Detection and Defense of SYN Flood Attacks Based on Dual Stack Network Firewall

Ding Pengfule[1], Tian Zhihong[1], Zhang Hongli[1], Wang Yong[2], Zhang Liang[2]* and Guo Sanchuan[2]

1. Harbin Institute of Technology, Harbin 150001 2. CNCERT/CC

Email: dpfl2012@163.com, tianzhihong@hit.edu.cn, zhanghongli@hit.edu.cn, wangyong@cert.org.cn, zl@isc.org.cn, guosc@mail.nankai.edu.cn

***Abstract*—The extensive use of Internet technology has brought great convenience to modern society, however, more and more severe problems regarding to network security have also emerged at the same time. Especially the DDoS attacks, represented by SYN Flood, pose massive threats to the network security. This paper discusses an algorithm which could detect SYN Flood attack quickly under large scale network: the adaptive threshold algorithm. Then we propose "Slow detection, Fast recovery" mechanism on basis of adaptive threshold algorithm. Finally, we implement the attack detection and defense algorithms in dual-stack firewall, and test the validity and performance respectively. The results indicate that the methods of detecting and defending SYN Flood proposed by this paper can improve the system efficiency substantially when firewall is attacked, while consuming only a small amount of extra memory.**

Keywords*—Dual Stack; Firewall; SYN Flood attack; Adaptive threshold; Fast recovery mechanism

I. INTRODUCTION

Due to the bad design of IPv4, there is a series of defects and shortages, such as lack of IPv4 address space, rapid expansion of IPv4 routing table, lack of support for the network layer security, lack of support for the mobile network and the network quality of service [1-5] and so on. In order to solve these problems, many countries in the world and the institutions are beginning to increase research funding and investment in the next generation network, namely IPv6 network [6, 7].

However, it needs a slow process that IPv6 technology completely replaces IPv4 protocol. During the transition from IPv4 to IPv6, the two protocols will inevitable appear to co-exist for a long time, and it will also lead to the existence of IPv4 and IPv6 traffic in the network at the same time. Currently, the transition from IPv4 to IPv6 mainly has three techniques: dual-stack, tunneling, and translation. And dual-stack technique is the most widely used in IPv6 transition technology, the tunneling and translation need to be implemented on the basis of dual-stack.

Denial-of-service (DoS) attack or distributed denial-of-service (DDoS) attack is that the attackers exhaust the computer or network resources to make it unavailable to provide normal service. Over the past few years, the DDoS attack repeatedly appeared in the network security events and under the increasing popularity of IPv6 background, DoS and DDoS attack also can use the IPv6 network. So we must attach great importance to the research of DDoS attack to strengthen the security of today's Internet.

And now the most prevalent attack in the DoS and DDoS is SYN Flood attack. Therefore, our paper mainly study the SYN Flood attack in dual-stack, to explore a model that can effectively resist the SYN Flood attack and improve the defense capability of the firewall itself.

The rest of this paper is organized as follows: Section 2 presents a brief overview of related work. Section 3 introduces the principle of SYN Flood attack and the adaptive threshold algorithm which can detect SYN Flood attack quickly under large scale network. In Section 4, we propose an IP decision algorithm based on the victims of the sampling window and a fast recovery mechanism based on adaptive threshold algorithm to design our defense system. Finally, in Section 5 we test our detect algorithm and defense system, and we make the summary and present our future work in Section 6.

II. BACKGROUND

Kang[8] pointed out that in the DDoS attack there are more than 94% attacks are based on TCP, while either UDP or ICMP attack is about 2% of the total. And the SYN Flood attack is one of the most important attack ways which are based on TCP, and it accounts for about 90%.

Since 2000, there have been a large number of articles which were about SYN Flood detection and defense method in academia at home and abroad.

Takada[9] used a non-parametric recursive algorithm (Cumulative Sum, CUSUM) to monitor the new source IP address appearing in data traffic per unit time. And when the number of new IP address increases anomaly, they decide there is an attack. However, this method just simply considers the increase of the number of single IP address as the judgment of attack, rather than the comprehensive changes of network, so there is a big false positive rate.

Lin[10] also used the non-parametric CUSUM algorithm, but they abstracted the ratio of the number of new IP address appearance in the unit time and all the number of IP address as a stochastic model. This algorithm can detect attack when the attackers forge the source IP address. But it will produce false positives when the server receives large amounts of different new IP address packets in a short time.

