Detect SYN Flooding Attack in Edge Routers

Yun Ling
Zhejiang Gongshang University, Hangzhou, Zhejiang, P. R. China
yling@zjgsu.edu.cn

Ye Gu
Zhejiang Gongshang University, Hangzhou, Zhejiang, P. R. China
zhuge011@126.com

Guiyi Wei
Zhejiang Gongshang University, Hangzhou, Zhejiang, P. R. China
weigy@ zjgsu.edu.cn

Abstract

Distributed Denial-of-Service (DDoS) attacks pose a serious threat to Internet security. Traditional detection methods rely on passively detecting an attacking signature and are inaccurate in the early stages of an attack. In this paper, we propose a novel defense mechanism that makes use of the edge routers that connect end hosts to the Internet to store and detect whether the outgoing SYN, ACK or incoming SYN/ACK segment is valid. This is accomplished by maintaining a mapping table of the outgoing SYN segments and incoming SYN/ACK segments and establishing the destination and source IP address database. From the result of simulation, the approach presented in this paper yields accurate DDoS alarms at early stage.

1. Introduction

Distributed Denial-of-Service (DDoS) is that 1) attackers exploits the technology of client/server, and 2) combine with multiple computers that are used as an attack platform, and 3) launch the attack of Denial-of-Service (DoS) for one or several targets, and hence 4) it will increasingly amplify the power of the DoS attack and make the target consume much system resources, finally, 5) the goal can not work normally. It mainly aims at larger site, such as commercial company, search engine and government site, and many famous website, such as Yahoo, eBay, CNN, Amazon etc., all suffer from the DDoS attack.

At present, most traditional defense mechanisms of against DDoS attack are effective only in later stages when attacking signatures are obvious. Therefore, there two disadvantages exist. First, the aggregation of numerous malicious packets on the victim server makes it difficult to launch an effective counterattack. Second, DDoS, at that time, has already brought great damage to the target server and wasted a lot of resources. Approaches deployed at the attack source-end, can mitigate the damage caused by DDoS attacks as much as possible. After detecting the attack, it can filter vicious data timely, and avoid waste of network resources. The most difficulty is that the number of malicious data is just a small part of whole data, and it is not easy to differentiate these from normal data. So it must record more network information, only in this way, can we reach an accurate detection effect. Certainly, recording more information means that it must consume some memory and calculative resources, but it is necessary to do that. Therefore, on the premise of accurate detection, we put forward an early detection method which be deployed on the edge router.

Most DDoS attacks exploit Transmission Control Protocol (TCP). The SYN flooding attacker always choose unreachable addresses as the spoofed source addresses of the attacking packets, which makes the compromised server cannot complete the necessary connection work. The method, which we propose, just judges the packet, which comes from edge router, whether it is malicious. Because all packets are emitted by end hosts. When sending a normal packet from a client (the client is not a puppet host but a normal host), if the network is also normal, this client will receive a relevant SYN/ACK packet from server. However, when sending a vicious packet from compromised client, this client can not receive a SYN/ACK packet from server. And then, the method which we propose determines whether the unavailable SYN/ACK packet is the result of SYN flooding or abnormal state of network, such as network congestion and so on.

To detect this SYN packet which is forwarded by edge router whether it is malicious, we use mapping table for noting down the record mapped by the hash function using the destination IP value of going SYN packet and ACK packet and the source IP value of coming SYN/ACK packet as input. At the same time, we also use counters for recording going SYN packet and ACK packet and coming SYN/ACK packet respectively. If the number of packets which is caused by the amount of going SYN packets subtracting the amount of going ACK packets, if these two kinds packet have same destination IP value, is more than the threshold given by the paper, our method will call detection module to judge whether attacker exists in the side of leaf router. Analogously, if the number of packets which is caused by the amount of coming SYN/ACK packets subtracting the amount of going ACK packets, if these two kinds packet have same destination IP value, is more than another threshold given by the paper, we can believe that one server is suffered from attack, and the attacker employ the innocent client IP (it maybe exist or not exist) as the spoofed source addresses of the attacking packets.

In order to detect SYN flooding attacks at its early stage, in this paper we have made the contributions as follows:

The approach adopts a novel mechanism to ensure detection SYN flooding attack at its early stage. Our approach is based on the fact that the number of SYN packets, SYN/ACK packets and ACK packets which are forwarded by leaf router are equal in normal network traffic. Further more, leaf router connect end hosts to Internet, so it can quickly detect some suspicious source address whether can be attained.

The approach mainly divides into two parts: storage and inspection. In order to send alarming information to victim server, we have to store the destination and source address of SYN packet and SYN/ACK packet. Therefore, we introduce a database instead of a memory data structure in our defense mechanism. Because it bears persistence and fault tolerance. And if any malfunction occurs, when the system goes up again, all data are still there and normal functionality can be resumed directly. The storage and inspection are mutual independent. Inspection is an active process, and the approach doesn’t depend on cooperation of other network devices.

