A Network Scanning Detection Method Based on TCP Flow State

Qiao Hong[1], Tian Jianwei[1], Ying Ying[*2], Tian Zheng[1], Zhu Hongyu[1] and Li Shu[1]

1 State Grid Hunan Electric Power Corporation Research Institute, Changsha, 410007
2 Zhejiang Economic Information Center, Hangzhou, 310006
*corresponding author's email: yy@zgb.com.cn

***Abstract*—Network scanning is always the beginning action of network attack and recent detection methods are hard to detect distributed scanning behavior. This paper proposes a Network Scanning Detection algorithm based on Flow State (NSCDFS) to precisely detect both traditional scanning and distributed scanning. The algorithm divides the state of flows into 6 stages and set the state of each flow according to the flag value of its package. And based on the situation of flows' state from the same source IP address, the traditional scanning and the distributed scanning can be detected precisely. And the experimental result shows the algorithm works well in the high speed network.**

Keywords- Network Scanning; Flow State; Distributed Scanning; High Speed Network

I. INTRODUCTION

The network attacks always use network scan tools to scan the targeted network and collect the network information including IP Address and opening port, used for providing network services. According to the feedback information of scanning, network attackers intrude the servers and network devices by corresponding methods and lead destructive effect. If the early scanning behavior can be detected and take measures to stop the attack activity, the network defense capabilities will improve evidently [1-6].

The way of scanning detection in recent NIDS (Network Intrusion Detection System) is based on the scan times from the same source IP address in a period time. Some distributed scanning attacks use multiple hosts scan targets simultaneously and constrain the scan frequency of each host under the alert threshold of NIDS, and may bypass the scanning detection. Hence, this paper proposes a Network Scanning Detection algorithm based on Flow State (NSCDFS) to solve the problem. NSCDFS considers the frequency of TCP connection requests in each time slide window to detect for traditional scanning behavior, and use the state of TCP flow for distributed scanning, which can detect the scanning more precisely. Besides, a scanning detection system based on NSCFS is designed and accomplished and achieve good effect.

This paper is organized as follow: Section II expresses NSCDFS algorithm; Section III describes System Design based on NSCDFS; Section IV expresses the experimental results; and finally is the conclusion.

.

II. ALGORITHM DESCRIPTION

Most network scanning methods utilize TCP three-way handshake protocol, such as TCP connection scanning, TCP SYN scanning, TCP ACK scanning, etc. Hence, to design scanning detection algorithm, we need know the TCP three-way handshake protocol.

A. TCP Three-way Handshake Protocol

TCP three-way handshake protocol is used to establish a TCP/IP connection before reliable data exchange between the server and the client. Like Figure 1, in the first handshake, the client sends packages with the SYN flag equals to 1 to the server and waits for the response from the server. Then, after receiving the SYN package, the server responses the client with both SYN flag and ACK flag equal to 1. Finally，the client sends the packages with ACK flag set to 1 to the server after receiving SYN+ACK package, and the process of handshake protocol finishes. After that, the client and the server can exchange data reliably.

Then, we can design the detection algorithm according to the handshake protocol.

B. NSCDFS Algorithm

The scanning behavior that sets illegal package flag can be easily detected by check the package head, since the flag configuration can't appear in the normal network transmission. For TCP connection scanning and SYN scanning especially distributed scanning, the detection method will be more complicated, because scanning packages mixes up with normal transmission packages. To handle this problem, we propose the NSCDFS algorithm.

To describe our algorithm clearly, we classify the state of the TCP flow into 6 stages, including first shake stage, second shake stage, third shake stage, data exchange stage, end stage and other stage, like Figure 2. The former 3 stages represent stages of TCP three-way handshake protocol. The data exchange stage means the two hosts finish three-way handshake and start to transmit message data. The end stage means the two hosts finish data exchange and want to end the communication. And the other stage means there are illegal flag set in the flow package.

