Defending against Flooding-Based Distributed Denial-of-Service Attacks: A Tutorial

Rocky K. C. Chang, The Hong Kong Polytechnic University

ABSTRACT

Flooding-based distributed denial-of-service (DDoS) attack presents a very serious threat to the stability of the Internet. In a typical DDoS attack, a large number of compromised hosts are amassed to send useless packets to jam a victim, or its Internet connection, or both. In the last two years, it is discovered that DDoS attack methods and tools are becoming more sophisticated, effective, and also more difficult to trace to the real attackers. On the defense side, current technologies are still unable to withstand large-scale attacks. The main purpose of this article is therefore twofold. The first one is to describe various DDoS attack methods, and to present a systematic review and evaluation of the existing defense mechanisms. The second is to discuss a longer-term solution, dubbed the Internet-firewall approach, that attempts to intercept attack packets in the Internet core, well before reaching the victim.

INTRODUCTION

A denial-of-service attack (DoS), which purports to deny a victim (host, router, or entire network) providing or receiving normal services in the Internet, can be launched in many different ways. One classic approach is to exploit system design weaknesses, such as in the ping-of-death and teardrop attacks. System patches are usually issued immediately after discovering such attacks. Another kind of DoS attack is to impose computationally intensive tasks on a victim, such as encryption and decryption computation, and secret computation based on Diffie-Hellman exchanges. Internet security protocols today therefore include mechanisms against such attacks, e.g., cookies in the Internet Key Exchange protocol and anti-replay algorithms in the IPSec protocols.

Flooding-based distributed DoS attack, or simply DDoS attack, is another form of DoS attack, and is the focus of this article.[1] Unlike the other two, DDoS attacks do not rely on particular network protocols or system weaknesses. Instead, they simply exploit the huge resource asymmetry between the Internet and the victim in that a sufficient number of compromised hosts is amassed to send useless packets toward a victim around the same time. The magnitude of the combined traffic is significant enough to jam, or even crash, the victim (system resource exhaustion), or its Internet connection (bandwidth exhaustion), or both, therefore effectively taking the victim off the Internet. The widely publicized DDoS attacks against Yahoo!, eBay, Amazon.com, and several other popular Web sites in February 2000 revealed the vulnerability of even very well equipped networks. In fact, DDoS attacks are prevalent events in the Internet today, but most of them go unreported. For example, a recent study observed more than 12,000 DoS attacks during a three-week period, and the actual number is most likely much higher [1].

It is agreed that DDoS attacks have already become a major threat to the stability of the Internet [2]. On one hand, launching a DDoS attack is made very easy by the availability of a number of user-friendly attack tools. On the other, there is still a lack of effective solutions to defend against them in terms of aborting an ongoing attack in a timely fashion and tracing back to the attack sources. Therefore, DDoS problems are expected to only become more severe and serious in the future. For instance, they could be used in cyber warfare to disable strategic business, government, public utility, and even military sites. They can also be used by cyber gangsters to blackmail companies that rely on Internet connectivity for their revenues. On the positive side, a number of new startup companies, such as Arbor, Asta, Entercept, Mazu, and Recourse, sprang up in the last two years to offer solutions to DDoS problems. Nevertheless, there is little public information describing these solutions. There is also a lack of evidence supporting the effectiveness of these approaches.

The rest of this article first describes up-to-

[1] *Although Internet worms may also overload hosts or even networks, they are not considered flooding-based DDoS attacks in this article. Moreover, a detail discussion on Internet worms deserves a separate article.*

0163-6804/02/$17.00 © 2002 IEEE

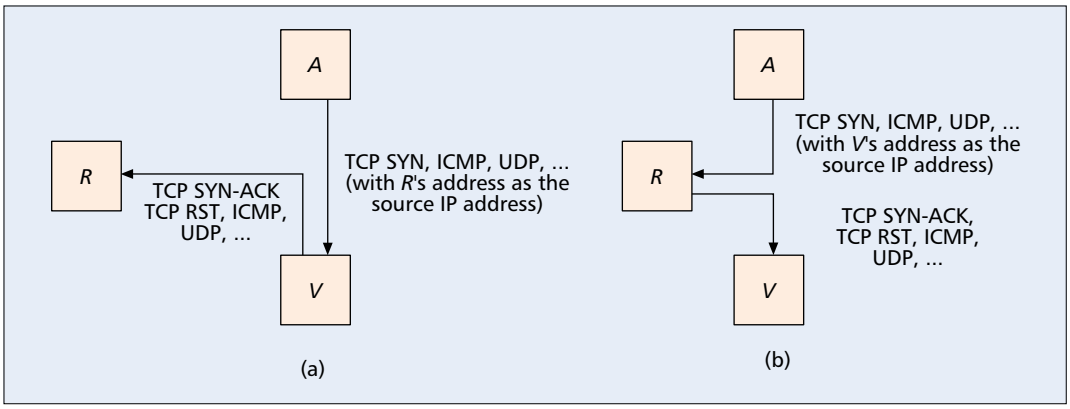

Figure 1. *Two types of flooding-based DDoS attack: a) direct; b) reflector.*

Before launching a direct attack, an attacker first sets up a DDoS attack network, consisting of one or more attacking hosts, a number of masters or handlers, and a large number of agents (also referred to as daemons or zombies).

date DDoS attack mechanisms, and then provides a systematic review and evaluation of the existing defense mechanisms. After that, this article also discusses an Internet-firewall approach which, unlike the existing solutions, attempts to intercept DDoS attacks in the Internet core. Two examples of this approach, route-based packet filtering and distributed attack detection, will be presented. Finally, this article ends with a comparison of four different approaches to defend against DDoS attacks.

The DDoS Problems

Direct Attacks

Broadly speaking, there are two types of flooding attacks: direct attacks and reflector attacks, as depicted in Fig. 1. In a direct attack, an attacker arranges to send out a large number of attack packets directly toward a victim. Attack packet types can be TCP, ICMP, UDP, or a mixture of them. In the TCP case, SYN flooding is the most well-known attack in which a large number of TCP SYN packets is sent to a victim's server port. If the port is actively listening for connection requests, the victim would respond by sending back SYN-ACK packets. However, since the source addresses in these attack packets are usually randomly generated (spoofed addresses), these response packets are sent elsewhere in the Internet (to *R* in Fig. 1a). Thus, the victim retransmits the SYN-ACK packets several times before giving up. However, these half-open connections will quickly consume all the memories allocated for pending connections, thus preventing the victim from accepting new requests. Besides SYN flooding, extraneous TCP state transitions and simultaneous open can be exploited to launch similar attacks.

