1 = X_2$, indicating same behavior. The larger similarity implies larger probability of the same scan intent or cause, e.g. two scanners aiming to the same service, using the same tool or infected by the same worm.

2.3 Vector Clustering

Primary vector clustering is the principle component of our center-based clustering algorithm. Suppose there are L scan groups, X_l denotes the center of group l and group l contains N_l vectors. For any suspect scanner X, once X^m is generated, we cluster it as one of the following two cases.

(i) $m = 1$

That means no point representing this scanner already exists. We compute the similarity between X^m and the center of each group l $(l = 1, 2, \ldots, L)$, then pick out group v whose center X_v is the most similar to X_m as:

$$Sim(X^m, X_v) = \max_{1 \le l \le L} (Sim(X^m, X_l)) .$$

Suppose T_s is a preset threshold, then:

- If $Sim(X^m, X_v) > T_s$, X^m will be put into group v and X_v should be adjusted to $X_{v'} = (x_{v',1}, x_{v',2}, \ldots, x_{v',J})$ as:

$$x_{v',j} = \frac{N_v x_{v,j} + x_j^m}{N_v + 1} \qquad (j = 1, 2, \ldots, J) .$$

Meanwhile, updated group v has $N_{v'} = N_v + 1$ points.

- If $Sim(X^m, X_v) \leq T_s$, a new group u will be created, with its center $X_u = X^m$ and one member X^m.

(ii) $m > 1$

Here X^{m-1} already belongs to certain group v with center X_v. What we need to do is replacing point X^{m-1} by X^m and adjusting X_v as:

$$x_{v',j} = x_{v,j} + \frac{x_j^m - x_j^{m-1}}{N_v} \qquad (j = 1, 2, \ldots, J) .$$

In this case, a question may arise that as any scanner's point is moving all along, should we reconsider which group it belongs to? Our scan vector model and clustering algorithm guarantee the distinct characters of different scan patterns, thus a point hardly has the chance to move from one group into another.

2.4 Optimizing Strategies

Besides primary vector clustering, some optimizing strategies are very important in reducing noise and improving clustering performance.

Vector Pre-checking. This strategy works on ΔX^m that satisfies $\Delta x_2^m = 1$ (only one destination $\langle IP, port\rangle$ pair) before X^m is computed. We record this $\langle IP, port\rangle$ pair. If the pair has been recorded for more than T_r times, we add it to a list called *service set*, and if it is already in service set, ΔX^m is ignored and X is not updated to X^m.

Simply speaking, the $\langle IP, port\rangle$ pairs in service set are connected by a number of sources that hardly make failed connection to other destinations. Therefore, these pairs are probably services opened to public and this strategy is used to reduce false alarms.

Group Merging. It is possible that the first several scanners of certain scan pattern are clustered into multiple groups, for a few points cannot offer enough information. As scanners increase and more data are gathered, we can merge these groups to reduce the amount of groups and improve the accuracy of scan pattern identification.

For any group u, group merging is operated every *checking interval* I_c. We pick out group u's most similar group v as:

$$Sim(X_u, X_v) = \max_{1 \leq l \leq L,\, l \neq u} (Sim(X_u, X_l)) .$$

If $Sim(X_u, X_v) > T_s$, group v will merge group u, which means all points in group u will belong to group v, and the center will move to $X_{v'}$ as:

$$x_{v',j} = \frac{N_v x_{v,j} + N_u x_{u,j}}{N_v + N_u} \qquad (j = 1, 2, \ldots, J) .$$

Group Obsoleting. Many scan groups, especially those representing individual attackers, are *active* (being updated) only in short period. The existence of these groups increases the computation cost. So we set an *obsolete period* T_o. If a group keeps inactive longer than T_o, it will be deleted from the group set.