The rest of paper is organized as follows. Section 2 discusses the related work. In section 3 the TCP-based DDoS attack will be discussed. Our detection method will be presented in Section 4. Some experiment results will be given in section 5 to evaluate the performance of the proposed method. In section 6 we will conclude our work and discuss future work.

2. Related works

A variety of attack detection and defense mechanisms to SYN flooding exist. Most of current DDoS attack detection and prevention schemes can be classified into four categories according to the location of the detector: at the victim server side, at the attack source side, between them, and at the innocent host side.

Detecting DDoS at the victim server side is more concerned by researchers, because people pay more attention to protection at this side. Wang [1] detected the SYN flooding attacks at leaf routers that connect end hosts to the Internet. The key idea of their method is the SYN-FIN pair's behavior should show invariant in normal network traffic and a non-parameter CUSUM method is utilized to accumulate these pairs. An approach utilizes the TTL (Time-To-Live) value in the IP header to estimate the Hop-Count of each packet in Jin C. [2] work. The spoofed packets can be distinguished by Hop-Count deviation from normal ones. Lemon J. [3] proposes an approach by using SYN Cache and Cookie method against SYN flooding attack. The basic idea is to use cache or cookies to evaluate the security status of a connection before establishing the real connection with a protected server. For those methods that provide detection at the victim server side, the main challenge is to consume all Cache and host when numerous connections are coming at the same time. The SYN Cookie can't keep any connection state of TCP, but calculate a certain Cookie value which is sent to the client. When client provide this Cookie in third round handshake, by calculating again, they can determine whether the Cookie value which is new is equal to the last Cookie value. Peng [4] compiled an IP address database of previous successful connections. When the network is suffering from congestion, an IP address that does not appear in the database seems more suspicious. A more proactive method was presented in [17] and this method applied an aggressive drop policy to identify attack traffic. The drop policies dropped a small number of packets and monitored the transient response from the clients. By comparing the dropping response to that predicted by the formula, it was possible to detect malicious traffic. Xiao [5] proposes an active detecting method against SYN flooding attack. This approach is based on the fact that the normal half-open connection maintained inside a server exists as a result of network traffic congestions while the half-open connections caused by a SYN flooding are launched only by attackers. But when attack comes across congestion, the accurate will be influenced. Sun [6] put forward a detection method which is based on link character. This method uses maximum likelihood estimation for estimate the distribution of link character and utilizes SOM neural network for link activity profile learning and anomaly link detection. Comparing estimated value and forecasted value which come from mentioned twotheories respectively, they can get a deviation value. If a deviation value of one link is more than a certain threshold, they can determine this link as abnormal link.

The methods deployed at the attack source side can filter attack data before these harm to Internet. Mirkovic [7] detected SYN flooding attacks at attack source. By monitoring outgoing and incoming data stream, and comparing with normal stream model, it can find action of attack. The RFC2827 [8] is to filter spoofed packets at each ingress router. Before the router forwards one packet to the destination, it will check whether the packet belongs to its routing domain. If not, this packet is probably a spoofed one and the router will drop it. SUN et al. [9] present a scheme to defend DDoS attacks based on the source and destination IP address database. The scheme establishes the source and destination IP address database (SDIAD) by observing the normal traffic and storages SDIAD in a three dimension Bloom Filter table. Then this paper cumulates and analyses the new pair of source and destination IP address based on the slide non-parametric cumulative sum (CUSUM) algorithm to detect the DDoS attacks quickly and accurately. Wang et al. [18] presents a simple and robust mechanism called SYN-dog to sniff SYN flooding sources. The core mechanism of SYN-dog is based on the protocol behavior of TCP SYN-SYN/ACK pairs. And due to its proximity to the flooding sources, SYN-dog can trace the flooding sources without resorting to expensive IP traceback.

Detection and prevention methods between these client and server sides mainly include traceback [10, 11], filtering [20], and pushback [12]. Traceback attempts to identify the real location of the attacker and mitigate the influence of the DDoS. During a DDoS attack, the source IP addresses are often forged and can not be used to identify the real location of the attack source. Most of the traceback schemes are to mark some packets along their routing paths or send some special packets, together with monitored traffic flow. With these special marks, the real routing path can be reconstructed and the true source IP can be located. With the identification of real path of the spoofed packets, pushback techniques can be applied to perform advanced filtering. The pushback is always performed at the last few routers before traffic reaches target.

Detection at innocent side is more hidden since it is deployed apart from attacking path. Attackers always choose IP addresses of innocent host as the spoofed source addresses of the attacking packet. B. Xiao and et al. [13, 14, 15] propose a novel cooperative system for producing warning of a DDoS attack. The system consists of a client detector and a server detector. The client detector is placed on the innocent client side. The server detector can actively assist the warning process by sending requests to innocent hosts. Besides that, they [16] also propose a rate-limit counteraction

3. TCP-based DDoS attack

For a DDoS attacking packet, its source IP address is usually forged. While Most DDoS attacks exploit TCP control packets by spoofing the three-way handshake between the source and the destination server. In this section we analyze the behavior of TCP control packets first in a normal three-way handshake and then in a spoofed three-way handshake.

