Detecting SYN Flooding Attacks Near Innocent Side*

Yanxiang He[1], Wei Chen[1], and Bin Xiao[2]

[1] Computer School, The State Key Lab of Software Engineering, Wuhan University, Wuhan 430072, Hubei, China
{yxhe, chenwei}@whu.edu.cn
[2] Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
csbxiao@comp.polyu.edu.hk

Abstract. Distributed Denial-of-Service (DDoS) attacks seriously threat the servers in the Internet. Most of current research is focused on the detection and prevention methods at the victim side or the source side. However, defense at the innocent side, whose IP is used as the spoofed IP by the attacker, is always ignored. In this paper, a novel method at the innocent side has been proposed. Our detection scheme gives accurate detection results using little storage and computation resource. From the result of experiments, the approach presented in this paper yields accurate DDoS.

1 Introduction

Distributed Denial-of-Service (DDoS) attacks are a large-scale cooperative attack, launched from a large number of compromised hosts. DDoS attacks have posed a major threat to internet since 1990's and they have caused some popular web sites on the world, such as Yahoo, eBay, Amazon, become inaccessible to customers, which caused huge financial losses. Current events have shown that DDoS attacks continue to bring increasing threats to the internet. While many methods have been proposed, there still is a lack of efficient defense.

Most DDoS attacks exploit Transmission Control Protocol(TCP)[1]. It has been shown that more than 90% of the DoS attacks use TCP[2]. The most efficient and commonly-used SYN flooding attacks[3, 4] exploit the standard TCP three-way handshake in which the server receives a client's SYN (synchronization) request and replies with a SYN/ACK (synchronization/acknowledge) packet. The server then waits for the client to send the ACK (acknowledge) to complete the handshake. While waiting for the final ACK, the server maintains a half-open connection. As more and more half-open connections are maintained on a victim server, DDoS attacks will deplete the server's resources.

* This work is supported by the National Natural Science Foundation of China under Grant No. 90104005 and partially by HK Polyu ICRG A-PF86 and CERG Polyu 5196/04E.

X. Jia, J. Wu, and Y. He (Eds.): MSN 2005, LNCS 3794, pp. 443–452, 2005.

Lots of research work has been done to detect and prevent the TCP based DDoS attack. According to the deployment location of defense systems, the current DDoS detection and prevention methods can be classified into three categories: defense at the source-end, at victim-end or at intermediate-network. However, the information at the innocent host, whose IP is utilized as the spoofed IP, is totally ignored. Each kind of mechanisms has its limitations, for example detection at the side of a victim server can hardly produce an alarm at the early stage because abnormal deviation can only be easily found until the DDoS attack turns to the final stage. Furthermore, it is difficult for victim side to take efficient response after DDoS is detected due to numerous malicious packets aggregating at this side. Providing an early DDoS alarm near the source is a difficult task because the attack signature is not easy to capture at this side. The information at the innocent host side will be used in our approach because the innocent side will receive abnormal TCP control packets during a DDoS attack. Compared to other detection mechanisms, defense at the innocent host has two main advantages:

- Detection at innocent side is more hidden since it is deployed apart from attacking path. To avoid being detected, attackers usually sniffer and deceive defense systems deployed around victims and attacking source before launching attacks. It is difficult for attackers to be aware of the existence of detection mechanism at the innocent side.
- It has little vulnerability to DDoS attack. The burden of monitoring numerous attacking packets congesting at the victim side makes the defense system itself vulnerable to DDoS attack. Defense at innocent side will avoid this problem due to limited attack streams near innocent side. This enables the defense system itself to have little risk of becoming potential target of DDoS attacks.

Detection at innocent side has several challenges. On the one hand, accurate detection is not easy to achieve since the abnormal signature is distributed in a backscatter way [2]. To capture the small quantity of attack signatures, more packets should be monitored and recorded for analysis, which implies expensive storage cost. On the other hand, defense at innocent side is deployed far from the victim side and detection effect depends on the number of participant of Internet Service Providers(ISPs). To attract more ISPs to participate detection, the detection scheme should use limited resource in defense, which will not bring evident degradation to ISPs' service. In this paper, we provide a detection scheme which gives satisfying result with little storage and computation requirement.

In this paper we propose a novel detection method against SYN flooding attack near innocent side. We summarize our contributions as follows:

- The detector performs detection at the side of innocent hosts because this side can provide valuable information. Another benefit of such deployment is to make the detection system itself invulnerable to DDoS attacks.
- A Bloom filter based detection scheme is proposed. We apply Bloom filter to DDoS detection, which can record huge traffic information on high speed network.