For a normal visiting host, it always needs to exchange data with servers and its TCP flows always finish three-way handshake stages and arrive at data exchange stage and end stage. For a scanning host, it always just send connection request and waits for the response and doesn't transmit business data. And most of scanning hosts' will arrives at first shake stage, second shake stage, third shake stage or other stage. Hence, we can detect the scanning behavior from the number of connect request from a host and the ratio of flows' state that doesn't arrive at data exchange stage.

We consider a TCP flow is bidirectional and use fi= (src,sp, des,dp) to represent the TCP flow i, where src is the source IP address, sp is source port, des is destination IP, dp

978-1-7281-1366-1/18/$31.00 ©2018 IEEE
DOI 10.1109/ICSCSE.2018.00089

is destination port . This means fi = (i1, p1, i2, p2) and fj=(i2, p2, i1, p1) represents the same TCP flow, that is, fi = fj .

We use structure Ti= (fi, src, st, tm) to represent state information of TCP flow i, where fi represent the flow i; src represents the source IP; st represents current stage flow i and is a 6 bits variable, and each bit will be set to 1 if receiving corresponding packages; tm represents the state update timestamp. For example, when we capture a TCP package of flow fi= (src,sp, des,dp), we firstly check whether we have store state information for flow fi. If we have stored it before, we set its stage st according to the value of SYN flag, ACK flag, FIN flag, PSH Flag and RST flag. Specifically, if SYN=1 and ACK = 0, st will be set to 100000; if SYN = 1 and ACK = 1, st will be set 010000; if SYN = 0 and ACK = 1, st will be 001000; if PSH = 1 and ACK = 1, st will be set to 000100; if ACK = 1 and FIN = 1， st will be set to 0000010;Otherwise, st will be set 000001. Of course, if the flow has received multiple types packages, corresponding bits will be all set to 1, such as st will be set to 1100000 if receiving two packages with SYN =1, ACK =0 and SYN = 1, ACK = 1 respectively.

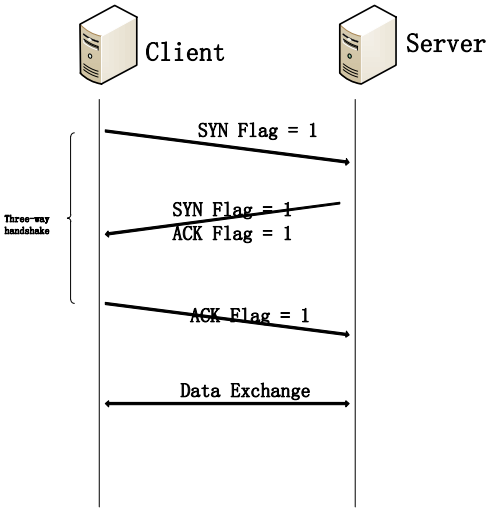

Figure 1. Three-way handshake

If we haven't store the flow information before, we will check if the package is a TCP connect request package, that is, SYN = 1, ACK = 0. If it is, we will create structure Ti= (fi, src, st, tm) to store it and set st to 100000; otherwise the package will be ignored.

To detect the scanning behavior more precisely, we classify the scanning behavior into two categories. One is traditional scanning, another is distributed scanning.

For traditional scanning, we define tr_win and tr_period as the slide time windows and calculation period of traditional scanning respectively. It means every tr_period seconds the algorithm will statistically analyze the TCP connection activating state in last tr_win seconds. And we use flag_count(src,st,tr_win) represents the number of TCP flow from source IP src in stage st in the last time slide windows. And we use abnormal_count to represent the number of flows that just don't exchange message data and it can be calculated as:

$$abnormal\_count = \sum_{st \in T} flag\_count(src, st, tr\_win) - flag\_count(src, 111110, tr\_win) \quad (1)$$

where T represents all state stages of flows. And formula (1) means the abnormal number of flows from src equals to the number of all flows from src minus the number of flows that have exchanged message data.

Define tr_thresh_count as the abnormal trigging threshold of traditional scanning, if abnormal_count > tr_thresh_count, we can judge the IP address src is scanning the network.