Port Cutting. Although X is a much long vector, in the implementation of our model we can only record port i that has $y_i > 0$ by using a link, which greatly reduces the computation cost. In group center $X_c = (x_{c,1}, x_{c,2}, x_{c,3}, y_{c,0}, \ldots, y_{c,I})$, if y_i is fairly small, port i cannot represent the essential feature of this group but increase computation cost. Therefore, we set a threshold T_p. If $y_{c,i} < T_p$, then set $y_{c,i} = 0$. This checking is operated every *checking interval*.

2.5 Scan Patterns Summary

Above algorithm clusters all the vectors into groups. By two types of features, these groups provide a summary of scan patterns.

The first type of features are represented by group center. Because scanners in one group have similar scan behavior, group center reveals the common character of them and we can identify a scan pattern from it.

The other type of features are statistics that assess a scan group's severity or threat on network. In our implementation, these statistics are computed along with clustering and reported together with group center every *report interval* I_r. Following six features are defined, of which some are long term features, and others are restricted in one report interval:

start time when the group is created, indicates its duration.

srcs number of suspect scanners active (scanning) in the current report interval, reveals the prevalence of this scan pattern.

cnnts total number of failed connections in the current report interval, reveals the strength of this scan pattern.

sinc difference between the number of active sources for current and previous report interval, reveals the prevalence trends.

tsrcs total number of scanners since this group was created, reveals long term prevalence.

tcnts total number of failed connections since this group was created, reveals long term strength.

As mentioned above, normal network activities also generate failed connections. Because such connections are mostly random and independent, they are much likely to be clustered into small groups with a few scanners and connections. Since the feature *tcnts* reveals the strength of a scan pattern, we can simply use it to select *major groups* and corresponding *major scan patterns*. If the value of a group's feature *tcnts* is larger than threshold T_m, it is a major group.

In summary, a combination of the two types of features describes the behavioral characteristics and assesses the severity or threat of each scan pattern. Summary of all major groups outlines a scene of port scans in networks, implying certain aspects of network security situation or trends.

3 Evaluation

The evaluation in this section will validate the low false alarm rate, large detection coverage and ability of scan pattern identification of our method. In addition, some issues related to implementation will also be discussed.

3.1 Data Description

We use three datasets gathered at the ingress of two subnets (A and B) in Tsinghua University campus network, both have an address space of $3 * /24$, with average daytime bandwidth of 100Mbps. Each trace l, a line in a dataset, representing an inbound SYN packet or an outbound SYN-ACK packet, contains the fields of time stamp t, source IP s, source port q, destination IP d, destination port p and TCP flag f, thus each trace can be written as a 6-tuple $l = \langle t, s, q, d, p, f \rangle$. Both of the two subnets have only one ingress, so we can observe bidirectional packets of a connection. Table 1 summarizes our datasets.

Table 1. Summary of datasets

Dataset	Period	SYN Packets	SYN-ACK Packets
A-1	Nov 23-Nov 24	4,146,139	356,754
A-2	Mar 4-Mar 7	5,290,970	492,330
B-1	Nov 23-Nov 24	4,507,755	346,431

In our evaluation procedure, each trace is processed sequentially. Values of all the parameters are summarized in Table 2 .

Table 2. Summary of parameter values

Parameter	Value	Parameter	Value
I_m	5 minutes	T_s	0.5
I_c	5 minutes	T_r	3
I_r	5 minutes	T_p	0.05
w_1	0.1	T_m	10
w_2	0.15	T_a	3 seconds
w_3	0.05	T_o	2 hours
α	0.5		

3.2 Result Analysis

According to our method, a summary of major active scan groups is reported periodically. Table 3 excluding group 8 is a report sample for dataset A-1. In the "Scan Behavior" part, "B", "A" and "R" in "x_1" column respectively represents feature $x_1 = 2$, 1 and others. Due to page limit, we will only present the analysis on the results of dataset A-1, and the results of other two datasets are similar.