Chen[11] proposed a method that using CUSUM algorithm to detect the difference between the number of SYN and FIN (RST active) packets which into the attacked network. Then they successively judged whether the network is receiving abnormal TCP connections. This method can certainly improve the on-line detection speed, and significantly improve the detection effect of distributed SYN Flood attack. However, due to the RST passive packet, this algorithm cannot solve the problem of false positive rate.

978-1-5090-1192-6/16 $31.00 © 2016 IEEE
DOI 10.1109/DSC.2016.108

Chen[12] analysed the data by change-point detection method. This algorithm can just consume a small amount of resources to achieve satisfactory detection effect. Its weakness is that it would be mistaken for the DDoS attacks when normal congestions occur in the network.

After studying researches about SYN Flood in recent years, we found that the common defense strategies are mainly about: defense strategy based on concession and negative feedback, gateway interception, and defect of TCP protocol stack.

The defense strategy based on concession and negative feedbacks sacrifices a part of the server resources or modifies the server configuration to implement the SYN Flood attack defense[13]. The principle of gateway interception is recording the IP address and its validity to decide whether to discard the request packets. Common ways are network trace, Push-back, SYN kill [14], and Ingress Filter. In consideration of TCP protocol stack's defect, Daniel J. Bemstein proposed the SYN Cookie[15]. The SYN Cookie method has played a very significant effect in defense SYN Flood attack, and already has practical implementation and development in the TCP protocol stack of Linux operating system.

In addition, Liu[16] proposed a model based on the adaptive threshold state to defense the SYN Flood attack. The way that they use the threshold to preprocess and analyse the traffic is also a good reference to design the SYN Flood defense strategy.

III. Research SYN Flood Detection Algorithms on Dual-stack Firewall

SYN Flood attack is one of the most popular DDoS attacks. It uses the flaw of TCP "thee-way handshake". The attackers send a large number of forged TCP connection requests to exhaust the being attacked object's resource (such as, make CPU full load or fill memory), so that the being attacked server can't provide normal service to users.

A. SYN Flood Attack

TCP is a connection-oriented transport protocol. Before transmitting data, the two communication sides must shake hands for three times to confirm each other's information, and such method can guarantee data integrity and consistency in the process of communication.

The complete process of TCP three-way handshake is shown as follows:

First Shake: Client sends connection request to Server and randomly chooses an initial sequence number j for the SYN packet;

Second Shake: When Server receives the connection request, it will send an ACK confirm packet and a SYN packet to Client, and the sequence number of the two flag are ACK=j+1 and SYN=k respectively;

Third Shake: Finally Client sends an ACK packet to Server after it receives Server's ACK+SYN packet, and then Client changes to "Established" state.

After Server receives the last ACK packet, it also changes to the "Established" state and the three-way handshake is completed.

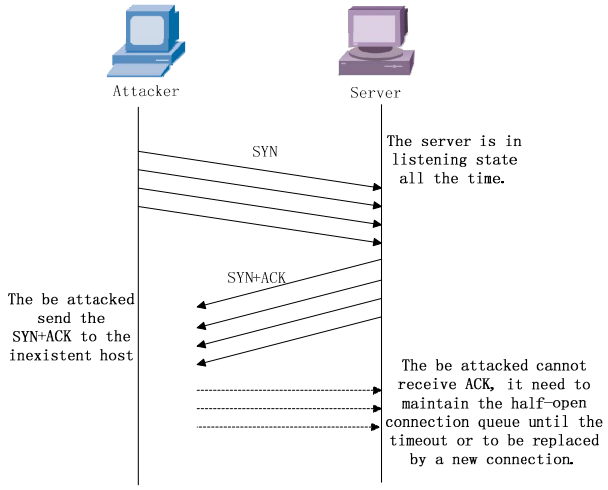

Figure 3-1: The SYN Flood attack process

When the malicious host launches the SYN Flood attack, they would control the zombie machines to send a large number of forged TCP connection request packets (namely SYN packets of the first handshake) to the target machine.

Then the target machine will allocate resource in the protocol stack for the connection requests, such as recording information in the half-open connection queue, and responding SYN+ACK packet to the attacker. However, the attacker no longer responds an ACK packet to complete the TCP three-way handshake, but to send many other SYN connection request packets. So that the victim host will allocate a lot of memory resources for the incomplete three-way handshake in a short time.

