Contents lists available at SciVerse ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Review

Analyzing well-known countermeasures against distributed denial of service attacks

Hakem Beitollahi *, Geert Deconinck

Katholieke Universiteit Leuven, Electrical Engineering Department, Kasteelpark Arenberg 10, Leuven, Belgium

ARTICLE INFO

Article history:
Received 1 October 2011
Received in revised form 5 April 2012
Accepted 6 April 2012
Available online 18 April 2012

Keywords:
Distributed denial of service attacks
DDoS
Flooding attacks
Countermeasures
Network security

ABSTRACT

This paper reviews and analyzes well-known countermeasures against distributed denial of service (DDoS) attacks. This paper provides an in-depth analysis of each countermeasure and enumerates strengths and challenges of each technique. If it is possible, the paper designs a countermeasure against each defense mechanism from the attacker's point of view. We believe that this survey is the most complete survey that analyzes the most cited DDoS defense techniques in detail. We expect that this survey will assist the potential victims to choose suitable countermeasures against DDoS attacks based on the analysis presented here and as well as the capabilities that they have to implement the techniques. The analysis done in this paper provides a great opportunity for both academic and industrial researchers to improve the state of the art countermeasures against DDoS attacks.

1. Introduction

Denial of service (DoS) attacks are characterized by an explicit attempt to prevent the legitimate use of a service. These attacks are one of the major threats against the availability in the Internet. Perpetrators of DoS attacks typically target various sites and services and their impact varies from minor inconvenience to users of a web site to serious financial losses for companies that rely on their online availability to do business. High-profile servers such as web servers of banks, credit card payment gateways, root nameservers, famous search engines (e.g., Yahoo, Google), government's web servers, broadcasting news servers, commercial servers (e.g., eBay, Amazon), social sites (e.g., Twitter, Facebook), critical infrastructure servers (e.g., power plants), etc., are the most likely targets for perpetrator of DoS attacks [1].

DoS attacks can be roughly divided into two categories: application-bug level and infrastructure level attacks [2,3]. In application-bug level attacks, the attacker for example renders a computing resource unavailable by modifying the system configuration, or overloads an application by sending it abusive workload which leads to the crashing of the application, or sends one or more careful crafted packets that exploit the application vulnerability in the target machine. The classical example of this type of attacks is the "ping-of-death" attack [4] in which the attacker sends a ping packet larger than the maximum IPv4 packet size, which is 65,535 bytes. Historically, many operating systems could not handle such a ping packet and consequently operating systems crash, freeze, or reboot due to buffer overflow. Application-bug level attacks are generally treated through patching known vulnerabilities, security policies, etc. Infrastructure-level attacks, often take place in a distributed manner (distributed denial of service (DDoS) attacks) where an attacker employs a large number of hijacked zombie machines to attack the victim. In this attack, any zombie machine sends a flood of packets to overwhelm the bottleneck resources of the victim (e.g., network bandwidth, TCP buffers and CPU cycles). Infrastructure-level DDoS attacks are also well-known as DDoS flooding attacks. In infrastructure-level attacks, the attacker only needs to know the victim's IP address. This paper focuses only on infrastructure-level attacks.

DDoS flooding attacks can be launched in two forms of attacks: direct attacks and reflector attacks. In direct attacks, the attacker directly sends a flood of bogus packets toward the victim through the zombie machines. Direct DDoS attacks are classified into two categories: network-layer DDoS attacks and application-layer DDoS attacks. Network-layer DDoS attacks encompass: TCP flood, UDP flood, ICMP flood and SYN flood. Application-layer DDoS attacks encompass: HTTP flood, HTTPS flood, FTP flood, etc. In reflector attacks, the attacker sends request messages to reflector machines through zombie machines, spoofing the source IP address of the victim server. As a result, the reflector machines send their replies to the given address causing packet flooding at that site which is the victim server. The well-known reflector attacks are ICMP ECHO reply flood, SYN ACK (RST) flood, DNS flood, Smurf attack [5] and Fraggle attack [6]. Fig. 1 shows direct DDoS and

* Corresponding author. Tel.: +32 484143830.
E-mail addresses: Hakem.Beitollahi@esat.kuleuven.be (H. Beitollahi), Geert.Deconinck@esat.kuleuven.be (G. Deconinck).

0140-3664/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.comcom.2012.04.008

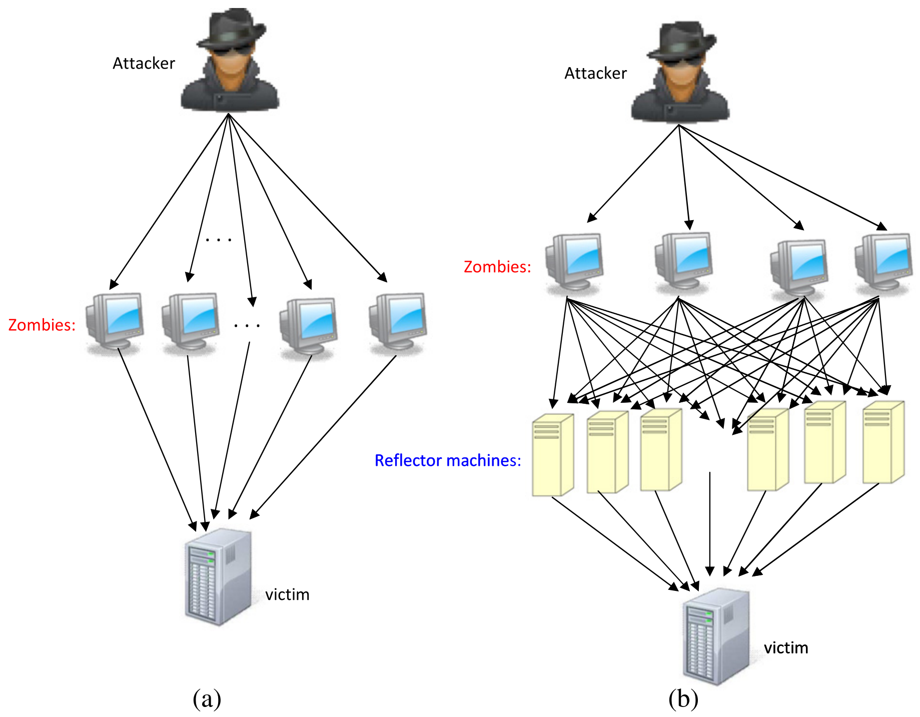

Fig. 1. The architecture of flooding DDoS attacks: (a) direct, (b) reflector.

reflector DDoS attacks. We have provided a more complete literature about DDoS attacks including classification of DDoS attacks, attack toolkits, several statistical reports about DDoS attacks, evolution of DDoS attacks, etc., in [7].

Several countermeasures have been proposed since year 2000 when almost DDoS attacks started at that year [1]. Countermeasures against DDoS attacks roughly classified in three categories: survival techniques, proactive techniques and reactive techniques [8]. This paper presents a survey on all categories and evaluates each technique in depth. Although some surveys on DoS countermeasures have already been published, none of them analyzes countermeasures in detail and moreover none of them presents a countermeasure against each defense mechanism in the attacker's point of view. Mirkovic and Reiher [8] presented a taxonomy of DDoS attacks and defense mechanisms. Molsa [9] provided a tutorial on mitigating DDoS attacks. Chen et al. [10] discussed the characterization of defense mechanisms against DDoS attacks. Carl et al. [2] provided a survey on DoS detection techniques.

The contribution of this paper is threefold. First, we provide overall issues and the characterization of defense mechanisms. Second, we review the most well-known and famous countermeasures and enumerate their characterization according to the first step. Third, we analyze and evaluate each countermeasure in detail and if it is possible, we provide a countermeasure against each of them in the attacker's point of view.

The rest of paper is organized as follows. Section 2 provides overall issues of defense mechanisms. Section 3 discusses survival techniques. Section 4 analyzes proactive techniques. Section 5 evaluates reactive techniques. Section 6 presents a discussion and finally Section 7 concludes the paper and provides some suggestions for the researchers on this area.

2. Overall defense issues and characterizations

To address how a defense method is designed, deployed or evaluated, the following five questions should be answered:

1. Where is the attack detected?
2. How is the attack detected?
3. What is the response mechanism?
4. Where should the response mechanism be applied?
5. Where is the control (decision) center?

Discussing the first question: there are two locations for attack detection: victim server and routers (either routers close to traffic sources or core routers or routers close to the victim server).

Discussing the second question: this question needs a detailed response which is presented in Section 2.3.

Discussing the third question: there are two mechanisms to respond to DDoS traffic: filtering and rate-limiting. There are two types of filtering techniques. In the first method, the attack traffic is totally filtered based on an attack signature. In the second method, the good traffic is passed through the filter based on a good traffic signature while the rest of the traffic is totally filtered. In the rate-limit technique, the attack traffic is not totally filtered, but is rate-limited. In some rate-limiting techniques, all traffic toward a victim is rate-limited whether traffic is legitimate or malicious.

Although the filtering technique is a promising technique, it has two serious challenges: (1) generating attack signatures or (even) good traffic signatures is not always possible and in a sophisticated attack, the victim server cannot generate such signatures; (2) filtering is a costly task because upstream routers must process all packets toward the victim server and this imposes a large overhead on the routers. In contrast, in those rate-limiting techniques that rate-limit traffic regardless of whether traffic is legitimate or not, (1) there is no need to have or generate signatures and (2) the cost of the mechanism is much less than filtering technique. However, if rate-limit is done at points close to the victim, the legitimate traffic is severely punished.

Discussing the fourth question: according to the infrastructure of the Internet, the response mechanism can be applied in four different points: source-end, core-end, victim-end and distributed. Fig. 2 shows the defense points against DDoS attacks.

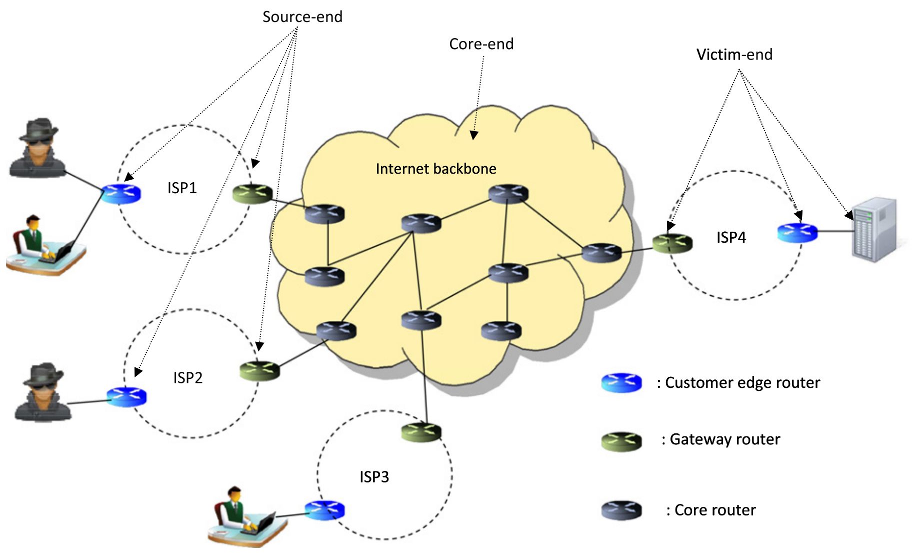

Fig. 2. Defense points.

Source-end points: source-end points refer to the points which are close to the source of traffic. Normally, gateway (inter-provider edge) routers or customer edge routers of the ISP which have traffic toward a victim server are considered as source-end.

Core-end points: these points refer to high-performance backbone routers of the Internet.

Victim-end points: these are the points that are close to the victim server. Normally, gateway routers of the ISP which hosts the victim server, customer edge router(s) of the ISP which the victim server connects to the ISP through it (them), or sometimes the victim server, itself, are considered as victim-end points.

Distributed points: distributed points consist of combination of source-end, core-end and victim-end points.

Discussing the fifth question: control (decision) locations refer to points where the filtering rules, rate-limit amounts, starting and stopping the defense are taken. There are three points for control centers: source-end, core-end and victim-end.

2.1. Classifying DDoS defense techniques based on defense points

Based on the above discussion and the points of defense, defense methods can be classified in four categories: source-end, core-end, victim-end and distributed.

2.1.1. Source-end defense techniques

These techniques are deployed at source-end points. These nodes autonomously detect malicious packets and then filter or rate-limit outgoing malicious packets toward a victim server. Source-end defense points are the best points to filter or rate-limit malicious traffic because minimum damage occurs for legitimate traffic. Moreover, any source-end defense point experiences a small portion of traffic and consequently it needs minimum amount of resources (e.g., processing power and buffers) for the defense. However, distinguishing malicious traffic from legitimate traffic is a serious challenge. Moreover, a source-end defense technique cannot observe its own effect on suspicious traffic (e.g., rate-limit or filter) at the victim side simply because it has no interaction with the victim. Hence, the percentage of false positive and false negative could be high. D-WARD [11] is an example of a source-end defense technique.

2.1.2. Core-end defense techniques

In these techniques any core router independently tries to detect malicious traffic and then filter or rate-limit the traffic. As core routers handle a large volume and highly aggregated traffic, they are not likely points to detect and filter traffic for each possible victim, separately. In fact, due to large volume of network traffic, it is not likely that core routers assign a part of their CPU cycles or their memory to detect any possible DoS traffic and then filter the traffic based on an attack signature. Another challenge for these techniques is that discriminating malicious traffic from good traffic is not an easy task. Core routers are promising points to rate-limit traffic toward a victim server regardless of the type of traffic. The architecture proposed by Zhang and Parashar [12] is an example of this type.

2.1.3. Victim-end defense techniques

Victim-end defense points can easily discriminate DoS traffic from legitimate traffic (see Section 2.3). The major problem with victim-end defense techniques is that victim-end defense points are not good points for rate-limiting or filtering attack traffic because the bandwidth might be saturated. Approaches proposed in [13,14] are victim-end defense methods.

2.1.4. Distributed defense techniques

As discussed above source-end points are promising points to rate-limit or filter malicious traffic; core-end points are promising points to only rate-limit traffic regardless of type of traffic and finally victim-end points are promising points to detect and discriminate DoS traffic from legitimate traffic. Hence, a cooperative mechanism between source-end and victim-end, or between core-end and victim-end, or between source-end, core-end and victim-end can be favorite defense techniques against DDoS attacks. Victim-end points can detect DoS traffic, generate an attack signature, generate good traffic signature and then ask upstream routers (e.g., source-end points) to rate-limit or filter traffic according to the signature. Distributed defenses seem a proper solution to tackling DDoS threats. However, they are infrastructural solutions that need cooperation from multiple ISPs and administrative domains. Security and authenticating the communication channels also are hard problems that should be solved.

2.2. Classifying DDoS defense techniques based on reaction time

Based on the reaction time to DDoS attacks, the countermeasures can be classified into three categories: survival, proactive and reactive.

Survival techniques suggest that the victim server tackles DDoS attacks by enlarging bottleneck resources. Proactive techniques prevent happening of DDoS attacks by some modifications in gateway routers, core routers, etc. Reactive techniques fight with DDoS attacks after happening of DDoS attacks. In Sections 3–5, we provide an in-depth analysis of well-known techniques in each category.

2.3. DDoS attack detection

The aims of DDoS detection methods are (a) detection of occurrences of an attack and (b) distinguishing legitimate packets from DDoS packets. In this section, first, we discuss where and how a DDoS attack is detected; then we discuss why a detection technique might be required and finally we introduce some well-known detection techniques. The detection mechanisms depend on the type of attacks (network-layer or application-layer) and on the location of detection. There are two locations to detect a DDoS attack: (a) victim server and (b) routers (either routers close to traffic sources, or core routers, or end-routers close to the victim server).