Figure 1(a) shows a normal three-way handshake. First client C sends a $SYN(k)$ request to the serve S_1, which replies with a packet containing both the acknowledgement $ACK(k+1)$ and the synchronization request $SYN(j)$ and waits with a half-open connection in its memory space for the acknowledgement from the client C. Upon receiving both $ACK(k+1)$ and $SYN(j)$ client C will finish building the connection by sending $ACK(j+1)$. When server S_1 get $ACK(j+1)$, it removes previously stored half-open connections in its memory space. The released memory space on server S_1 makes it possible to handle further connection requests from clients and a network can run smoothly.

k And j are respectively sequence numbers produced randomly by the server and the client during the three-way handshake.

SYN, SYN/ACK and ACK all appear at both the edge router R_c .

Figure 1(b) shows a spoofed three-way handshake. The packet at the first round of a valid authentication process is a malicious one with a spoofed IP address. The edge router R_a in the attacker domain forwards the SYN packet with the spoofed address P_i , the IP address of the innocent host I , to the server S_2 . The server S_2 replies with an ACK/SYN packet and a half-open connection is pending. This ACK/SYN will be sent to the innocent host I because the server S_2 regards the SYN packet from I according to the spoofed source IP P_i . The edge router R_i on the innocent host side will receive the ACK/SYN packet but as no previous SY N request had been forwarded by the client detector at R_i , the ACK/SYN packet is dropped. The pending half-open connection on the server S_2 is maintained for a long time. More accumulated half-open connections will quickly consume all the memory space reserved for handling TCP requests and the server S_2 will deny any new requests. It is difficult to trace back the attacker's true address because the innocent host I , whose IP is used as the spoofed source IP, is usually not in the same domain as the attacker.

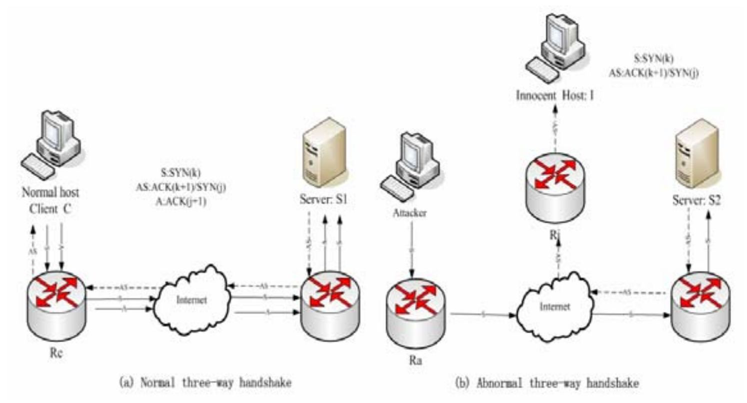

Figure 1. Normal and Spoofed three-way handshake

4. Defense mechanism

One of the main tasks for the defense mechanism is to monitor the TCP control packets that comes in and goes out of the domain. The approach mainly divides into two parts: storage module and inspection module. The storage module utilizes hash function to store source and destination address information into the database. The inspection module by using mapping table which is constructed in the storage process detect whether there have some abnormal cases. If some suspicious incidents exist, the module will call detection module to differentiate whether these abnormal events are caused by the SYN flooding attack. The structure of defense mechanism is shown in Figure 2.

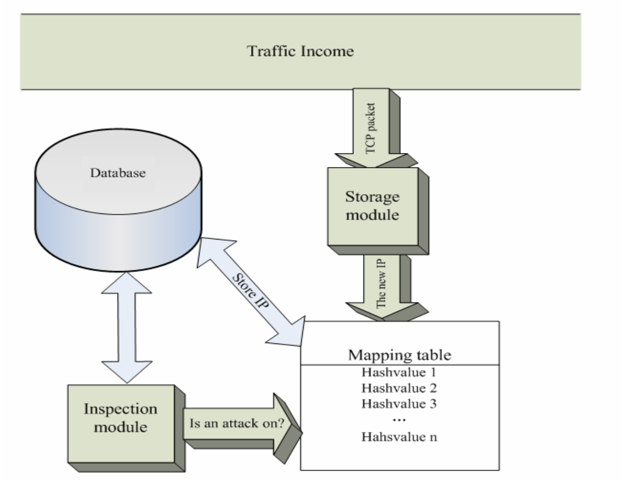

Figure 2. The structure of defense mechanism

4.1. Storage module

In this part, in order to record each destination address of SYN packet, this paper constructs HASH function for calculating mapping value of destination address. While the HASH function consists of k hash functions, which are mutually independent. Let $hash(i)$ be the hash value of s bit of destination address. Therefore, as for the address a, the total hash value can be computed by $HASH_a = \sum_{i=1}^{k} hash(i)$. However, the values computed by different hash functions are possible to be the same and their sums might be equal too. Thus, this will lead to the hash value collision. In order to improve this situation, we allocate different weight w_i to sequential hash functions, because do thus can reflect the difference among hash functions. This idea is similar to the thought of the Attenuate Bloom Filter [19]. In attenuate Bloom filters, higher filter levels are attenuated with respect to earlier filter level and it is a lossy distributed index. In fact, the advantage of out approach is that even if hash value encounters collision, inspection process of attack will have little influence. There are two main reasons. On the one hand, under normal condition, most three-way handshake can be finished in a very short period at the beginning phrase of the connection. The leaf router can get a SYN/ACK packet from server soon. So, it can avoid this condition about mistaking normal action for abnormal action. On the other hand, under attacking condition, if some normal server address has same hash value with one victim server, as long as the network works normally, we can only send alarming packet to victim, server.