For distributed scanning, we define dt_win and dt_period as the slide time windows and calculation period of distributed scanning respectively. It means every dt_period seconds the algorithm will statistically analyze the TCP connection activating state in last dt_win seconds. Because distributed scanning will restrain its scanning frequency, we can't detect the scanning behavior according to the number of abnormal flows. But, the scanning behavior seldom exchange data among hosts, and most of its flow will not arrive at data exchange stage. So we can calculate the ratio of abnormal flows to all flows from src to detect the scanning behavior. And we define tr_thresh_ratio as abnormal ratio trigging threshold of traditional scanning. And the ratio of abnormal TCP flows from src can be calculated as：

$$abnormal\_ratio = 1 - \frac{flag\_count(src, 111110, tr\_win)}{\sum_{st \in T} flag\_count(src, st, tr\_win)} \quad (2)$$

where T represents all state stages of flows. If abnormal_ratio> tr_thresh_ratio, we can also judge IP address src is scanning the network.

To sum up, the NSCDFS algorithm can be described as Algorithm 1.

III. SYSTEM DESIGN

The network scanning system is mainly consist of three parts: Network Probe module, Package Process module and Alert View module, like Figure 3.

The Network Probe is used for capturing network package without loss. For special network processor, it has good performance for its optimized package process component. But its price is high and its transferability is bad for hardware development. And network package capture method based on general PC architecture has good extendibility and cheap price, but its performance is not suitable for high speed network, such as TcpDump and Wireshark. To capture packages from high speed network without loss, we use SNORT [7] and PFRING technique to capture network packages. SNORT can be used for preliminary analyze the package from IP layer to application layer, and PFRING can be used to reduce the

CPU and system interruption while capturing network packages, resulting in better performance. The Network Probe can be deployed distributed to capture the network package comprehensively.

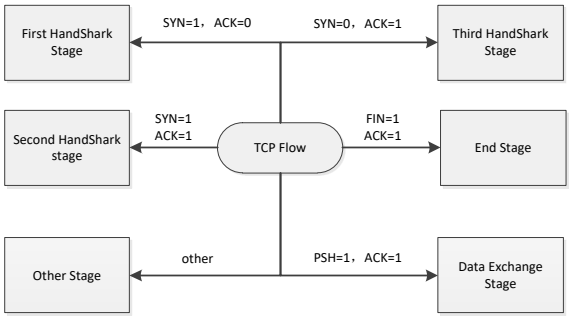

Figure 2. TCP flow state

Algorithm 1. Network Scanning Detection algorithm based on Flow State (NSCDFS)

Input: All flows state structure *T*

Output: Scanning IP address Set *S*

S←*Φ*

1. **Every** *tr_period* **do**

   for each *src* of flows state structure in *T* **do**

   calculating *abnormal_count* according to formula (1)

   if *abnormal_count* > *tr_thresh_count* **do**

   add IP address *src* to scanning set *S*

2. **Every** *dt_period* **do**

   for each *src* of flows state structure in *T* **do**

   calculating the number of TCP flow from source IP *src* in first stage or other stage *flag_count (src,100000,dt_win), flag_count (src,100001, dt_win), flag_count (src,000001,dt_win)* respectively

   then calculating ab*normal_ratio according to formula (2)*

   if ab*normal_ratio* > *dt_thresh_ratio,* **do**

   add IP address *src* to scanning set *S*

After preliminary analysis by SNORT, the result will be transmitted to Package Process module, mainly consist of FLUME, KAFKA and STORM. FLUME is used to buffer message data from distributed probes and transmit to KAFKA in batches. Here, because network traffic is unstable, the number of messages data transmitted to Package Processor from each probe is always changing. And KAFKA is good for stable incoming messages. If each probe straightly transmit message data to KAFKA, the process performance will reduce. Hence, we use FLUME to transmit message data to KAFKA steadily.

KAFKA dispatches message data to the STORM by considering process load of each node in the STORM cluster. And the NSCDFS algorithm is implemented in the STORM to detect the scanning behavior. The analysis result will store into MySQL and ES (elastic search) database. MySQL database is mainly used to store configuration and alarm information. And ES database is mainly used to store flow package information.