Let us first discuss attack detection at the victim server. A victim server can **easily** detect network-layer DDoS attacks; but it can **hardly** detect application-layer DDoS attacks. A network-layer DDoS attack is detected easily at the victim server because DDoS packets are bogus packets, mostly with random payloads and invalid sequence and acknowledgment numbers. Thereby, the victim server can easily distinguish these bogus packets from legitimate packets. Legitimate users always establish a TCP connection and then through systematic and controlled sequence numbers (TCP flow control) send packets which include legitimate and correct payload. While in a network-layer DDoS attacks, zombie machines do not establish TCP connections and simply flood the server with bogus packets that include forged source IP addresses and random (meaningless) payloads. Thereby, when the number of these bogus packets increases and the total traffic volume at the victim server exceeds a threshold rate, the victim sever can detect that it is under **network-layer DDoS attacks** and meanwhile it can easily distinguish legitimate packets from DDoS attack packets.

Unlike network-layer DDoS attacks that can easily be detected at a victim server, a victim server cannot easily detect application-layer DDoS attacks. The reason is that in application-layer DDoS attacks, all zombie machines establish TCP connections and similar to legitimate users send legitimate requests (e.g., downloading web pages). In application-layer DDoS attacks, the adversary has to disclose the IP address of all its zombie machines; otherwise, the zombie machines cannot establish TCP connections. But, the interesting point is that due to the large number of zombie machines that the adversary has, the adversary does not worry about disclosing IP addresses of its zombie machines. **The most challenging issue** for a victim server here is how it can detect a flash crowd[1] from an application-layer DDoS attacks. Several innovative techniques have been proposed to assist a victim server to how distinguish a flash crowd from an application-layer DDoS attack [15–17]. However, evaluating and discussing these techniques are out of the scope of this paper.

Let us now discuss attack detection at routers. Routers can easily detect neither network-layer DDoS attacks nor application-layer DDoS attacks. The reason is that the only criterion that a router can judge about a packet is the header part of packets (the payload part of packets for routers is meaningless). On the other hand, in network-layer DDoS attacks, sophisticated attackers send legitimate-like packets toward the victim server; i.e., routers cannot distinguish between headers of bogus packets from headers of legitimate packets. Moreover, in application-layer DDoS attacks, the DoS packets are completely legitimate. However, routers close to the victim server may have a better chance to detect a DDoS attack than core routers and routers close to traffic sources because the DDoS traffic becomes aggregated at routers close to the victim server; consequently, these routers can collect more specifications for attack packets.

As discussed above, a victim server can easily detect network-layer DDoS attacks; now the question is why a victim server may still need detection techniques. The main reason for this is that if a victim server could detect an attack before it could make a serious damage, then the victim server can win more time to react effectively. Why may routers need detection techniques? The reason is that if they could detect a DDoS attack, then they can filter traffic before it could reach the victim server (these techniques are well-known as proactive techniques and we discuss them throughout the paper). Two important characteristics of detection techniques that designers of such techniques should consider are as follows. The first issue is that a detection technique should detect the attack in real-time, otherwise, the DDoS attack reaches the victim server and may damage the server; in this case the existence of detection technique would be useless. The second issue is that detection technique should be very accurate otherwise either false positives would be high (i.e., legitimate traffic is dropped mistakenly) or false negatives would be high (i.e., malicious traffic is not dropped and reaches the victim server). Some well-known DDoS detection mechanisms are as follows. In this paper, we are not going to evaluate these techniques (the paper only focuses deeply on countermeasures).

- **Sequential change-point detection:** Change-point detection techniques [18,19] isolate a traffic statistics' change caused by attacks. The mechanisms normally use address, port, or protocol for change-point detection and store the resultant flow as time series. The time series can be considered a time domain representation of a cluster's activity. If a DoS flooding attack begins at time λ, the time series will show a statistical change either around or at a time greater than λ.
- **Wavelet analysis:** Wavelet analysis [20,21] describes an input signal in terms of spectral components. Wavelets are used to isolate characteristics of signals via a combined time–frequency representation. Wavelet analysis allows the predictable and ambient part of traffic (legitimate part), normally appearing as low-frequency components in the wavelet spectrum, to be removed. In the next step, the time-localized anomaly signals that normally appear as high and middle-frequency components in the wavelet spectrum can be detected using statistical methods.
- **Neural networks:** Neural networks [22,23] combine machine learning (training) and information visualization techniques to detect DDoS attacks. The neural network is trained with logs of network traffic (including several attributes) and then through visualization techniques (e.g., U-Matrix [24]), we can see the existence of possible DDoS attacks. Some well-known neural networks to detect DDoS attacks are Self-Organization Maps (SOMs) [24], Learning Vector Quantization (LVQ) [22], Radial Basis Function (RBF) [25] and Back Propagation (BP) [26].
- **Statistical techniques:** These techniques [27,28] measure various statistical properties of specific fields in the packet headers during normal conditions. Then, using these statistical properties, the system creates a reference model for normal traffic. When an attack occurs, the computed statistical properties for current traffic show differences from the reference model and using these differences the DDoS packets are detected. Some

[1] Flash crowd is the sudden increase of the workload in a server when many legitimate users simultaneously (or in a short time) connect to the server.

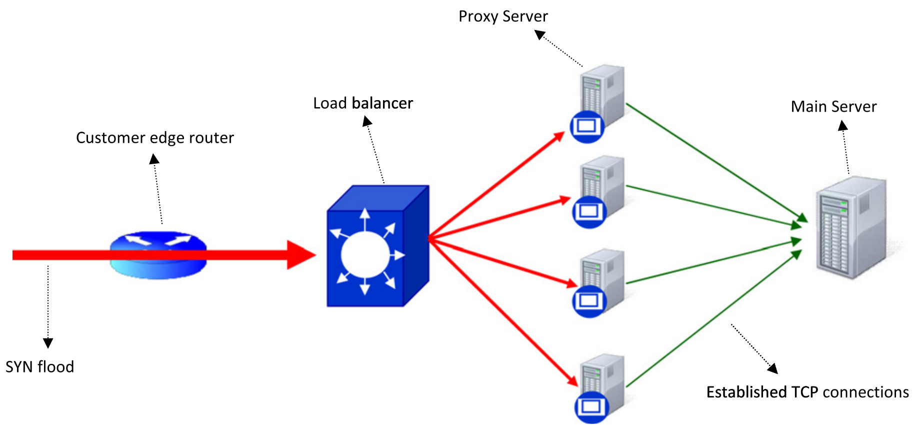

Fig. 3. The proxy servers technique against SYN flood attack.

well-known statistical methods to detect DDoS attacks are Entropy [27], Chi-Square [27] and Kolmogorov–Smirnov (KS) [28].

Some other techniques to detect DDoS attacks are data mining techniques [29] and Fuzzy systems [30]. However, several reactive defense techniques do not rely on the above detection techniques and do not need those detection techniques. In fact, in several reactive countermeasures, the attack detection is **threshold-based**[2]; i.e., whenever traffic rate at the victim server exceeds a threshold rate, the defense mechanism is initiated.

3. Survival techniques

These techniques are victim-end defense techniques. In these methods, a victim counters DDoS attacks through enlarging the resources such as bandwidth, memory, CPU power, TCP buffers, replication, etc. Enlarging the resources causes the victim to receive and process both legitimate and malicious traffic without any degradation in response to legitimate clients. Enlarging the resources can be done either statically or dynamically. In the static manner, the victim normally purchases more resources; while in the dynamic manner, the victim replicates its services in several public servers. The replication procedure can effectively protect static services from DDoS attacks. However, the replication procedure cannot protect dynamic services (e.g., web pages) from DDoS attacks because of updating and management issues. We note that mostly dynamic services are targets for attackers; thereby the replication procedure cannot be considered as a defense technique. Here we present a few simple survival techniques against SYN flood attacks.

Multiple proxy servers: as a server has limited TCP buffers, a flood of SYN packets can paralyze the server. In this simple technique, a set of proxy servers are located in front of main server and a load balancer distributes the arriving SYN packets between the proxy servers (see Fig. 3). A proxy server redirects a request to the main server if and only if all three steps of the 3-way handshake have been completed successfully. As multiple proxy servers can have TCP buffers far more than a single server, it can handle a relatively large SYN flood. However, when the attacker attacks the server through hundreds thousands of zombie machines, this technique can never protect the main server from being downed.

Enlarging the backlog queue: the backlog queue is a large memory structure used to handle incoming packets with the SYN flag set until the moment the three-way handshake process is completed. An operating system allocates a part of the system memory for every incoming connection. The backlog queue controls how many half-open connections can be handled by the operating system at the same time. When the queue is saturated, subsequent requests are silently dropped by the operating system. Under a SYN attack, we can enlarge the size of the backlog queue to support more connections in the half-open state without denying access to legitimate clients. Increasing the backlog queue size requires that a system reserves additional memory resources for incoming requests. If a system has not enough memory for this operation, it will have an impact on system performance.

Changing the timeout for connection requests: operating systems such as Microsoft Windows hold a SYN packet in the backlog queue for a time something between 45 s and 360 s. When there is no SYN attack, this time does not cause any problem; but when the system is under attack, this time is a bottleneck challenge. Since 90% of all transferred TCP packets have a RTT (Round Trip Time) below 500 ms [31], and only 15% of connections have a median RTT greater than one second [32], in case of attack against a system, the timeout time can be reduced to a time for example less than 2 s. Therefore, a buffer is released far faster when the ACK packet does not arrive on time.

Combination of all techniques: a combination of the above survival techniques can be a better approach against SYN flood attack. However, when the SYN flood attack overwhelms the bandwidth, none of above techniques can protect the server.

3.1. Analysis of the technique

Due to deployment difficulties of techniques of the two other categories (proactive and reactive), today, victims normally counter DDoS attacks by survival techniques. Although survival techniques can protect the victim's resources from small scale DDoS attacks, they cannot counter large scale DDoS attacks. We note that attackers can easily employ hundreds of thousands zombie machines and attack the victim through them; thereby victim servers cannot rely on such techniques.

4. Proactive techniques

These techniques rely on prevention techniques. In these techniques, the attack is prevented before it can damage a victim. In these techniques, victims do not notice that they were a target for a DDoS attack. Routers play a main role in these techniques by detecting and filtering malicious traffic. In some techniques [11], any router autonomously tries to distinguish malicious traffic

[2] It is worth note that the threshold-based detection mechanism, formally locates at the category of sequential change-point detection.

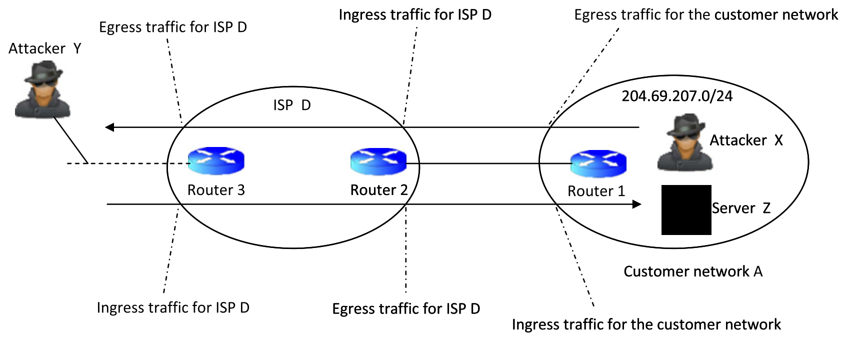

Fig. 4. An example of ingress filtering based on RFC 2827.

from legitimate traffic and then filters malicious traffic and in some techniques [12] a cooperation procedure is set up between routers to accurately distinguish legitimate traffic from malicious traffic and effectively filter or rate-limit the malicious one. Below, we study and analyze some well-known proactive techniques in detail.

4.1. Ingress/egress filtering

Ingress filtering means filtering the traffic coming into your network and egress filtering means filtering the traffic leaving your network. To see how ingress/egress filtering is done, let us discuss the example given in RFC 2827 [33]. As shown in Fig. 4, ISP *D* provides Internet access to the customer network *A* which can be a university or an enterprise network. Router 1 is the edge router of the customer network *A*, router 2 is the edge router of the ISP and router 3 is another edge router of the ISP that connects to other networks. Suppose that the ISP defines the address space of 204.69.207.0/24 for the customer network *A*. Also suppose attacker *X* resides in the customer network *A* and attacker *Y* resides in another customer network which is interconnected to the ISP *D* through some intermediate ISPs.

The ISP can design an ingress filter, install it in the **input** port of router 2 that is connected to the customer network *A* and set up the filter such that it drops any packet coming from the customer network *A* beyond the address space of 204.69.207.0/24. So, if attacker *X* sends packets with spoofed source IP addresses beyond the address space of 204.69.207.0/24, then the packets are dropped by the ingress filter installed on router 2. Similarly, the customer network *A*, itself, can design an egress filter, install it in the **output** port of router 1 which connects the customer network to ISP *D* and drop any packet which has source IP address beyond the defined space.

In another scenario, ISP *D* can design an ingress filter, install it in input port of router 3 which connects the ISP to other networks (e.g., other ISPs) and drop any packet coming with obvious wrong source IP addresses. For instance, if attacker *Y* sends packets with source IP addresses such as 10.*.*.* which are invalid source IP addresses, the packets are dropped at router 3. However, if router 2 provides the same functionality and drops packets with obvious wrong source addresses leaving the ISP is called egress filtering. If this functionality is done by router 1, it is called ingress filtering.

4.1.1. Analysis of the technique

Specification of the technique: Detection location: routers close to traffic sources; detection mechanism: checking source IP addresses; type of defense: source-end; response mechanism: filtering technique; control (decision) center: source-end; type of attack that it can protect against: both direct and reflector attacks.

Evaluation: it is easy for an ISP to have knowledge of the address space that is assigned to each customer network. Hence, it is easy for an ISP to design such ingress and egress filters and prevent IP spoofing. Furthermore, filtering can be done not only based on IP addresses, but also protocol type, port number, or any other criteria of importance in both ingress and egress filters. The main challenges with this technique are as follows:

1. Universal deployment: this technique can only be effective when almost the ISPs in the world implement these filters. If attackers are aware of lack of such filters in some ISPs, they can use zombie machines of those ISPs to attack the target.
2. Spoofing IP addresses from the subnet: an attacker still can use IP spoofing if it uses spoofed IP addresses from the subnet range. For example, in Fig. 4, if attacker *X* uses spoofed IP addresses from the range 204.69.207.0–204.69.207.255, the ingress filter installed on router 2 does not detect IP spoofing.
3. Non-spoofed attacks: today, several DDoS attacks do not use IP spoofing to hide the location of zombie machines. By exploiting a large number of zombie machines, an attacker can set up various DDoS attack types without using IP spoofing. For instance, in HTTP flood attack, zombie machines use their own IP addresses to download the web-pages. In this case, ingress/egress filter is never effective.
4. Administrative overhead: network ingress filtering imposes some administrative overhead, especially to initially deploy as well as to maintain the filters.
5. Performance cost: network ingress filtering may impose a performance cost to a router implementing the ingress filtering because the router must check new rules for every packet. In many cases, ingress filtering can be implemented using one or two light rules that can be checked without noticeable performance degradation of the router. However, this is not true for all networks. The approach may have a particularly large performance overhead on routers that support a very large number of different address ranges. In addition, it may lead to performance degradation in routers that are very close to their maximum load. In this case, such routers should be upgraded to new hardware tools to not impose noticeable performance degradation to their customers.
6. Problem with mobile and multi-home IP addresses: in mobile IP environments, where a roaming host must use a “home” IP address in a foreign network, the ingress/egress filtering poses problem.
7. No benefit for ISPs: ingress filtering has no benefits for the ISPs that install these filters.