Another type of TCP-based attack is to congest a victim's incoming link. Under these attacks, the victim usually responds with RST packets, except when the attack packets are also RST packets. ICMP messages (echo requests and timestamp requests) and UDP packets may also be used to achieve the same result. In these cases, the victim usually responds with the corresponding ICMP reply and error messages, and UDP packets. Based on a backscatter analysis performed on the response packets sent back by attack victims, Moore *et al.* have measured DoS activities in the Internet [1]. One notable observation reported is that most attacks used TCP packets (over 94 percent), followed by UDP packets (2 percent) and ICMP packets (2 percent). The TCP-based attacks are observed mainly based on SYN-ACK packets, RST packets, and ICMP error messages sent back by victims in response to attacks. The SYN-ACK response packets are clearly an indication of SYN flooding attacks. However, the other two types of response packets are inadequate for determining the occurrences of other TCP-based attack incidents.

Before launching a direct attack, an attacker first sets up a DDoS attack network, consisting of one or more attacking hosts, a number of masters or handlers, and a large number of agents (also referred to as daemons or zombies), as shown in Fig. 2a. The attacking host is a compromised machine used by the actual attacker to scan for vulnerable hosts and to implant specific DDoS master and agent programs, such as Trinoo, Tribe Flood Network 2000, and Stacheldraht. Each attacking host controls one or more masters, and each master in turn is connected to a group of agents. A detailed description of the entire process of building a DDoS attack network is given in [3]. With an attack network ready, the attacking host may launch a DDoS attack by issuing an attack command with the victim's address, attack duration, attack methods, and other instructions to the masters. This communication is based on TCP in Trinoo, and the messages are even encrypted in Stacheldraht. Each master, upon receiving the instructions, then passes them to its agents for execution. Today's DDoS attack tools can launch attacks against multiple victims at the same time, and use various types of attack packets.

Reflector Attacks

A reflector attack is an indirect attack in that intermediary nodes (routers and various servers), better known as *reflectors*, are innocently used as attack launchers. An attacker sends packets that require responses to the reflectors with the packets' inscribed source addresses set to a victim's address. Without realizing that the packets are actually address-spoofed, the reflectors return response packets to the victim according to the types of the attack packets. As a result, as illustrated in Fig. 1b, the attack packets are essentially reflected, in the form of normal packets, toward the victim, and the reflected packets can flood the victim's link if the number of reflectors is large enough.

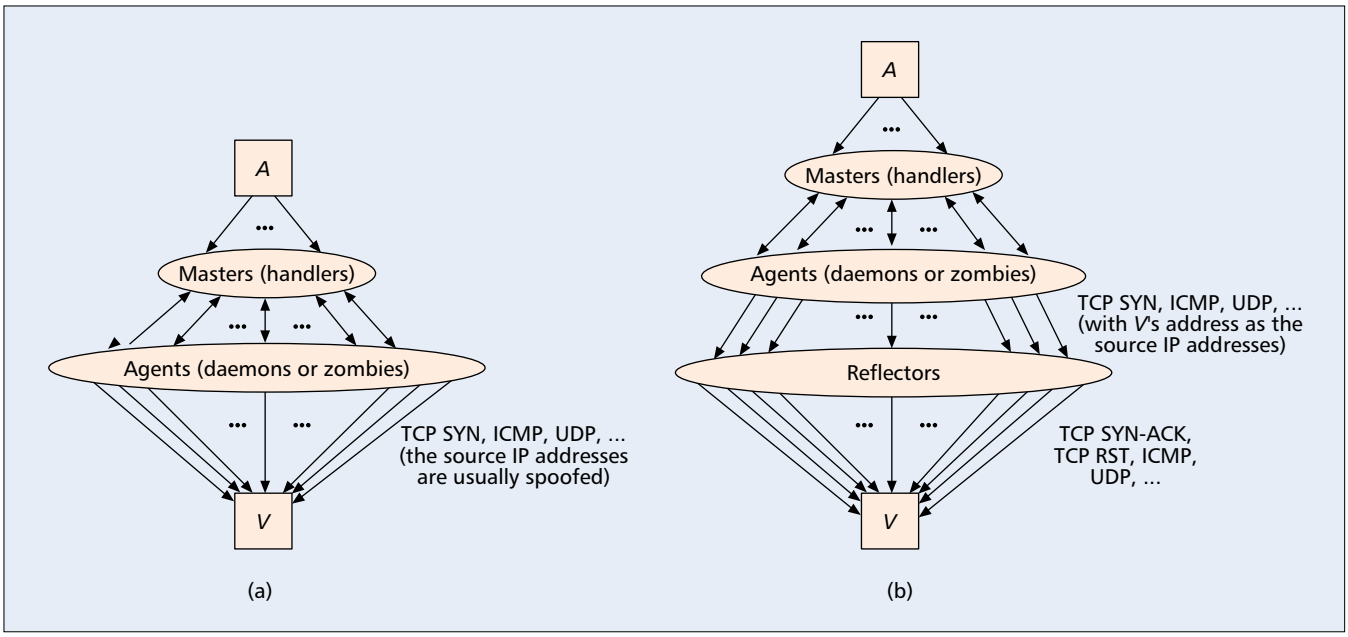

Figure 2. *DDoS attack architectures for a) direct and b) reflector attacks.*

Reflector attack is not entirely new. Smurf is a classic reflector attack that is triggered by sending an ICMP echo request to a subnet-directed broadcast address with the victim's address as the source address. The victim would therefore be overwhelmed by the ICMP echo replies sent from all nodes in that subnet. Strictly speaking, Smurf is not a DDoS attack, because the attack sources (the reflectors) are all confined to a subnet. Moreover, Smurf attacks can be prevented by filtering packets with subnet-directed broadcast addresses.

As explained before, reflector attacks are based on reflector's ability of generating messages in response to other messages. Thus, any protocol that supports this type of "automatic message generation" can be exploited to launch reflector attacks, including TCP and UDP packets, various ICMP messages, and application protocol messages. When TCP attack packets are used, a reflector may respond with either SYN-ACK packets (in response to SYN packets) or RST packets (in response to illegitimate TCP packets). When SYN-ACK packets are involved, the reflector in fact behaves like a victim of SYN flooding attacks, because it also maintains a number of half-open connections. However, the situation may not be as severe as the direct attack, because each reflector usually contributes only a small part to the entire attack. Moreover, in contrast to SYN flooding attack, this SYN-ACK flooding attack does not exhaust the victim's ability to accept new connections, because the SYN-ACK packets can easily be recognized as illegal by checking the ACK flag. Instead, this reflector attack and also the flooding of RST packets are aimed at clogging the victim's network link [4].