Certainly, storage module needs two tables: mapping table (Table.1) and storing table (Table.2). The format for each table is provided as follows.

Table 1. Mapping table structure

hash value	counter (syn)	counter(syn/ack)	counter(ack)	explore mark T	attack mark E

Table 2. Storing table structure

hash value	source address	destination address	suspicious mark S	accurate mark A

In the mapping table, it contains hash value, counter(syn), counter(syn/ack), counter(ack), explore mark T and attack mark E. T refers to a explore mark. Initially, this field is set to 0, when the difference exceeds threshold, the value of this field is set to 1. E refers to an attack mark. Initially, this field is also set to 0. When we deduce attacking maybe happen by the method, the field is set to 1. Under other cases, such as the jam of network, the field is set to 2.

In order to get result quickly, the paper adopt array to store information. The first field in the mapping is hash value. We allocate different weight to sequential hash functions to compute the value. Do thus may be made the mapping table discontinuous, and waste the space of store. However, some conflicts also exist in the calculation of hash value. Therefore, these useless items can be used to store a verification of C. The verification value can be the value of the k hash function. For example, when malicious attack is found to one server, we let C equal to the value of the k hash function of the address of the victim server. Some SYN packets flow into leaf router, and we check the hash value of this packet. If it conflicts with the hash value of the victim server, we will further check whether the value of k function of this packet equals to C, and to determine whether the packet is sent to the affected server.

In the storing table, it contains hash value, source address, destination address, suspicious mark S, and accurate mark A. S refers to suspicious mark. Initially, this field is set to 0, when a SYN/ACK packet coming and the first round of three-way handshake dose not exists, this field is set to 1. Similarly, A is set to 0 at first, when normally finish the three-way handshake, the field is set to 1. The algorithm for storage module is described in Figure 3.

```
Storage_ Process (Input  packet p)
 if p is a SYN packet
 {
     h = hash(destination ip)
    if E= =1 or E= =2 then
    if  (hash(k)=  =C)     //k is the last hash function; C is a
                            verification value of victim server.
   drop p and break;
    else counter(syn)++;
      else h write into the mapping table;
      <h/source ip/destination ip> write into the storing table;
      counter(syn)++;
  }
  else if p is SYN/ACK packet
  {
      h = hash(source ip);
  }

  if already correlative destination ip in the database
  { count(syn/ack)++;}
  else
  {
      set S=1 and count(syn/ack)++;
          //S belong to storing table
  }
  if p is ACK packet  h = hash(source ip);
  if S=1  then S=0;
  set A=1;   //A belongs to storing table
  count(ack)++;
  }
```

Figure 3. Storage algorithm

4.2. Inspection module

The inspection module mainly divides into two parts: detection module and disposal module. The inspection module by checking mapping table which is constructed in the storage process detect whether there have some abnormal cases. If some suspicious incidents exist, the module will call detection module to differentiate whether these abnormal events are caused by the SYN flooding attack.

On the one hand, let d_1 be the difference which is the result of the number of SYN packets subtracting the amount of ACK packets. If d_1 is more than the threshold given in this paper, we define this instance as abnormal action. Generally, we view getting ACK packet as the accomplishment of three-way handshake. So, our method will call detection module.

In the detection module, the method chooses source address which was recorded in database as the destination address of the sniffing packet, and selects the IP address of leaf router as source address of the sniffing packet. And then this sniffing packet is sent to the destination host. By this means, similar to a 'ping' operation, it knows whether objective host exists. If something reply message returns, it will prove the host is living host and we can think that the SYN flooding attack have not taken place. Otherwise, we believe that the SYN flooding attack have taken place. Under this condition, method will call disposal module. In this module, we get the source and destination addresses from database, and construct RST packet to release half-open connections maintained by the victim server, set 1 to E (attack mark) in the mapping table, and generate verification C.