Due to these serious challenges ingress/egress filtering is not a promising technique against DDoS attacks. Moreover, ISPs’ administrators have not been convinced to install ingress/egress filters at customer edge routers of ISPs.

4.2. Route-based distributed packet filtering (DPF)

Park and Lee [34] extend ingress filtering to core routers of the Internet. DPF uses routing information to determine if a packet arriving at a router— e.g., a border router at an AS (Autonomous System)— is valid with respect to its inscribed source/destination addresses, given the reachability constraints imposed by routing and network topology. The fact behind the DPF approach is that for each link in the core of the Internet, there is only a limited set of source addresses from which traffic on the link could have originated. Let us explain the DPF approach through an example. Consider the graph depicted in Fig. 5 that each node shows a border router in a different AS. Solid lines show route from node 2 to all other nodes. Assume a host belonging to AS 6 is attempting a DoS attack targeted at a server residing in AS 3 by using a forged source IP address belonging to AS 2. A border router belonging to AS 5 at the peering point with AS 6—if cognizant of the route topology—would recognize that a packet originating from AS 2 destined to AS 3 would not enter through link (6, 5) implying that its source address must be spoofed. Such packets could be discarded at AS 5, thus proactively protecting AS 3 from the DoS attack.

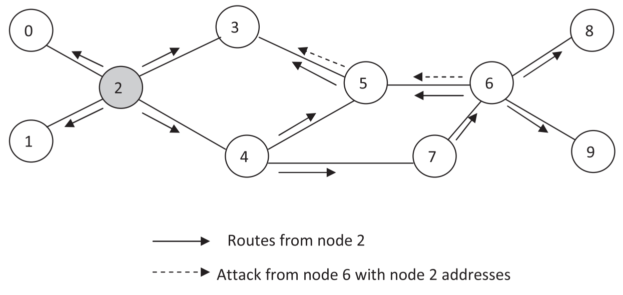

Fig. 5. Route-based distributed packet filtering (example from [34]).

4.2.1. Analysis of the technique

Specification of the technique: Detection location: core routers; detection mechanism: checking routing information; type of defense: core-end; defense mechanism: filtering technique; control (decision) center: core-end; type of attack that it can protect against: both direct and reflector attacks.

Evaluation: if all border routers of all ASs are equipped with this scheme, then spoofed IPs are significantly prevented. Park and Lee show that even if DPF is implemented on 18% of ASs, a significant percentage of those spoofed IPs that have been generated randomly are prevented. However, DPF has several challenges:

1. Universal deployment: DPF is an effective mechanism if most of ASs in the Internet implements DPF which seems an onerous task.
2. Performance degradation: since border routers of ASs must check source IP address of any packet, the performance of the Internet may decrease (see item 5 of Section 4.1).
3. Collateral damage: sometimes, it is possible that legitimate traffic is mistakenly filtered (false positive). For instance, in Fig. 5, suppose traffic cannot be routed through links (2, 3) and (4, 5) due to congestion problem, link failure or etc. In this case a border router of AS 2 must route traffic of a host residing in AS 2 via path 2–4–7–6–5–3 to a server residing in AS 3. In this case the border router of AS 5 at the peering point with AS 6 drops the traffic because it imagines that the source IP address of the traffic is spoofed. In fact, due to dynamic nature of Internet, the percentage of collateral damage could be high.
4. Sophisticated IP spoofing: a sophisticated attacker can still spoof IP addresses based on the network topology, especially, if only some ASs employ DPF technology. For instance, in Fig. 5, if AS 2 does not employ the DPF filter, then if an attacker residing in AS 1 wants to attack to a server residing in AS 4, he can use IP spoofing via IP addresses of AS 0, AS 1 or AS 2. In fact, with DPF, IP spoofing is still possible, if an attacker carefully chooses source IP addresses. Moreover, similar to ingress filtering, IP spoofing also is still possible with IP addresses of subnets.
5. Non-spoofed DoS attacks: as discussed above, today, several DDoS attacks use non-spoofed IP addresses; thereby employing DPF becomes useless.

Due to above challenges, it seems that DPF technology is ineffective approach against DDoS attacks.

4.3. D-WARD

D-WARD [11] is a firewall that is deployed at source-end networks, and autonomously detects and stops DoS attacks originating from these networks. D-WARD gathers two-way traffic statistics from the border router at the source network and compares them to network traffic models built based on application and transport protocol specifications, reflecting legitimate, suspicious, and attack behavior. Models are built at two granularities: for each communication with a single destination (called a "flow") and for individual connections. D-WARD applies a rate limiting to all outgoing traffic for a given destination which seems to be under attack, preferring legitimate connection traffic, slightly slowing down suspicious traffic, and severely slowing down attack connections.

4.3.1. Analysis of the technique

Specification of the technique: Detection location: routers close to traffic sources; detection mechanism: statistical methods; type of defense: source-end; defense mechanism: rate-limit technique; control (decision) center: source-end; type of attack that it can protect against: both direct and reflector attacks.

Evaluation: the experiments shown at [11] indicate that D-WARD has the ability to quickly detect those attacks that create anomalies in two-way traffic, such as heavy floods, including some on–off or pulsing attacks. It stops attacks at the source networks. Hence, the damages caused by DoS attacks on the victim server are limited. However, D-WARD has serious challenges:

1. Universal deployment: D-WARD can only effectively stop DDoS attacks if it widely is deployed in the Internet which seems unrealistic.
2. No benefit to the deployer: installing D-WARD has no benefit to the deployer ISP; thereby administrators of ISPs are not motivated to install D-WARD on border routers.
3. Performance degradation: D-WARD significantly degrades performance of the system. As D-WARD polices any traffic flow, the customers behind the D-WARD filter experience slower Internet connections which are not favorable for them.
4. Large overhead: D-WARD imposes a large overhead on the routers because the router must permanently (all 86,400 s a day) monitor traffic, classify traffic based on destination IP addresses, measure statistics for each category, compare statistics with a reference model and then rate-limit traffic. This implies that routers must have sufficient processing power, memory, etc., while a border router may lack such requirements. Even if future routers are equipped with sufficient processing power, memory, etc. the question is why an ISP's administrator must pay for such routers when D-WARD has no benefits for the ISP.
5. Collateral damage: since distinguishing legitimate traffic from attack traffic is not accurate at the source networks, the percentage of collateral damage (false positive) might be high. The percentage of false negative can be high as well.

6. Weak against application-layer DDoS attacks: D-WARD is a weak technique against application-layer DDoS attacks such as http flood.

Due to the above challenges we believe an ISP's administrator is not encouraged to install D-WARD on the border routers of the ISP. The serious question is why should all customers of an ISP experience slow internet links only because of a possible DoS attack against a victim located in another ISP domain? We note that D-WARD always polices traffic regardless of whether is an attack against a server or not; so another question is why customers of the ISP should suffer from slow links when there is no DoS attack against a server?.

4.4. Proactive cooperative defense

Zhang and Parashar [12] suggest that border routers of autonomous systems (ASs) autonomously and locally detect DoS traffic and generate an attack signature for DoS traffic. Then border routers of different ASs compose a peer-to-peer network and exchange attack signatures using a gossip-based communication mechanism to achieve an accurate attack signature. Next, the border routers negotiate and decide how much rate-limit should be applied to the attack traffic.

4.4.1. Analysis of the technique

Specification of the technique: Detection location: routers close to the victim server; detection mechanism: statistical methods; type of defense: distributed; defense mechanism: rate-limit technique; control (decision) center: distributed; type of attack that it can protect against: direct attack and partially reflector attacks.

Evaluation: the cooperative mechanism helps to reduce the percentage of false positives and in fact, helps to generate a more accurate attack signature. Cooperation also assists routers to achieve an accurate rate-limit amount. However, the approach suffers from the same challenges as D-WARD suffers. Moreover, the overhead imposed on routers in this approach is higher than for the D-WARD technique (reasons: (a) core routers experience more traffic than edge routers in D-WARD and (b) overlay communication overhead). Consequently, overall internet speed is decreased for all users because a border router should permanently monitor traffic, generate an attack signature for a possible attack, exchange the attack signature with other border routers and finally decide a suitable rate-limit. Undoubtedly, this overhead affects the standard task of a router (standard routing) which leads to a slower internet speed.

4.5. Internet indirection infrastructure (i3)

i3 [35] is an overlay-based communication technique to defeat DDoS attacks. i3 was followed by Si3 [36]. i3 consists of a set of nodes that store triggers and forward packets (using IP) between i3 nodes and to end-hosts. Packets are of the form $(id, data)$ and triggers are of the form $(id_t, addr)$, where id is the identification of the packet, $data$ is the payload of the packet, id_t is the trigger's identification and $addr$ indicates a node's address which consists of an IP address and a port number. A trigger $(id_t, addr)$ indicates that all packets with an identifier id which have the same prefix id_t should be forwarded (at the IP level) by the i3 infrastructure to the node identified by $addr$. The communication between a server and clients is as follows: the server that expects connections from arbitrary clients inserts triggers whose identifiers are well-known in some i3 nodes. These triggers are called public triggers. Next, when a client wishes to send a packet, e.g., $(id, data)$ to the destination, it delivers the packet to an i3 node (suppose the client knows at least one i3 node). If the contacted node did not store the matching trigger $(id, data)$, the packet is forwarded via IP to another i3 node. This process continues until the packet reaches the i3 node that stores the matching trigger. The packet is then sent to the destination via IP. Once a client contacts the server through its public trigger, they exchange a pair of identifiers which they use for the remainder of the communication. Triggers corresponding to these identifiers are referred to as private triggers. An attacker also can send traffic toward the server because the attacker also can explore public triggers of the server but if a server receives flood traffic from a particular trigger, it can simply stop receiving packets from the trigger by simply removing that trigger. Fig. 6 illustrates the communication between two nodes.

4.5.1. Analysis of the technique

Specification of the technique: Detection location: victim server; detection mechanism: threshold-based; type of defense: distributed and overlay-based; defense mechanism: filtering technique; control (decision) center: victim-end; type of attack that it can protect against: both direct and reflector attacks.

Evaluation: i3 was the first overlay-based mechanism against DDoS attacks that clients indirectly connect the server using a rendezvous primitive. The architecture is simple. The management of the system is easy and it has a low communication overhead; but it has several serious challenges:

1. The architecture withstands DDoS attacks if we assume that the whole overlay network is completely trusted and attackers have no means to find the IP address of trigger nodes[3] which is not realistic. If we assume that the overlay network is not totally trusted and attackers have the power of traffic analysis, then a simple traffic analysis attack [37–39] can break the architecture.
2. i3 could potentially be overwhelmed by a bandwidth attack on the firewall that allows only packets from triggers passing through it.
3. The architecture cannot protect the server against meek attacks where there are a large number of attackers but each one attacks the server with small attack traffic. In this case the server either stops or rate-limits traffic from all triggers which punish also legitimate traffic or does not stop or rate-limit traffic from triggers which leads to saturation of the server.

4.6. Secure overlay services (SOS)

In the SOS architecture[4] [40], first, traffic of users is authenticated in SOAP (Secure Overlay Access Points) nodes; then it is routed through the Chord overlay network to the beacon nodes. Next beacon nodes deliver traffic to secret Servlet nodes and finally Servlet nodes deliver traffic to the victim server which is protected via filters. The filters only accept those packets of which source IP addresses match the IP address of Servlet nodes.

4.6.1. Analysis of the technique

Specification of the technique: Detection location: victim server; detection mechanism: threshold-based; type of defense: distributed and overlay-based; defense mechanism: filtering technique; control (decision) center: victim-end; type of attack that it can protect against: all types of DoS attacks.

[3] The authors of the paper explicitly mention that "we assume here that it is very hard for the attacker to find the IP address of trigger nodes by other means" [36].

[4] The architecture of SOS is constructed in proactive time, but the victim server switches to this architecture after the attack happens. So, the SOS architecture also can be categorized in the reactive techniques as well.

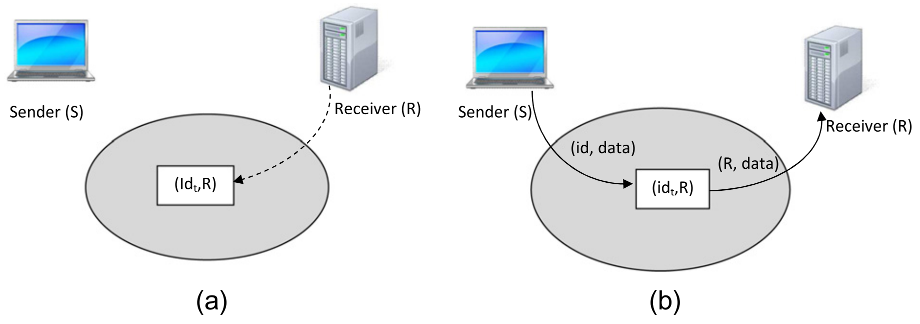

Fig. 6. Communication architecture between sender and receiver in i3: (a) the receiver R inserts trigger (id,R), (b) the sender S sends packet (id, data).