– With this space-efficient data structure, detection scheme monitors abnormal handshake with little computation overhead. The detection scheme only requires simple hash function operation, addition operation and subtraction operation, which bring little overhead to present computers.

The remainder of the paper is organized as follows: Section 2 will introduce some related works in spoofed DDoS detection. In Section 3, the TCP-based DDoS attack will be discussed and analyzed. The techniques for DDoS detection at innocent side will be proposed 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 The Related Work

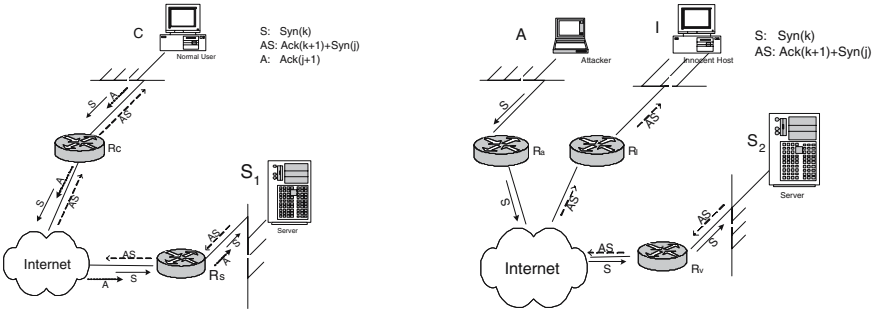
Hash table is a high performance data structure used for quick data look up and is versatilely applicable in network packet processing. The Bloom filter is a kind of space-efficient hash data structure, which is first described by Burton Bloom [5]. It is originally used to reduce the disk access times to different files and other applications and now it has been extended to network packet processing. Song present a hash table data structure and lookup algorithm using extended Bloom filter, which can support better throughput for router applications based on hash tables. NetFlow [6] maintains a hash table of connection record in DRAM and monitors network traffic. The concept of multiple hashing, which is similar to Bloom filter, was used to track large flows in network traffic.

Hash table is also used to defend DDoS attack. Snoeren [7] present a hash-based technique for IP traceback that generates audit trails for traffic within the network, and can trace the origin of a single IP packet delivered by the network in the recent past. Hash table is employed to look for an imbalance between the incoming and outgoing traffic flows to or from each IP address [8]. IDR [9] is a router equipped with DDoS protection mechanism, which uses Bloom filter to detect DDoS attack.

3 The TCP-Based DDoS Attack

In this section, we analyze the difference between normal TCP handshakes and spoofed one. During spoofed DDoS attack, the source IP address of attacking packet is usually modified, which is not the attacker's IP address anymore. The normal three-way handshake to build a connection would be changed consequently.

The normal three-way handshake is shown in Figure 1(a). First the client C sends a $Syn(k)$ request to the server S_1. After receiving such request, server S_1 replies with a packet, which contains both the acknowledgement $Ack(k+1)$ and the synchronization request $Syn(j)$. Then client C sends $Ack(j+1)$ back to finish the building up of the connection. k and j are sequence numbers produced randomly by the server and client respectively during the three-way handshake.



(a) Normal three-way handshake (b) Spoofed three-way handshake

Fig. 1. The process of the TCP three-way handshake

In the remainder of the paper, SYN shows a message sent to server S inside the TCP control packet during the first round of the three-way handshake protocol. A packet containing both $Ack(k+1)$ and $Syn(j)$ (denoted as ACK/SYN in the following sections) is delivered back from server S in the second round. A control package with $Ack(j+1)$(denoted as ACK) involves in the third round. During the normal three-way handshake procedure, SYN, ACK/SYN and ACK can be observed at the edge router(R_c in Figure 1(a)) near the client.

If the packet at the first round is a malicious one with a spoofed IP address, a valid authentication process is modified. As Figure 1(b) shown, the innocent host I, whose IP is used as spoofed source IP, is usually not in the same domain with the attacker host A. In other words, the edge router working for attacker host A does not route for the innocent host I. In fact, to avoid being traced back, the attacker usually uses the IP address belonging to other domains to make a spoofed packet. Under this assumption, there exists difference between the normal TCP three-way handshake and the spoofed one. In Figure 1(b) 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 sever S_2 replies with an ACK/SYN packet. This ACK/SYN will be sent to the innocent host I because the server S thinks the SYN packet is from I according to the spoofed source IP P_I in it. So the edge router R_I at the innocent host side will receive the ACK/SYN packet from victim server S_2. But there is no previous SYN request forwarded by the client detector at R_I. This scenario is different from the normal one. Our approach is proposed on the base of this difference.