On the other hand, let d_2 be the difference which is the result of the number of SYN/ACK packets subtracting the amount of ACK packets. If d_2 is more than the threshold given by this paper, we also define this instance as abnormal action. This phenomenon shows that we always receive SYN/ACK packets from someone victim server. Then, we can infer that in the area of leaf router, there has an innocent host. So, our method also calls disposal module and sends RST packet to release half-open connections maintained by the victim server, and informs server what has happened. The algorithm for inspection module is defined in the Figure 4.

```
Inspection( Input mapping table)
For (i =1; i<= m; i++)  // m is the length of mapping table
{
    $d_1$ =Ri.count(syn)- Ri.count(ack);
        //Ri is record of mapping table
    $d_2$ = Ri.counter(syn/ack)- Ri.counter(ack);
    if( $d_1 > k_1$ && $d_2 < k_2$ )  set T=1;
        // $k_1, k_2$ are thresholds;
    if( $d_1 < k_1$ && $d_2 > k_2$ )  set S=1;
        if( $d_1 > k_1$ && $d_2 > k_2$ )  set T=1 and S=1;
        if( $d_1 < k_1$ && $d_2 < k_2$ ) then
         if (T= =1//E==1//E==2)
         set T=0; E=0;

 }
```

Figure 4. Inspection algorithm

4.3. Main process

From the process of SYN flooding attack, we can see in the area of leaf router there have three kinds of host, namely, normal host, innocent host, and attacker. Normal hosts always send normal packets; innocent hosts which may exist or may not exist in the source-end always receive abnormal packets; attackers always send vicious packets. Therefore, we set the defense mechanism on the leaf router. Its network topology is shown in figure 5.

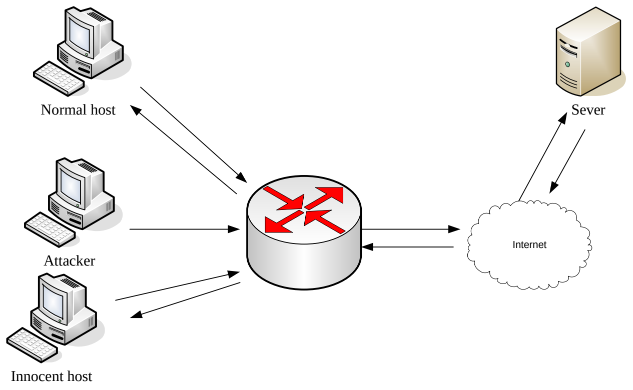

Figure 5. The network topology in which the proposed defense method can be installed.

The main object of our method is guarantee that each packet sent by client is valid as much as possible. So, the process is showed as follows.

Under the normal condition, the client sends a SYN request to the server. And then, the mechanism maps the destination IP address of this packet and stores the hash value, destination address and source address into the database, and counter of SYN packet increases one. Then, router receives a SYN/ACK packet from server. At the same time, it maps the source address of SYN/ACK packet. If the hash value already exists in the mapping table, we can compare the source and destination address of the replying packet with relevant address which has stored in database. If its address is matching, we just forward the packet and let counter of SYN/ACK packet increase one. Otherwise, we can store its addresses to database and forward it in the last. Then client send ACK packet to finish the three-way handshake. Similar, we map the destination address of ACK packet, and check the storing table, and let counter of ACK packet increase one. If its address has store in the database, we set 1 to A (Accurate mark).

Under attacking condition, attackers always choose many unreachable addresses as the spoofed source addresses of the attacking packet. Out detect mechanism record all SYN packet from the attackers to the database. But the leaf router can not receive any SYN/ACK packet from the victim server. When the inspection module finds that d_1 which has mentioned in the part 4.1 is more than a threshold. So, our method will call detection module, because we can't realize what reason causes that. The main task of detection module is to ascertain whether these addresses stored in database are true. If these addresses are living, we can think this phenomenon is caused by network reason. Otherwise, we can infer that attacker is living in the area of the leaf router. Then, we send a alarming packet, such as RST packet, release half-open connections maintained by the victim server, and inform the victim server what has happened, at last, we will stop hosts from sending TCP control packet to this victim server in a bit of time. Because attacking is starting at this time, but do thus also sacrifices benefits of some normal client which will sent packet to this victim server. However this is worthy to do that, because attacker will mistakenly think the network is congestion. Under the condition of jam network, launching a SYN flooding attack is meaningless and it will betray itself. So, attacker will stop attack action. If the phenomenon of not receiving SYN/ACK packet is the reason of congestion network, we also filter the packet to this server, because that can save the time of inspection.

Under the other condition, leaf router always receives SYN/ACK packets from the same victim server. Because we map the source addresses of these packet. We will find that d_2 which has mentioned in the part 4.1 is more than a threshold, but this destination address can't have relevant source address in the database. So, we will get the destination address from database and send an alarming packet to victim server. One is order to release the half-open connection. Other is order to inform the victim server what has happen. The algorithms for detection module and disposal module are defined individually in the Figure 6 and Figure 7.

```
Detection (Input mapping table )
for(i =1;i<=m;i++) // m is the length of mapping table
{
   if (T==1) { construct a sniffing packet;}
   if host exist  { E=2;
                    generate verification C;
                   }
   else { E=1;
           generate verification C;
          }
 }
```

