Deployable Overlay Network for Defense against Distributed SYN Flood Attacks

Yuichi Ohsita
Graduate School of
Information Science and Technology,
Osaka University
E-mail: y-ohsita@ist.osaka-u.ac.jp

Shingo Ata
Graduate School of Engineering,
Osaka City University
E-mail: ata@info.eng.osaka-cu.ac.jp

Masayuki Murata
Graduate School of
Information Science and Technology,
Osaka University
E-mail: murata@ist.osaka-u.ac.jp

***Abstract*— Distributed denial-of-service attacks on public servers have recently become more serious. To assure that network services will not be interrupted, we need faster and more accurate defense mechanisms against malicious traffic, especially SYN floods. But single point defense (ex. firewalls) lack a scalability to catch up the increase of the attack traffic.**

In this paper, we introduce a distributed defense mechanism using overlay netwoeks. This mechanism detects attacks near the victim servers and alert messages are sent via the overlay networks. Then defense nodes identify legitimate traffic and block malicious ones. The legitimate traffic is protected via the overlay networks. We simulate and verify our proposed method can effectively block malicious traffic and protect legitimate traffic. We also describe the deployment scenario of our defense mechanism.

***Index Terms*— Distributed Denial of Service (DDoS), SYN Flood, Overlay Network, TCP Proxy**

I. Introduction

The recent rapid growth and increasing the wide use of the Internet are making Internet security issues increasingly important. Distributed Denial-of-service (DDoS) attacks are one of the most serious problems. The DDoS attack causes serious damage at the victim server by increasing the number of hijacked nodes even if the rate of attack traffic generated by each node is quite small.

Recently, there are many kinds of DDoS attacks such as Smurf attacks [1], UDP floods [2], and SYN flood attacks [3]. In SYN Flood attacks, attackers send so many connection requests to one (i.e., victim) server that end users cannot connect to it. Because attackers can easily put servers into a denial-of-service state this way, it is reported that about 90% of all DoS attacks are SYN Flood attacks [4].

Therefore, several methods against SYN Flood attacks have been proposed so far. SYN cache [5] and SYN cookie [6] are mechanisms deployed in server (victim) nodes.

However, these single-point defense mechanisms have a fundamental problem with respect to scalability. In DDoS attacks, attack nodes are widely distributed all over the world. Attack traffic from attack nodes is aggregated into a very high rate attack at the server. At this point, a DDoS attack is highly scalable because the amount of attack traffic can increase in proportion to the number of attacker nodes. On the other hand, single-point defense mechanisms lack scalability commensurate with the attack traffic increase. That is, high-rate attacks from widely distributed nodes can overwhelm the firewalls or the servers regardless of the implemented server-side defense mechanism (e.g. SYN Cache or SYN Cookie) is implemented. For this reason, a distributed defense mechanism is needed to defend servers from distributed attacks.

Implementing a distributed defense mechanism, such as a cooperation of distributed nodes, is more difficult than a single-point approach. D-WARD [7] has been proposed as a way to stop DDoS attacks near their source. In this method, an edge node detects attacks and limits the rate of traffic addressed to the victim server. However, detecting distributed attack traffic near attacker nodes is quite difficult when attack nodes are highly distributed and each attacker node generates a small amount of attack traffic. We believe that it is more practical to detect attacks near a victim node and alert other nodes deployed near attacker nodes. In pushback [8], a router detecting an attack requests upstream routers to limit the amount of traffic bounded to the victim node. This method can set a rate limit near attackers by recursively requesting the limitation from upstream routers. This is an effective countermeasure to attacks exhausting network links, but a rate limit is not an effective way to prevent attacks, such as SYN flood attacks, which exhausting the resources of servers. DefCOM [9] has been proposed as a framework that allows DDoS defense nodes to communicate with each other; however, the framework was reported without any description of a specific method to detect or block attack traffic. In PacketScore [10], edge nodes compute a per-packet score which estimates the legitimacy of a packet. Core nodes perform score-based selective packet discarding. This method can effectively mitigate attacks when the characteristics of attack traffic differ from those of legitimate traffic. However, legitimate traffic may be mistakenly identified as attacks and blocked by this method. This can seriously impair the communication between the victim and legitimate clients.