4 The Bloom Filter Based Detection Scheme

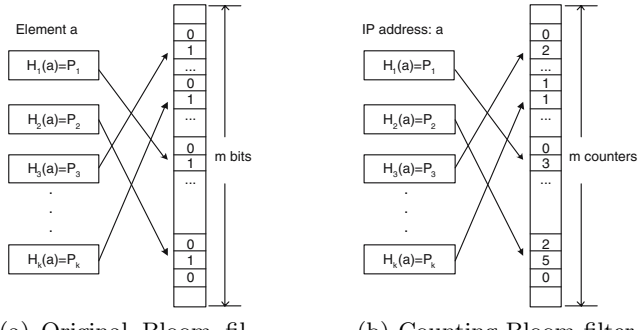
In order to detect DDoS, the TCP control packets for handshakes are monitored and analyzed at the edge router of innocent side. For example, the detector will be installed on the router R_I in Figure 1(b). To save storage cost, a Bloom filter based method is applied to monitor two-way traffic between innocent side and

the rest of Internet. It checks the TCP control packets flowing through the edge router. When it captures suspicious handshakes, the alarm about the potential DDoS attack will be launched.

4.1 The Bloom Filter Based Hash Table

The basic idea of innocent-side detection is to monitor two-way TCP control packets and capture abnormal handshakes. A TCP connection may hold for several seconds or even for several minutes but most three-way handshake can be finished in a very short period(e.g., less than 1 seconds) at the beginning phrase of the connection. However, it is expensive to keep a record for each handshake considering numerous traffic volume on the high speed link network. To record useful information with limited storage, Bloom filter, a kind of space-efficient hash data structure, is applied in our method.

Original Bloom Filter. Bloom filter is first described by Burton Bloom [5] and originally used to reduce the disk access to differential files and other applications, e.g. spell checkers. Now it has been extended to defend against DDoS attack [7, 8, 9]. The idea of Bloom filter is to allocate a vector v of m bits, initially all set to 0, and then choose k independent hash functions, $h_1, h_2, \ldots, h_k$, each with range $\{1, \ldots, m\}$. For each element $a \in A$, the bits at positions $h_1(a), h_2(a), \ldots, h_k(a)$ in v are set to 1(Figure 2). Note that a particular bit might be set to 1 multiple times which may cause potential false result. Given a query for b we check the bits at positions $h_1(b), h_2(b), \ldots, h_k(b)$. If any of them is 0, then certainly b is not in the set A. Otherwise we conjecture that b is in the set. However there is a certain probability that Bloom filter give false result, which is called a "false positive". The parameters k and m should be chosen such that the probability of a false positive is acceptable.



(a) Original Bloom filter

(b) Counting Bloom filter

Fig. 2. Bloom filter uses independent hash functions to map input into corresponding bits

Counting Bloom Filter. The original Bloom filter is not suitable to monitor handshakes. We will use a variant of a Bloom filter called *Counting Bloom filter* which substitutes m bits with countable integers table as shown in Figure 2(b).

After using counters table to replace m bit array, all the counts are initialized to 0. When a key is inserted or deleted, the value of count is incremented or decremented by 1 accordingly. When a count changes from 0 to 1, the corresponding bit is turned on. When a count changes from 1 to 0 the corresponding bit is turned off. The value in the count indicates the current statistic results of traffic.

4.2 Detection Scheme

To detect attacking traffic with spoofed source IP, the destination IP(the server's IP) is recorded in the hash tables. When a SYN packet, the TCP control packet for the first round handshake, is captured from the outgoing traffic, the destination IP(the server's IP) is hashed into the hash table by k independent hash functions. For the output of each hash function, if the corresponding counter is 0, the corresponding counter is turned on. If the counter is already turned on, the counter is incremented by 1 accordingly. If corresponding ACK/SYN packet for the second round of handshake is soon captured in the incoming traffic. The source IP(the server's IP) is hashed into the hash table again. But this time the corresponding counter is decremented by 1 for each independent hash function. When a count changes from 1 to 0, the corresponding counter is turned off. For a normal TCP handshake, ACK/SYN packet can be hashed to a turn-on counter by each independent hash function because the corresponding counter has already been turned on by previous SYN packet. The counter will keep unchanged if the first two rounds of three-way handshake are completely captured at the ingress and egress router at the innocent side. The detection scheme is depicted in Figure 3. These counts are reset to 0 for every period t.