In addition to ICMP echo messages, many ICMP error messages can be taken advantage of by an attacker in a reflector attack. For example, attack packets inscribed with inactive destination ports would trigger hosts to send ICMP port unreachable messages. Attack packets with very small time to live (TTL) values would trigger routers to send ICMP time exceeded messages. An even more effective approach is based on bandwidth amplification in which an attack packet (e.g., DNS recursive queries) triggers a reflected packet of a much larger packet size (e.g., DNS replies) [5]. These reflector attack methods are summarized in Table 1. Paxson has also analyzed the vulnerability of Gnutella and T/TCP protocols in a reflector attack [6]. Unlike direct attacks, reflector attacks cannot be observed by a backscatter analysis, because attack victims do not send back any packets in response. As a result, except for some isolated reports, such as the ones in [4], the prevalence of reflector attacks in the Internet is largely unknown.

As shown in Fig. 2b, the attack architecture for launching reflector attacks is very similar to that for direct attacks. However, there are several important differences. First of all, a reflector attack requires a set of predetermined reflectors, including DNS servers, HTTP servers, or even routers. Therefore, the magnitude of the attack is based on the size of the reflector pool (instead of the agent pool in direct attacks), and the transmission frequency and size of the reflected packets. The attack sources (the reflectors) could also be more dispersed on the global Internet, because the attacker does not need to implant agents to launch such attacks. Moreover, the reflected packets are in fact normal packets with legitimate source addresses and packet types, which therefore cannot be filtered based on address spoofing or other route-based mechanisms.

How Many Attack Packets Are Needed?

In a SYN flooding attack, each half-open connection is held for a certain amount of time before giving up. If a victim has resources to admit N half-open connections, its capacity of processing incoming SYN packets can be modeled as a $G/D/\infty/N$ queue, where G is a general arrival process for the SYN packets, and D refers

	Packets sent by an attacker to a reflector (with a victim's address as the source address)	Packets sent by the reflector to the victim in response
Smurf	ICMP echo queries to a subnet-directed broadcast address	ICMP echo replies
SYN flooding	TCP SYN packets to public TCP servers (e.g., Web servers)	TCP SYN-ACK packets
RST flooding	TCP packets to nonlistening TCP ports	TCP RST packets
ICMP flooding	• ICMP queries (usually echo queries) • UDP packets to nonlistening UDP ports • IP packets with low TTL values	• ICMP replies (usually echo replies) • ICMP port unreachable messages • ICMP time exceeded messages
DNS reply flooding	DNS (recursive) queries to DNS servers	DNS replies (usually much larger than DNS queries)

Table 1. *A summary of some reflector attack methods.*

to the deterministic lifetime for each half-open connection if not receiving the third handshaking message. This infinite-server queuing model with finite capacity yields the minimal rate of SYN packets required to exhaust the server's resources for pending connections. Figure 3 shows the results for three types of servers: Microsoft Win2000 Advanced Server (WIN), BSD, and Linux kernel 2.2.9-19. The three systems employ similar exponential backoff mechanisms to retransmit apparently lost SYN packets. BSD systems set the retransmission timeouts to 6, 24, and 48 s, and usually give up the retransmissions after a total of 75 s. On the other hand, both Linux and WIN systems' first retransmissions start at 3 s after the initial transmission, and the subsequent timeout period is double the pervious one (i.e., the timeouts are given by 3, 6, 12 s, etc.). However, the WIN system retransmits SYN packets at most twice; therefore, it will give up the connection if not receiving ACKs within 9 s. The Linux system, on the other hand, allows up to 7 retransmissions, amounting to a total of 309 s before giving up the connection.

Based on the maximum lifetimes of half-open connections, the WIN system is apparently tuned to offer better protection against SYN flooding attacks. Compared with the other two systems, a much higher rate of SYN packets is therefore required to exhaust a WIN system's resources for accepting new connection requests. If each SYN packet is 84 bytes long (including the Ethernet frame header and interframe gap), a 56 kb/s connection is sufficient to stall both Linux and BSD servers with $N \leq 6000$. Moreover, a 1 Mb/s connection is sufficient to stall all three servers with $N \leq 10{,}000$. In reflector attacks, the SYN-ACK flooding is also very effective to clog the victim's network link, because the reflectors involved keep on retransmitting SYN-ACK messages before giving up the connections. As recalled, the Linux system retransmits SYN packets most aggressively among the three systems; therefore, it also causes more damage when used in a SYN-ACK flooding attack.

In other flooding attacks aimed at jamming a victim's incoming link, an aggregated attack traffic rate has to be at least 1.544 Mb/s to jam a T1 link. For example, in a direct ICMP ping flooding attack, around 5000 agents are needed to flood a victim's T1 link if each agent sends a query every second. The required number of agents is further decreased if the frequency of sending queries is increased. In a corresponding reflector attack, the number of reflectors is also 5000 but the number of agents can be much fewer, if each agent is responsible for sending ICMP echo requests to a number of reflectors.

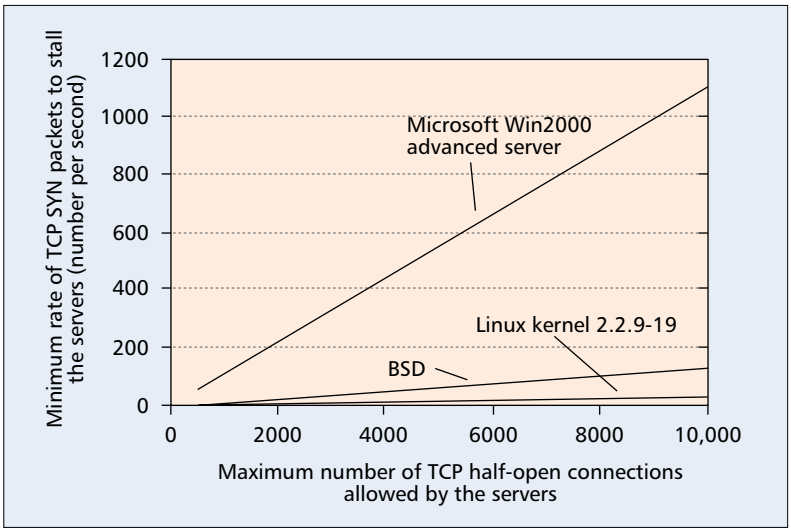

Figure 3. *Minimal rates of SYN packets to stall TCP servers in SYN flooding attacks.*

Solutions to the DDoS Problems

In reference to the start and end of a DDoS attack, there are three lines of defense against the attack: *attack prevention and preemption* (before the attack), *attack detection and filtering* (during the attack), and *attack source traceback and identification* (during and after the attack). Since there is no one-size-fits-all solution to the DDoS problem, a comprehensive solution should include all three lines of defense, each of which is further elaborated in the following.