In addition, when the subsequent SYN packet arrives, the victim host needs to search the half-open connection queue to confirm whether there has this packet's shake hands information before. While this certainly will consume lots of CPU resources of the victim host, thus the victim host's system efficiency is greatly reduced and the system may even crashes. Figure 3-1 shows the SYN Flood attack process.

Our dual-stack firewall is based on IPv4 and IPv6 mixed network traffic, and even in normal circumstances the firewall also needs to deal with very large network traffic, so the detection algorithm in our paper must satisfy the following conditions: high efficient and rapid, less calculation, and shouldn't use excessive CPU and memory resources.

B. Adaptive Threshold Detection Algorithm

The adaptive threshold detection algorithm can change the threshold of abnormal decision by analyzing the dynamic characteristics of current traffic. It is a simple and rapid algorithm, and it's very suitable for SYN Flood attack detection under the large traffic network environment.

The basic idea of this algorithm is determining whether the system is under attack by judging whether the network traffic exceeds threshold. And the value of the threshold is calculated by the recent changes of network traffic. When detecting SYN Flood

attack, the traffic characteristic here refers to the number of SYN packets. This algorithm firstly calculates the number of SYN packets and the statistics of total flow. Then we compare the calculated result with the calculated threshold intervals to determine whether the system is under SYN Flood attack.

The judgment condition of whether a host is attacked is given by:

$$x_n \geq (\alpha + 1)\overline{f}_{n-1} \quad (1)$$

where x_n indicates the number of SYN Flood packets in the time interval n, $\bar{f}_{n-1}$ indicates the estimated average traffic in the time interval $n-1$. The parameter α indicates the ratio of the condition when number of SYN packets is large than the average traffic when there is an attack.

At the same time, $\bar{f}_n$ can be calculated by $\bar{f}_{n-1}$ and x_n using the Exponential Weighted Moving Average (EWMA), namely:

$$\overline{f}_n = \beta\overline{f}_{n-1} + (1-\beta)x_n \quad (2)$$

where β is the exponential of EWMA.

However, it may cause a higher false positive rate if we directly use the above judging formula in practice.

Siris[17] proposed an improved method of the above judging formula, namely:

$$H_i = \begin{cases} 1 & x_i \geq (\alpha+1)\overline{f}_{i-1} \\ 0 & x_i < (\alpha+1)\overline{f}_{i-1} \end{cases} \quad (3)$$

And Boolean variable H_i represents whether the attack condition of the formula 3-1 is satisfied during the ith time interval.

The judgment of attack is optimized as the condition when the successive valid times of formula 3-1 is larger than the given threshold k, as follows:

$$\sum_{i=n-k+1}^{n} H_i \geq k \quad (4)$$

The threshold k is a constant which is bigger than 1, and represents the minimum of the number of time intervals which continuously exceed the threshold.

In the specific implementation of the algorithm, we can detect whether there is an attack by letting a timer achieve every once in a while.

Initialized $\bar{f}_{n-1}$ to 0, and the process of the adaptive threshold detection algorithm is as follows:

1) Judging whether the timer exceeds the time limit;
2) If there is not a timeout, it means that the time interval is not reached, we go to the normally capturing packet and protocol reassembling procedures. Next step, we analyze the captured packets, if it is SYN packet, update the value of x_n; otherwise, we do nothing.
3) If the timer exceeds the time limit, the algorithm should judge whether they are SYN Flood attack as follows: computing $\overline{f}_n = \beta\overline{f}_{n-1} + (1-\beta)x_n$, then determine whether the condition $\sum_{i=n-k+1}^{n} H_i \geq k$ is satisfied. If satisfied, it indicates that the firewall is SYN Flood attack, then we start the defense module to protect host. If not satisfied, go back to the step 1.

However, to make the adaptive threshold detection algorithm operate normally, we should set the following parameters: the parameter α which indicates the ratio of the number of SYN packets larger than the average traffic when there is an attack, β which is the exponent of EWMA, and k which is the minimum of the number of time intervals that continuously exceed the threshold.

Therefore, we conducted a lot of experiments on our system. The experimental results show that when $\alpha = 0.7$ and $\beta = 0.96$, our system can achieve the best performance and lowest false positive rate. And we set the time interval to 0.25 seconds for the large network traffic of our project.