Given the above state of affairs, we clearly need a defense mechanism that (1) has enough scalability accommodate any the increase in the distributed attack traffic, (2) can detect attacks accurately on the victim sides, and (3) can correctly protect legitimate traffic. For the first issue, blocking attacks at distributed points has higher scalability than defense at a single node. Next, we can reliably identify legitimate packets by receiving the ACK packets corresponding to SYN/ACK packets. We use a proxy approach which responds to the acknowledgements of SYN packets on behalf of the victim

node, and passes SYN packets only when the proxy receives the ACKs of SYN/ACK packets. For the last problem, the TCP traffic whose requests are identified as legitimate by the proxy is delivered via the overlay network, in which proxies are logically connected. By giving a priority to the traffic which is transferred on the overlay network, the legitimate traffic is thus protected against the attack traffic.

Moreover the possibility of deployment is also important issue, because the defense against DDoS attack is very urgent and serious problem which should be solved as soon as possible. The solution is also required to deploy easily and not affect any negative impacts to current IP frameworks.

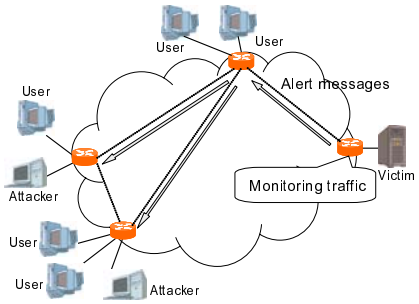
In this paper, we propose a new distributed defense system using overlay networks against distributed SYN Flood attacks. In this system attacks are detected by victim-side nodes which can detect attacks easily. After an attack is detected, alert messages are forwarded to all nodes via the overlay networks. The edge defense nodes which received the alert begin to identify and block attack packets. At the same time, the defense nodes protect legitimate packets by forwarding them via the overlay networks. Note that the detection mechanism may also cause a false detection (i.e., falsely detect the legitimate traffic as the attack). However, the actual identification is not performed at the victim side, but at the attacker side by the proxy. Therefore, the traffic falsely detected as attack is identified as the legitimate traffic at the proxy, and forwarded to the server via the overlay network.

In Section II we explain the overview of our detection mechanism and discribe the detailed operation. In Section III, we explain the deployment senario of our mechanism. In Section IV, we show some simulation results that our method can effectively block attack traffic and protect legitimate traffic. In Section V we conclude by briefly summarizing the paper and mentioning some of the future works we intend to do.

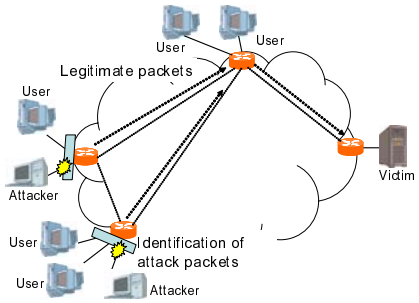
II. Distributed defense mechanism using overlay networks

Figure 1 shows an overview of our proposed architecture. We place *defense nodes* at the edge of a network (we call this network a *protected network*). Each defense node logically connects to one or more other defense nodes, and constructs an overlay network among the defense nodes. To identify legitimate SYN packets, defense nodes act as a SYN proxy which returns a SYN/ACK packet instead of the victim node doing so. The SYN packet is relayed only when the defense node receives the ACK packet of the SYN/ACK packet from the client (Figure 2). Once a flow (i.e., packets having the same (src IP, dest IP, src port, dest port, protocol) fields) is identified as legitimate traffic, packets of the flow are transferred via the overlay network and distinguished from attack traffic.