Attack Prevention and Preemption

The first line of defense is obviously to prevent DDoS attacks from taking place. On the passive side, hosts may be securely protected from master and agent implants. There are indeed known signatures and scanning procedures to detect them. Another is to monitor network traffic for known attack messages sent between attackers

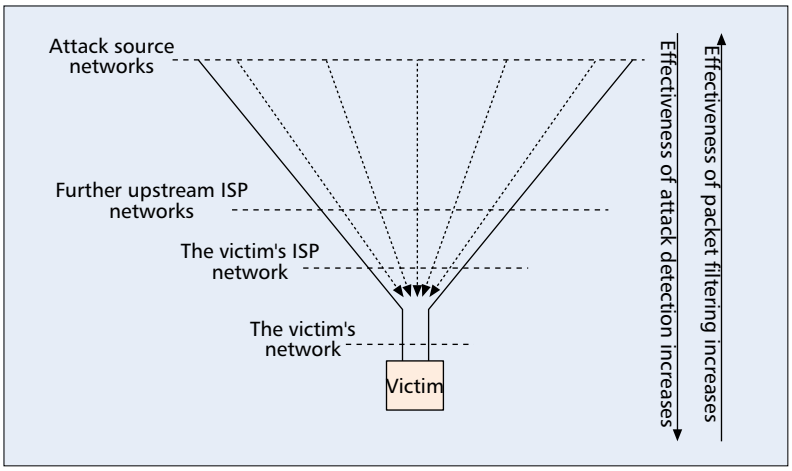

Figure 4. *Possible locations for performing DDoS attack detection and filtering.*

and masters. On the active side, cyber-informants and cyber-spies can be employed to intercept attack plans. For instance, Gibson vividly described how he successfully spied on attack plans among a group of agents [7].

This line of defense alone is clearly inadequate. There are always don't-care users and careless users, who leave their hosts vulnerable to DDoS agent implants. Internet service providers (ISPs) and enterprise networks do not have incentives to monitor for attack packets. Furthermore, spying on attack plans, such as the one described in [7], requires in-depth knowledge of particular ways of launching DDoS attacks, which may also be altered later on to avoid spying.

Attack Source Traceback and Identification

Attack source traceback and identification is usually an after-the-fact response to a DDoS attack. IP traceback refers to the problem, as well as the solution, of identifying the actual source of any packet sent across the Internet without relying on the source information in the packet. There are generally two approaches to the IP traceback problem. One is for routers to record information about packets they have seen for later traceback requests [8]. Another is for routers to send additional information about the packets they have seen to the packets' destinations via either the packets [9] or another channel, such as ICMP messages.

However, it is infeasible to use IP traceback to stop an ongoing DDoS attack. First, current IP traceback solutions are not always able to trace packets' origins (e.g., those behind firewalls and network address translators). Moreover, IP traceback is ineffective in reflector attacks in which the attack packets come from legitimate sources. Even if the attack sources can be successfully traced, stopping them from sending attack packets is another very difficult task, especially when they are scattered in various autonomous systems (ASs). Nevertheless, IP traceback could be very helpful in identifying the attacker and collecting evidence for post-attack law enforcement.

Attack Detection and Filtering

A third approach consists of two distinguishable phases: DDoS attack (packet) detection and attack packet filtering. The detection part is responsible for identifying DDoS attacks or attack packets. After identifying attack packet flows or attack packets, the filtering part is responsible for classifying those packets and then dropping them (rate-limiting is another possible action). The overall performance of this detect-and-filter approach clearly depends on the effectiveness of both phases. The false positive ratio (FPR) and false negative ratio (FNR) can quantitatively measure the effectiveness of the attack detection. The FPR is given by the number of packets classified as attack packets (positive) by a detection system that are confirmed to be normal (negative), divided by the total number of confirmed normal packets. The FNR, on the other hand, is given by the number of packets classified as normal (negative) by a detection system that are confirmed to be attack packets (positive), divided by the total number of confirmed attack packets. Effective DDoS attack detection should yield very low ratios.

The effectiveness of packet filtering, on the other hand, refers to the level of normal service that can be maintained by the victim during a DDoS attack by filtering the attack packets. It is very important to first point out that effective attack detection does not always translate into effective packet filtering. Because of the distributed nature of the attack, the detection phase can only use the victim's identities, such as IP address and port number, as the signatures of the attack flows. As a result, packet filtering usually drops attack packets as well as normal packets because both match the signatures. As a result, packet filtering does not always help restore the victim's service. Quantitatively, the effectiveness of packet filtering can be measured by *normal packet survival ratio* (NPSR), which gives the percentage of normal packets that can make their way to the victim in the midst of a DDoS attack. An effective packet filtering mechanism should be able to achieve a high NPSR during a DDoS attack.

Figure 4 shows that the detect-and-filter approach can be performed in four places on the paths between the victim and the agents or reflectors. As depicted in the diagram, a DDoS attack resembles a funnel in which attack packets are generated in a dispersed area, like the top of a funnel. The victim, like the narrow end of a funnel, receives all the attack packets generated. Thus, it is not difficult to see that detecting a DDoS attack is "relatively" easy at the victim network, because it can observe all the attack packets. In contrast, it is less likely for an individual source network, where attack sources (agents and reflectors) are located, to detect the attack unless a large number of attack sources are located in that network. Opposite to the case of attack detection, it is most effective to filter attack packets closer to the agents or reflectors. The effectiveness of packet filtering declines rapidly as attack packets are dropped closer to the victim, because more normal packets would also be dropped. During typical large-scale DDoS

attacks, a victim network or its ISP network usually drops all packets destined to the victim network, thus reducing the NPSR close to zero.

At Source Networks — Although source networks usually cannot detect a DDoS attack, they can still filter attack packets, as well as other illegitimate packets, based on address spoofing. ISP networks that are directly connected to source networks can effectively ingress-filter spoofed packets [10]. Thus, a ubiquitous deployment of the ingress packet filters can drop all attack packets in direct attacks, and all attack packets sent from agents to reflectors, if all attack packets use spoofed addresses. Even if this is not the case, the attack agents can be traced easily in direct attacks. However, tracing to attackers in reflector attacks may still be very difficult even when valid addresses are used. Of course, ensuring all ISP networks in the Internet to install ingress packet filtering is an impossible task in itself.

