An Active Detecting Method Against SYN Flooding Attack *

Bin Xiao† Wei Chen‡ Yanxiang He‡ Edwin H.-M. Sha§

†Department of Computing
The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
‡Computer School, Wuhan University, Wuhan 430072, Hubei, China
§Department of Computer Science
University of Texas at Dallas, Richardson, Texas 75083
E-mail: csbxiao@comp.polyu.edu.hk, {chenwei,yxhe}@whu.edu.cn, edsha@utdallas.edu

Abstract

SYN flooding attacks are a common type of Distributed Denial-of-Service (DDoS) attack. Early detection is desirable but traditional passive detection methods are inaccurate in the early stages due to their reliance on passively sniffing an attacking signature. The method presented in this paper captures attacking signatures using an active probing scheme that ensures the efficient early detection. The active probing scheme–$DARB$ obtains the delay of routers by sending packets containing special Time-to-Live set at the IP headers. The results of the probe are used to perform SYN flooding detection, which is reliable and with little overhead. This approach is more independent than other methods that require cooperation from network devices. Experiments show that this delay-probing approach distinguishes half-open connections caused by SYN flooding attacks from those arising from other causes accurately and at an early stage.

1 Introduction

The issue of defending Distributed Denial of Service(DDoS) [13] attacks have been addressed in recent years. Traditional passive detection methods are inaccurate in the early stages due to their reliance on passively sniffing an attacking signature, an approach that is effective only in later stages when attacking signatures are obvious. There is still a lack of efficient methods for detecting such attacks at an early stage.

Most DDoS attacks exploit Transmission Control Protocol(TCP) [9]. In the TCP case, the SYN flooding attack [19, 15] is the most efficient and common-used one which exploits the standard TCP three-way handshake. In the TCP three-way handshake, when the server receives a client's SYNrequest, it replies with a SYN/ACK packet and then waits for the client to send the ACK to complete the three-way handshake. While waiting for the final ACK, the server maintains a half-open connection. Since the SYN flooding attacker always chooses unreachable addresses as the spoofed source addresses of the attacking packets, the server will not receive the anticipated final ACK from the client. Given that the server has limited resource for new connections, it is unable to provide service to the forthcoming connection request.

It is important to detect SYN flooding attacks at an early stage before there are a large number of half-open connections maintained by the protected server. Early detection also allows sufficient time for defense responses such as filtering, pushback and traceback.

To improve detection efficiency, an active approach is preferred to a passive one. Thought traditional passive methods can give accurate detection result at the later stage when attack signatures become evident, they are inaccurate in the early stages due to their reliance on passively sniffing an attacking signature. Given this, an active approach to detection is preferred to announce early DDoS attacks and the delay probing method $DARB$ is proposed in this paper.

An active detection approach begins by determining whether a half-open connection is the result of SYN flooding. The normal half-open connections are usually caused simply by network congestion while half-open connection originated from a SYN flooding attack has no relevancy to the network traffic congestion. If the delay between the server and the client is much longer than the normal delay, the likely cause is a congested router, but if no signs of congestion feature are found, the half-open connection is re-

*This work is partially supported by HK Polyu ICRG A-PF86 and CERG Polyu 5196/04E, by the National Natural Science Foundation of China under Grant No. 90104005, and also by TI University Program, NSF ETA 0103709, Texas ARP 009741-0028-2001 and NSF CCR-0309461.

garded as abnormal and maybe the result of a SYN flooding attack.

The delay between the server and the client is estimated using a delay probing method–$DARB$. By setting different time-to-live(TTL) values at the IP headers, the packets will die at different routers. Information about the death of packets is sent by the relevant router and this provide information for estimating the delay along the path. A score is then given to the delay value which allows an evaluation of the probability of the half-open connection being the result of a SYN flooding attack.

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

- The approach adopts a novel mechanism to ensure detecting SYN flooding attack at its early stage. Our 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.

- The detection method is active and independent. The approach take a more active mechanism to search SYN flooding feature instead of passively sniffing attack signature. Such active approach is deployed at the server side which does not depend on cooperation of other network devices.