In the ideal situation, the defense node should handle all arriving packets and pick up legitimate packets from among them. However, this process causes processing overhead, and the defense node will become a performance bottleneck. To minimize the defense node overhead, it is desirable to identify only those packets going to the victim node. For this purpose, we use a mechanism for detecting a SYN flood attack. The mechanism has two phases, *attack detection mode* and *defense mode*. In the *attack detection mode*, the defense node monitors



(a) Attack detection mode



(b) Defense mode

Fig. 1. Distributed difense using overlay networks

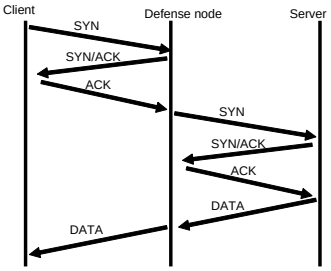


Fig. 2. Delegation of SYN/ACK packets

packets outbound from the protected network and checks whether the arriving traffic is attack or not. If the defense node detects attack traffic, the defense node alerts other defense nodes of the address of the victim node. Upon receiving the alert, each defense node moves into the *defense mode* for the victim's address. In the defense mode, the defense node delegates SYN/ACK packets to identify legitimate traffic. The defense mode is continued until the attack ends.

In defense mode, the defense node performs the following operations:

1) Detecting attacks
2) Alerting all defense nodes
3) Delegation of SYN/ACK packets
4) Relay of legitimate packets
5) Ending the defense mode

In the following sections, we describe these operations in detail.

A. Detecting attacks

It is difficult to detect highly distributed attacks at edge routers or core networks because the number of attack packets is very small there. For this reason, we detect attacks at the server side where we can detect attacks comparatively easily.

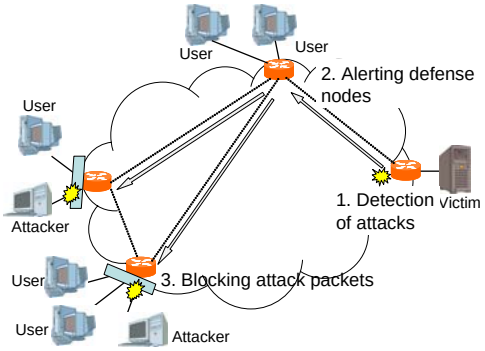


Fig. 3. Steps of alerting

TABLE I
DATA STRUCTURE USED TO IDENTIFY FLOWS

Source address 32 bit	
Destination address 32 bit	
Initial sequence number (receiver) 32 bit	
Initial sequence number (sender) 32 bit	
Source port 16 bit	Destination Port 16 bit
Timer	reserved for future use

There are several proposals to detect attacks. In this paper, we use the method [11] we proposed before. This method detects attacks by comparing the SYN arrival rates with normal distributions. This method can detect attacks fast regardless of time variation of traffic.

B. Alerting all defense nodes

Figure 3 shows the steps to alert all defense nodes after an attack is detected. Once the attack is detected, the IP address of the victim node is sent to all defense nodes as alert messages via the overlay network. The defense nodes that receive the alert then move into the *defense mode*, and begin to return SYN/ACK packets for SYN packets whose destination addresses are that of the victim server. These alert messages are generated at the event of attack detections, and forwarded once for each defense node. No other alerts are forwarded during the defense mode, except the events of ending the defense mode. These alerts therefore do not strictly affect on the network bandwidth availability.

Note that the propagation of the alert message depends on the topology of the overlay network. However, the problem of how to construct an effective overlay network is a separate research topic beyond the scope of this paper.

C. Delegating SYN/ACK packets

In the defense mode, the defense node delegates SYN/ACK packets. When the defense node receives a SYN packet, it checks whether the destination address of the received packet is the IP address of the victim node. If the packet is intended for delivery to the victim node, the defense node returns a SYN/ACK packet to the address specified in the source address of the received packet. Then, after the defense node receives the acknowledgement for the SYN/ACK packet, it tries to establish a connection to the victim server. To identify whether the received ACK packets are acknowledgements of SYN/ACK packets, the defense node uses the data shown in Table I for each flow.