Evaluation: SOS simplifies the filter rules and it proposes a lightweight filter to tackle DDoS attacks. To achieve this goal, SOS proposes to receive data only from secret Servlet nodes and only checks the source IP address of packets. However, SOS suffers from the following challenges:

1. SOS is very weak against traffic analysis attacks [37–39] in such a way a simple traffic analysis attack immediately finds the location of Servlet nodes. SOS does not use proper cryptography and moreover, SOS uses static routing through the Chord overlay network. In architectures such as SOS, discovering only one Servlet node equals to collapsing the architecture.
2. If the adversary could monitor traffic of α percent of the Internet (e.g., 20%), he has α percent chance to discover the IP address of a Servlet node without any further traffic analysis.[5] Additionally, if the server has k Servlet nodes, then the adversary has a probability of $1-(1-\alpha)^k$ percent to discover at least the IP address of one servlet node (e.g., when $\alpha = 0.2$ and $k = 5$, this probability is 70%). On the other hand discovering the IP address of one servlet node is enough to bypass the filter.
3. If the adversary could passively monitor traffic nearby the victim server (e.g., by compromising the gateway router of the victim server), the algorithm completely fails because the adversary can easily discover the IP address of Servlet nodes. In this case, the victim cannot protect the location of Servlet nodes by changing their location.[6]
4. SOS could potentially be overwhelmed by a bandwidth attack on the firewall that allows only approved packets. In fact, the challenge is where to deploy the network filter. The link capacity can be saturated if the network filter is situated too close to the victim, while moving it toward the core raises scalability and security concerns as the number of routers on which the filter should be installed on is increased. The reason for scalability problem is that anytime the location of even one Servlet node changes, firewalls installed on all routers should be notified and modified. The reason for security concern is that increasing the number of routers which know filter rules increases the chance of attackers to disclose location of Servlet nodes because if even one router is passively monitored by the attacker, the SOS architecture fails.
5. Routing traffic through Chord is not robust against massive failure. In fact, if the adversary could bring down 40% of overlay nodes simultaneously, more than 70% of packets are lost [44]. This means that traffic of clients cannot reach the Servlet nodes.
6. SOS cannot protect global servers (e.g., Yahoo, Google) from DDoS attacks as any client must be authenticated in SOAP nodes.
7. A compromised client can easily find the location of a Servlet node. For instance, suppose that Chord has N nodes; the Servlet node is one of these nodes. As SOS says, the attacker may know the IP addresses of all the overlay nodes but does not know which of them is a Servlet node. Now, in a loop, the compromised client encapsulates its request in a packet and at each iteration of the loop inserts the IP address of one of the overlay nodes in the source IP address of the packet and sends the packet directly to the victim server. The IP address on which the compromised client gets a response from the victim server is the IP address of the Servlet node because the packet has successfully passed through the firewall. Our simulation results show that if an overlay network has 1000 and 10 000 nodes respectively, the attacker can disclose the location of a Servlet node in less than 18 and 50 s respectively. Moreover, SOS uses multiple Servlet nodes (e.g., 5 nodes) due to fault-tolerance. Attackers are happy with this policy of SOS because their brute force technique can discover a Servlet node much faster. We note that discovering only one Servlet node is enough to overwhelm the victim server with a flooding attack.
8. If an attacker could passively monitor traffic of a legitimate client and measure the response time, he can discover the location of a Servlet node through an adaptive flooding. Consider Chord has N nodes. If the attacker floods the victim server with N packets, each packet uses one IP address of overlay nodes as source IP address, then one packet passes the filter and reaches the victim server and thereby consumes limited resources of the victim server. Suppose the victim server has an ability to receive/process maximum M packets per seconds. Also suppose that the attacker can generate a certain number of packets per second, say F; in most DDoS attacks, $F \gg M$. In the first round, the attacker divides N IP addresses of overlay network into n groups and sends F packets toward the victim server using the first set of IP addresses (i.e, N/n IP addresses). If the IP address of Servlet node is inside the first set of IP addresses, then nF/N packets pass the filter and reach the server. If the attacker sees that the client receives service from the server slowly, it knows that the IP address of the Servlet node is inside this

[5] By checking the destination address of packets, an adversary can easily discover the IP address of Servlet nodes because the adversary recognizes its packets and on the other hand the only legitimate nodes that send packets toward the server are Servlet nodes.

[6] This is a strong attack model. However, previous studies [38,39,41] indicate that it is possible for professional and strong attackers to passively monitor traffic of some part of Internet (e.g., nearby the victim server). Previous studies [42,43] also indicate that attackers have repeatedly demonstrated their ability to compromise routers and passively monitor traffic.

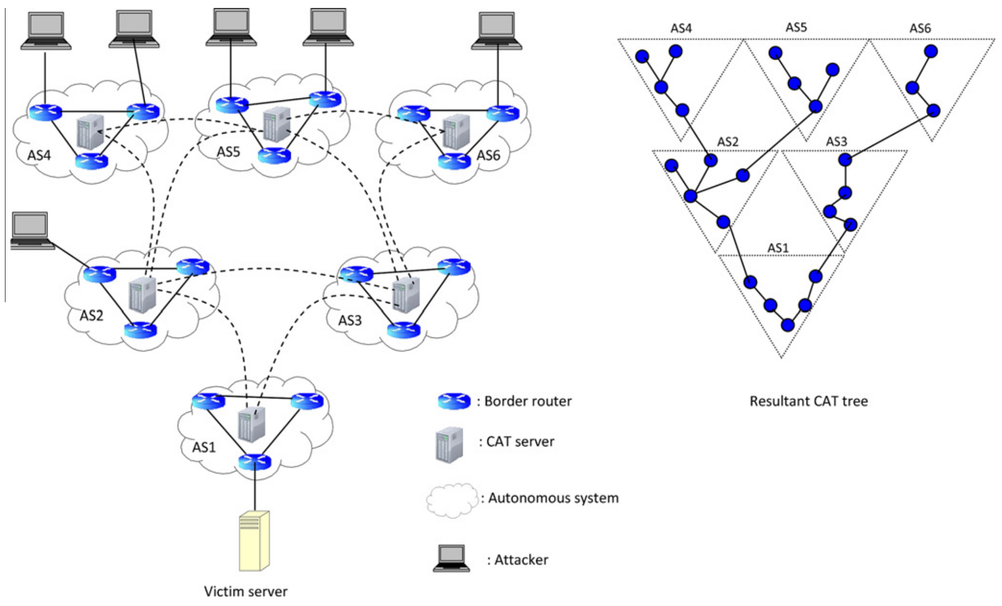

Fig. 7. The DCD architecture (an Example 6-domain global CAT tree construction environment [18]).

set because when DDoS traffic reaches the server and consumes the limited resource of the bandwidth, it causes the service becomes slow. However, if the Servlet node was not in the first set of IP addresses (i.e., the attacker does not notice the slow response), then the attackers flood the victim server with the next set of IP addresses until it notices the Servlet node belongs to which group. When the attacker found the group which the IP address of Servlet node belongs, then the attacker recessively divides that group into some small groups and continues the above procedure until it uniquely finds the IP address of Servlet node. As discussed above, due to fault tolerance, SOS uses multiple Servlet nodes. This assists the attacker to find the location of at least one Servlet node faster. Finding the location of one Servlet node is enough to bombard the victim server.

4.7. Collaborative detection of DDoS attacks

Chen et al. [18] propose the DCD (Distributed Change-point Detection) architecture in order to construct a change aggregation tree (CAT) for early detection of DDoS attacks. In this scheme, several autonomous systems (AS) such as ISPs should cooperate and meanwhile, every AS should permanently allocate a central CAT server for this work. The system has a hierarchical detection architecture as follows. At the lowest layer, individual border routers of ASs function as sensors to monitor local traffic fluctuations for every possible domain. To achieve this goal, a change point detection program is executed on each router. When a DDoS attack happens, routers raise an alarm and send anomaly reports to their central CAT server. The second layer runs at each CAT server. Any CAT server is responsible to construct a CAT subtree for its AS according to alerts collected from routers. Next, at the highest layer the CAT servers at different ASs form an overlay network. CAT servers communicate via VPNs (virtual private networks). All CAT servers send the generated CAT subtrees to the CAT server of the domain where the victim server resides. In fact, the CAT server of the domain on which the victim server resides is the root of the tree. Finally, the root CAT server merges all the CAT subtrees from cooperative ASs; thereby the destination server has a global picture of the attack. Fig. 7 shows an Example 6-domain global CAT tree construction environment.

4.7.1. Analysis of the architecture

Specification of the technique: Detection location: routers including all routers close to traffic sources, core routers and routers close to the victim server; detection mechanism: change-point detection; type of defense: distributed; defense mechanism: unspecified; control (decision) center: victim-end; type of attack that it can protect against: both direct and reflector attacks.

Evaluation: it is desirable and crucial to detect DDoS flooding attacks at their early launching stage before widespread damages are done to legitimate applications on the victim server. In this case, a victim server can win more time to effectively react to the attack. Therefore, the goal of this work is appreciable. However, the DCD architecture suffers from the following limitations and challenges.

1. Similar to the previous proactive techniques, it needs a universal deployment to be effective. In this architecture, several ASs should permanently monitor traffic for various domains.
2. The scheme may impose large overheads to some routers of an AS. The router should permanently, every window time (e.g., 100 ms), measure the parameter of deviation from the average (DFA) for every input port interface and for every domain. To calculate the DFA, the router should first for every input port and for every domain calculate the average number of packets destined to that domain as well as the standard deviation. Next, the router should compare the DFA parameter computed for a domain with a reference threshold parameter (β). This overhead could affect the standard task of a router and also affect the Internet speed.
3. Every AS should permanently allocate a dedicated server to this work. The cost of maintaining the server and security issues of the server is another overhead that is imposed to ASs.
4. As the paper discusses, the report (the constructed subtree) from each CAT server reveals the internal structure of an AS. Therefore, if an attacker could compromise a CAT server or more desirably a few CAT servers, he can acquire the internal structure of several autonomous systems. This vulnerability does not motivate ASs to cooperate. In fact, any AS simply does not like to reveal its internal structure to others.

5. The scheme has no benefits for ISPs that cooperate; consequently the resultant of items 2, 3, 4 and 5 indicates that several ISPs may not cooperate.
6. We are afraid that the architecture cannot detect the early happening of a DDoS attack. The reason is that the attack is detected only when the complete CAT tree is constructed. The complete CAT tree is constructed at the AS that hosts the victim server by merging all CAT subtrees received from other ASs. At each AS, a CAT server needs time to construct a CAT subtree. This time overhead depends on the number of border routers at each AS. Large ASs need more time to construct a CAT subtree. Consequently, the root CAT server needs to wait for receiving all CAT subtrees and then construct the complete CAT tree. When a DDoS attack is large and for example hundreds of thousands routers are involved in the attack, constructing every CAT subtree and emerging all CAT subtrees need a significant large time. Unfortunately, the paper has not discussed time overhead for constructing a complete CAT tree. The detection time is the most important parameter that this paper should discuss, but it has not been discussed. Furthermore, before merging all CAT subtrees, the AS that hosts the victim server, itself, should construct a CAT subtree; in this case, the attack already has reached the victim server and in fact, the ultimate goal of this architecture is not met.
7. The architecture falsely detects flash crowds as DDoS attacks.
8. The architecture either does not detect a meek attack[7] at all or detects it very late.

5. Reactive techniques

In these techniques a DDoS attack occurs against a victim; the victim detects the occurrence of the attack and then appropriately responds to it. In fact, in these techniques, any victim, itself, is responsible to detect an attack and fights with the attack. Most DDoS defense techniques belong to this category. Below, we discuss and analyze the most famous works in this category.

5.1. PushBack

Mahajan et al. [46] studied recursive pushback of max–min fairness[8] rate limits, starting from the congested routers toward upstream routers. In the PushBack technique, first, a local Aggregate Congestion Control (ACC) detects the congestion at the router level and devises an attack signature, or more appropriately, a congestion signature, that can be translated into a router filter. The signature defines a traffic aggregate as a group of traffic flows with a common property (e.g., traffic flows with same destination address). Then, a local ACC determines an appropriate rate limit for this aggregate. Then, the congested router asks its adjacent upstream routers to rate-limit the aggregate. The congested router only sends rate-limit request to those neighbors that send a significant fraction of aggregate traffic toward it. Moreover, the congested router determines the rate-limit amount for each of its upstream neighbor routers based on the max–min fairness algorithm. Next, the receipt routers recursively propagate pushback further upstream. Fig. 8 depicts the mechanism of the PushBack technique.

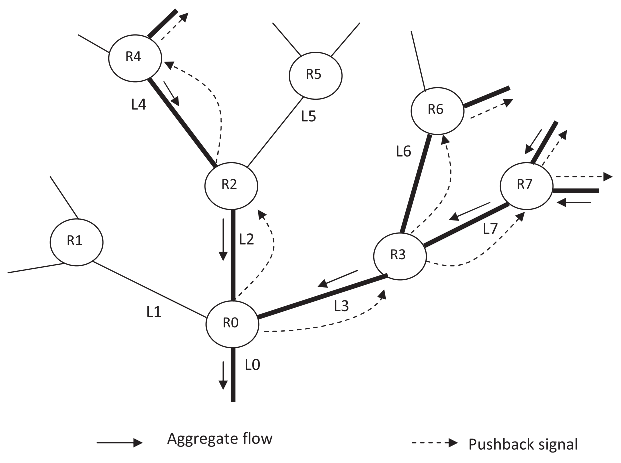

Fig. 8. The architecture of Pushback (the thick lines show links through which significant of aggregate reaches the server).

5.1.1. Analysis of the technique

Specification of the technique: detection location: routers close to the victim server; detection mechanism: threshold-based; type of defense: distributed; defense mechanism: rate-limit technique; control (decision) center: victim-end; type of attack that it can protect against: only direct attacks.

Evaluation: pushback was the first cooperative mechanism against DDoS attacks. PushBack can effectively mitigate DDoS attacks when the attacker's machines are gathered in few places. In fact, the effectiveness of this technique depends on the distribution form of legitimate users and attackers. PushBack has several challenges which we enumerate below:

1. When attackers are widely distributed over the Internet, the legitimate traffic also is rate-limited and PushBack will not be successful. Normally, congested routers are close to the victim server; thereby as resource sharing is started at points close to the victim server and then recursively pushed back to upstream routers, good traffic is severely punished because good traffic aggregates to a large volume close to the victim server.
2. Pushback cannot protect the victim server against a meek attack.
3. Traffic of cumulative legitimate users is severely punished.
4. PushBack potentially hurts innocent sources which share same paths with the attack sources.
5. As all routers between the congested routers and traffic sources are involved in the recursive pushback mechanism, the cost of the technique is high.
6. Any router must maintain a state about traffic flows and periodically inform it to its downstream neighbors; thereby the cost of communication is high as well.
7. It is not guaranteed that all routers accept to install a leaky-bucket[9] for the victim server and push back the rate-limit signal toward upstream routers. The reason is that installing leaky-buckets, dividing the rate-limit in max–min fashion between upstream neighbor nodes, measuring total incoming traffic and reporting it to downstream routers need sufficient computational

[7] In a meek attack, a zombie machine sends traffic toward the victim server at a rate similar to the rates at which normal users send toward the victim server. In fact, in a meek attack, any zombie machine behaves like a normal user. However, in this attack type, the adversary should compromise many hosts (take over many zombie machines) to launch a meek attack. A recent report [45] shows that a group of hackers could capture between 1 to 10 million zombie machines; so meek attacks are possible.

[8] Max–min fairness: a division of the bandwidth resources is said to be max–min fair when: firstly, the minimum data rate that a dataflow achieves is maximized; secondly, the second lowest data rate that a dataflow achieves is maximized, etc. As can be seen the max–min fairness simply means allocating the same share to all.

[9] The leaky-bucket is a technique used in computer networks to guarantee that data transmissions conform to defined limits on bandwidth and burst transmission. Incoming data flows into a buffer (the ''bucket''), then ''leaks'' out at a steady rate, which is designated as constant bit rate traffic.

and memory resources. Moreover, several servers might be under attack at the same time and a router cannot install leaky-bucket for each server. All these evidence show that it is possible that a router may reject such request. If a router rejects this request, the traffic from that path can have a negative effect on other traffic in the downstream routers. In fact, if the pushback signal cannot reach the upstream routers close to traffic sources, this technique cannot protect a victim well.

5.2. K-MaxMin

Yau et al. [47] viewed DDoS attacks as a resource management problem and proposed to distribute the bottleneck resource (e.g., bandwidth) of the victim as max–min fashion between level-*k* routers.[10] In fact, *K*-MaxMin is the enhancement of the recursive max–min (PushBack technique). The authors use AIMD (Additive Increase Multiplicative Decrease) technique to distribute the bottleneck resource as max–min fashion between level-*k* routers.

5.2.1. Analysis of the technique

Specification of the technique: Detection location: victim server; detection mechanism: threshold-based; type of defense: distributed; defense mechanism: rate-limit technique; control (decision) center: victim-end; type of attack that it can protect against: only direct attacks.

Evaluation: the experiments shown in [47] indicate that level-k max–min acts better than recursive max–min (PushBack technique [46]). Moreover, the deployment cost of level-*k* max–min is much lower than for the PushBack technique. However, similar to the PushBack technique, the effectiveness of this technique also depends on the distribution form of legitimate users and attackers. Below, we discuss challenges regarding the *K*-MaxMin technique.