Table 3. A report sample(reported at Nov 23 16:01:07)

Scan Behavior

Group No.	x_1	x_2	x_3	port i ($y_i \geq 0.8$)	port i ($0 < y_i < 0.8$)
1	B	8.7	8.7	445	44445 135 1957
2	B	13.5	13.4	135	445
3	B	3.0	2.2	1023 5554	1022 445 44445
4	A	3.1	1.0	6129 3127	2745 80
5	R	92.0	92.0	4899	
6	R	5.7	5.7	21	248
7	R	1.5	1.5	1433	
8	*R*	*1.0*	*1.0*	*23672*	

Severity Assessment

Group No.	*srcs*	*cnnts*	*sinc*	*tsrcs*	*tcnts*	*start* (Nov 23)
1	58	3060	-2	369	308638	00:01:44
2	5	232	-1	51	40819	07:00:58
3	3	11	0	10	375	12:06:55
4	2	4	1	100	317	07:31:47
5	1	92	1	1	92	15:57:24
6	1	3	1	19	956	10:47:05
7	1	1	0	37	58	08:24:53
8	*1*	*2*	*1*	*21*	*46*	*15:31:59*

False Alarm Analysis. We investigated into all the 37 major groups through out dataset A-1. Table 4 summarizes them as 4 parts and 10 subcategories. Note that some groups may actually represent one scan pattern emerging at different time, due to *group obsoleting* operation. The 31 groups in the first three parts are important scan patterns in networks, with different intents or causes.

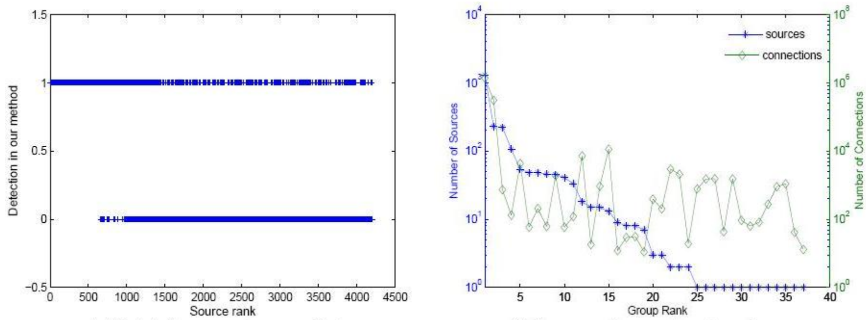
However, the last part is undetermined. Groups with port 113 may be normal authentication service accesses, and the last four groups with non-well-known ports possibly represent normal applications such as P2P. Thus, the last 6 groups may be false alarms. However, since they only involve 1.2% of all the scanners and much less portion of the connections, the false judgements hardly influence our macro assessment on network security.

In fact, the design of *service set* greatly reduces false alarms. Group 8 in Table 3 is a false alarm generated if service set is not performed. We looked over all the reports and found some similar cases. Totally, when service set is performed, the number of sources in dataset A-1 judged as scanner is reduced by 42%, of groups is reduced by 45%. In other two datasets the results are similar.

Detection Coverage. We use following method to obtain a ground truth and compare our approach's result with it: (i) we set *measurement interval* to the whole period of the dataset and pick out all the suspect scanners, (ii) then we sort the scanners in descending order of the number of failed connections. Here

Table 4. Investigation on major active groupes of dataset A-1

	typical target ports	x_1	groupes	scanners	connections
Vulnerability	445,139	B	3	1307	1358768
	135	B	4	261	325111
	6129,3127,1433,...	A,R	9	567	1408
Related	1023,5554,9898	B,R	7	64	9484
	4899	R	3	3	236
Service	21	R	2	60	2788
Searching	80,1080,8080,...	R	1	1	273
By Administrator	some important ports	B	2	2	1986
Undetermined	113	R	2	17	43
	non-well-known ports	R	4	11	84
Total		—	37	2293	1700181



(a) Detected scanners in our method (b) Sources and connections in each group

Fig. 2. Scanners detection and distribution in groups

the hypothesis is that failed connections provide strong evidence of scans, and the much long measurement interval greatly decreases the false alarms caused by failed normal service accesses.