Figure 6. Detection algorithm

```
Disposal (Input mapping table)
for (i =1;i<=m;i++)  m is the length of mapping table
{
    if (E==1) {
        send an alarming packet to victim server;
        filter packet of this address;
     }
     if (S==1){
        send an alarming packet to victim server;
        set S=0;
        set E=1;
        generate C;
     }
}
```

Figure 7. Disposal algorithm

5. Experiment

We develop a simulation system in a Linux software router and test our methods against several DDoS attacks. We use the libpcap facility to capture information about every packet and update the flow and connection statistics. A separate thread classifies connections and flows and determines the appropriate rate limit. Information about good connections and the desired rate limit is inserted into the kernel module through system calls. The module detects and forwards packets belonging to good connections from rate-limited flows and enforces the rate limit on the rest of the flow. At large packet rates, the libpcap monitoring facility cannot capture all packets. More accurate statistics are then obtained from the kernel module for rate-limited flows and good connections. Attack traffic mixture (relative ratio of TCP SYN, ICMP

ECHO and UDP packets), packet size, attack rate, target ports, spoofing techniques and attack dynamics can be customized.

Our experiment shows the time before the attack is detected with increasing rate attack. (as illustrated in figure 8)

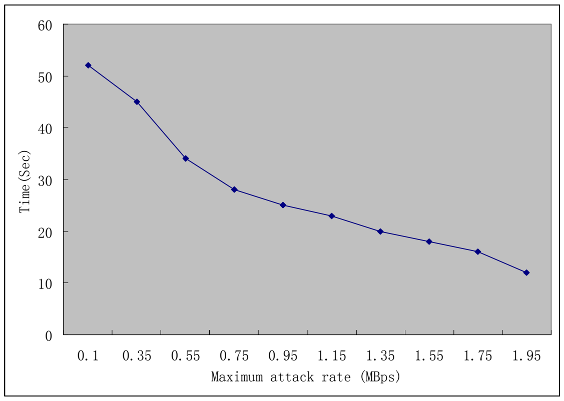

Figure 8. Time before the attack is detected

We determine the level of false positives by measuring the number of flow and connection misclassifications. We report this measure relative to the total number of flow and connection classifications performed during the trace. Table 3 presents the results for several traces.

Table 3. Percentage of false positives

Trace Number	Connection Misclassifications
1	0.103%
2	0.062%
3	0.025%
4	0.018%
5	0.015%
6	0.019%
7	0.008%
8	0.014%
9	0.006%
10	0.005%

To evaluate the defense mechanism, we also design some experiments to test whether the method can detect DDoS effectively.

When our defense method was applied, Fig.9 shows that the normal packets were forwarded to some sever. No matter how much the threshold k_1 and k_2 are, the difference d_1 and d_2 are always maintained 0. The result is showed in Figure 9(a), (b). Besides attacking, other reasons bring one difference more than one threshold, and the difference also can be recovered to 0 quickly, and amend the false alarm caused by the network, as showed in Figure 9(c).

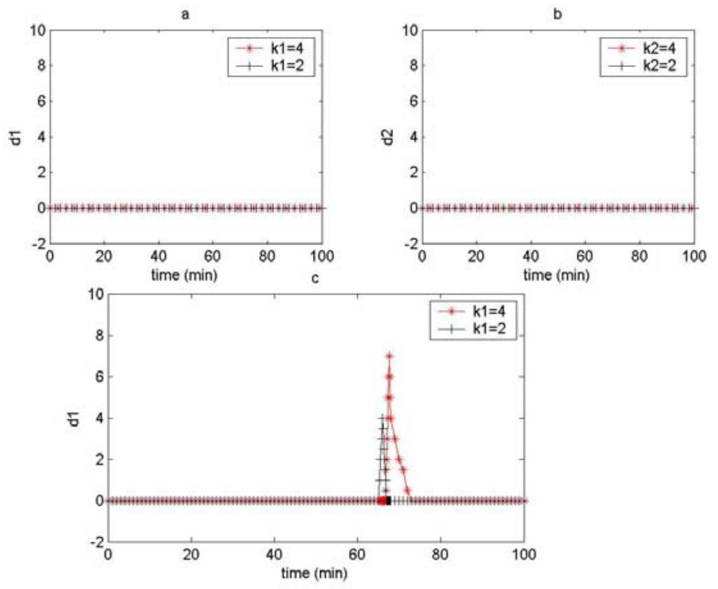

Figure 9. Under no attacking

Our defense method is deployed at leaf router, and users also can be divided into attacker, and the innocent.

Under attacking, attacker always chooses unreachable addresses as the spoofed source addresses of the attacking packets and sent them to victim server. Quickly, it breaks the symmetry relationship between SYN and ACK and the difference d_1 is more than threshold k_1 . In the Figure 10, legitimate users sent 20 normal SYN packets per second to the server. After 10s, attacker sent 400 abnormal SYN packets per second to the server. The attack lasts 100s. From the Figure 10(a), we can see that attacking flow increase quickly, and d_1 is more than k_1 rapidly, so the method calls detect module, and filters these attacking packets finally. Our method detects abnormal packets at 15s. Figure 10(b) show the speed of packets.