- A reliable mechanism with little overhead is applied to probe the delay between the server and the client. Our $DARB$ probing method is a more reliable mechanism than the direct probing method like `ping`. The selective probing scheme reduce the overhead involved in the proposed detecting technique.

The rest of the paper is organized as follow. Section 2 discusses the related work. Our active detection method will be discussed in Section 3. Section 4 describes a simulation designed to evaluate the performance of the approach. Section 5 offers our conclusion.

2 Related Work

Defences against DDoS attacks can be classified into three categories: prevention, detection and counterattack. These are three defense strategies against different stages of DDoS attacks. Prevention emphasizes either preemptive measures or the tolerance to attack of the protected server [8, 3, 7]. Detection emphasizes the capturing of intrusion signatures so that an accurate alarm can be raised once an attack has begun. Counterattack schemes, including the filtering [20, 18, 17], pushback [4] and traceback [11, 14, 16], seek to mitigate the influence of the DDoS and identify the attack source. Our work is in the area of attack detection. The following outlines relevant approaches.

Schuba gave an analysis of SYN flooding attacks and proposed a `Synkill` mechanism to detect the condition of a SYN flooding [15]. Having observed a bad or evil IP address, `Synkill` sends RST packets to the protected server to lessen the impact of the attack. In [19] Wang detected the SYN flooding attacks at leaf routers that connect end hosts to the Internet. Their method is based on the fact that in normal network traffic the SYN and FIN pair should appear symmetrically in normal network traffic. They use a non-parameter CUSUM method to accumulate these pairs. In Cheng's work [5], their approach utilized the TTL(Time-To-Live) value in the IP header to estimate the Hop-Count of each packet. Using the Hop-Count deviation, it is possible to distinguish spoofed from normal packets. Peng [12] 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 [6] and this method applied a 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.

3 The Active Detecting Approach

Our mehtod captures attacking signatures using an active probing method $DARB$. It obtains the routers delay values by sending packets containing special TTL set at the IP headers. The results of the probing are used to perform SYN flooding detection. We first analyze the difference of half-open connections originated from DDoS attacks and normal network traffic congestion. Such distinction provides the basis of the presented delay probing method - $DARB$. After that, the network delay values are actively probed by $DARB$ and half-open connections are evaluated with these delay values. At last the half-open can be categorized into *normal half-open* and *abnormal half-open*.

3.1 Analysis of Half-open Connection

A half-open connection is a connection state in which one connection end is established but the other connection end is unreachable or has deposed of its connection information. This state is normally caused by an uncompleted TCP three-way handshake. The server maintains the half-open connection state for a period of time and tries to recover the full-open connection by sending 'ACK/SYN' packets. When these half-open connections are the result

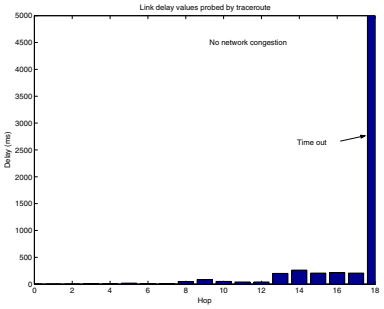
of network congestion and error, we refer to them as *normal half-open* connections.

Half-open connections can also be observed on the victim server during a SYN flooding DDoS attack. The attacker sends the victim server packets with spoofed IP addresses to victim server. The server will accept these 'SYN' requests, turn to the 'syn-received' state and send a 'ACK/SYN' packet to the destination according to the source IP. This type of half-open connection caused by SYN flooding is different from that caused by network congestion. For the server,however, it is hard to tell which is caused by congestion and which by attack. As a result, the victim server will wait until time is out and will try several times by resending 'ACK/SYN' packets. We shall refer to half-open connections of this type as *abnormal half-open* connections.

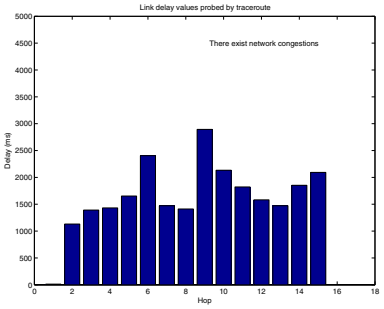
The center problem is to distinguish the *abnormal half-open* state from the *normal half-open* state. A basic distinction is that most of *normal half-open* connections arise from network congestion whereas *abnormal half-open* connections have nothing to do with congestion. i.e. the delay between the network routers is probably as same as the delay under network normal status. If the half-open connection is caused by congestion, the routing path between the server and the client should show some features of congestion, for example, increased packet delay, a rising packet loss rate, and a near-capacity queue at the congested router.