Fig. 2(a) shows the detection coverage of our method. The x-axis is the rank of sources in the above order, and the y-axis shows whether a source is detected by our method. If the capacity of operators paying attention and taking steps on suspect scanners is 656 or less (in fact, the 656th scanner makes only 8 failed connections and 656 scanners are fairly an large amount to deal with), our method's detection coverage is 100%. On the other hand, many scanners that make a few failed connections are detected in our method, for we make use of the correlations between scanners' behavior other than just the number of connections. Therefore, our method can detect not only the heavy scanners but also the scanners ignored in the usual scan detection deployment. This point will be illustrated more specifically in the next subsection.

Scan Pattern Identification. From Table 3 we can see that 7 major active groups briefly summarize 587 scanners (71 are active at the moment), and each has its own distinct features.

Group 1 and 2 are related to common worm (Sasser, Blaster, Nachi, etc) scans. Group 3 are probably backdoor scans performed by viruses or attackers. Group 4 also represents worm (Phatbot and its variants) scans. Port 4899 of group 5 is a frequently used backdoor port, port 21 of group 6 is the most common FTP port and port 1433 of group 7 is for Microsoft SQL-Server.

Through out the whole dataset A-1, Table 4 reveals that subcategory 1 and 2 occupy more than 99% of the scan traffic due to their scan preference for addresses in the same /16, while subcategory 3 (including group 4 in Table 3) occupies more than 25% of the scanners. Although the scanners in subcategory 3 are far away from the protected subnet, they imply potential threat, therefore we think they are worthy of notice. However, since each scanner initiates only a few connections to protected subnet on average, approaches just analyzing a single scanner's behavior probably miss such scans.

Therefore, it is an important advantage that our clustering based method prevents important scan patterns – especially those with low scan rate or total connections – from being biased by certain scan patterns involving most heavy hitters. Concept of scan group provides an effective way to assess importance or threat of scans. Many small scans (a few connections observed) can compose a noticeable group, representing important scan pattern, while small (non-major) groups are really negligible.

3.3 Discussion

Fig. 2(b) plots the rank of each major group against its number of scanners and failed connections. Except the first two groups with both dominant scanners and connections, other groups have no proportional relation between their numbers of scanners and connections, which reveals the necessity of characterizing scan behavior with more than one feature. Also, network administrators should combine various features and their most concerns on network security to assess the importance of certain scan patterns.

We have mentioned four basic types of scans: vertical scans, horizontal scans, coordinated scans and stealth scans. The latter two are difficult to detect in most previous work. Although there are no instances in this evaluation, we believe in our method's ability in identifying a great part of such scans. Probably, the scanners participating in one attack of scan have similar behavior and are clustered into one scan group. Then adding in scanners' address feature, we may identify this coordinated scan episode. For stealth scans, they are difficult to detect because of their low scan rate. In our method, obsolete period T_o is 2 hours. Thus, as long as scan rate is larger than 0.5 failed connections per hour, the scan can be detected and attract attention when it belongs to a major group.

Computation cost is another major issue we are concerned about. The cost relies on many factors, such as bandwidth, traffic structure and total scans. Now our method has already been implemented as an online functional module in the security monitoring system for subnet A and B. In this evaluation, the computation on each dataset requires about 15 minutes on a 2.6GHz Intel-based

PC. Thus from the point of computation cost, our approach has the potential to scale up to higher speed environment.

4 Conclusion and Future Work

In this paper, we have proposed an approach to diagnose port scan behavior. Our work aims to provide a succinct summary of important scan patterns in networks, which is useful for effectively monitoring network security, but little explored in the pervious work. Our method is based on the fact that scans have strong correlations because of scanners' same or similar intents. We model any suspect scanner as a moving point in a high dimensional space and a center-based clustering algorithm clusters scanners of similar behavior into one scan group, which represents a major scan pattern in networks.

We evaluate our method by real network data. All important scan patterns in our datasets are identified, with negligible false alarms and low computation cost. The results validate our method's ability in effectively diagnosing port scan behavior in networks.