We set parameter k equal to 4 so that we can try to reduce the computational burden of the firewall, while reducing the false positive rate. Thus, the formula (4) can be simplified as follows:

$$\sum_{i=n-3}^{n} H_i \geq 4$$

IV. DEFENSE SYSTEM DESIGN

Currently the two most popular SYN Flood defense technologies are SYN Cookie and SYN Proxy. However, both of them not only need a lot of extra calculations which will greatly reduce the efficiency of system, but also can't solve the problem of DoS attack essentially. Therefore, these two technologies cannot satisfy our requirements for SYN Flood attack defense.

The characteristic of SYN Flood attack is the large-scale attack traffic in a short time. If we can identify these attacks traffic as soon as possible, and try to discard the attack traffic directly rather than submit it to the protocol stack, then the ability of firewall to defense SYN Flood attack will be greatly improved. So this lead to the question: How to judge the attack traffic?

Sun[18] proposed a tree structure with M-MULTOPS to manage data. The M-MULTOPS used 256 trees to cover all IPv4 addresses, and each node of the tree is a table which store no more than 256 records. But for IPv6, its address space is 2^{96} times that of IPv4. So if we use the M-MULTOPS tree structure to defense SYN Flood in IPv6 that will need an enormous memory, which is almost impossible to realized in practice. Therefore, we propose a fast sample-based victim IP decision algorithm.

A. Sample-based Victims IP Decision Algorithm

Because the dual-stack firewall process both IPv4 and IPv6 traffic, so we need the victim IP decision algorithm to detect the two types of traffic simultaneously.

Our algorithm is based on the following facts: When there is a SYN Flood attack, we will find that there is only one attacked target in most cases, and then a large number of SYN packets which have the same destination IP address will appear in a short time. At this moment, if we make a sampling process to the SYN packets which are received by firewall, then the number of

attacked target IP addresses in the sampling window will far exceed the average of that in normal traffic.

Therefore we can count for the destination IP of the two types of traffic in sampling time window in order to do a quick analysis of the statistical results after sampling. Then we make the decision that the destination IP address, whose number of occurrences is more than a certain threshold, is the attacked target. In the subsequent process, we can directly discard the packet whose destination IP is the decided attacked target IP, and process the packets that doesn't contain the attacked target IP with program's normal procedure. Thus, this method implements the orientation filter to attack traffic, and no matter the attack traffic is launched by IPv4 or IPv6, it is effective.

w is the size of the sampling window, and n is the number of the kinds of destination IP address in the sampling window, so $1 \le n \le w$. c_i is the occurrence number of the ith $(1 \le i \le n)$ kinds of IP, then

$$w = \sum_{i=1}^{n} c_i$$

and

$$p_i = \frac{c_i}{w}$$

is the probability of occurrence of the ith kinds of IP. So if given a threshold t, when $p_i \ge t$, we can assert that the ith kinds of IP is the attacked target.

Otherwise, due to the complexity of the algorithm is $O(w^2)$ (w is the size of sampling window), the value of w should not be too big or too small. It will affect the efficiency of firewall if it is too big, while we cannot decide the victims IP because the amount of data is not enough if w is too small. The threshold t is the number of victims IP of the sampling window. We can estimate the threshold t under the normal network traffic by the statistics of the same IP addresses' mean frequency of the sampling window.

So in the actual implementation, we set w to 192 at first, namely sampling the IP destination address of 192 SYN packets. But when the system directly went into "Tolerance State" from "Discard State", we update the value of w to 512. We choose 5% for t, so that the threshold of c_i is 10 when $w = 192$, and increases to 26 after w update. And considering the memory alignment to make the access storage much faster, we finally set the threshold of c_i to16 or 32.

In order to realize the function of sampling and decision attack IP, we add two data structures: sample IP table and dangerous IP table. Sample IP table used to store the destination IP address of SYN packets when system sampling. And the dangerous IP table store the attack IP address which decision by sampling. Note that: In the program, for each thread for IPv4 and IPv6 are respectively need to maintain a sample IP table and a dangerous IP table. Table 4-1 shows the sampling algorithm.

Since we are not dealing with the attack traffic when judging the victim IP, we can call the data sampling state as "Tolerance State".