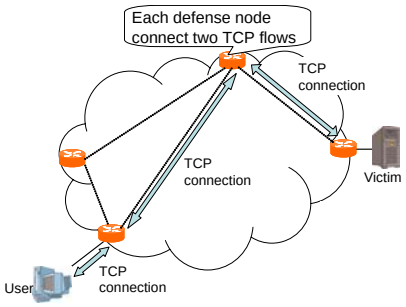


Fig. 4. Relaying the legitimate packets

However, the defense node must hold a number of structures equal to the number of delegating SYN/ACK packets in the attack mode. The defense nodes should save their resources such as memory or CPU load while they hold legitimate connection requests even if they receive a number of SYN packets.

To save resources we use the same approach as the SYN cache mechanism. The SYN cache uses a hash table to search the data structures. The hash value is computed from the source and destination IP addresses and the source and destination port numbers. Entries having the same hash value are kept on a forward linked list. The length of the list is limited. When the list is full (i.e., the length of the link is equal to the maximum value) and a new connection request is received, the oldest (i.e., the head) entry in the list is dropped and a new request is appended at the tail of the list.

D. Relaying legitimate packets

We identify flows which complete the 3-way handshake as legitimate traffic. Once a flow is identified as legitimate traffic, the defense node relays packets of the flow to the server via the overlay network. In this paper, we use the TCP Proxy [12] to relay legitimate traffic.

TCP Proxy is a method which controls transmission quality at the transport layer. TCP Proxies construct overlay networks and establish the connections to the next-hop TCP Proxies, which are determined according to the destination addresses. TCP Proxies relay packets by using hop-by-hop connections established via the overlay networks.

Figure 4 shows an overview of how legitimate packets are relayed using TCP Proxies. The defense nodes establish the hop-by-hop connections. Each node relays packets by connecting the flow from the previous hop and the flow to the next hop.

Authors in [12] show that the hop-by-hop TCP connections can gain more throughput rather than a single TCP connection between the same end nodes. However, hop-by-hop connections require additional resources such as sender/receiver buffers on intermediate defense nodes. Since hop-by-hop connections have both pros and cons, we can use both types of TCP connections based on the administrative policy.

We also need to consider about the security of the overlay network. A malicious user may send spoof packets in order to attack to the defense node, or to inject attack traffic to the overlay network. To avoid such problem, it is necessary to introduce some authentication mechanisms to verify the peer defense node.

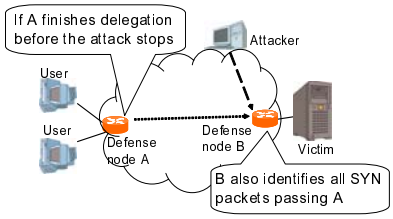


Fig. 5. Problem in finishing the defense mode

E. Ending the defense mode

Since the resources of the defense node are limited, the defense mode should be terminated as soon as the attack ends. To enable this, it is necessary to detect the end of an attack at the defense node.

To detect the end of an attack, the defense node counts the number of connection requests (i.e., SYN packets) which time out or are dropped. When the number becomes 0, the attack is considered to have ended. Unlike attack detection, detection of the end of an attack does not have to be particularly fast since a long defense mode does not disturb legitimate connections.

A problem arises, though, when all of the defense nodes independently detect the end of an attack. This situation is shown in Figure 5. In this figure, all of the attack traffic is passed via defense node D_B. If the detection is performed independently, defense node D_A first detects the end of the attack and stops delegating SYN/ACK packets. After finishing the delegation at D_A, all of the SYN packets passing D_A are subject to identification at D_B. The load on D_B thus increases because the total number of SYN packets increases, and D_B may drop some SYN packets because of the SYN cache limit on D_B. This degradation of performance will not occur if D_A continues to delegate SYN/ACK packets until D_B detects the end of the attack (i.e., the attack has completely ended in this case).