In our method, the path delay between the server and the client is probed by a method similar to `traceroute`. A very long delay is regarded as a congestion feature and in this circumstance a half-open connection is considered normal. Otherwise, the half-open will be dealt with as abnormal.

To verify whether the network delay is suitable to be used as the sign of traffic congestion, we probed about 100 web site from Hong Kong and Wuhan respectively hop-by-hop to obtain the delay of network. We found the average delay for a healthy network is about 100ms. Figure 1(a) shows the average hop-by-hop delay values of a healthy network. The delay values of first several hops are always below 20ms, which seems to be zero in the figure, and delay values of middle hops is about 100-200ms. If the probing result is time-out, we use an infinite value to substitute delay value. Time-out results are always found at the last hops of delay probing because the probing packets may be filtered out by the firewall at the server side. Figure 1(b) depicts the delay value of a congested network, which is infected by the worm virus. The clients were informed by ISP that the service might degrade because they were being attacked by some kind of worm virus. The worm virus affected the ISP at the Hop 2 and the average delay after the Hop 2 is is far above 1000ms. From such experiments, we believe the delay value is suitable to distinguish a healthy network from a congested one.



(a) A healthy network



(b) A congested network

Figure 1. Delay values probed hop-by-hop

3.2 $DARB$–the DelAy pRoBing Method

Delay is estimated using a method called $DARB$(DelAy pRoBing). The $DARB$ traces outgoing paths toward network destinations by sending packets with special time-to-live (TTL) fields in the IP layer and then recording their time of deaths. The IP TTL field limits the lifetime of packets transmitted across the Internet and is decremented by each forwarding device(routers). If the TTL field reaches zero before the destination host is reached, the router drops the offending packets and transmits an ICMP(Internet Control Message Protocol) [1] TTL exceeded in transit error message to the original host, informing the original host of the packet's timeout. If the packet has been created appropriately, the destination host should return a final packet to the original host when the packet reaches its destination. The time-stamps of both the sent-out packets and ICMP-replied packets are recorded to calculate the delay between the original host and each router. The adopted $DARB$ is similar to `traceroute` [2], which works by sending packets with progressively longer TTL value. The `traceroute` technique sends arbitrary Layer 4 packets to a destination host with an IP TTL of 1 and then monotonically incrementing the TTL field after each response. Our proposed $DARB$

method sets TTL values using a selective algorithm, which will be presented in the following paragraph.

There is an important assumption that underlies our probing method: there is no serious network congestion near the victim side at the early stage. As Moore reported[9], most of DDoS attack durations last from 3-20 minutes. Even if there exists a burst of flooding traffic at the beginning of attack, most of victims can survive for the first several minutes and serious network congestion appears gradually instead of appearing suddenly. We believe $DARB$ can send out probing packets at the early stage when there is no serious network congestion. $DARB$ can get probing results within several seconds which can guarantee the sufficient time to give alarms and take response before DDoS attack becomes serious.

$DARB$ selects special routers along a routing path to probe delays rather than, like `traceroute`, probing all the routers hop-by-hop. The selection scheme shown in Figure 2 is a binary search algorithm. Since this delay probing will not guarantee successful probing of all routers along the path, the scheme seeks to probe a router as far as possible. The **far_hop** defines the largest TTL value and the packet with this TTL value will hop to the farthest route in probing. The **near_hop** defines the nearest router. Probing process can start from the gateway routers which work for the protected server because we think the protected server could be aware of the congestion status within its Autonomous System. The **current_hop** means the current hop that the packet will traverse. Firstly, the TTL is set large enough to ensure that the packet is able to reach the destination host, which is equal to a 'ping' operation. If an ICMP echo reply message returns, it means the destination is a living host. In a SYN flooding attack, to ensure the attack efficiency, attackers prefer to use unreachable IPs as the spoofed IP. Otherwise, when a living host receives unexpected 'ACK/SYN' from server, it will discard it or send a 'RST' to server. Therefore, a living destination can be safely regarded as the legitimate user. If, on the other hand, there is no direct reply from the destination host, delays will be probed on routers along the path. The probing packet with **current_hop** as the TTL will first seek a successful reply from the correlative router. Upon receiving that, it will probe the farther router. If, however, it fails to get a reply, i.e. the request times out, it will try to probe a nearer router. This probing process will continue until it is possible to probe the available router.