B. Fast Recovery Mechanism Based on Adaptive Threshold

After the firewall judges the victim IP in "Tolerance State", it changes to "Discard State". However, the time of the firewall in the "Discard State" cannot be too long. Because the longer it takes to discard packet, the more packets the firewall have to discard, and the greater impact it will have on the network communication for normal users. So it is necessary to bring in a fast recovery mechanism in the "Discard State".

Table 4-1: Sampling algorithm

Algorithm 1 Sampling to decide victim IP

INPUT:

packet_info: A packet's information which include source and destination IP address and port number, and some other information

OUTPUT:

Update sample ip tables and dangerous ip tables

```
01 PROCEDURE cache_sample_ips(packet_info)
02     IF packet_info.version equal to 4 THEN
03         ipv4_info ← get_ipv4_info(packet_info)
04         update sample_ipv4 by ipv4_info
05         IF sample_ipv4.count[ipv4_info] > t THEN
06             added ipv4_info in dangerous_ipv4
07         END IF
08     ELSE IF packet_info.version equal to 6 THEH
09         ipv6_info ← get_ipv6_info(packet_info)
10         update sample_ipv6 by ipv6_info
11         IF sample_ipv6.count[ipv6_info] > t THEN
12             add ipv6_info in dangerous_ipv6
13         END IF
14     END IF
15 END PROCEDURE
```

This mechanism must make the firewall rapidly return to "Normal State" from "Discard State" in order to minimize the packets being discarded and reduce the negative impact on the upper-application.

Fast recovery mechanism is improved on the basis of adaptive threshold algorithm: reducing the time interval of adaptive threshold algorithm to increase frequency of attack detection, and immediately return to "Normal State" once found there is no attack any more. We can use the simple decrease progressively method or the multiplicative decrease method, such as that is used in TCP congestion control, to reduce the time interval. But if the time interval is too short, the firewall will frequently switch between the states, and it will also affect the efficiency of the system. So we should restrict the times of time interval from decreasing. In our program, the times of reduction is less than 3, namely the time interval won't change after reducing for 3 times. Finally, if the fast recovery mechanism still detect SYN Flood attack, the firewall re-enter the "Tolerance State" and re-sampling to decide the victim IP, and the time interval of adaptive threshold algorithm should return to the initial value.

C. System Operating Process

At first, the system is in "Normal State". When it is detected as being attacked, it enters into "Tolerance State". In the "Tolerance State", system samples the received SYN packets. After finishing sampling, system judges out the victim IP and changes to "Discard State".

In the "Discard State", the firewall only discard the SYN packet whose destination IP is as the same as the victim IP. And it means that the "Discard State" of IPv4 module and IPv6 module in the protocol stack is separate. But the IPv4 and IPv6 should enter into the "Tolerance state" at the same time when the adaptive threshold algorithm detects SYN Flood attack. And it starts the fast recovery mechanism in "Discard State" to make the firewall quickly return to "Normal State".

Table 4-2: System operating process

Algorithm 2 Detection and Defense Algorithm

INPUT:

packet_info_streams: Date stream's information

OUTPUT:

What to do for this packet

```
01 PROCEDURE det_and_def(packet_info_streams)
02   FOR all packet_info_streams THEN
03     packet_info ← current packet_info_streams
04     IF sys.state is Tolerate State THEN
05       cache_sample_ips(packet_info)
06       check the dangerous_ipv4 or dangerous_ipv6 is empty or not
07       IF dangerous ip table is empty THEN
08           set sys.state to Normal State
09       ELSE
10           set sys.state to Discard State
11       END IF
12     ELSE IF sys.state is Discard State THEN
13       check whether this packet's ip is dangerous ip
14       IF packet_info.ip in dangerous ip table THEN
15           discard this packet
16       END IF
17       monitor the speed of current traffic
18       IF traffic.speed < k THEN
19           go to Normal State
20       ELSE
21           update k but less than 3 times
22           go to Tolerate State
23       END IF
24     ELSE
25       Detect whether our system is under attack
26       IF traffic.speed > k THEN
27           go to Tolerate State
28       END IF
29     END IF
30   END FOR
31 END PROCEDURE
```

Algorithm 2 (Table 4-2) shows the detection and defense algorithm of our system, where k is the threshold of traffic speed.

V. Testing Results and Analysis

In this section, we test the method of detecting and defending the SYN Flood attack in dual-stack network firewall that we proposed in the previous section.