On the contrary, in the scenario of the spoofed handshake, suspicious ACK/SYN packet should meet at least turn-off counter since previous SYN

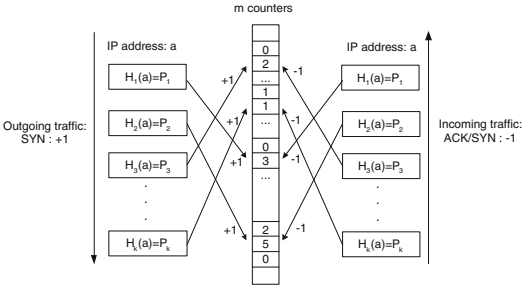


Fig. 3. The detection scheme increases or decreases the value of the count according to the three-way handshake

```
Detection_Scheme (INPUT: P) {
if P is a SYN packet then
  for i = 0; i < k; i++ do
    j=Hash_i(P)
    Counter_j++
  end for
else if P is a ACK/SYN packet then
  for i = 0; i < k; i++ do
    j=Hash_i(P)
    //Check whether exists turn-off counter
    if Counter_j == 0 then
      Report Suspicious Alarm(SA)
      RETURN
    end if
  end for
  //If no turn-off counter, do subtraction
  for i = 0; i < k; i++ do
    j=Hash_i(P)
    Counter_j- -
  end for
end if
RETURN }
```

Fig. 4. The Bloom filter based Detection Scheme

packet has turned on the counter. During a DDoS attack, the second round handshake packets, ACK/SYN, are backscattered according to the spoofed source IP of attacking packets. The innocent side will receive part of these ACK/SYN packets and try to hash them with k independent hash functions. It is possible for the abnormal ACK/SYN to hit one turn-on counter since there may exist collision for one of hash functions. However, the possibility of hitting all k turn-on counters is rather low since it is not likely that all k independent hash functions have collisions at the same time. If ACK/SYN packet hits one turn-off counter, the detailed information of this packet is recorded for further analysis. The detection scheme only requires addition and subtraction operations. These operations bring little overhead to system considering today's computation ability.

When a new Suspicious Alarm(SA) is reported, the detector will analyze the source IP distribution of SAs in database. Assumed a DDoS attack takes place, the detector will find asymmetric ACK/SYN packets sent from victim server with its IP P_{victim} as the source IP. When SAs with the same source IP P_{victim} are reported in a short period, there probably exists a DDoS attack targeting the host P_{victim}. But if each SA has a different source IP, it is most likely caused by some reasons other than a DDoS attack. To evaluate the distribution of the source IP of the alarms, an expression is presented below:

$$score = \sum_{s \in IPList} (|X_s| - 1)^2$$

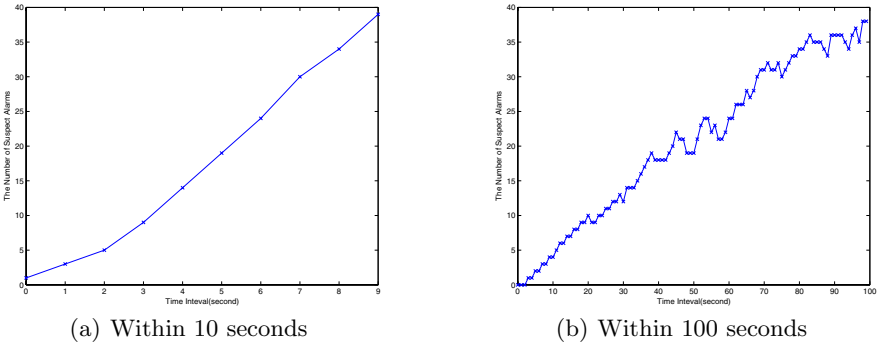
Where X_s stands for a subset of the total SA set. All the elements in X_s are SAs that have the same IP value s in a certain period. The score will increase dynamically when the number of SAs with the same source IP increases. On the other hand if each of the SAs has a different source IP, the score will reach minimum.