The advantages of using $DARB$ is that it avoids the negative influences of firewalls or gateway filters. The simplest method is to send a ICMP echo request message to the destination as the `ping` mechanism. However, firewalls and gateways set at some end hosts may filter out an ICMP echo message, which makes the simple `ping` request unreliable. Alternatively, we use a delay probing method $DARB$.

A delay value table T_{delay} is built to store the network delay values according to internet Autonomous Systems(ASes). After every time interval T, the T_{delay} extracts network delay values from the successive TCP three-way handshakes. It can also actively probe network delay when system is idle and there is a lack of successive handshakes for certain ASes. These delay values are collected in T_{delay} and refreshed at every time interval T. The delay value collection and congestion estimations are performed according to the ASes because the congestion status does not change much within a same AS. With the support from T_{delay}, $DARB$ first searches the table for a specified IP before it probes the network. If a matched delay value is found, which is from the same AS as this IP, the probing process can be omitted to save probing time.

A static size table structure is used in T_{delay}. A dynamic size table structure may be more flexible but may become the potential victim of a DDoS attack. 10k entries

```
Probe(Input: A Half-Open Connection; Output: Delay Value)
Create Probing Packet P
Set P.TTL=128 //It is equal to a 'ping' operation
Send(P)
if Receive ICMP echo reply message M_e then
   Return 0 // The probed destination is a living host and
   // it can be safely regarded as the legitimate user
end if
//If cannot get a direct 'ping' result, do probing initialization
Set far_hop = 32
Set near_hop =the hops to victim's gateways router
Set current_hop = (far_hop + near_hop) / 2
Set delay = ∞
while far_hop > near_hop do
   Set P.TTL= current_hop
   Send(P)
   if Receive ICMP TTL exceeded message M_t then
      delay = M_t response time
      near_hop = current_hop + 1
      current_hop = (far_hop + near_hop) / 2
   else
      far_hop = current_hop -1
      current_hop = (far_hop + near_hop) / 2
   end if
end while
if delay = ∞ then
   Return -1 // Fail to probe any router along the path
else
   Return delay
end if
```

Figure 2. The Selective Probing Scheme

are recorded in the table and each entry occupies 32 bits. 24 bits are used to record the first three octets of IP address, which contains enough information to identify an AS. 8 bits of entry is for storing the delay value of probing. The total size of table is about 40k bytes, which does not mean an expensive storage cost for modern computers.

3.3 The Probability Estimation

The delay values obtained by the $DARB$ method are used to classify half-open connections as either normal or abnormal. If the returned result by the $DARB$ method has a large delay value, this half-open connection has a high probability to be normal. Whereas a small delay value indicates the network is in a benign traffic status, which does not yield a half-open connection. The tested half-open connection is abnormal and may be caused by a SYN flooding attack.

We first obtain the average network traffic delay when a network runs smoothly. In a time period t, a sample is selected to contain some half-open connections in a server. The sample is denoted as S. Let x_i be the delay returned by the $DARB$ method for the ith half-open connection in S. The average delay value $\overline{x}$ is calculated by

$$\overline{x} = \frac{\sum_{i \in S} x_i}{|S|}$$

We now provide a function $f(x)$ to evaluate probed delays. Let $\beta = \overline{x}$ and x be a random variable to denote the delay value of a half-open connection probed by the $DARB$ method. Thus, $f(x)$ indicates the probability of a half-open connection belonging to *abnormal half-open*.

$$f(x) = \begin{cases} \frac{1}{\beta} e^{-\frac{x}{\beta}} & x > 0 \\ 0 & x \leqslant 0 \end{cases}$$

The probability density function $f(x)$ is modeled by an exponential distribution because it reflects the relationship between the network traffic delay and the chance of existing of an abnormal half-open connection. When the probed delay by the $DARB$ method is short for a half-open connection, which means x is nearly 0, such connection has a high probability to be an abnormal half-open one from the analysis in Section 3.1.