Therefore, the defense node should stop delegating SYN/ACK packets only when there are no attack packets at either the defense node or intermediate defense nodes on the way to the victim node.

Figure 6 shows the steps to stop delegating SYN/ACK packets. First, the defense node nearest to the victim node detects the end of the attack. This defense node sends a message indicating the end of the attack to all adjacent nodes (i.e., those logically connected from the defense node). A defense node receiving the message still delegates SYN/ACK packets until it detects the end of the attack. Upon detecting the end of the attack, each defense node ends the defense mode and forwards the message to the downstream adjacent defense nodes. The defense is completely ended after all defense nodes have received the message and ended the defense mode.

III. Deployment Scenario

In this section, we explain how our mechanism can be deployed in the Internet. We deploy our method in a phased manner because it is impossible to deploy in the whole Internet at once. In our mechanism, the unit of deployment is the Autonomous System (AS). In this paper, we refer to an AS in which our mechanism is deployed as a *protected AS*. All edge routers are defense nodes in a *protected AS*. Otherwise, an AS is referred to as *unprotected*. Figures 7 through 9

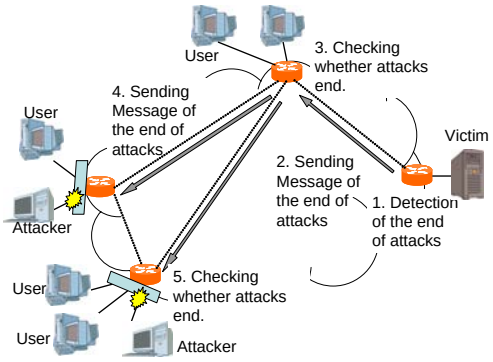


Fig. 6. Steps to finish the defense mode

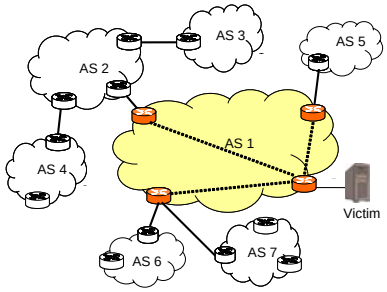


Fig. 7. First stage of deployment

show the strategic scenario for the deployment of our defense mechanism. There are three stages as follows.

1st stage (Fig.7): Only one AS is *protected*. Others are *unprotected*.
2nd stage (Fig.8): Several ASes are *protected*.
final stage (Fig.9): All ASes are *protected*.

At the first stage, we consider our method to be deployed in only one AS, as shown in Figure 7. In this figure, AS 1 (the yellow cloud) is *protected*. Outside AS 1 all attack traffic to the victim node is first delivered to the victim node. The defense node nearest to the victim node then detects the attack traffic, and alerts the other defense nodes of the attack. Attack traffic is therefore blocked at the defense nodes placed at the edge of AS 1. In the case shown in Figure 7, our method enables AS 1 to block attack packets at three points. This means that our defense mechanism can defend against attack traffic up to three times as effectively as a single-point defense mechanism.

At the second stage of deployment (Figure 8), our method is deployed in several ASes which cooperate with each other. In the case shown in Figure 8, AS 1, AS 6, and AS 7 are *protected*. After an attack alert, the delegation of SYN/ACK packets is performed at the edge of the *protected* ASes. As a result, attack traffic generated in AS 6 and AS 7 is blocked at the egress edges of these ASes. Attacks from AS 2, AS 3, and AS 4 are blocked at the edge of AS 1 (the defense node for the link to AS 2). Attacks from AS 5 are also blocked at the edge of AS 1. Increasing the number of *protected* ASes means that attack traffic is blocked at more defense nodes. Moreover, the amount of legitimate traffic that our mechanism can protect may increase.