1. Traffic of cumulative legitimate users is severely punished.
2. Similar to the PushBack technique, traffic of innocent sources which share the same paths with the attack sources is punished.
3. *K*-MaxMin cannot protect a victim server against a meek attack.
4. *K*-MaxMin cannot protect a victim server against an incremental step attack[11] (see our experiments in [48]).
5. The AIMD technique is not an accurate technique; hence, the system may be stabilized too late or even in case of an incremental step attack may be never stabilized.
6. Another challenge is that the victim server will not survive during the distribution of the bottleneck resource. As discussed above in the *K*-MaxMin technique, the system is stabilized too late. During this time the bottleneck resource of the victim is saturated and thereby the victim server goes down. The authors of the paper assume that either the victim server remains active and healthy even when its capacity was overwhelmed or a remote machine issues control signals for rate-limiter routers on behalf of the victim server. The former is not realistic in practice because control signals are lost and may never reach the rate-limiter routers; the latter requires that a remote machine is permanently aware about the status of the victim server. In this case, the remote machine can be a target for the DDoS attack.
7. It is not guaranteed that all routers of level-*k* accept to cooperate. It is possible that some routers of level-*k* reject to rate-limit traffic for the victim due to lack of sufficient resources, buffers, etc. The *K*-MaxMin algorithm has no solution for such cases.
8. The authors of the paper do not consider the situation in which the traffic of a level-*k* router passes through another router at level-*k*. In this case a traffic flow may be rate-limited several times and as a result legitimate traffic can be severely punished.

5.3. Hop count packet filtering (HCF)

Jin et al. [14] proposed the hop counting packet filtering approach that is based on observing the TTL values (time-to-live) of packets. The victim observes the TTL value of a packet and guesses the initial TTL value that was placed in the packet at the sender. There are only a few such values[12] that operating systems use and they are fairly different, which facilitates correct guesses. The hop count is then the difference between the initial TTL and the observed one. During normal conditions, the victim creates a table of the most frequent legitimate clients and their relevant hop counts. Legitimate clients are those clients that established a TCP connection successfully. During the attack time, the spoofed packets are those packets that either are not in the table or their hop counts do not match with their source addresses according to the table. Next, the victim drops the spoofed packets and does not assign its resources (e.g., TCP buffers, etc.) to spoofed packets.

5.3.1. Analysis of the technique

Specification of the technique: detection location: victim server; detection mechanism: threshold-based; type of defense: victim-end; defense mechanism: filtering technique; control (decision) center: victim-end; type of attack that it can protect against: direct DDoS attacks.

Evaluation: HCF filtering is a simple, light-weight and cheap technique that can noticeably protect a victim server from direct DDoS attacks. The reason is that if an attacker wants to bypass the HCF filter, he must guess the correct TTL value to insert into a forged packet, so that the deduced hop count matches the expected one. In spite of the simplicity and light-weight attributes of this filtering technique, HCF suffers from the following challenges:

1. HCF cannot protect the victim server from a bandwidth flooding attack. In fact, flooding attacks based on overwhelming the link coming into the machine that is checking the TTL values cannot be tolerated. We note that most of DDoS attacks are bandwidth flooding attacks.
2. If the adversary could monitor traffic of α percent of the Internet (e.g., 20%), he has the chance to observe the IP address of α percent of users. In this case, the adversary can bypass the filter using the following technique. The adversary can obtain the hop-count information of these revealed legitimate users (α percent), using the IP SOURCEROUTE option and simple traceroute tools. After obtaining the hop-count information, the adversary adjusts the source IP address of attack packets with the source IP address of these users and then adjusts the TTL value appropriately based on the hop-count information. To obtain the hop-count information, the adversary should acquire the number of routers between the legitimate users and the victim server. To achieve this goal, the adversary first obtains the IP address of gateway routers of the legitimate users using simple Traceroute tools. Next, the adversary issues another traceroute command by setting the IP SOURCEROUTE option on; traceroute packets are forced to traverse the gateway routers of legitimate users and reach the victim server. Thereby, the

[10] Routers of level-*k* contain all routers that are either *k* hops away from a target or less than *k* hops away from the target but are directly connected to a host.

[11] In incremental step attacks, an attacker does not activate all zombie machines in one mass, but step-by-step and incrementally increases the number of attack machines according to a frequency speed.

[12] Most modern operating systems use only a few selected initial TTL values, 30, 32, 60, 64, 128, and 255.

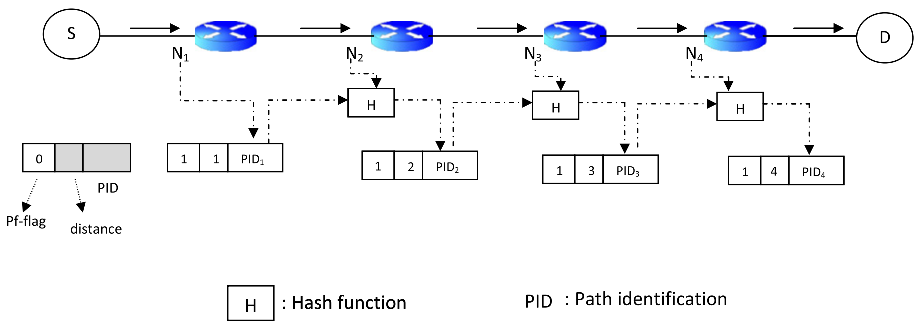

Fig. 9. ANTID: the proposed path fingerprint.

adversary can successfully obtain the number of routers between legitimate users and the victim server (i.e., hop count information).

3. If the adversary could passively monitor traffic nearby the victim server, then he can discover all pairs of (legitimate source IP, hop count); thereby, HCF totally fails.
4. Challenge with dynamic IP addresses: the technique is valid only for static IP addresses. A significant portion of users in the Internet have dynamic IP addresses [49]. Dial-up connections receive a new IP address each time a new connection is made. In broadband connections, an ISP changes the IP addresses of users from time to time (e.g., every few days) due to some policies. In fact, for broadband users, HCF works only when the IP address of a user has not been changed. Somebody may say, the technique keeps prefix of IP addresses in the table (e.g., the most 16 significant bits of the IP address). In this case, firstly, the percentage of both false positive and false negative is increased; secondly, for a particular prefix, the table may need to keep several hop counts because several IP addresses with the same prefix may have different hop counts. This increases the complexity of the technique.
Furthermore, the recent popularity of laptop computers indicate that number of mobile hosts increases and a laptop user may connect to the Internet in different places; thereby it gets a new IP address at every different place. Another challenge with dynamic IP addresses is that a portion of users may connect to the server through different internet cafés distributed across a city; it is not clear how the algorithm recognizes the IP addresses of such users.
5. A significant portion of users are behind NAT (Network Address Translator) devices [50,51]. All users located behind a NAT device (both malicious and legitimate) get the same IP address (i.e., the IP address of the NAT device). It is not clear how the algorithm differentiates legitimate users and zombie machines that are located behind the same NAT device.
6. During the attack time, legitimate users whose IP addresses are not in the table are refused to the connection. For example, all new users that are completely legitimate are rejected by the system.
7. Collateral damage even for those legitimate users whose IP addresses are in the filter's table can happen. The reason is that as a result of the DDoS attack against a victim server, it is highly possible that congestions happen in routers near the victim server or near the ISP that hosts the victim server. In this case, the congested router may route traffic of legitimate users through alternative paths. Consequently, the legitimate packets reach the filter with hop counts different from those inside the filter's table.

Peng et al. [52] proposes a history-based IP filtering (HIF) to tackle DDoS attacks. This technique is similar with HCF. In fact, HCF is the improved version of HIF. In the HIF technique, the edge router of the victim server keeps a history of all the legitimate IP addresses which have previously appeared in the network. HIF has all the challenges which we enumerated for HCF (even, in the attack model of item 2, the attacker does not need traceroute tools).

5.4. Anti-DDoS (ANTID)

The ANTID technique [13] is similar to the HCF technique, but instead of inspecting TTLs to discover spoofed IPs, it uses path fingerprint. In this scheme, each IP packet contains a unique path fingerprint representing the route an IP packet has traversed. An IP packet with incorrect path fingerprints is considered spoofed. The victim server holds a table of pairs of source IP address and fingerprint. To generate a path fingerprint, it is assumed that each router assigns to each of its network interfaces (i.e., a port interface) an n-bit random number and these random numbers are kept secret. A path fingerprint of an IP packet is composed of two fields: a d-bit distance field and an n-bit path identification field (PID), where the former represents the number of intermediate routers that the packet traverse and the latter denotes an identifier derived from the random numbers associate with the traversed network interfaces in the route. At each router the PID field of a packet is computed using the hash function of the current value of the PID field with the unique value associated with the network interface of the router that the packet arrives from it. Fig. 9 shows the architecture of ANTID.

5.4.1. Analysis of the technique

Specification of the technique: Detection location: victim server; detection mechanism: threshold-based; type of defense: victim-end; defense mechanism: filtering technique; control (decision) center: victim-end; type of attack that it can protect against: direct DDoS attacks.

Evaluation: ANTID has the same capabilities as HCF has. However, it suffers from several challenges.

1. It suffers from all challenges which we enumerated for the HCF technique except item 2.
2. Universal deployment: any router in the Internet should be updated to support the path fingerprint.
3. Due to the dynamic nature of Internet, it is possible that the percentage of false positives is high.
4. The cost of a technique should be proportional with the performance of the technique. The ANTID technique has the same abilities as HCF has. On the other hand, all the drawbacks (except one drawback) of higher HCF are also drawbacks of the ANTID technique (see item 1). But, the cost associated with the ANTID technique is much higher than the cost associated with the HCF technique because in the former, every router on the path should do a hashing oper-

ation and embeds the hash information in the packets. Our note is that the cost associated with this technique is not proportional with the abilities that the technique has.

5. This paper uses the IP identification field (16 bits) for embedding the fingerprints into IP packets. Other techniques such as [53,54] propose the use of IP identification field for IP traceback. So, if tomorrow, routers' manufactures have to decide on choosing one of the above ideas for implementing, they will choose the second one because it has many benefits compared to the first one; for instance, the latter one can determine the true location of spoofed packets while the former one only determines which packets have spoofed IPs (even not for all packets, but only for the most frequent packets). In summary, although the authors have done a valuable research, their idea can never be implemented. So, such ideas cannot practically help solving DDoS attacks.
6. Performance degradation: as any router in the Internet adds a fingerprint (a hash operation) to a packet, the overall speed of the Internet decreases.

5.5. Traceback techniques

The goal of traceback techniques is to find the true location of the attacker's machines (zombie machines). As a significant fraction of DDoS attacks use IP spoofing [53,8,55], analyzing the source IP address of packets cannot help to find the true location of the attacker's machines. To find the true location of attackers, various IP traceback techniques have been proposed of which we analyze the most famous ones. Traditional IP traceback techniques such as link testing (including both Input debugging [56] and Controlled flooding [57]) and packet logging [58] have been evaluated in [53]; thereby we omit to discuss them here.

Probabilistic Packet Marking (PPM): these techniques assume that all routers in the Internet mark packets probabilistically. Hence, a victim server can construct attack paths by collecting a significant number of packets. The main idea of PPM techniques is that each router probabilistically embeds its IP address into the packets traversing between source and destination. Based on the embedded path information in the packets, when the destination receives a sufficient number of such packets, the path can be reconstructed. However, there is no specific field in IPv4 packets to embed the IP address of routers in the packets. Savage et al. [53] proposed that the 64 bits (32 bits IP address of the router + 32 bits hash of the IP address of the router) should be divided into 8 parts. Every marked packet carries one part of the edge value. When the victim collects a sufficient number of malicious packets, the reconstruction program can combine the parts of the edge values with the help of the fragmented ID and distance. Hence, it can construct the attack path. The initial proposal by Savage et al. [53] did not have any provisions for authentication of those markings, but a later proposal added authentication and integrity checking [59]. Several techniques along these lines have been proposed, including single-bit techniques [60].

Algebraic based traceback technique: the algebraic traceback technique [54] also embeds partial tracing information into IP packets at the router level. This technique encodes path information as points on polynomials. Then it uses algebraic methods from coding theory to reconstruct these polynomials at the victim. As the authors of the paper claim, this scheme offers more flexibility in design and more powerful techniques that can be used to filter out attacker generated noise and separate multiple paths compared to scheme proposed by Savage et al. [53].

5.5.1. Analysis of the techniques

Specification of the technique: Detection location: victim server; detection mechanism: threshold-based and observing bogus packets; type of defense: distributed; defense mechanism: unspecified; control (decision) center: victim-end; type of attack that it can protect against: only direct attacks.

Evaluation: traceback techniques, themselves, do not stop DDoS attacks. If perfect, they only determine the edge networks containing all DDoS sources. It is not clear, after determining the source of traffic which countermeasure is taken. One option could be to dynamically deploy filters or rate limiters close to the identified sources. However, these techniques have several challenges as follows:

1. These techniques severely suffer from scalability. Ref. [61] shows that the best PPM scheme proposed can only effectively trace fewer than 100 attackers. In fact, traceback solutions often become unacceptably complex and expensive when there are large numbers of sources or when sources are well distributed.
2. Regarding PPM traceback techniques: as routers do not deterministically mark packets, it has the potential drawback that an attacker may impede traceback by sending packets with spoofed marking field values. In this case, several wrong paths can be generated.
3. Regarding the algebraic traceback technique: it burdens more overhead to the all routers compared to other traceback techniques.
4. These techniques need a universal deployment in the Internet to be practical because in these techniques, all routers in the Internet should mark packets. It is important to know that an Internet-wide deployment can be difficult to achieve due to political, financial, and administrative reasons, or different technology preferences.
5. All routers in the Internet are involved in deploying these techniques; hence, the cost is high.
6. Survivability of the victim server: all these techniques assume that the victim server can survive during the construction of the attack tree which takes a few minutes; thereby after constructing the attack trees, the victim server can issue control signals for routers close to the attack sources to rate-limit or filter the traffic. However, this is not true in practice because the bandwidth of the victim server is saturated and thereby the control packets are lost and do not reach the rate-limiter routers.
7. Today, due to the large number of zombie machines, attackers do not need to use spoofing techniques. They attack the victim directly by the real IP address of zombies. This means that traceback techniques which try to find true source of spoofed packets become useless.
8. As any router in the Internet probabilistically marks a packet, the overall speed of the Internet may slightly decrease.
9. These techniques cannot trace the true source of attackers in reflector DDoS attacks. In fact, in reflector DDoS attacks, these traceback techniques return the true source of reflector machines and NOT the true source of zombie machines.
10. These techniques can easily be paralyzed (i.e., the true source of malicious packets cannot be found), if an attacker attacks the target with packets larger than MTU (Maximum Transmission Unit) size. In this case, the IP identification field is used for fragmentation. Hence, routers cannot mark packets. For a such situation, Savage et al. [53] propose that routers send marking information through ICMP packets; but it has the following limitations:

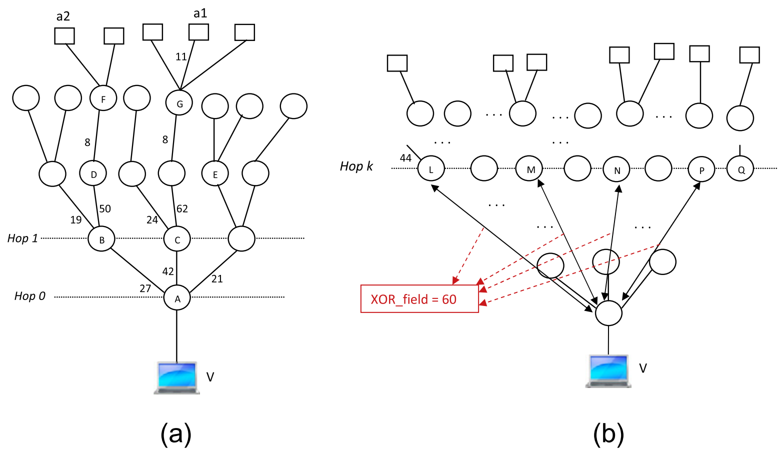

Fig. 10. PAD: the proposed path fingerprint.