3.4 Active Method to Detect DDoS Attacks at the Early Stage

Not every half-open connection maintained on a protected server is necessary to be probed using the $DARB$ method. Only suspicious half-open connections, which are maintained on the server longer than a predefined time period t that can be adjusted, will be probed. Periodic snapshot of server's half-open connections is captured at every interval t. We denote S_T as the snapshot set of half-open connections at time T and S_{T-t} as the set at time $T-t$. Two adjacent snapshot sets S_T and S_{T-t} can be compared. If a half-open appears in both S_{T-t} and S_T, we will perform a delay probing for this half-open, i.e., the elements in the intersection $S_{T-t} \cap S_T$ will be probed by the $DARB$ method. If there are too many half-open connections in intersection $S_{T-t} \cap S_T$, a sample will be selected from $S_{T-t} \cap S_T$ and probed.

A threshold T is predefined and used to compare with the probability calculated by $f(x)$. If $f(x) > T$, the half-open connection is suspicious and classified as *abnormal half-open*. Otherwise, the half-open connection is legitimate and regarded as a *normal half-open* connection.

When the number of *abnormal half-open* exceeds a predefined threshold N, a DDoS alarm will be notified. The threshold N depends on how many half-open connections a server can tolerate. If the server reserve more resource for half-open connections, the threshold N can be set larger. Otherwise, the N is set to a smaller value.

4 Experiment Result

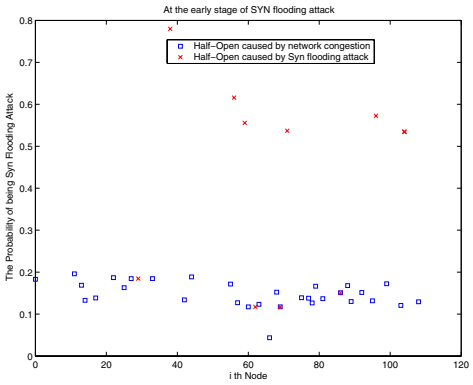
An experiment was designed to evaluate the performance of our detection method. The network topology is simulated using the method presented in [10]. In this experiment, routers are simulated as 110 nodes and each node has a maximum of five edges for connecting with its neighbor routers. Each edge is assigned a weight to stand for the latency between two nodes. One nodes is assumed to be the protected server. Then there exists the shortest path from this server node to each of the other nodes. When the network is uncongested, the weight of each edge is set at no more than 10, which means the delay between two adjacent nodes is less than 10ms. The average latency for all shortest paths is calculated statically and is used to make further comparisons.

The experiment next arranges for nodes to be randomly congested. Once the path that includes these nodes is congested, the delay is increased by a predefined value. The more congested nodes that a path includes, the longer delay it will suffer. If the total delay exceeds a predefined TIMEOUT value, the packet will cause half-open connection on the server node.

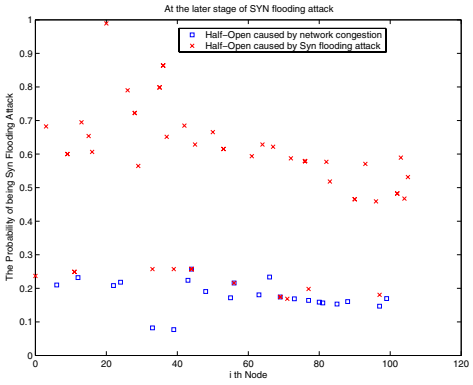
At the same time that the nodes are congested, a SYN flooding attack is simulated. The attack randomly selects unreachable IP addresses as the spoofed addresses. Upon receiving spoofed 'SYN' request packets from the attacker, the server replies with 'ACK/SYN'. These 'ACK/SYN' packets are transmitted by the nodes according to the shortest path tree and arrive at the target node. These 'ACK/SYN' request do not, however, receive a corresponding reply from the client. This causes a half-open connec-

tion on the server host.