And the Alter View module will read information from database and display the detect result.

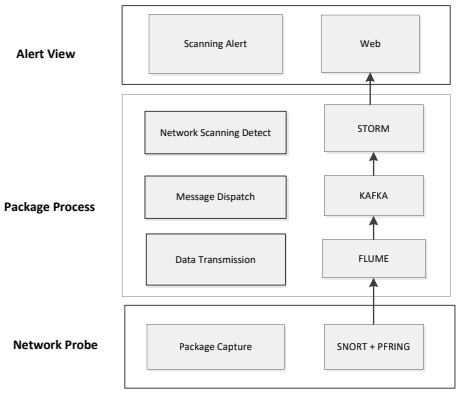

Figure 3. System architecture

IV. EXPERIMENTAL RESULT

To evaluate the performance of NSCDFS algorithm, we use Unbuntu 16.04 as the server and run it on a 1000Mbps local network area. Besides, we use another 6 hosts to run NMAP to scan the server. NMAP has a timing option to configure the parallel or serial scanning and the scanning speed. And the option ranges from T0 to T5, higher level with faster speed. Specifically, T0, T1, T2 means serial scanning and speed is low and they can be used for simulating distributed scanning, and T3, T4, T5 means parallel scanning and speed is high and they can be used for traditional scanning . The experimental result shows as Table 1.

Table 1 Scanning Detection Result

No	Scanning IP	Server IP	Timing Option	NSCDFS Result
1	192.168.219.3	192.168.219.1	T0	No
2	192.168.219.4	192.168.219.1	T1	Yes

3	192.168.219.5	192.168.219.1	T2	Yes
4	192.168.219.6	192.168.219.1	T3	Yes
5	192.168.219.7	192.168.219.1	T4	Yes
6	192.168.219.8	192.168.219.1	T5	Yes

From the result, we can see that NSCDFS can detect all kinds of scanning of NMAP except for T0 level. However, in the T0 level, NMAP will use serial scanning and the scanning interval is 5 minutes. Because T0 is too slow, it is unpractical in the real network and we can consider it as one way of APT (Advanced Persistent Threat), which will not be discussed in this paper. Hence, NSCDFS can achieve good performance in the network.

V. CONCLUSION

To detect network scanning attack precisely, this paper proposes a NSCDFS (Network Scanning Detection algorithm based on Flow State) algorithm. The algorithm divides the state of flows into 6 stages and set the state according to the flag value of flows' package. And based on the number of flows' state from the same source IP address, the traditional scanning and the distributed scanning can be detected. And the experimental result shows the algorithm works well in the high speed network.

ACKNOWLEDGEMENTS

This work is sponsored by State Grid Hunan Electric Power Company Limited Technology Project (No. 5216A516002T).

REFERENCES

[1] A.Sridharan, T. Ye, et al. "Connectionsless Port Scan Detection on the Backbone." Proceedings of the 25th IEEE International Performance, Computing, and Communications Conference, 2007, pp.566-576.

[2] W. J. Wang, B. J. Yang, et al. "Detecting Subtle Port Scans Through Characteristics Based on Interactive Visualization." Proceedings of the 3rd annual conference on Research in information technology, 2014, pp.252-269.

[3] BouHarb E, Debbabi M, et al. "On Fingerprinting Probing Activities." Computers&Security, 2014, vol.43, no.6, pp.35-48.

[4] D.M.Farid, M.Z.Rahman. "Anomaly Network Intrusion Detection Based on Improved Self Adaptive Bayesian Algorithm." Journal of Computers, 2010, vol.5, no.1, pp.23-31.

[5] L. J. Liu and J. P. Huai. "Research of a Network Scan Detection Algorithm Based on the FSA Model." Journal of Computer Research and Development, 2006, vol.43, no.3, pp.417-422.

[6] H. Y. Zhu, T. Zheng, et al. "Practice of Automatic Monitoring Tool for Boundary Port of Electric Power Information Network." Hunan Electric Power, 2017, vol. 4, pp.49-52.

[7] Martin Roesch.snort. http : www.snort.org , 2004.