At the final stage of deployment (Figure 9), all ASes are *protected*. In the case shown in Figure 9, no attack packets reach AS 1 because all attack packets are blocked inside each AS. The attack traffic is no longer delivered to the core

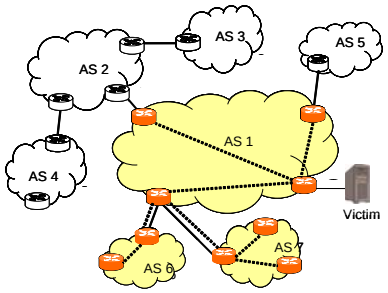


Fig. 8. Seccond stage of deployment

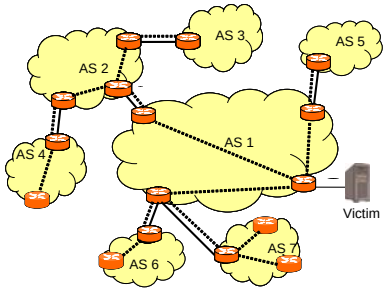


Fig. 9. Final stage of deployment

network when detected.

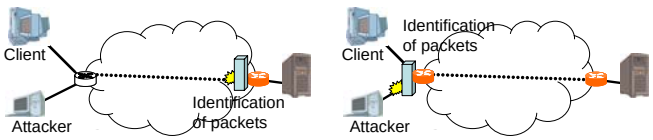
IV. Evaluation

In this section, we evaluate the performance of proposed defense mechanism through simulation. We perform two types of evaluation. First, we show the effectiveness of attacker-side defense by comparing the dropping rate of legitimate traffic where a single defense node is placed at the attacker-side with the one where the node is placed at the victim-side. Second, we place defense nodes at both victim and attackers sides to evaluate the effectiveness of distributed defense mechanism.

A. Effectiveness of attacker-side defense

To demonstrate the effect of our method, we compared the probability of dropping legitimate SYN packets when deploying an attacker-side defense mechanism (Figure 10(b)) with that when deploying a victim-side defense mechanism (Figure 10(a)). We assumed that the average round-trip time (RTT) between the clients and the victim server was 200 ms, and the average RTT between the clients and the attacker-side defense node was 20 ms. We also assumed that all attack packets were received by the same defense node. From the result described in [11], the SYN arrival rates of normal traffic would follow a normal distribution with a mean of 100 SYNs/sec. We set the SYN Cache parameters to the values used in FreeBSD.

Figure 11 shows the probabilities of legitimate SYN packets being dropped based on the rate of attack traffic. We have plotted three results: (1) without a defense mechanism, (2) with victim-side defense, and (3) with attacker-side defense. As shown, the attacker-side defense protected legitimate packets much better than the victim-side defense. This was because the RTTs between clients and the attacker-side defense node were much shorter than the RTTs between clients and the victim-side defense node. The average holding time for each connection request on the SYN cache was also short, which increased the availability of the SYN cache.



(a) Victim-side defense (b) Attacker-side defense

Fig. 10. Environment supposed in our simulations

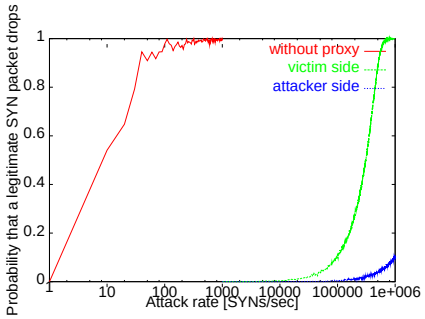


Fig. 11. Probability of dropping legitimate SYN packets vs. attack rate

[4] reports observing attacks whose rates exceeded 600,000 SYNs/sec. In the event of such high-rate attacks, victim-side defense cannot protect legitimate packets and the probability of dropping legitimate SYN packets rises to almost 1. On the other hand, if we deploy attacker-side defense, the probability of dropping legitimate SYN packets would be less than 0.1.