At the Victim's Network — Unlike the case for source networks, a DDoS victim can detect (by a router, intrusion detection system, or network operator) a DDoS attack based on an unusually high volume of incoming traffic (of certain packet types) or degraded server and network performance. In fact, a number of recent startup companies have offered products based on this traffic anomaly approach to detecting DDoS attacks. These commercial detection systems are usually placed in a network under protection or in a service provider's network. Although the details of the attack detection algorithms are not disclosed, statistical approaches employed by other intrusion detection systems, notably EMERALD, have been presented and discussed in the past [11].

Besides, other defense mechanisms that do not use the detect-and-filter approach have been proposed or deployed to protect a victim host from DDoS attacks. One such mechanism is known as *IP hopping* or the *moving target defense*, in which a host frequently changes its IP address or changes its IP address when a DDoS attack is detected. However, it is not difficult to add to the attack tools a DNS tracing function, which could attack the new address after discovering the address change. Another approach is to tackle SYN flooding attacks by proxying TCP connection requests, so the actual host does not need to handle half-open connections. However, this approach does not hold up under large-scale DDoS attacks. Moreover, if an incoming link is jammed by attack packets, a victim practically cannot do anything but shut down its network and ask the upstream ISP to filter the packets involved [7].

At a Victim's Upstream ISP Network — Frequently, an upstream ISP is requested (through telephone calls) by a DDoS victim to filter attack packets. To speed up and automate this process, a victim network may send to an upstream ISP router an intrusion alert message, which specifies the signatures of the attack packet flows, as soon as detecting a DDoS attack. Such intrusion alert protocol needs to be designed carefully. If the protocol is based on TCP, the victim network is unable to receive acknowledgments in the midst of an attack. The messages also have to be protected by strong authentication and encryption algorithms. Similar to the previous discussion on victim networks, it is not effective to filter attack packets at the victim's upstream ISP network, because most, if not all, normal packets would also be dropped in the process. As a result, the victim network is still declared unusable. Moreover, the upstream ISP network may also be jammed under sufficiently large-scale DDoS attacks.

At Further Upstream ISP Networks — In principle, one could extend the backpressure approach just described to further upstream ISP networks. That is, the victim network is responsible for detecting DDoS attacks, and the upstream ISPs are then notified to filter those packets matched with the signatures of the attack packet flows. In other words, packet filtering is pushed as upstream as possible. Similar to the ubiquitous ingress filtering, this approach is effective only if ISP networks are willing to cooperate and to install packet filters upon receiving intrusion alerts.

It is not effective to filter attack packets at the victim's upstream ISP network, because most, if not all, normal packets would also be dropped in the process. As a result, the victim network is still declared unusable.

AN INTERNET FIREWALL?

Driven by the need for quick deployment of defense mechanisms against DDoS attacks, the current detect-and-filter approaches, as reviewed in the last section, are implemented mainly at source networks and victim networks. However, this bipolar defense scheme cannot possibly achieve both effective attack detection and effective packet filtering, as discussed in the last section. In response to this shortcoming, proposals to deploy a global defense infrastructure, or an "Internet firewall," to protect the entire Internet from DDoS attacks have recently been made. That is, this Internet firewall attempts to detect DDoS attacks in the Internet core so that it can drop the suspected attack packets well before reaching a victim. If implemented correctly and effectively, this approach has the potential to maintain a victim's normal service during an attack. The rest of this section discusses two such proposals, both of which employ a set of distributed nodes in the Internet to perform attack detection and packet filtering.

A ROUTE-BASED PACKET FILTERING APPROACH

The *route-based packet filtering* (RPF) approach, proposed by Park and Lee, essentially extends the ingress packet filtering function to the Internet core [12]. This approach employs a number of distributed packet filters to examine whether each received packet comes from a correct link according to the inscribed source and destination addresses, and the BGP routing information. A packet is considered an attack packet (or other illegitimate packet) if received from an unexpected link and is therefore dropped. However, note that the dropped packet may still be legitimate due to a recent route change. Simulation experiments have shown that a significant fraction of spoofed packets can be filtered by implementing the packet filters in about 18 percent of ASs in the Internet. Moreover, the effectiveness of the approach is quite sensitive to the underlying Internet AS connectivity structure.

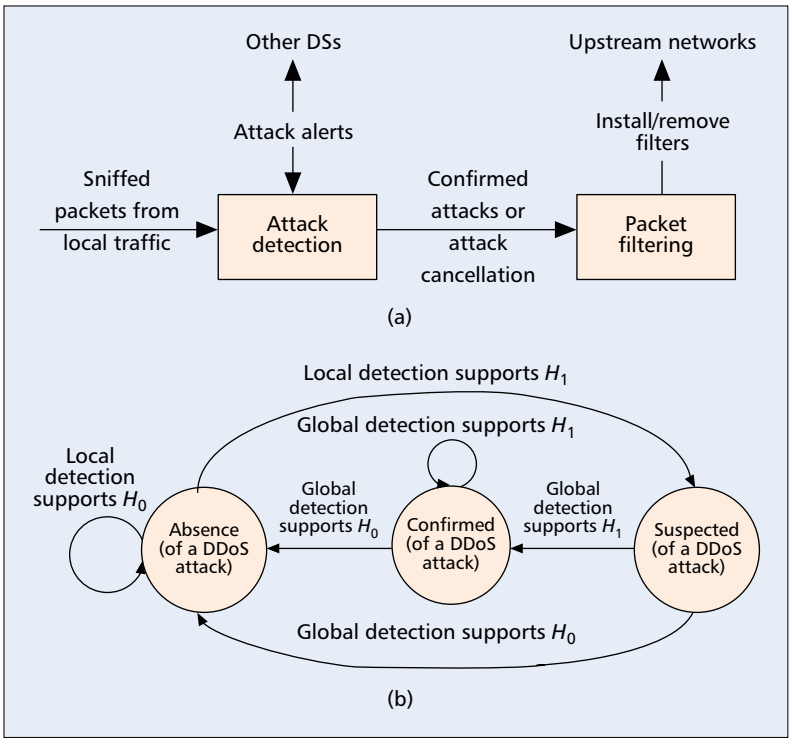

Figure 5. *a) High-level DS architecture and b) a state diagram of two-level attack detection in the distributed detection approach.*

On the practical side, the major drawback of this approach is to require BGP messages to carry source addresses. Inclusion of the source information would significantly increase the BGP message size and the message processing time. Moreover, although the RPF approach significantly reduces the number of filters required for ubiquitous deployment of ingress packet filtering, the number after reduction is still very high for global deployment. With the number of ASs currently exceeding 10,000, filters need to be placed in at least 1800 ASs in order to be effective! Moreover, the number of ASs is also expected to increase continuously. Finally, as in the ingress packet filtering approach, the RPF approach cannot filter attack packets using valid source addresses, such as reflected packets.