– The victim may miss several of those ICMP packets due to congestions at routers caused by DDoS attacks.
– Some networks do not allow ICMP messages to travel across their border router; hence, the attack tree cannot be constructed accurately.

5.6. DefCOM

DefCOM [62] is a cooperative DDoS defense technique that combines source-end, victim-end, and core-network defenses. It is composed of three types of nodes: alert generators that detect an attack (these nodes are located at the victim-end); rate limiters that enforce simple rate limits on all traffic going to the attack's target (these nodes are located at the core-end); and classifiers that rate-limit traffic, separate legitimate packets from suspicious ones and mark each packet with its classification (these nodes are located at the source-end). DefCOM makes an overlay network between defense nodes. Data packets such as attack signature, rate-limit amount and other control packets are exchanged using this overlay network.

5.6.1. Analysis of the technique

Specification of the technique: Detection location: routers close to the victim server; detection mechanism: statistical methods; type of defense: distributed; defense mechanism: rate-limiting technique; control (decision) center: victim-end; type of attack that it can protect against: direct attacks.

Evaluation: DefCOM is an interaction of detection at the victim-end, rate limiting at the core-end, and blocking of suspicious/attack traffic at the source-end points. DefCOM claims to handle flooding attacks while inflicting little or no harm to legitimate traffic. However, the following drawbacks are seen in the DefCOM's design:

1. Distinguishing attack traffic from legitimate traffic is a challenging task at points close to the traffic sources; thereby classifier nodes cannot accurately distinguish legitimate traffic from attack traffic.
2. DefCOM uses D-WARD filters as the classifier nodes. As discussed above, a D-WARD filter is a very expensive filter and it, itself, suffers from several challenges. In fact all challenges of the D-WARD technique can be included here as well.
3. DefCOM uses packet stamping to tell rate-limiter routers downstream from the classifier nodes to not drop the traffic that has been policed by the classifier nodes. However, packet stamping either can be faked by attackers or it is prone to replay attacks. The next point is that if the architecture uses hashing function for the stamping (DefCOM does not mention how to stamp packets), the cost associated to check the stamp of packets would be too high for routers.
4. Traffic of several rate-limiter routers passes through each other which means that traffic of a user may be punished several times.
5. DefCOM does not specify the proper amount of rate-limits and in fact it is not clear how much an overlay node must rate-limit the traffic.
6. Handling damaged or subverted nodes in the overlay network may be hard, and DefCOM is likely to operate badly if they are not handled.

5.7. A divide and conquer strategy

Chen et al. [63] propose two novel techniques, namely AD (Attack Diagnosis) and PAD (Parallel Attack Diagnosis), to traceback DDoS attackers. These techniques integrate the concepts of Pushback and packet marking. Unlike traceback techniques of Section 5.5 that routers embed their IP addresses into IP packets, in the AD and PAD techniques, routers embed the identifier of port interfaces on which receive traffic toward the victim server into IP packets. In these techniques, any input port interface of a router has a unique identifier that is called PID. In these techniques, the marking field of IP packets (17 bits) divides into three parts: the hop-count field (5 bits), the PID field (6 bits) and the XOR field (6 bits). In these techniques, a router can be set in one of the following marking modes: ADMM (Active Deterministic Marking Mode) and PDMM (Passive Deterministic Marking Mode). When a router is set in the ADMM mode, it must (1) set the hop-count field to zero and (2) copy the PID of the port interface on which it receives traffic toward the victim server into PID field and as well as the XOR field. When a router is set in the PDMM mode, it must (1) increase the hop-count field by one and (2) compute the bit-by-bit XOR value of the PID field with the value inside the XOR field and then write the result back into the XOR field.

Let us explain the AD technique with an example. Consider the graph depicted in Fig. 10a, where round nodes show routers and square nodes show hosts. Suppose a_1 and a_2 are attackers. The AD technique can handle only one attacker at each time; hence, let us only discuss about a_1. First, the victim server (V) sends a request to router A to set all input port interfaces in the ADMM mode. Router A does the request and therefore V knows the attack comes from which port interface of router *A* (in our example, PID number 42). Next, *V* asks router *A* to set PID number 42 in the PDMM mode and also asks router *A* to send a request to the router which is connected to port number 42 (in our example, router *C*). The request asks router *C* to set port interfaces in the ADMM mode. When routers *A* and *C* do PDMM and ADMM actions, respectively, the victim server detects that traffic is coming from port interface number 62 of router *C*. This procedure is iterated until node *G* receives the request and then filters traffic of port number 11. After tracing attacker a_1, *V* continues the same procedure for attacker a_2 and so on.

In the AD technique at each round, only one attacker is traced. The PAD method is the extension version of the AD method that multiple attack paths are traced simultaneously. The victim server can distinguish interfaces of upstream routers several hops away from it through XOR field. For instance, in our example, the packet that comes from port 50 of router *B* has the value of 41 in the XOR field because PID 50 at hop 1 is grouped with PID 27 at hop 0 ($41 = 50 \oplus 27$). In the same manner, the packet that comes from port 24 of router *C* has the value of 50 in the XOR field because PID 24 at hop 1 is grouped with PID 42 at hop 0 ($50 = 24 \oplus 42$). This process is repeated for every hop until the request is received by the edge routers, *G* and *F*.

5.7.1. Analysis of the technique

Specification of the technique: Detection location: victim server; detection mechanism: threshold-based; type of defense: distributed; defense mechanism: filtering technique; control (decision) center: victim-end; type of attack that it can protect against: only direct attacks.

Evaluation: The major advantage of PAD technique over traceback techniques of Section 5.5 is that, it does not need universally deployment. The mechanism only executed when the DDoS attack happens against a sever; while in traceback techniques of Section 5.5 routers permanently should mark packets. The main difference between PAD and traceback techniques of Section 5.5 is that PAD is a down to up technique (traceback is started from the victim server and continues toward traffic sources); while methods of Section 5.5 are up to down techniques (i.e., traceback is started from routers close to traffic sources and continue toward the victim server). Although, AD and PAD seem attractive techniques, they have the following limitations and challenges:

1. In bandwidth DDoS attacks, when the bandwidth of the victim server was saturated, the victim server does not survive to incrementally issue requests for upstream routers. In other words, the victim server's requests are simply lost and do not reach upstream routers.
2. The techniques need cooperation of all routers that locate between the victim server and attackers. If some routers along the attack paths do not support the ADMM and PDMM procedures, the victim server will not be able to trace attackers. For instance in our example, if node C rejects to participate in this procedure, the victim cannot determine location of attacker a_1.
3. The cost of architecture is too much because all routers along the attack paths do ADMM and PDMM procedures and moreover a large communication is needed between upstream routers and the victim until the attack paths are determined.
4. If attackers evenly distributed in the internet which attackers and legitimate clients are few hops away from each other, the percentage of false positive is severely increased. For instance, experiments in [63] shows that when only 4000 attackers participate the percentage of false positive is about 70%.
5. The AD technique can trace only one attacker in a round; thereby the victim should run AD techniques in several consecutive rounds to trace all attackers which take very long time. For instance, when number of zombie machines are at the range of even thousands zombie machines, the AD technique is never practical.
6. The PAD technique fails to trace the location of attackers when the value of XOR field is identical for several paths that have the same hop-count value (note that the XOR field is only 6 bits). For instance in Fig. 10b, the victim does not know the packet that has value of 44 in PID field is coming through node *L*, or *M*, or *N* or *P*.

5.8. Client-puzzle protocols

Client-puzzle protocols [64,65] force clients to solve a mathematical puzzle before establishing a connection with the victim server. After solving the puzzle, the client would return the solution to the server, which the server would quickly verify, or reject and drop the connection. The puzzle is made simple and easily solvable but requires at least a minimal amount of computation on the client side. A client-puzzle is a computable cryptographic problem formulated using the time, a server secret, and additional client request information. For instant, the server asks clients to find x such that the rightmost n bits of SH-1(client IP$\|$client port$\|t\|n\|x$) are zero ($\|$ denotes concatenation and t denotes time). The parameter n is a tunable parameter that changes based on the attack volume. Clients need $O(2^n)$ time to find the answer. This increases the clients' workload. Consequently, a client cannot establish several TCP connections with the server as solving puzzles imposes a large overhead to its system.

5.8.1. Analysis of the technique

Specification of the technique: Detection location: victim server; detection mechanism: threshold-based; type of defense: victim-end; defense mechanism: filtering technique; control (decision) center: victim-end; type of attack that it can protect against: only SYN flood attacks.

Evaluation: this technique is a promising technique against SYN flood in which attackers try to exhaust the TCP/IP stack. The victim server does not allocate a TCP buffer to a client unless the client solves the puzzle correctly. The challenges and drawbacks for these techniques are as follows:

1. It cannot protect the victim server against a bandwidth saturation attack. An attacker can simply overwhelm the bandwidth by flooding the victim server by bogus packets.
2. To achieve high-level security, it incurs a significant computation overhead to the clients, which can be undesirable for certain applications, especially when mobile devices are involved. However, if the puzzle is simple with a low overhead to clients, then the attacker can easily solve the puzzle and gets resources of the victim server.
3. The client puzzle mechanism itself can become the target of a denial-of-service attack. In most systems either the puzzle creation or verification operation (or both) requires the server to perform a cryptographic hash computation. This opens the possibility that the puzzle verification mechanism itself will be the target of a denial of service attack, in which an attacker floods the server with bogus puzzle solutions that the server has to process.
4. Patience of the users: if a website employs client puzzles, then a user who wants to visit the site has to wait for his computer to solve a puzzle before accessing the site. Thus puzzles use up not only computer time, but also users' time, which is often much more valuable. Since many users have little patience for website delays, a site that imposes long puzzle delays can drive away legitimate users. This means

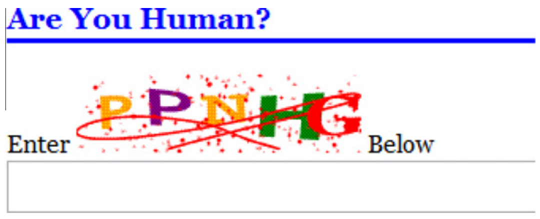

Fig. 11. A sample of CAPTCHA test.

that a puzzle that costs the attacker some fixed price to solve will cost legitimate clients much more, due to the higher cost of human time for real clients.

5. Client-puzzle solution techniques need client-side software. A software program must be installed on the client-side to solve the puzzles. However, this problem cannot be a serious challenge because such a program can be built into a browser, made available as a plug-in, or distributed by any one of a variety of other means.

5.9. CAPTCHA puzzles

A CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) puzzle [66,67] is a type of challenge-response test used in computing as an attempt to ensure that the response is generated by a human and not by a machine. Many websites use CAPTCHA tests during registration and initial login. CAPTCHAs can be deployed to protect systems vulnerable to e-mail spam, such as the webmail services of Gmail, Hotmail, etc. CAPTCHAs are also used to minimize automated posting to blogs, forums, etc. Today, several DDoS victims use CAPTCHA technology against application-layer DDoS attacks such as HTTP flood attacks. Fig. 11 shows an example of a CAPTCHA puzzle.

5.9.1. Analysis of the technique

Specification of the technique: Detection location: victim server; detection mechanism: threshold-based; type of defense: victim-end; defense mechanism: filtering technique; control (decision) center: victim-end; type of attack that it can protect against: application-layer DDoS attacks such as HTTP flood and SYN flood attacks.

Evaluation: this technique is a promising technique against application-layer DDoS attacks where an attacker floods the victim server not by bogus packets, but by legitimate requests. However, in spite of its effectiveness against attacks such as HTTP flood attacks, it has serious challenges as listed below.

1. This technique cannot protect a victim server against a bandwidth flood attacks (e.g., TCP flood, UDP flood) and reflector attacks.
2. As non-human machines cannot solve these tests, if a server has legitimate non-human users, they cannot access the server. Thereby, the server misses its legitimate non-human users.
3. Patience of the users: if a website employs CAPTCHAs, then a user who wants to visit the site has to solve the test and wait for the response before accessing the site. Several reports [68,69] show that these tests annoy the users and they are not user-friendly. Since many users have little patience to solve a CAPTCHA test and wait for the response, a site that uses CAPTCHAs may drive away legitimate users.
4. Breaking techniques: today, several image recognition techniques have been proposed to break CAPTCHAs [70]. In these techniques, first, background noise and anomalies are removed from a CAPTCHA image. Next, the image is segmented and passed to character and shape recognition algorithms. In order to defend against image processing CAPTCHA attacks, modern CAPTCHAs include background noise, juxtaposition [71], and animation [72] that is harder to recognize by computer software. Unfortunately, overuse of these countermeasures can sometimes make it too difficult for a human to read the image and therefore limit their practical usage. On the other hand, it annoys the people much more.
5. Insecure implementation: some CAPTCHA protection systems can be bypassed without using OCR (Optical character recognition) simply by re-using the session ID of a known CAPTCHA image. A correctly designed CAPTCHA does not allow multiple solution attempts at one CAPTCHA. This prevents the reuse of a correct CAPTCHA solution or making a second guess after an incorrect OCR attempt. Finally, some implementations use only a small fixed pool of CAPTCHA images. Eventually, when enough CAPTCHA image solutions have been collected by an attacker over a period of time, the CAPTCHA can be broken by simply looking up solutions in a table.
6. Labor attack: today even some reports [72,73] indicate that there is free or cheap 3rd party human labor to break CAPTCHAs. They highlighted the economy of breaking CAPTCHA in third world countries with cheap labor.

5.10. Sharing beliefs

Peng et al. [74] propose a distributed approach for detecting reflector attacks using sharing beliefs among potential reflectors. In this approach, every potential reflector monitors the incoming packets and broadcasts a warning message to other potential reflectors if any abnormal traffic is observed. The warning message contains a description of the abnormal traffic it has observed. A detection decision can be made based on the information from multiple potential reflectors. Next, the potential reflectors ignore incoming request packets that have a source address equals to the victim's IP address.

5.10.1. Analysis of the technique

Specification of the technique: detection location: victim server; detection mechanism: threshold-based; type of defense: distributed; defense mechanism: filtering technique; control (decision) center: distributed; type of attack that it can protect against: only reflector attacks.

Evaluation: this technique cannot be deployed in practice due to the following challenges:

1. There is no way by which a particular reflector knows the group of reflectors participating in ongoing attacks, such that it can share its beliefs with them.
2. Even if attack detection is possible, there is no way a reflector can distinguish legitimate packets from malicious packets.
3. Attackers can abuse the scheme by sharing their own beliefs with many other innocent reflectors to drop legitimate requests received by them.