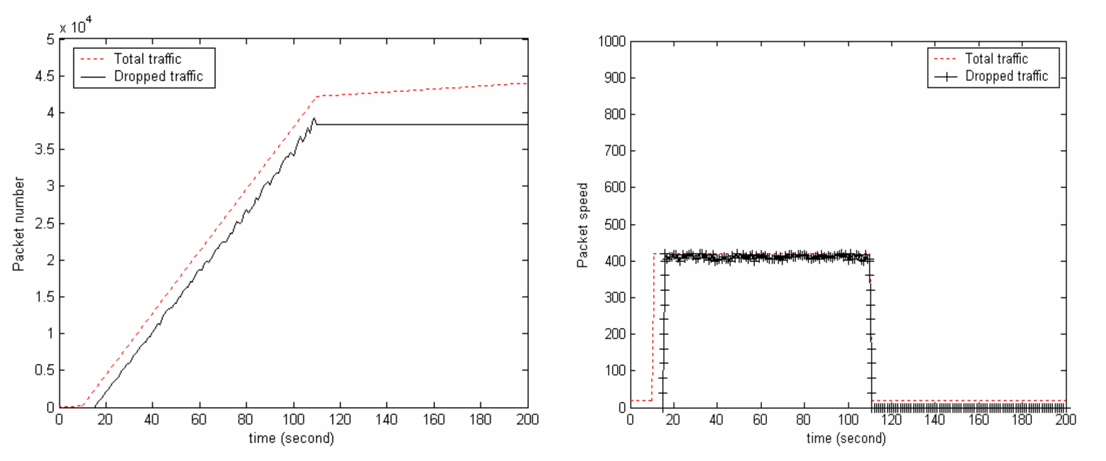

Figure 10. SYN Flooding attack (a) packet number figure (b) packet speed figure

The innocent host always receive SYN/ACK packet from outside one server. So, the symmetry relationship between SYN/ACK and ACK is broken very quickly and d_2 is more than k_2. The rate of the attack packets rises from 100 packets/sec to 400 packets/sec, the attack lasts 300s. The variety of d_2 is shown in Figure 11. In the simulation only a small part of packets were received by the victim server are designed to the router which are deployed with defense mechanism, but it is enough to generate alarming packet to victim server.

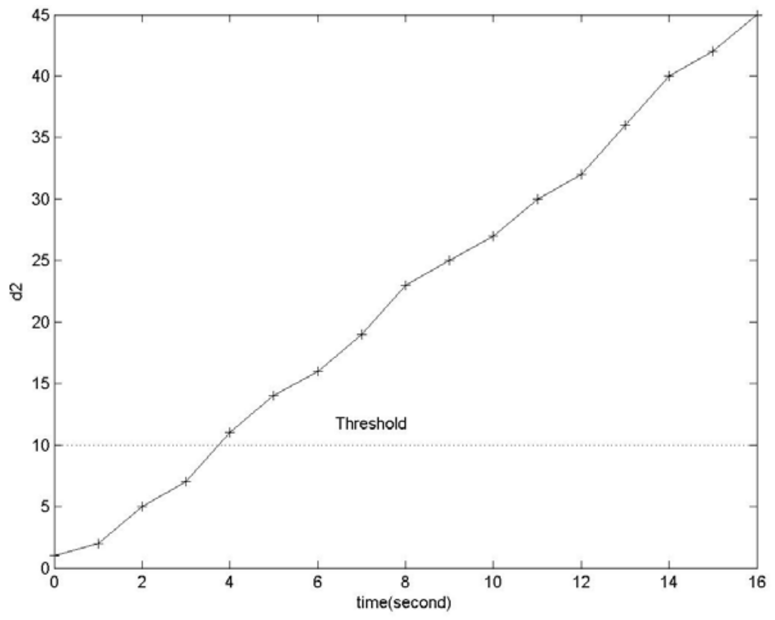

Figure 11. An alarm is generated when a curve goes beyond the threshold.

6. Conclusion and future work

This paper presents a response to SYN flooding attacks at early stage. The defense mechanism is deployed at leaf router, which can catch the character of TCP control packet easily. Using parameter d_1, we can determine whether the SYN flooding attack has happened. The advantage of our method is guarantee that each packet sent by client is valid as much as possible and can response to SYN flooding attacks at early stage. We can image that if every leaf router deploys this defense mechanism, the influence of SYN flooding attack will dramatically be weaken.

In the future work, the establishment of defense mechanism is still a priority. An effective defense mechanism is the key to deal with DDoS attacks. We can consider adopting new rules to determine the anomaly. For example we can record the average times from edge router to one server, or can consider cooperation with other edge routers, and share the database of source and destination IP. From another perspective, since we already know that there is an attacker in the edge router side, so we can determine which host emits malicious packets by the time interval of each host to router.

7. Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grants 60803161 and 60673179, Natural Science Foundation of Zhejiang Province of China under Grant number Z106727, Grand Science Project of Zhejiang Province of China under Grant number 2007C13068.