A Distributed Attack Detection Approach

Just as the RPF approach extends the ingress packet filtering function to the Internet core, *distributed attack detection* (DAD), another Internet firewall approach, extends a typical intrusion detection system's functions to the Internet core. That is, the DAD approach detects DDoS attacks based on network anomalies and misuses observed from a set of distributed *detection systems* (DSs). Anomaly detection involves both determining a set of normal traffic patterns and detecting traffic patterns that "significantly" deviate from the normal ones. For example, traffic intensity of particular types of packets may serve as a parameter for detecting anomalies in connection to DDoS attacks. Misuse detection, on the other hand, aims to identify traffic that matches a known attack signature. For example, it is known that an attacker using Trinoo communicates with a master via a TCP port of 27665, while a master communicates with an agent via a UDP port of 27444. Since DSs are expected to rely mainly on traffic anomaly detection to discover DDoS attacks, the rest of this section will concentrate the anomaly detection function.

In the DAD approach, a number of DSs are placed in "strategic" locations in the Internet, and they nonintrusively monitor and analyze the traffic passing through them for possible DDoS attacks.[2] Since each DS can usually observe only partial anomalies (or none), the DSs cooperatively detect DDoS attacks by exchanging attack information derived from local observations. Thus, the main difference between the DAD and RPF approaches is that the former is stateful in respect to the presence (or absence) of DDoS attacks, while the latter processes each packet independently. As a result, the DAD approach requires sophisticated mechanisms to correlate the information received from packets passing through the DSs, and a separate channel for the DSs to communicate.

Designing an effective and deployable architecture for the DAD approach is a challenging task that involves many algorithmic and engineering design issues. For instance, what is the minimum number of DSs required in this infrastructure? Without going into a detailed analysis, one can expect that the number of DSs can be much smaller than that required by the RPF approach, because the DAD approach does not rely on routing information to detect DDoS attacks. Furthermore, a larger set of DSs would translate into a longer delay in response to DDoS attacks. Optimal placement of the DSs, which is closely related to the size of the DS set, is also an important problem, and its objective is to ensure that the set of DSs can observe most of the attack scenarios. Potential detection locations include network access points, Internet exchanges, and backbone ISP networks. Other issues are related to the design of the individual DSs and communication among themselves, to be discussed next. Wan and I have discussed these issues in more detail and proposed solutions to address them [13].

DS Design Considerations — One major challenge in DS design is to process packets at very high speeds, especially when placed in backbone networks. Improving the design of hardware, systems, and algorithms for high-speed packet processing continues to be a very important and active research area. In the case of monitoring local traffic for DDoS attacks, a high-speed packet classifier may be dispatched to quickly distribute packet streams to different DSs for further processing. Figure 5a shows a high-level architecture of such a DS that performs attack detection and packet filtering.

Due to the distributed nature of a typical DDoS attack, each DS can observe only partial traffic anomalies. As a result, the entire attack detection process consists of two levels: *local detection* and *global detection*. There are two hypotheses to test on both levels: H_1 for the presence of a DDoS attack, and H_0, a null hypothesis. The binary hypothesis is tested on a set of packet flows, of which the packets share the same destination IP address (the victim's address) and possibly other information in the packets, such as packet types, TCP flags, and

[2] *Although the DAD approach is described here as an Internet-wide deployment, the approach can be scaled down to protect an arbitrary set of networks in the Internet.*

port numbers. As soon as the local detection supports H_1, the DS involved floods an attack alert to all other DSs, signaling a possible DDoS attack. Each DS then independently consolidates and analyzes its local detection result with attack alerts received from other DSs to make a global detection decision (see the attack detection box in Fig. 5a). For this purpose, each attack alert should include a *confidence level* that quantifies the amount of evidence supporting the suspected attack. If a DDoS attack is confirmed, the DS notifies the packet filtering component to install packet filters for the corresponding packet stream, and it may also notify the upstream networks to filter the attack packets there (see the packet filtering box in Fig. 5a). Figure 5b shows the state diagram of two-level attack detection.

Packet filtering, however, degrades switches' performance significantly, especially during an ongoing DDoS attack. Therefore, one approach to alleviating the workload is to install filters only on *suspected* switch interfaces. Specifically, as soon as a DDoS attack is confirmed (in the CONFIRMED state), the DS concerned installs packet filters on all switch interfaces. The DS, at the same time, starts analyzing traffic received from the switch interfaces to identify the attack interfaces that are responsible for receiving a significant amount of attack packets. Filters will therefore be installed only on those attack interfaces.

Another important consideration is to ensure that any DS can reliably flood attack alert messages to other DSs. Therefore, the most important requirement for this network of DSs is that all DSs must always be "connected," which includes both physical connectivity and usability of the paths among the DSs. There are quite a number of technical issues to address in order to fulfill this requirement. For example, what is the best logical topology for the DS network, and how can a partitioned DS network be reconnected? When a DS is under a DDoS attack itself, how would the DS send attack alerts to other DSs? In terms of the communications protocols and intrusion language specification, possible candidates are the Intrusion Detection Exchange Protocol and Intrusion Detection Message Exchange Format, both works in progress under IETF [14].

A Quickest Detection Problem Formulation — The two-level DDoS attack detection can be formulated as a *quickest detection problem*, which has been studied in the areas of signal processing, quality control, and wireless channel monitoring. For the time being, first consider the problem formulation for local detection in which a DS computes the instantaneous traffic intensity for a particular packet flow periodically. Let the ith sample of the instantaneous traffic intensity be A_i, $i \geq 1$. Further assume that DDoS attack packets reach the DS between the $(k-1)$th and kth sample, such that the distribution of A_i follows P_0 for $1 \leq i < k$ but follows P_1 for $i \geq k$. The objective of the quickest detection problem is to detect this "abrupt" change in the distribution ($P_0 \rightarrow P_1$) as soon as possible, subject to some constraints on the occurrences of false alarms (i.e., supporting H_1 in the absence of DDoS attacks). The event responsible for the change in distribution is usually called a *disorder* (the arrivals of attack packets), and the time of the disorder occurrence is known as *change time*.