In summary, the attacker-side defense can catch up more legitimate traffic than the victim-side defense. This effectiveness is more obvious when the difference between attackers and the victim becomes larger.

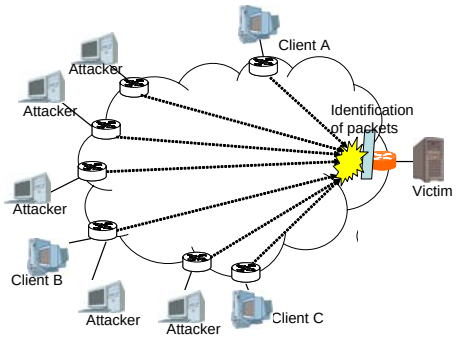
B. Effectiveness of distributed defense

We next consider the effectiveness of the distributed defense mechanism. We consider the three scenarios shown in Figure 12. Case 1 is a single-point defense mechanism. In Case 2, there are several defense nodes on the edge of the network, but not all edge nodes are defense nodes. In Case 3, all edge nodes except one (the ingress node for Client A) are defense nodes.

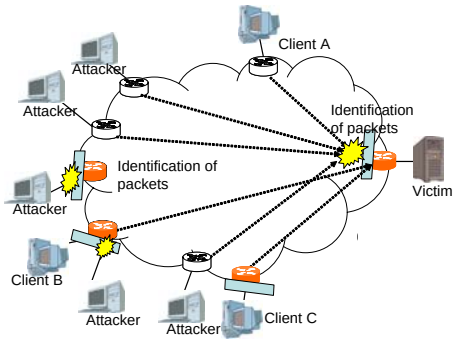
We injected attack traffic after a certain period of time from the beginning of the simulation. Each attacker sent 200,000 attack packets per second, and the attack began 100 sec from the simulation start and ended at 200 sec from the simulation start. Each client sent 30 SYNs/sec.

We plotted the time dependent variation of the probability of dropping legitimate SYN packets at three points for each case. Client A was connected to a non-defense node. Client B had a attacker node on the same segment. Client C was connected to a defense node in Cases 2 and 3, and there was no attacker node on the same network.

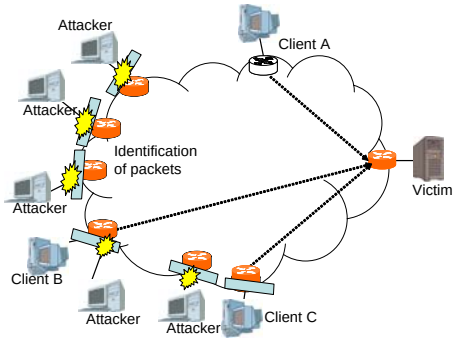
In Case 1, none of the clients could send legitimate SYN packets to the victim node. On the other hand, in Case 2, the probability of packet loss for Client B and Client C became very low soon after the attack started while the probability for Client A remained high. This is because our method quickly detects attacks and distinguish legitimate packets from attack packets. In Case 3, the probabilities were very low for all clients. This result shows that our method can effectively



(a) Case1: Single-point defense



(b) Case2: Some of ASes including attackers deploy our method



(c) Case3: All ASes including attackers deploy our method

Fig. 12. Environment supposed in our simulations

protect legitimate packets and block attack packets.

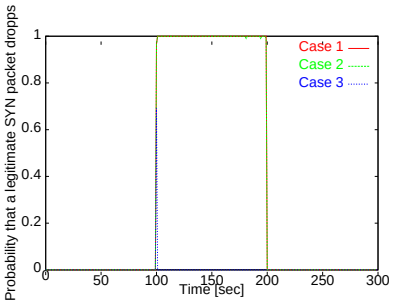
V. Conclusion and Future Work

We have proposed a distributed defense mechanism against distributed SYN Flood attacks. Our mechanism is based on the collaboration of distributed defense nodes by constructing overlay network. Overlay network is effective to alert attacks and protect legitimate traffic. We have also shown the effect of attacker-side defense and the effectivity of our method. One of our future works is to identify attack packets at the points where the routes of packets may vary.