5.11. Reflector attack detection (RAD)

Kline et al. [75] propose two techniques based on message authentication code (MAC) to resist against reflector DDoS attacks. Local RAD (L-RAD) uses MACs to mark outgoing requests at their sources; therefore, the target of a reflector attack is able to differentiate between replies to legitimate requests and spoofed requests. In L-RAD, MACs can be validated either at the target machine or at the gateway router(s) of the target's network. In L-RAD, MACs are generated using hashing over source and destination IP addresses, source and destination port numbers and a 32 bits counter value (the counter counts every two seconds and it is used against replay attacks). L-RAD uses SHA-1 for generating the MACs and inserts the rightmost 32 bits of MACs in the SYN place. Core RAD (C-RAD) which is deployed at the autonomous system (AS) level, handles larger attacks that overwhelm L-RAD. The source AS marks each packet it sends using MACs and then core ASs verify the MAC of each packet and drops the packets that carry incorrect MAC. In C-RAD, MACs are generated by hashing over the packet's payload, source and destination IP addresses, port numbers and a secret key associated with the source AS. SHA-1 is used for generating MACs and the rightmost 16 bits of MACs is inserted in the IP identification field of packets.

5.11.1. Analysis of the technique

Specification of the technique: Detection location: victim server; detection mechanism: threshold-based; type of defense: L-RAD: victim-end and C-RAD: core-end; defense mechanism: filtering technique; control (decision) center: victim-end; type of attack that it can protect against: only reflector attacks.

Evaluation: the goal of RAD is to allow individual networks to have more control over the packets that they send due to own protection. Unlike ingress filtering that needs a large universal deployment (each one should do its part), C-RAD only needs the cooperation of core ASs. By deploying both L-RAD and C-RAD a defense system of two layers is constructed and reflector attacks are effectively tackled. However, RAD has the following limitations and challenges:

1. Both L-RAD and C-RAD impose large overhead (cryptographic overhead) to border routers of ASs because border routers of ASs should verify the MAC of each packet destined to the victim server. Previous benchmarks [76] show that the algorithm of HMAC (SHA-1) needs 11.9 cycles for operation on one byte. It means that an Intel Core two 2.4 GHz processor can handle 192 MB/s hash operation. The L-RAD technique does hash operation over 45 bytes for each packet (the C-RAD does hash operation over more bytes because it includes also the payload of packets). This means that if a border router totally consumes its processor time to verify the MACs, it can verify the MACs of 4.4 million packets per second. Therefore, in L-RAD, when number of DDoS packets exceeds this threshold rate, the border router may not be able to verify all MACs and simply fails to protect the victim server. The clue point for the adversary is that he should attack using packets with small length. We note that in reflector attacks such as SYN-ACK (RST), the size of packets is small and it is 40 bytes. In other reflector attacks, the attacker can attack with small packets. In this case, a relative large attack, e.g., 5 Gbit/s, generates more than 16 million packets per second. As can be seen border routers in L-RAD cannot handle this number of MACs and also border routers in C-RAD encounter significant performance degradation. So, verifying the MACs is the most serious bottleneck for this technique.[13]
2. Regarding L-RAD: like previous victim-end defense techniques, no matter how well L-RAD, it will eventually be overwhelmed with bandwidth flooding attack.
3. Regarding C-RAD: C-RAD partially needs a universal deployment. In fact all core routers should cooperate. If some ASs do not cooperate (e.g., due to large overhead of verifying MACs), then the attackers can use the reflector machines which their traffic toward the victim server passes through those ASs to attack the victim server.
4. Regarding C-RAD: in C-RAD, every AS needs to know the unique secret key associated with any other ASs; these keys regularly should be updated. Therefore, the problem of key distribution is another challenge for this technique.
5. Regarding C-RAD: in C-RAD, the source AS shares one secret key with all core ASs; thereby if and only if one border router of one of ASs is compromised, the C-RAD simply fails because the key is revealed. The source AS simply cannot share a unique and separate key with each AS because the source AS does not know the returned reply packets to the target pass through which core ASs to use the suitable key for generating the MAC of each packet.
6. Regarding C-RAD: in C-RAD, border routers of the AS that hosts the victim server generate MAC for any request that has been originated from that AS. Thereby, the technique simply cannot protect the victim server from the spoofed requests originated from the zombie machines that locate inside the source AS (i.e., the same AS that the victim server locates). If the AS that hosts the victim server is relatively large, then it could have relative large number of users. In this case, the attacker might have relatively large number of zombie machines in this AS. Consequently, simply C-RAD fails to protect the victim server.
7. L-RAD is not vulnerable against replay attacks because the output of a counter is used for generating the hash (the counter counts every two seconds and it is hard for an attacker to generate replay attacks during two seconds). But, C-RAD is vulnerable against replay attack as it does not and cannot use counters (the reason why a counter cannot be used is that the source AS should share and synchronize a unique counter with all core ASs that looks unrealistic).

[13] One possible solution to handle this serious challenge is that border routers of ASs are equipped with cryptographic hardware tools in order to verify the MACs because these tools can handle more MACs/s (this is the proposed solution of the authors of the RAD technique). But the big question is whether ASs' administrators are convinced to spend money in order to equip the routers with these hardware tools only due to reflector attacks against a victim that locates in another domain. On the other hand equipping routers with these hardware tools has no benefit itself for the ASs.

5.12. SYN cookies

The SYN cookie technique [77] is the most promising technique against SYN flood attacks. The idea of SYN cookies is that the server instead of storing the initial sequence number (ISN) of SYN packets in the backlog queue, stores authentication information in its sequence number of the SYN/ACK packets. When the server replies with a SYN/ACK packet, it uses a specially designed formula to calculate its sequence number (used as an authentication cookie) and stores it into the SYN/ACK packet. The cookie value is calculated using the hash of the source address, the source port, the destination address, the destination port, the maximum segment size value (MSS) that the server would have stored in the SYN queue entry, a counter that is changed approximately every minute and a secret value that changes at every server boot. Normally MD5 is used as hash function. When the server receives a packet with the ACK flag set (the last stage of the three-way handshake process) then it verifies the cookie. When its value is correct, it creates the connection, even though there is no corresponding entry in the SYN queue.

5.12.1. Analysis of the technique

Specification of the technique: Detection location: victim server; detection mechanism: threshold-based; type of defense: victim-end; defense mechanism: filtering technique; control (decision) center: victim-end; type of attack that it can protect against: only SYN flood attacks.

Evaluation: this technique is the most promising technique against SYN flood attacks. Moreover, the use of SYN Cookies does not break any protocol specifications, and therefore should be compatible with all TCP implementations. The drawbacks for this technique are:

1. The countermeasure is not robust against a SYN flood that overwhelms the bandwidth.
2. The server must reject all TCP options such as large window scaling, selective ACKs, RFC 1323 because the server discards the SYN queue entry where that information would otherwise be stored.
3. The server cannot resend the lost SYN/ACK packet because the information is no more available.
4. A connection may freeze when the final ACK of the three-way handshake is lost and the client first awaits data from the server (i.e. the client has completed the three-way handshake, the server did not receive the client's ACK and thus has not actually opened the connection).
5. In case of large SYN flood attack, it is possible that the computational resources of the server are exhausted as the server needs to calculate a hash value for each SYN packet.

6. Discussion

We can summarize the main benefits of proactive defense techniques as follows. (1) If these techniques are universally deployed, i.e., every ISP in the Internet accepts to participate, then a large percentage of today's DDoS attacks can be successfully defeated, but not all of them. (2) With proactive techniques, victims do not worry about a significant percentage of today's DDoS attacks and they do not need to think about defense mechanisms separately. In fact, proactive techniques make the Internet safer for all possible victims from a DDoS attack point of view. (3) Most proactive techniques are inclusive techniques that protect victim servers against both direct and reflector attacks; while most reactive techniques either control direct attacks or reflector attacks. (4) In proactive techniques, response to DDoS attacks is much faster than reactive techniques because DDoS traffic is dropped before it reaches the victim server.

We can summarize the main challenges of proactive techniques as follows. (1) These techniques impose large overhead to routers (such as processing overhead, memory overhead, etc.) because routers permanently need to process packets (any packet for any destination). The overhead might affect the normal routing functionality of a router. (2) Proactive techniques can tackle DDoS attacks if and only if they are implemented universally. (3) They have no benefits for the ISPs themselves that implement the techniques, while even they may decrease the overall speed of the Internet. The question is why ISPs should implement something when there is no benefits for them and why Internet users should suffer performance reduction due to a DoS attack against a possible victim server (even in a worse scenario, when there is no DoS attack against a server). (4) Attack traffic cannot be easily distinguished from legitimate traffic. Hence, the percentage of false positive might be high and legitimate traffic might be mistakenly filtered (collateral damage); or vice versa, the percentage of false negatives might be high and malicious traffic is not filtered, reaches the target and damages the target. The only criterion on which routers distinguish legitimate packets from malicious packets is the header part of packets [63]. Today, expert attackers adjust the parameters of the header part of the packets in such a way that the malicious packets look like the legitimate packets; thereby routers either are unable to detect malicious traffic or can hardly detect them.

Due to the above challenges the ISPs' administrators have not been convinced to employ proactive techniques and it seems that in the future they also will not be motivated to employ such techniques. In fact, proactive techniques are only effective if most ISPs employ them.

In contrast to proactive techniques, in reactive techniques, any victim itself must design and implement a defense solution. The main benefits of reactive techniques are as follows. (1) Reactive techniques impose lower overhead to routers (both processing and memory-based overheads) compared to proactive techniques because these defense mechanisms are only activated in parts of the Internet and only when a real attack happens. (2) They do not need universal deployment.

Reactive techniques have some difficulties and challenges as follows. (1) Unlike proactive techniques, most reactive techniques are not inclusive techniques; they either control direct attacks or reflector attacks. In some cases such as victim-end defense techniques, they can only handle a few types of DDoS attacks such as SYN floods and HTTP floods. (2) As these defense techniques are activated after occurrence of the attack, the attack may cause some damages before it is eliminated. In this case, employing detection techniques may help. (3) Handling bandwidth flooding attacks needs cooperation of upstream routers and there are five main challenges. (i) It needs a cooperation protocol between the owners of the routers which are ISPs. The ISPs should be motivated to cooperate. Perhaps, one of the main reasons that an ISP is not convinced to cooperate is that there is a lack of economic incentives for ISPs to invest money in security and add new features to their routers to mainly protect other networks. (ii) How should upstream routers trust a signal coming from a downstream router which asks for rate-limit/filter traffic for a destination? It is worth noting that without a reliable authentication, an attacker can simply ask upstream routers to do rate-limit/filter which is another type of DDoS attacks. Hence, a reliable mechanism for managing trust and authentication in a distributed environment is required. (iii) If upstream routers should rate-limit traffic or filter traffic for a victim beyond their own ISP's domain, then these techniques should be light-weight enough such that they do not affect the performance of the routers (i.e., the users locating on domains of those ISPs do not feel low performance). (iv) How to manage any risks of liability if an upstream router makes an incorrect decision? (v) The final issue is how to tackle scalability when several thousands upstream routers should be involved in the rate-limit/filter action.

Due to the challenges which both proactive and reactive techniques have, today, most of servers use survival techniques and the rest of them are defeated by DDoS attackers. Today, social web-sites such as Facebook, Twitter and Yahoo messenger and software tools such as VoIPs have attracted a huge number of Internet users with little knowledge about computer security. These unprofessional users stay on-line for a long time on the Internet. Today, also improvements of technology have provided high speed (high bandwidth) Internet for these users. All of the above evidences show that the power of attackers to perform a DDoS attack is rapidly expanding. This indicates that the security community and ISPs must overcome challenges to provide practical and effective techniques against DDoS attacks in spite of valuable research that have been done in this area till now.

7. Conclusions and suggestions

This paper presents a comprehensive study and analysis of many well-known DDoS countermeasures. For each countermeasure, the paper provides the specification of the technique, enumerates strengths and challenges of the technique, and if it is possible presents a countermeasure against the technique from attacker's point of view. The paper also provides the overall characterization of DDoS defense techniques, the role of detection techniques and briefly discusses some detection techniques. Our analysis shows that both proactive and reactive techniques suffer from some serious challenges such that in spite of considerable research efforts into defenses against DDoS attacks, there has been only limited progress in solving the DDoS problem in the real world. Due to serious challenges of both proactive and reactive techniques, today, most of servers use survival techniques against DDoS attacks and the rest of them are beaten by DDoS attackers.

This paper has the following benefits. (1) It assists servers which are frequently targeted by attackers to select their defense mechanism according to the analysis provided in this paper and the abilities that they have to implement a DDoS defense mechanism. (2) It helps the researchers to improve the state of the art defense mechanisms against DDoS attacks and to design vulnerability patches for current techniques. (3) It helps novice researchers in the DDoS field to quickly find their research route. (4) It surveys the main efforts of the last decade to control and mitigate DDoS attacks and encourages the security community and internet service providers to take more fundamental steps to tackle DDoS attacks in practice.

7.1. Suggestions

Below, we propose some suggestions for future research on this area:

– We believe DDoS attackers are going to switch from network-layer DDoS attacks to application-layer DDoS attacks (the new generation of DDoS attacks). The major goals for an application-layer DDoS attack are saturating the bandwidth through outbound traffic (e.g., downloading files), exhausting TCP/IP stack, memory, CPU cycles and I/O devices. In application-layer DDoS attacks, the real IP addresses of zombie machines are disclosed but the adversary does not worry about them as it has control over many zombie machines (see above). Most current techniques cannot handle application-layer DDoS attacks. As discussed above, CAPTCHA puzzles are the most promising techniques against these attacks. But as mentioned above, these techniques suffer from several serious challenges. Thereby, we recommend researchers to do more research in this area and bring new ideas, the techniques rather than CAPTCHA puzzles, to handle this type of attacks though a few techniques [15,78] but not so secure have already been proposed.

– Most research on defending against DDoS attacks has been done for direct DDoS attacks, while reflector DDoS attacks have been discussed less. The reason is that the power of direct DDoS attacks is effectively more than reflector DDoS attacks. However, 14 years (first anti-DoS paper dates back to 1998) research on defending against direct DDoS attacks has caused this research area to approach near saturation. While, research on reflector DDoS attacks is still young and has more opportunities. For instance, one of the active areas for future research in reflector attacks is to trace the true location of zombie machines. Current traceback techniques are unable to trace the correct location of zombie machines in reflector attacks.

–There is lack of research on cooperation protocols between ISPs. It has been shown that without cooperation of upstream routers, many DDoS attack types cannot be solved. Upstream routers can belong to different ISPs. The question is according to which criteria ISPs cooperate to block traffic for servers that locate in other ISPs. Why should an ISP cooperate? Why should an ISP invest to add new features to its routers mainly to protect other networks? What are the benefits for the ISP? Are benefits mutual? How much should ISPs cooperate? If economic incentives encourage an ISP to cooperate, who is responsible to pay them and how?

–For designing and inventing new techniques against DDoS attacks, researchers should have in mind the following clues:

- The technique should be deployable.
- It is most promising that the technique covers most DDoS attack types.
- The cost of a technique should be proportional to its abilities.
- Its overhead to routers and side-effects such as negative impact on the Internet's speed should be minimized (especially in proactive techniques).
- The designers should have in mind that if a task is allocated to routers, the task should be as light as possible and needs minimum level of resources. Otherwise, it is very probable that the router rejects the task. For instance a simple comparison is a light task and it is very likely that a router accepts it. Try to avoid asking routers to do cryptographic operations such as verifying signatures or hashing operations.
- False positives and false negatives should be as low as possible.

Acknowledgments

We thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper.