The first experiment determines whether the detection system can distinguish between *normal half-open* and *abnormal half-open*, using delay probing to estimate whether a half-open connection is the result of an SYN flooding attack. Figure 3 shows that *abnormal half-opens* have much higher score than *normal half-opens*. This makes it easy to classify the half-open as either normal or abnormal. In this experiment, if the probability of being a SYN flooding attack packet exceed 0.5, this half-open is regarded as abnormal. Yet response time delays may also result if the 'ACK/SYN' packet sent to a spoofed address happens to meet a congested router. In such a case, the probability score of an *abnormal half-open* will be as low as that of a *normal half-open* and it will bring potential false negatives. But this case does not happen frequently and will not influent the accuracy of result.



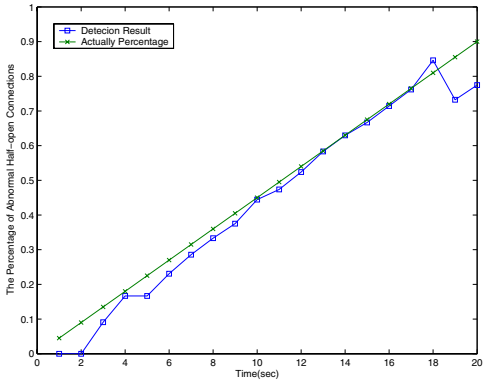
(a) Experiment result when attack is at its early stages



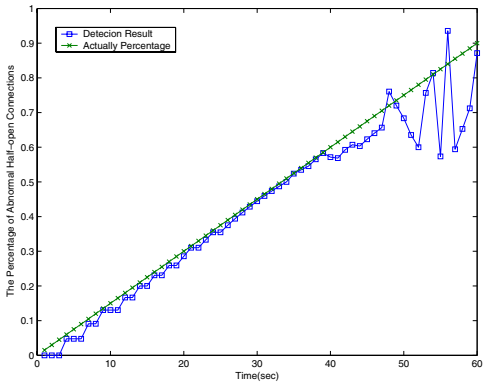
(b) Experiment result when attack is at its later stages

Figure 3. The probability that a half-open indicates an SYN flooding attack. Such a half-open produces a score distinguishable from that of a half-open caused by congestion

The second experiment tests the detection rate by launching SYN flooding attacks at various rates. Under various attack rates, 90% of the server's connections will be *abnormal half-open* in, respectively, 20 and 60 seconds. After the half-open connections being probed and the probability score being evaluated, the percentage of *abnormal half-open* connections contained in all half-open connections is calculated. The detection result is shown in Figure 4.



(a) Within 20 sec



(b) Within 60 sec

Figure 4. Percentage of abnormal half-opens on the server. Abnormal half-opens reaches 90% at different rates shown in (a)(b)

Figure 4 shows that the detection results are accurate in the early stages of the SYN flooding attack. At later stages, there are vibrations around the actual percentage. These vibrations are caused by the use of a sample in probing. As we mentioned in section 3.4, if there is too many half-open connections, only a sample will be selected to perform detection, which affects the accuracy of detection. The greater accuracy in the early stages is explained by the existence of fewer half-open connections, which can all be probed.

5 Conclusion and Future Work

This paper presents a novel response to SYN flooding attacks. The approach works by classifying half-open connections as either abnormal half-open or normal half-open. Normal half-open connections, those caused by network congestion, will exhibit features not present in abnormal half-open connections. Network congestion feature are identified using a delay probing method $DARB$ that collects data on the delay between the server and the client. If the delay is much longer than average value, it is regarded as a congestion feature and the corresponding half-open is classified as normal. Otherwise, the half-open is regarded as abnormal.

There are several factors impacting the accuracy of detection and bringing potential false alerts. When the spoofed IP address of an attacking packet happens to meet an actually congested router, our method cannot find this malicious packet. If there is congestion during attacks, the accuracy will be influenced because it is difficult for our method to distinguish the attack from the network congestion. Though these cases do not happen frequently, these will bring some false negatives. Besides congestion and SYN flooding attack, some other reasons to cause the failure of the three-way handshake, such as errors in routers, will also bring potential false positive. More precise mechanism will be studied to give more accurate distinguish between network congestion and SYN flooding attack.