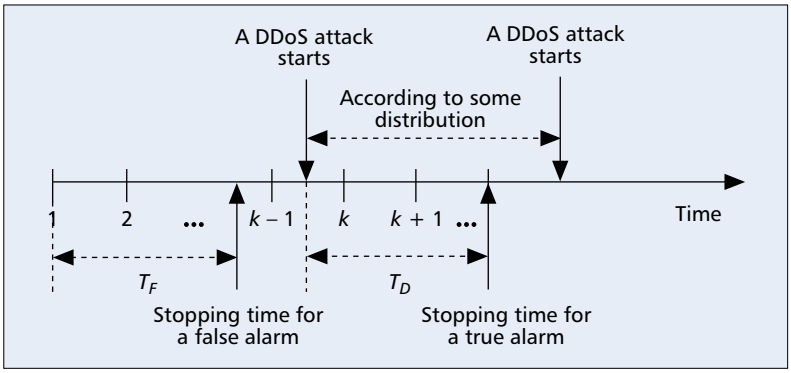

Figure 6. *Relevant events in DDoS attack detection.*

There are generally two approaches to mathematically formulate the quickest detection problem. The first is a Bayesian formulation in which the change time is assumed to have a known prior distribution, such as geometric distributions. The objective is to minimize the expected delay of detecting the disorder after its occurrence (denoted by T_D in Fig. 6), subject to a lower bound on the expected time between false alarms before the disorder (denoted by T_F in Fig. 6). In the second approach, the change time is unknown but nonrandom. That is, no prior distribution is given for the change time. The objective and constraints are the same as the first approach. Since DDoS attack occurrences generally do not follow a particular distribution, the second problem formulation is preferred. A simple local detection algorithm based on the second approach is a threshold-based decision rule [15]. The DS will move the state of the packet flow to the SUSPECTED state if the sum of a number of traffic intensity measurements exceeds a local threshold.

As for global detection, for simplicity assume that all DSs' local decisions (0 for supporting H_0 and 1 for supporting H_1) are available to all DSs instantly and their decision times are synchronized. A DS can then formulate a likelihood ratio based on these local decisions. Under certain assumptions about the local decisions, a test statistic for the global detection can be obtained and a DDoS attack is confirmed if the test statistic exceeds a global threshold [15].

Limitations and Open Problems — The approach of detecting DDoS attacks distributedly based on traffic anomalies has its own limitations, as well as a few open problems. On one hand, there are a set of theoretical issues related to the detection algorithms presented in the last section, such as the choices of local and global thresholds, traffic modeling, and admitting multilevel local detection results, but they are not further discussed here due to their mathematical nature. On the other hand, there is another set of issues concerning other performance aspects of the DAD approach, which are briefly outlined below.

Since the two-level detection induces a certain amount of delay to reach a global detection decision, this DS network is not very useful for DDoS attacks of very short durations. For example, the DS network should be designed to han-

	Ubiquitous ingress packet filtering (UIPF)	Route-based packet filtering (RPF)	Local attack detection (LAD)	Distributed attack detection (DAD)
1. Detection locations	All ISP networks that are connected to leaf networks in the Internet	A set of packet filters distributed in the Internet	Potential victims' networks and/or their upstream ISP networks	A set of detection systems distributed in the Internet
2. Filtering locations	Same as the detection locations	Same as the detection locations	Same as the detection locations and further upstream ISP networks if backpressure is used	Same as the detection locations and other upstream networks
3. Attack signatures	Spoofed source IP addresses	Spoofed source IP addresses according to the BGP routing information	Traffic anomalies and misuses detected by local intrusion detection systems	Mainly traffic anomalies observed from the set of distributed detection systems
4. False positive ratio (FPR)	= 0	= 0 if the BGP routes are correct	≥ 0 (= 1 in a sufficiently large-scale DDoS attack)	≥ 0 (high if the detection algorithms are overly sensitive)
5. False negative ratio (FNR)	≥ 0 (= 0 if all attack packets use spoofed addresses)	≥ 0 (small if most attack packets use spoofed addresses)	≥ 0 (= 0 in a sufficiently large-scale DDoS attack)	≥ 0 (high if the detection algorithms are not sensitive enough)
6. Normal packet survival ratio (NPSR)	≥ 0 (= 1 if all attack packets use spoofed addresses)	≥ 0 (large if most attack packets use spoofed addresses and the number of the AS nodes involved in the packet filtering is sufficiently large)	≥ 0 (= 0 in a sufficiently large-scale DDoS attack)	≥ 0 (high if both the false negative and positive ratios are low, and the set of detection systems are placed optimally in the Internet)
7. New communication protocols	Not required	Modifications to BGP protocols	Attack alert protocols between victims and their upstream ISP networks if backpressure is used	Protocols between detection systems
8. Computation requirement	Low	Moderate	Low	High
9. Deployment difficulty	Very high	High	Moderate without backpressure mechanisms	High
10. Technical complexity	Low	High	Moderate without backpressure mechanisms	High

Table 2. *A comparison of four approaches to detecting and filtering DDoS attack packets.*

dle DDoS attacks longer than 5 min, which take up around 75 percent of all the attacks measured in a recent study [1].

Flash crowds on the Internet can trigger false alarms in the detection systems. The events triggering the flash crowds can be unpredictable (e.g., news reporting on September 11, 2001), predictable but nonrepetitive (e.g., World Cup soccer games), and predictable and repetitive (e.g., transaction requests when the stock market just opens). For predictable and repetitive events, a different normal traffic pattern may be used at the time the flash crowd event occurs. For the other two, the detection algorithms are required to adapt to the new "normal" traffic patterns as soon as they detect a number of false alarms.

Besides DDoS attacks, new attack patterns continue to emerge, such as degradation of service (DeS) attacks that use "pulsing agents" to send out short bursts of attack packets, instead of steady packet streams in DDoS attacks. Moreover, a different set of agents may be used the next time to send attack packets, which makes it more difficult to detect them and to trace to the attack agents. Unlike DDoS attacks, the purpose of DeS attacks is only to degrade the victim's service; therefore, traffic anomaly patterns other than abrupt changes in traffic rates need to be identified in order to detect these attacks.

A Comparison of Four Detect-and-Filter Approaches

Based on the previous discussion, there are four detect-and-filter approaches to defend against general flooding-based, DDoS attacks in the Internet: ubiquitous ingress packet filtering (UIPF), local attack detection (LAD), RPF, and DAD. The UIPF approach refers to the perfect ingress filtering scenario where every packet in the Internet is subject to ingress filtering. Although UIPF is quite impossible to realize, it is included here for the sake of completeness in the comparison study. The LAD approach refers to the attack detection at a victim's network and/or its ISP networks. Other defense mechanisms for specific attacks, such as SYN flooding, are not considered here.