8. References

[1] Wang H., Zhang D., Shin K.G.. Detecting SYN flooding attacks, Proceedings of t he Annual Joint Conference of the IEEE Computer Society and Communications Society (INFOCOM), New York, N Y, USA, 2002, 3 : 1530 ~ 1539.

[2] Jin C., Wang H. N., Shin K. G., Hop 2count filtering: An effective defense against spoofed DDoS traffic. In: Proceedings of the 10th ACM Conference on Computer and Communication Security (CCS). Washington, DC, USA, 2003, 30-41.

[3] Lemon J., Resisting SYN flood DoS attacks with a SYN cache. In: Proceedings of the BSDCon 2002 Conference, San Francisco, CA , USA , 2002 , 89-97.

[4] T. Peng, C. Leckie, and R. Kotagiri, Protection from distributed denial of service attack using history-based IP filtering. In IEEE International Conference on Communications, Anchorage, Alaska, USA, volume 1, pages 482–486, May 11-15 2003.

[5]Bin Xiao, Wei Chen, Yanxiang He, Edwin H.-M. Sha, "An Active Detecting Method Against SYN Flooding Attack", in Proc. the 11th International Conference on Parallel and Distributed Systems (ICPADS'05), Volume I, pp 709-715, Fukuoka,Japan,20-22 July 2005.

[6]Hongjie Sun, Binxing Fang, Hongli Zhang, DDoS attacks detection based on link character. Journal on Communication, Vol.28, No.2, February 2007, pp.88-93.

[7] Mirkovic J ., Prier G., Attacking DDoS at t he source. In: Proceedings of the 10th IEEE International Conference on Network Protocols, Paris, France, 2002, 312-321.

[8] Ferguson P., Senie D., Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing. May 2000. [Online].http://www.ietf.org/rfc/rfc2827.txt

[9]Sun, Li, Defending DDos Attacks Based on the Source and Destination IP Address Database. Journal of Software, Vol.18, No.10, October 2007, pp.2613-2623.

[10] Branigan S., Burch H., Cheswick B., Wojcik F.. What can you do with Tracerouter IEEE Internet Computing, 2001,5: 96.

[11] Dequan Li, Purui Su, Dengguo Feng, Notes on packet marking for IP traceback. Journal of Software, 2004, 15(2):250-258.

[12] Ioannidis J., Bellovin S.M., Implementing pushback: Routerbased defense against DDoS attacks. In: Proceedings of the Network and Distributed System Security Symposium (NDSS),San Diego, California , 2002.

[13] Bin Xiao, Wei Chen, Yanxiang He, "A Novel Technique for Detecting DDoS Attacks at the Early Stage", Journal of Supercomputing (Springer),Vol. 36, pp.235-248, 2006.

[14] Bin Xiao, Wei Chen, and Yanxiang He, "A Novel Technique for Detecting DDoS Attack at its Early Stage", in Proc. Second International Symposium on Parallel and Distributed Processing and Applications (ISPA'2004) (LNCS), pp 825-834, Hong Kong, China, 13-15 Dec. 2004.

[15] Yanxiang He, Wei Chen and Bin Xiao, "Detecting SYN Flooding Attacks Near Innocent Side", Proc. International Conference on Mobile Ad-hoc and Sensor Networks (MSN 2005) (LNCS), pp 443-452, Dec. 2005.

[16] Bin Xiao, Wei Chen, and Yangxiang He, "An Autonomous Defense against SYN Flooding Attacks: Detect and Throttle Attacks at the Victim Side", Journal of Parallel and Distributed Computing (JPDC - Elsevier), Volume 68, Issue 4, April 2008, Pages 456-470.

[17] M. Kalantari, K. Gallicchio, and M. Shayman. Using transient behavior of TCP in mitigation of distributed denial of service attacks, Proceedings of the 41st IEEE Conference on Decision and Control 2002, volume 2, pages 1422–1427, 10-13 Dec. 2002.

[18]Wang H, Zhang D, Shin G., SYN-dog: sniffing SYN flooding source. Proceedings of the 22nd International Conference On Distributed Computing Systems (ICDCS'02), July 2002, p.421-429.

[19]Rhea S.C., Kubiatowicz J., Probabilistic location and routing, Proceedings of the IEEE INFOCOM, 3 (2002) 1248-1257.

[20] A. Yaar, A. Perrig, D. Song, SIFF: A stateless internet flow filter to mitigate DDoS flooding attacks, in: Proceedings of the IEEE Symposium on Security and Privacy, IEEE Computer Society, IEEE Computer Society Press, Silver Spring, MD, 2004, pp. 130–143.

Authors

Guiyi Wei. He received the MA Degree from the Zhejiang Gongshang University, China, in 2000 and the Ph.D. Degree in Computer Science from the Zhejiang University in 2006. He is now an Associate Professor in the Department of Computer Science at the Zhejiang Gongshang University. His research interests are in the areas of grid computing, peer-to-peer computing, and electronic commerce. He is a member of the IEEE.