We deploy the firewall in real network environment, and use the real network traffic to test how the detection-module and defense-module of the firewall would impact on the system performance. So the traffic that the firewall receives is uncontrollable and the result cannot be reproduced, then we can just give a statistical result by collecting data regularly. And we call it passive testing.

The network topology of our test is shown in Fig.5-1.

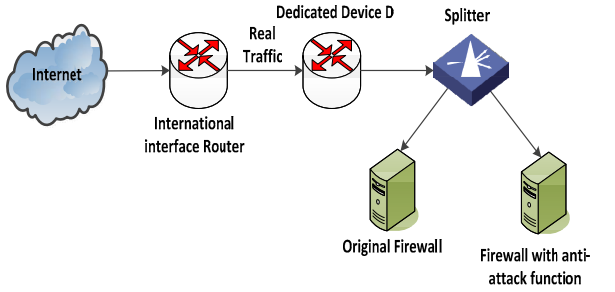

Figure 5-1: Network topology of passive testing

As shown in the network topology, we use a splitter to forward two copies of real traffic to the two firewalls. After the two systems start and run stably, we let the systems run continuously for 24 hours. And at this time, we check and record the memory usage rate and CPU usage rate every minute by using shell script.

Figure 5-2 shows the traffic of real network. Figure 5-4 is the change of the system's CPU usage rate.

In Fig.5-2 and Fig.5-3 we can see that the CPU usage of the two systems don't have big difference under the normal network. But when it is attacked, the original system's CPU usage rate would change sharply, while the other system's is obviously less than it, as Fig.5-4 shows. Furthermore, at most of the time, the CPU usage of the two systems is stable, and the usage of our procedure is very little. Those means our system is stably and our procedure occupies small amount of memory space.

In conclusion, the detection module and defense module only occupy very small amount of CPU, but they significantly improve system performance when it is attacked.

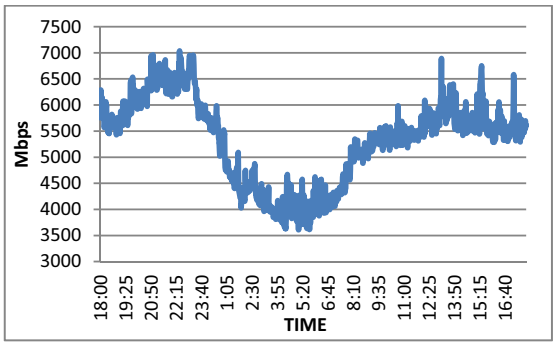

Figure 5-2: Real traffic

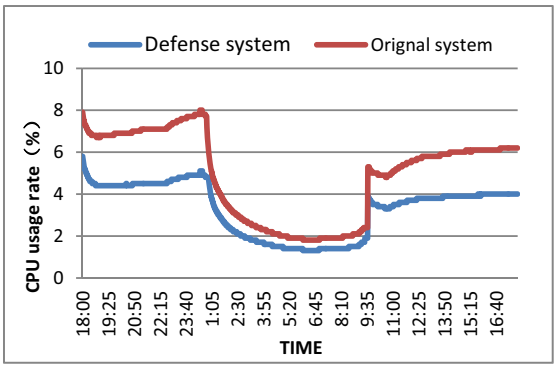

Figure 5-3: The comparison of two systems' CPU usage rate

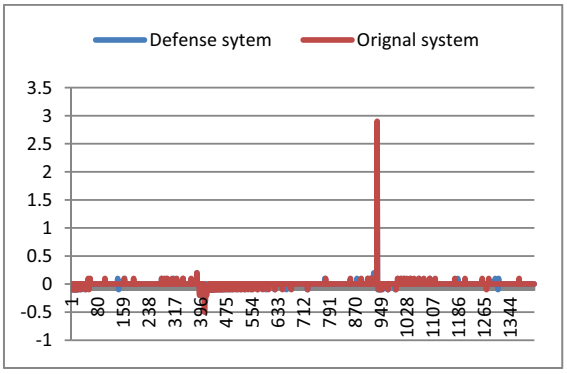

Figure 5-4: The change of two systems' CPU usage rate

VI. CONCLUSIONS

We introduce the theory of SYN Flood attack and a detection algorithm that can rapidly detect the SYN Flood attack under the large-scale network environment, namely an adaptive threshold detection algorithm. In order to satisfy the efficiency and safety of the firewall, we propose a sample-based victims IP decision algorithm, and a fast recovery mechanism, which are based on the adaptive threshold detection algorithm, to reduce the influence of discarding packets on the upper-application. Finally, we test the correctness and performance of the detection module and defense module. And the test results show that the increased detection module and defense module only occupy very small amount of memory and CPU, but they significantly improve system performance when it is attacked.