Table 2 compares the four approaches on three sets of criteria: detection and filtering mechanisms (items 1–3), effectiveness of attack detection and packet filtering (items 4–6), and other technical requirements and deployment

effort (items 7–10). Both UIPF and RPF are listed side by side because they are similar in their detection parts (compare items 2 and 3), detecting attack packets based on spoofed source addresses albeit their differences in the actual detection mechanisms (compare items 1 and 3). Likewise, LAD and DAD are similar in their detection parts (compare items 2 and 3), because both detect DDoS attacks based on anomaly detection and misuse detection.

Clearly, UIPF can achieve zero FPR and zero FNR if all attack packets use spoofed addresses. Furthermore, since attack packets can all be dropped immediately after exiting source networks, all normal packets can arrive at the attack victim (i.e., NPSR = 1). Its deployment difficulty, on the other hand, is obviously the highest among the four. Since RPF is in essence an ingress packet filtering approach implemented in the Internet core, it shares similar performance characteristics as UIPF (compare items 4–6). The deployment difficulty for UIPF is somewhat eased by having a smaller number of ASs involved in the packet filtering, but the number is still high and is also expected to increase.

As discussed before, packet filtering in LAD is very ineffective in the midst of a sufficiently large-scale attack (see items 4–6). However, this approach is most deployable among the four because the detection activities are centralized in the victim network or its ISP network. Finally, DAD's effectiveness in detecting attacks and filtering attack packets depends very much on the performance of the distributed detection algorithms. Unlike LAD, this approach has the potential of achieving a high NPSR during a large-scale DDoS attack. The deployment difficulty and technical complexity for both RPF and DAD can be considered high, but DAD poses a higher computation requirement on DSs due to the continuous analysis of the network traffic for DDoS attacks.

Conclusion

The current defense mechanisms reviewed in this article are clearly far from adequate to protect Internet nodes from DDoS attacks. The main problem is that there are still many insecure areas in the Internet today that can be compromised to launch large-scale DDoS attacks. This situation will perhaps last for a long while, if not forever. Coupled with the fact that attack mechanisms and tools continue to improve and evolve, more effective detect-and-filter approaches must be developed in addition to the use of ingress packet filtering and other existing defense mechanisms and procedures. One promising direction is to develop a global defense infrastructure, or an Internet firewall, to protect the entire Internet from DDoS attacks. This article describes and contrasts two deployable proposals of implementing this Internet firewall approach, namely route-based packet filtering and distributed attack detection. While the route-based packet filtering approach extends the ingress packet filtering function to the Internet core, the distributed attack detection approach extends a typical intrusion detection system's functions to the Internet core.

Acknowledgments

The work described in this article was partially supported by a grant from the Research Grant Council of the Hong Kong Special Administrative Region, China (Project no. PolyU5080/02E). The author would also like to thank Samuel Choi and Kitty To for studying the SYN flooding issues in Linux and Microsoft Win2000 Advanced Server. The author is especially indebted to Kalman Wan for starting an investigation on this problem.

References

[1] D. Moore, G. Voelker, and S. Savage, "Inferring Internet Denial-of-Service Activity," *Proc. 10th USENIX Sec. Symp.*, 2001.

[2] Comp. Emergency Response Team, "Results of the Distributed-Systems Intruder Tools Workshop," http://www.cert.org/reports/dsit_workshop-final.html, Nov. 1999.

[3] D. Dittrich, "The DoS Project's 'Trinoo' Distributed Denial of Service Attack Tool," http://staff.washington.edu/dittrich/misc/trinoo.analysis, Oct. 1999.

[4] S. Gibson, "Distributed Reflection Denial of Service: Description and Analysis of a Potent, Increasingly Prevalent, and Worrisome Internet Attack," http://grc.com/dos/drdos.htm, Feb. 2002.

[5] Comp. Emergency Response Team Incident Note IN-2000-04, "Denial of Service Attacks Using Nameservers," http://www.cert.org/incident_notes/ IN-2000-04.html, Apr. 2000.

[6] V. Paxson, "An Analysis of Using Reflectors for Distributed Denial-of-Service Attacks," *ACM Comp. Commun. Rev.*, vol. 31, no. 3, July 2001, pp. 38–47.

[7] S. Gibson, "The Strange Tale of the Denial of Service Attacks Against GRC.COM," http://grc.com/dos/grcdos.htm, Mar. 2002.

[8] A. Snoeren *et al.*, "Hash-Based IP Traceback," *Proc. ACM SIGCOMM*, Aug. 2001, pp. 3–14.

[9] S. Savage *et al.*, "Practical Network Support for IP Traceback," *Proc. ACM SIGCOMM*, Aug. 2000, pp. 295–308.

[10] P. Ferguson and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks Which Employ IP Source Address Spoofing," RFC 2827, May 2000.

[11] P. Porras and A. Valdes, "Live Traffic Analysis of TCP/IP Gateways," Proc. Net. and Distrib. Sys. Sec. Symp., Mar. 1998; http://www.sdl.sri.com/projects/emerald/live-traffic.html

[12] K. Park and H. Lee, "On the Effectiveness of Route-Based Packet Filtering for Distributed DoS Attack Prevention in Power-Law Internets," *Proc. ACM SIGCOMM*, Aug. 2001, pp. 15–26.

[13] K. K. Wan and R. Chang, "Engineering of a Global Defense Infrastructure for DDoS Attacks," *Proc. IEEE Int'l. Conf. Net.*, Aug. 2002.

[14] IETF's Intrusion Detection Exchange Format Working Group, http://www.ietf.org/html.charters/ idwg-charter.html, Apr. 2002.

[15] R. Crow and S. Schwartz, "Quickest Detection for Sequential Decentralized Decision Systems," *IEEE Trans. Aerospace and Electronic Systems*, vol. 32, no. 1, Jan. 1996, pp. 267–83.

Biography

ROCKY K. C. CHANG (csrchang@comp.polyu.edu.hk) received his Ph.D. degree in computer systems engineering from Rensselaer Polytechnic Institute (RPI) in 1990. Prior to that, he received his Master's degrees in electrical engineering, and operations research and statistics from RPI in 1985 and 1987, respectively. From 1991 to 1993 he was with IBM Thomas J. Watson Research Center working on performance evaluation methodologies and tools. Since 1993 he has been with the Department of Computing, The Hong Kong Polytechnic University, where he is now an associate professor. He has led research projects on stability analysis of polling models, optimal node placement in WDM networks, interconnection network design, ATM survivability schemes, transport-level proxies, scalable multicast routing, and high-performance TCP. More recently, he has worked on distributed detection of denial-of-service attacks, per-queue stability, on-demand multicasting, peer-to-peer networks, and Bluetooth.

Packet filtering in LAD is very ineffective in the midst of a sufficiently large-scale attack. However, this approach is most deployable among the four because the detection activities are centralized in the victim network or its ISP network.