References

[1] L. Garber, Denial-of-service attacks rip the internet, IEEE Computer 33 (4) (2000) 12–17.
[2] G. Carl, G. Kesidis, Denial-of-service attack detection techniques, Journal IEEE Internet Computing 10 (1) (2006) 82–89.
[3] A. Hussain, J. Heidemann, C. Papadopoulos, A framework for classifying denial of service attacks, in: Proceedings of ACM SIGCOMM, Karlsruhe, Germany, 2003, pp. 99–110.
[4] Denial of service attack via ping. <http://www.cert.org/advisories/CA-1996-26.html> (accessed 08.11).
[5] CERT Coordination Center, Smurf IP Denial-of-Service Attacks, March 2000. <http://www.cert.org/advisories/CA-1998-01.html>.
[6] SARA (TM). Possible DoS (fraggle) Problem, 2000. <http://www-arc.com/sara/cve/Possible_DoS_problem.html>.
[7] H. Beitollahi, G. Deconinck, Denial of Service attacks: a tutorial, Tech. Rep. 08-2011-0115, Electrical Engineering Department (ESAT), University of Leuven, August 2011. <http://www.esat.kuleuven.be/electa/publications/fulltexts/pub_2261.pdf>.
[8] J. Mirkovic, J. Martin, P. Reiher, A taxonomy of DDoS attacks and DDoS defense mechanisms, Computer Communication Review 34 (2) (2004) 39–53.
[9] J. Mosla, Mitigating denial of service attacks: a tutorial, Journal of Computer Security 13 (6) (2005) 807–837.
[10] L. Chen, T.A. Longstaff, K.M. Carley, Characterization of defense mechanisms against distributed denial of service attacks, Elsevier Journal of Computer & Security 23 (8) (2004) 665–678.
[11] J. Mirkovic, P. Reiher, D-WARD: a source-end defense against flooding denial-of-service attacks, IEEE Transaction on Dependable Secure Computing 2 (3) (2005) 216–232.
[12] G. Zhang, M. Parashar, Cooperative defence against DDoS attacks, Journal of Research and Practice in Information Technology 38 (1) (2006) 69–84.
[13] F. Lee, S. Shieh, Defending against spoofed DDoS attacks with path fingerprint, Elsevier Journal of Computers & Security 24 (7) (2005) 571–586.
[14] H. Wang, C. Jin, K.G. Shin, Defense against spoofed IP traffic using hop-count filtering, IEEE/ACM Transactions on Networking 15 (1) (2007) 40–53.
[15] Y. Xie, S.-Z. Yu, A large-scale hidden semi-Markov model for anomaly detection on user browsing behaviors, IEEE/ACM Transactions on Networking 17 (1) (2009) 54–65.
[16] K. Li, W. Zhou, J. Hai, J. Liu, Distinguishing DDoS attacks from flash crowds using probability metrics, in: Proceedings of the Third International Conference on Network and System Security, Gold Coast, Australia, 2009, pp. 9–17.
[17] T. Yatagai, T. Isohara, I. Sasase, Detection of HTTP-GET flood attack based on analysis of page access behavior, in: Proceedings of IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, Victoria, Canada, 2007, pp. 232–235.
[18] Y. Chen, K. Hwang, W.-S. Ku, Collaborative detection of DDoS attacks over multiple network domains, IEEE Transactions on Parallel and Distributed Systems 18 (12) (2007) 1649–1662.
[19] H. Wang, D. Zhang, K.G. Shin, Change-point monitoring for the detection of DoS attacks, IEEE Transactions on Dependable and Secure Computing 1 (4) (2004) 193–208.
[20] G. Carl, R.R. Brooks, S. Rai, Wavelet based denial-of-service detection, Elsevier Journal of Computers & Security 25 (8) (2006) 600–615.
[21] L.F. Lu, M.L. Huang, M.A. Orgun, J.W. Zhang, An improved wavelet analysis method for detecting DDoS attacks, in: Proceedings of Fourth International Conference on Network and System Security, Melbourne, Australia, 2010, pp. 318–322.

[22] J. Li, Y. Liu, L. Gu, DDoS attack detection based on neural network, in: Proceedings of 2nd International Symposium on Aware Computing, IEEE explore, Tainan, Taiwan, 2010, pp. 196–199.
[23] B.B. Gupta, R.C. Joshi, M. Misra, ANN based scheme to predict number of zombies in a DDoS attack, International Journal of Network Security 14 (1) (2012) 35–46.
[24] A. Mitrokotsa, C. Douligeris, Detecting denial of service attacks using emergent self-organizing maps, in: Proceedings of the Fifth IEEE International Symposium on Signal Processing and Information Technology, Athens, Greece, 2005, pp. 375–380.
[25] D. Gavrilis, E. Dermatas, Real-time detection of distributed denial-of-service attacks using RBF networks and statistical features, Elsevier Journal of Computer Networks 48 (2) (2005) 235–245.
[26] S. Khatta, R. Melhem, D. Mosse, T. Znati, Honeypot Back-propagation for mitigating spoofing distributed denial-of-service attacks, in: Proceedings of 20th International Symposium on Parallel and Distributed Processing, Rhodes, Island, 2006, pp. 1–8.
[27] L. Feinstein, D. Schnackenberg, R. Balupari, D. Kindred, Statistical approaches to DDoS attack detection and response, in: Proceedings of DARPA Information Survivability Conference and Exposition, Washington, DC, USA, 2003, pp. 303–314.
[28] A.L. Toledo, X. Wang, Robust detection of MAC layer denial-of-service attacks in CSMA/CA wireless networks, IEEE Transactions on Information Forensics and Security 3 (3) (2008) 347–358.
[29] R. Zhong, G. Yue, DDoS detection system based on data mining, in: Proceedings of the Second International Symposium on Networking and Network Security, Jinggangshan, China, 2010, pp. 62–65.
[30] H. Baig, F. Kamran, Detection of low intensity DoS attacks using fuzzy based intrusion detection system, in: Proceedings of International Conference on Electrical and Computer Engineering, Dhaka, Bangladesh, 2006, pp. 591–594.
[31] H. Jiang, C. Dovrolis, Passive estimation of TCP round-trip times, ACM SIGCOMM Computer Communication Review 32 (3) (2002) 75–88.
[32] J. Aikat, J. Kaur, F. Smith, K. Jeffay, Vaiability in TCP Round-trip Times, in: Proceedings of Internet Measurement Conference, Miami Beach, FL, USA, 2003, pp. 279–284.
[33] P. Ferguson, D. Senie, Network ingress filtering: defending denial of service attacks which employ IP source address spoofing, internet RFCs RFC 2827, 2000, pp. 1–10. <http://www.ietf.org/rfc/rfc2827.txt>.
[34] K. Park, H. Lee, On the effectiveness of route-based packet filtering for distributed DoS attack prevention in power-law internets, in: Proceedings of ACM SIGCOMM, San Diego, California, USA., 2001, pp. 15–26.
[35] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, S. Surana, Internet indirection infrastructure, IEEE/ACM Transactions on Networking 12 (2) (2004) 205–2018.
[36] K. Lakshminarayanan, D. Adkins, A. Perrig, I. Stoica, Taming IP packet flooding attacks, ACM SIGCOMM Computer Communication Review 34 (1) (2004) 45–50.
[37] J. Raymon, Traffic analysis: protocols, attacks, design issues, and open problems, in: Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability, LNCS, vol. 2009, Springer-Verlag, 2000, pp. 10–29.
[38] S.J. Murdoch, G. Danezis, Low-cost traffic analysis of Tor, in: Proceedings of the 2005 IEEE Symposium on Security and Privacy, Oakland, California, USA, 2005, pp. 183–195.
[39] T.J. Murdoch, P. Zieliński, Sampled traffic analysis by internet-exchange-level adversaries, in: Proceedings of the Seventh Workshop on Privacy Enhancing Technologies (PET'07), Ottawa, Canada, 2007, pp. 167–183.
[40] A.D. Keromytis, V. Misra, D. Rubenstein, SOS: an architecture for mitigating DDoS attacks, Journal on Selected Areas in Communications 22 (1) (2004) 176–188.
[41] M. Rennhard, MorphMix: a peer-to-peer-based system for anonymous internet access, Ph.D. thesis, Swiss Federal Institute of Technology, Zurich, Swiss, 2004.
[42] A.T. Mizrak, Y.C. Cheng, K. Marzullo, S. Savage, Detecting and isolating malicious routers, IEEE Transactions on Dependable and Secure Computing 3 (3) (2006) 230–244.
[43] C. Heffner, Remote attacks against SOHO routers, February 2010. <http://www.defcon.org/images/defcon-18/dc-18-presentations/Heffner/DEFCON-18-Heffner-Routers-WP.pdf>.
[44] H. Beitollahi, G. Deconinck, Making overlay networks more robust to massive failures, in: Proceedings of 15th IEEE Pacific Rim International Symposium on Dependable Computing (PRDC'09), Shanghai, China, 2009, pp. 214–221.
[45] E. Messmer, The Botnet World is Booming, Network World, April 2011. <http://www.networkworld.com/news/2009/070909-botnets-increasing.html>.
[46] R. Mahajan, S.M. Bellovin, S. Floyd, Controlling high bandwidth aggregates in the network, ACM SIGCOMM Computer Communication Review 32 (3) (2002) 62–73.
[47] D.Y. Yau, J.C.S. Lui, F. Liang, Y. Yam, Defending against distributed denial-of-service attacks with max–min fair server-centric router throttles, IEEE/ACM Transactions on Networking 13 (1) (2005) 29–42.
[48] H. Beitollahi, G. Deconinck, A cooperative mechanism to defense against distributed denial of service attacks, in: 10th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (IEEE TrustCom-11), Changsha, China, 2011, pp. 11–20.
[49] Y. Xie, F. Yu, K. Achan, E. Gillum, M. Goldszmidt, T. Wobber, How dynamic are IP addresses, ACM SIGCOMM Computer Communication Review 37 (4) (2007) 301–3012.
[50] A. Tekeoglu, N. Altiparmak, A. Tosun, Approximating the number of active nodes behind a NAT device, in: Proceedings of 20th International Conference on Computer Communications and Networks (ICCCN'11), Maui, Hawaii, USA, 2011, pp. 1–7.
[51] G. Armitage, Estimating the penetration of NAT/NAPT in the online game community, QuakeIII Arena, 2001. <http://gja.space4me.com/things/nat-quake3.html>.
[52] T. Peng, C. Leckie, K. Ramamohanarao, Protection from distributed denial of service attack using history-based IP filtering, in: Proceedings of IEEE International Conference on Communications, Anchorage, Alaska, 2003, pp. 482–486.
[53] S. Savage, D. Wetherall, A.R. Karlin, T.E. Anderson, Network support for IP traceback, IEEE/ACM Transactions on Networking 9 (3) (2001) 226–237.
[54] D. Dean, M. Franklin, A. Stubblefield, An algebraic approach to IP traceback, ACM Transactions on Information and System Security 5 (2) (2002) 3–12.
[55] D. Moore, C. Shannon, D. Brown, G. Voelker, S. Savage, Inferring internet denial-of-service activity, ACM Transactions on Computer Systems 42 (2) (2006) 115–139.
[56] R. Stone, Centertrack: an IP overlay network for tracking DoS floods, in: Proceedings of the 9th conference on USENIX Security Symposium, Berkeley, USA, 2000, pp. 199–212.
[57] H. Burch, B. Cheswick, Tracing anonymous packets to their approximate source, in: Proceedings of the 14th USENIX conference on System administration, New Orleans, LA, USA, 2000, pp. 319–328.
[58] G. Sager, Security Fun with OCxmon and cflowd, 1998. <http://www.caida.org/funding/ngi1998/content/security/1198/>.
[59] D.X. Song, A. Perrig, Advanced and authenticated marking schemes for IP traceback, in: Proceedings of Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, Anchorage, AK, USA, 2001, pp. 878–886.
[60] M. Adler, Tradeoffs in probabilistic packet marking for IP traceback, Journal of the ACM 52 (2) (2005) 217–244.
[61] M. Sung, J. Xu, J. Li, L. Li, Large-scale IP traceback in high-speed internet: practical techniques and information-theoretic foundation, IEEE/ACM Transactions on Networking 16 (6) (2008) 1253–1266.
[62] J. Mirkovic, M. Robinson, P. Reiher, Forming alliance for DDoS defenses, in: Proceedings of the New Security Paradigms Workshop, Ascona, Switzerland, 2003, pp. 11–18.
[63] R. Chen, J. Park, A divide-and-conquer strategy for thwarting distributed denial-of-service attacks, IEEE Transactions on Parallel and Distributed Systems 18 (5) (2007) 577–588.
[64] M. Fallah, A puzzle-based defense strategy against flooding attacks using game theory, IEEE Transactions on Dependable and Secure Computing 7 (1) (2010) 5–19.
[65] A. Michalas, N. Komninos, N. Prasad, V. Oleshchuk, New client puzzle approach for dos resistance in ad hoc networks, in: Proceedings of IEEE International Conference on Information Theory and Information Security, 2010, pp. 568 – 573.
[66] W.G. Morein, A. Stavrou, D.L. Cook, A.D. Keromytis, V. Misra, D. Rubensteiny, Using graphic turing tests to counter automated DDoS attacks against web servers, in: Proceedings of the 10th ACM Conference on Computer and Communications Security, Washington, DC, USA, 2003, pp. 8–19.
[67] J.-F. Podevin, Telling humans and computers apart automatically, Communications of the ACM 47 (2) (2004) 57–60.
[68] J. Fraser, Why you should never use a CAPTCHA, July 2010. <http://www.blogopreneur.com/2007/04/02/captchas-are-annoying/>.
[69] L.O. Caum, Why is CAPTCHA so annoying?, 2011. <http://lorenzocaum.com/blog/why-is-captcha-so-fing-annoying/>.
[70] G. Mori, J. Malik, Recognizing objects in adversarial clutter: breaking a visual CAPTCHA, in: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Madison, Wisconsin, 2003, pp. 134–141.
[71] J. Yan, A.S.E. Ahmad, A low-cost attack on a microsoft CAPTCHA, in: Proceedings of the 15th ACM Conference on Computer and communications security, Alexandria, VA, USA, 2008, pp. 543–554.
[72] E. Athanasopoulos, S. Antonatos, Enhanced CAPTCHAs: using animation to tell humans and computers apart, in: Proceedings of Communications and Multimedia Security, Heraklion, Crete, 2006, pp. 97–108.
[73] H.D. Truong, C.F. Turner, C.C. Zou, iCAPTCHA: the next generation of CAPTCHA designed to defend against 3rd party human attacks, in: Proceedings of IEEE International Conference on Communications, Kyoto, Japan, 2011, pp. 1–6.
[74] T. Peng, C. Leckie, K. Ramamohanarao, Detecting distributed denial of service attacks by sharing distributed beliefs, in: Proceedings of 8th Australasian Conference on Information Security and Privacy, Wollongong, Australia, 2003, pp. 214–225.
[75] E. Kline, M. Beaumont-Gay, J. Mirkovic, P. Reiher, RAD: reflector attack defense using message authentication codes, in: Proceedings of Annual Computer Security Applications Conference, Honolulu, HI, USA, 2009, pp. 269–278.
[76] W. Dai, Crypto++ 5.6.0 Benchmarks, 2009. <http://www.cryptopp.com/benchmarks.html>.
[77] D.J. Bernstein, SYN Cookies. <http://cr.yp.to/syncookies.html> (accessed 09.11).
[78] T. Thapngam, S. Yu, W. Zhou, G. Beliakov, Discriminating DDoS attack traffic from flash crowd through packet arrival patterns, in: Proceedings of 2011 IEEE Conference on Computer Communications Workshops, 2011, pp. 969–974.