The parameters t, T, N in Section 3.4 is important for the effectiveness of the proposed approach. In experiments, these parameters are decided by experience and maybe are not the optimal. The optimization of these parameters will be part of our future work.

References

[1] S. M. Bellovin. ICMP traceback messages. Technical report, March 2000.

[2] S. Branigan, H. Burch, B. Cheswick, and F. Wojcik. What can you do with Traceroute? *Internet Computing, IEEE*, 5:96, Sept.-Oct. 2001.

[3] P. Ferguson and D. Senie. Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing, May 2000.

[4] J. Ioannidis and S. M. Bellovin. Implementing pushback: Router-based defense against DDoS attacks. In *Proceedings of Network and Distributed System Security Symposium, Catamaran Resort Hotel San Diego, California*. The Internet Society, February. 2002.

[5] C. Jin, H. N. Wang, and K. G. Shin. Hop-count filtering: An effective defense against spoofed DDoS traffic. In *Proceedings of the 10th ACM conference on Computer and communication security(CCS)*, pages 30–41. ACM Press, October 2003.

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

[7] A. Keromytis, V. Misra, and D. Rubenstein. SOS: An architecture for mitigating DDoS attacks. *IEEE Journal on Selected Areas in Communications*, 22:176–188, January 2004.

[8] J. Lemon. Resisting SYN flood DoS attacks with a SYN cache. In *In Proceedings of the BSDCon 2002 Conference.*, 11-14 Feb. 2002.

[9] D. Moore, G. Voelker, and S. Savage. Inferring internet denial of service activity. In *Proceedings of USENIX Security Symposium*, Aug. 2001.

[10] P. Narvaez, K.-Y. Siu, and H.-Y. Tzeng. New dynamic SPT algorithm based on a ball-and-string model. *Networking, IEEE/ACM Transactions*, 9:706–718, Dec. 2001.

[11] K. Park and H. Lee. On the effectiveness of probabilistic packet marking for IP traceback under denial of service attack. In *INFOCOM*, pages 338–347, 2001.

[12] 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.

[13] Rocky.K.Chang. Defending against flooding-based distributed denial-of-service attacks: a tutorial. *Communications Magazine, IEEE*, 40(10):42–51, Oct. 2002.

[14] S. Savage, D. Wetherall, A. Karlin, and T. Anderson. Practical network support for IP traceback. In *Proceedings of the ACM SIGCOMM Conference*, pages 295–306. ACM Press, 2000.

[15] C. L. Schuba, I. V. Krsul, M. G. Kuhn, E. H. Spafford, A. Sundaram, and D. Zamboni. Analysis of a denial of service attack on TCP. In *Proceedings of the 1997 IEEE Symposium on Security and Privacy*, pages 208–223. IEEE Computer Society, IEEE Computer Society Press, May 1997.

[16] A. C. Snoeren. Hash-based IP traceback. In *Proceedings of the ACM SIGCOMM Conference*, pages 3–14. ACM Press, August 2001.

[17] M. Sung and J. Xu. IP traceback-based intelligent packet filtering: A novel technique for defending against internet DDoS attacks. *IEEE Transactions on Parallel and Distributed Systems*, 14(9):861–872, September 2003.

[18] U. Tupakula and V. Varadharajan. Counteracting DDoS attacks in multiple ISP domains using routing arbiter architecture. In *Networks ICON2003. The 11th IEEE International Conference*, pages 455–460, Sept. 28-Oct. 1 2003.

[19] H. Wang, D. Zhang, and K. G. Shin. Detecting SYN flooding attacks. In *Proceedings of IEEE INFOCOM*, volume 3, pages 1530–1539, June 23-27 2002.

[20] A. Yaar, A. Perrig, and D. Song. SIFF: A stateless internet flow filter to mitigate DDoS flooding attacks. In *Proceedings. 2004 IEEE Symposium, Security and Privacy*, pages 130–143, May 9-12 2004.