But there are also tow shortages in our paper:

We can make an active testing about our system to test the correctness of our system, and we can propose more performance indicator to measure the efficiency of the system completely in the later work.

The study of the application layer is not enough, and system still might be significantly affected under the application-layer attacks.

ACKNOWLEDGMENTS

The work presented in this paper is supported by the National "242" Information Security Program Funded projects No. 2012D100, National Natural Science Foundation of China under Grant No.61572153.

REFERENCES

[1] Kent S. Atkinson R. Security architecture for the internet protocol[S].IETF RFC 2401, November 1998.

[2] Weiser M. Whatever happened to the next-generation Internet?[J]. Communications of the ACM, 2001, 44(9): 61-69.

[3] Chen S, Nahrstedt K. An overview of quality of service routing for next-generation high-speed networks: problems and solutions[J]. Network, IEEE, 1998, 12(6): 64-79.

[4] Stoica I, Adkins D, Zhuang S, et al. Internet indirection infrastructure[C]. ACM SIGCOMM Computer Communication Review. ACM, 2002, 32(4): 73-86.

[5] Geoff Huston. BGP Table Statistics[EB/OL]. (2013-06-06) [2013-06-06]. http://bgp.potaroo.net/as1221.

[6] Geoff Huston, Anatomy-A look inside Network Address Translators[EB/OL]. (2004-08-01) [2013-06-06]. http://www.potaroo.net/papers/ipj/nats/anatomy.pdf.

[7] Deering S E. Internet protocol, version 6 (IPv6) specification[J]. 1998.

[8] Kang J, Zhang Z, Ju J. Protect e-commerce against DDoS attacks with improved D-WARD detection system[C].e-Technology, e-Commerce and e-Service, 2005. EEE'05. Proceedings. The 2005 IEEE International Conference on. IEEE, 2005: 100-105.

[9] Takada H H, Hofmann U. Application and Analysis of Cumulative Sum to Detect Highly Distributed Denial of Service Attacks Using Different Attack Traffic Patterns[EB/OL]. (2004-04-10) [2013-06-06]. http://www.istintermon.org.

[10] LIN Bai, LI Ou, LIU Qingwei. DDoS Attacks Detection Based on Sequential Change Detection[J]. Computer Engineering, 2005, 31(9): 135-137.

[11] CHENG Jun, LIN Bai, LU Jianzhi, LI Ou. Detection of SYN Flooding Attacks Based on Non-parametric CUSUM Algorithm[J]. Computer Engineering, 2006, 32(2): 159-161.

[12] CHEN Wei, HE Yan-Xiang, PENG Wen-Ling. A Light-Weight Detection Method Against DDoS Attack[J]. CHINESE JOURNAL OF COMPU TERS, 2006, 29(8): 1392-1400.

[13] SUN Chang-hua, LIU Bin. Survey on New Solutions Against Distributed Denial of Service Attacks[J]. ACTA ELECTRONICA SINICA, 2009,37(7):1562-1570.

[14] P Ferguson, D Senie. Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spooling[S]. RFC2667,1998.

[15] D J Bernstein. SYN Cookies [EB/OL]. (2000) [2013-06-06]. http://cr.yp.to/syncookies.html.

[16] LIU Qun-hua, MA Jin, XIA Zheng-min. A Stateful Inspection Model Based on Adaptive Threshold Algorithm for Defense of SYN Flood Attacks [J]. Information Security and Communications Privacy. 2010, 11(4): 84-86.

[17] Siris V A, Papagalou F. Application of anomaly detection algorithms for detecting SYN flooding attacks[C]. Global Telecommunications Conference, 2004. GLOBECOM'04. IEEE. IEEE, 2004, 4: 2050-2054.

[18] SUN Zhi-xin, TANG Yi-wei, GONG Jing. Novel wobble-defended M-MULTOPS structure and its application in detecting network abnormal traffic[J]. Journal on Communications, 2007, 28(8): 92-98.