In the future work, we will go on researching on how to design an optimal vector that catches more essential characters of scan behavior. Beyond TCP SYN scan, other scan techniques [14] will be also taken into account. Furthermore, as the information of any scan pattern reported at intervals also forms a timeseries, we will try to find whether some forecast models that have been widely studied and applied for traffic anomaly detection [2,5] can work on this scan related timeseries and draw meaningful conclusions about its developing trends.

Acknowledgement

This work is supported in part by the National High Technology Research and Development Program of China (863 Program) under Grant No. 2003AA142080 and the National Natural Science Foundation of China under Grant No. 60203004. The authors are grateful to Jianguang Di and Xueli Yu for their help in building experiment environment.

References

1. V.H. Berk, R.S. Gray, and G. Bakos: Using sensor networks and data fusion for early detection of active worms. In Proceedings of the SPIE AeroSense, 2003
2. J. Brutlag: Aberrant Behavior Detection in Timeseries for Network Monitoring. In Proceedings of USENIX Fourteenth Systems Administration Conference (LISA), New Orleans, LA, Dec 2000
3. J. Jung, V. Paxson, A. W. Berger, H. Balakrishnan: Fast Portscan Detection Using Sequential Hypothesis Testing. In Proceedings of 2004 IEEE Symposium on Security and Privacy, pages 211–225, Berkeley, CA, USA, May 2004
4. R. R. Kompella, S. Singh, and G. Varghese: On Scalable Attack Detection in the Network. In Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, pages 187–200, Taormina, Sicily, Italy, Oct 2004

5. B. Krishnamurthy, S. Sen, Y. Zhang, and Y. Chen: Sketch-based Change Detection: Methods, Evaluation, and Applications. In Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement, Pages: 234–247, Miami Beach, FL, USA, Oct 2003
6. C. Leckie and R. Kotagiri: A probabilistic approach to detecting network scans. In Proceedings of the Eighth IEEE Network Operations and Management Symposium (NOMS 2002), pages 359–372, Florence, Italy, Apr 2002
7. D. Moore, C. Shannon, G. M. Voelker, and S. Savage: Internet Quarantine: Requirements for Containing Self-Propagating Code. In Proceedings of IEEE INFOCOM, Apr 2003
8. V. Paxson: Bro: A System for Detecting Network Intruders in Real Time. In Proceedings of the 7th USENIX Security Symposium, 1998
9. S. Robertson, E. V. Siegel, M. Miller, and S. J. Stolfo: Surveillance detection in high bandwidth environments. In Proceedings of the 2003 DARPA DISCEX III Conference, pages 130–139, Washington, DC, Apr 2003
10. M. Roesch: Snort: Lightweight intrusion detection for networks. In Proceedings of the 13th Conference on Systems Administration (LISA-99), pages 229–238, Berkeley, CA, Nov 1999. USENIX Association
11. S. E. Schechter, J. Jung, A. W. Berger: Fast Detection of Scan Worm Infections. In Proceedings of the Seventh International Symposium on Recent Advances in Intrusion Detection, Sophia Antipolis, France, Sep 2004
12. S. Staniford, J. A. Hoagland, and J. M. McAlerney: Practical automated detection of stealthy portscans. In Proceedings of the 7th ACM Conference on Computer and Communications Security, Athens, Greece, 2000
13. V. Yegneswaran, P. Barford, and J. Ullrich: Internet intrusions: global characteristics and prevalence. In Proceedings of the 2003 ACM SIGMETRICS, volume 31, 1 of Performance Evaluation Review, pages 138–147, New York, Jun 2003. ACM Press
14. M. de Vivo, E. Carrasco, G. Isern and G. de Vivo: A Review of Port Scan Techniques. Computer Communications Review, 29(2), April 1999, pages 41–48
15. C. C. Zou, L. Gao, W. Gong, and D. Towsley: Monitoring and Early Warning for Internet Worms. In Proceedings of the 10th ACM conference on Computer and communications security, Washington, DC, USA, Oct 2003