5 Experiment

Experiments are designed to evaluate the detection method. In the experiment, 10 zombies are simulated to perform SYN flooding attacks toward the server. The rate of the attack packets rises from 10 packets/sec to 1000 packets/sec in 10 seconds and 100 seconds separately. The maximum rate is set to 1000 packets/sec because it is enough to shut down some services as Chang reported [10]. In simulation only 1% of ACK/SYN packets replied by the victim server are designed to arrive detectors near innocent side. These packets will trigger detectors to generate SAs. The number of SAs generated by the client detectors is shown in the Figure 5.

Although only 1% spoofed attack packets can be received by detector, detector still give accurate SAs. The number of SAs is enough for detector to give a further potential DDoS alarm at the early stage of the DDoS attack. The detection results are satisfying even when the DDoS attacker increases the attack packets slowly. From experiment results, the SAs number raises stably in the Figure 5(a) because the DDoS attack is launched in a short time. In the Figure 5(b) the number of SAs fluctuates a little because the attacking packets rise up at a much slower rate.

The storage and computation cost of our scheme is compared with another hash method used in Snort [11]. Snort uses a hash method to monitor network traffic and each connection is recorded as a 5-tuple entry. In simulation, 1000000 IP addresses are randomly generated and inserted into hash tables using our hash



(a) Within 10 seconds (b) Within 100 seconds

Fig. 5. The number of SA generated by the detector after receiving 1% of ACK/SYN packets from the protected server. The attacking packets rate reaches to 1000 packets/sec within (a)10s (b)100s.

Table 1. Comparison of time and storage consumption between 5-tuple hash and our hash method

Hash method	Time Consumption(sec)	Storage Consumption(byte)
5-tuple hash	0.328	14336
Our hash method	0.187	4096

method and the 5-tuple hash method. The simulation platform uses a Pentium 4 1.7G processor and 256M memory. The time and storage consumption are compared and list in Table 1. It shows that our method uses less memory and spends less time than 5-tuple hash method.

6 Conclusion and Future Work

In this paper a novel detection method against the DDoS attack is presented. The detection system is deployed at the innocent side, which is quite different from current methods which are often deployed at the sides of the victim server or the attacking source. Detection at innocent side makes defense more hidden to attackers. The basic idea in the detection is to differentiate between the normal TCP three-way handshake and the spoofed one. The proposed Bloom filter based detection scheme can give accurate detection with little storage cost. The experiment results are given in section 5 and the proposed approach is effective to detect DDoS.

In the future research work we will apply method to real environment to test the memory and computing cost for actual running. The optimization of k, m will also be studied.

References

1. Postel, J.: Transmission Control Protocol : DARPA internet program protocol specification,RFC 793 (1981)
2. Moore, D., Voelker, G., Savage, S.: Inferring internet Denial of Service activity. In: Proceedings of USENIX Security Symposium, Washington, D.C, USA. (2001) 9–22
3. Wang, H., Zhang, D., Shin, K.G.: Detecting SYN flooding attacks. In: Proceedings of Annual Joint Conference of the IEEE Computer and Communications Societies(INFOCOM). Volume 3. (2002) 1530–1539
4. Schuba, C.L., Krsul, I.V., Kuhn, M.G., Spafford, E.H., Sundaram, A., Zamboni, D.: Analysis of a denial of service attack on TCP. In: Proceedings of the 1997 IEEE Symposium on Security and Privacy, IEEE Computer Society, IEEE Computer Society Press (1997) 208–223
5. Bloom, B.H.: Space/time trade-offs in hash coding with allowable errors. Communications of the ACM **13** (1970) 422–426
6. Estan, C., Keys, K., Moore, D., Vargese, G.: Building a better NetFlow. In: ACM SIGCOMM. (2004) 39–42

7. Snoeren, A.C.: Hash-based IP traceback. In: Proceedings of the ACM SIGCOMM Conference, ACM Press (2001) 3–14
8. Abdelsayed, S., Glimsholt, D., Leckie, C., Ryan, S., Shami, S.: An efficient filter for denial-of-service bandwidth attacks. In: IEEE Global Telecommunications Conference(GLOBECOM'03). Volume 3. (2003) 1353–1357
9. Chan, E., Chan, H., Chan, K., Chan, V., Chanson, S., etc.: IDR: an intrusion detection router for defending against distributed denial-of-service(DDoS) attacks. In: Proceedings of the 7th International Symposium on Parallel Architectures, Algorithms and Networks 2004(ISPAN'04). (2004) 581–586
10. Rocky.K.Chang: Defending against flooding-based distributed denial-of-service attacks: a tutorial. Communications Magazine, IEEE **40** (2002) 42–51
11. Snort: (Open source network intrusion detection system, http://www.snort.org)