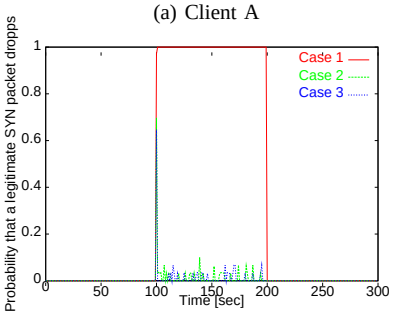
References

[1] "CERT advisory CA-1998-01 smurf IP Denial-of-Service attacks." available at `http://www.cert.org/advisories/CA-1998-01.html`, Jan. 1998.

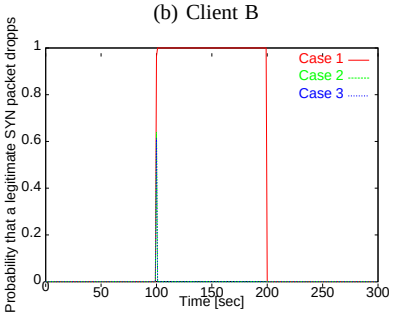
[2] "CERT advisory CA-1996-01 UDP port Denial-of-Service attack." available at `http://www.cert.org/advisories/CA-1996-01.html`.



(a) Client A



(b) Client B



(c) Client C

Fig. 13. Probability of dropping legitimate SYN packets

[3] "CERT advisory CA-1996-21 TCP SYN flooding and IP spoofing attacks." available at `http://www.cert.org/advisories/CA-1996-21.html`, Sept. 1996.

[4] D. Moore, G. M. Voelker, and S. Savage, "Inferring internet Denial-of-Service activity," in *Proceedings of the 2001 USENIX Security Symposium*, pp. 9–22, Aug. 2001.

[5] J. Lemon, "Resisting SYN flooding DoS attacks with a SYN cache," in *Proceedings of USENIX BSDCon'2002*, pp. 89–98, Feb. 2002.

[6] A. Zuquete, "Improving the functionality of SYN cookies," in *Proceedings of 6th IFIP Communications and Multimedia Security Conference*, pp. 57–77, Sept. 2002.

[7] J. Mirkovic, *D-WARD: DDoS network attack recognition and defence*. PhD thesis, Computer Science Department, University of California, Los Angels, June 2003.

[8] S. Floyd, S. M. Bellovin, J. Ioannidis, K. Kompella, R. Manajan, and V. Paxson, "Pushback messages for controlling aggregates in the network." draft-floyd-pushback-messages-00.txt, internet-draft, work in progress, July 2001.

[9] J. Mirkovic, M. Robinson, P. Reiher, and G. Kuenning, "Alliance formation for DDoS defense," in *Proceedings of the New Security Paradigms Workshop, ACM SIGSAC*, pp. 11–18, Aug. 2003.

[10] Y. Kim, W. C. Lau, M. C. Chuah, and H. Chao, "PacketScore: Statistics-based overload control against distributed denial-of-service attacks," in *Proceedings of IEEE INFOCOM 2004*, Mar. 2004.

[11] Y. Ohsita, S. Ata, and M. Murata, "Detecting distributed Denial-of-Service attacks by analyzing TCP SYN packets statistically," in *Proceedings of IEEE Globecom 2004*, vol. 4, pp. 2043–2049, Nov. 2004.

[12] I. Maki, G. hasegawa, M. Murata, and T. Murase, "Throughput analysis of TCP proxy mechanism," in *Proceedings of ATNAC 2004*, Dec. 2004.