ENGINEERING OF A GLOBAL DEFENSE INFRASTRUCTURE FOR DDOS ATTACKS[1]

Kalman K. K. Wan and Rocky K. C. Chang

Department of Computing
The Hong Kong Polytechnic University
Hung Hom, Kowloon, Hong Kong

ABSTRACT

Distributed denial-of-service (DDoS) attacks have recently emerged as a major threat to the stability of the Internet. By the very nature of the DDoS attacks, pure preventive and pure reactive approaches are not effective to defend against them. In this paper, we propose a global defense infrastructure to detect-and-respond to the DDoS attacks. This infrastructure consists of a network of distributed local detection systems (LDSes), which detect attacks and respond to them cooperatively. Because of the current Internet topology, this infrastructure can be very effective even if only a small number of major backbone ISPs participate in this infrastructure by installing fully configured LDSes. Moreover, we propose to use traffic volume anomaly for DDoS attack detection. A fully configured LDS monitors the passing traffic for an abnormally high volume of traffic destined to an IP host. A DDoS attack is confirmed if multiple LDSes have detected such anomalies at the same time. Our simulation studies have demonstrated that the proposed detection algorithms are responsive and effective in curbing DDoS attacks.

1 INTRODUCTION

A distributed denial of service (DDoS) attack overwhelms a targeted host with an immense volume of useless traffic from distributed and coordinated attack sources [2]. In February 2000, a number of the world's largest e-commerce sites were brought offline for days by DDoS attacks, even though they were designed to provide high availability services. The outages had caused a huge economic loss to both the victim sites and their users.

Because of the nature of the DDoS attacks, an Internet site has very limited mechanisms to defend against them. To provide a thorough solution to this problem, we argue that DDoS attacks must be effectively detected and handled before the attack packets arrive at the victim site. In this paper, we propose a global defense infrastructure (GDI), which consists of a number of distributed parties to detect and block such attacks cooperatively. Although the underlying idea is very simple, as we will show in the rest of this paper, engineering of such a GDI involves a wide range of technical issues.

1.1 Types of DDoS Attacks

There are various ways of launching a DDoS attack. In this paper, we only consider flooding attacks, which essentially exhaust a system's network bandwidth, buffers, or other resources by an abnormally high volume of IP packets, such as UDP packets (e.g., UDP port attacks [3]), ICMP packets (e.g., Smurf attacks [5]), and TCP SYN packets (SYN flooding attacks [4]). On the other hand, we do not consider attacks, which exploit system and application security flaws, which can be defended effectively by patching the software accordingly. Moreover, we emphasize on DDoS attacks with attacking hosts distributed at different Autonomous Systems (ASes), because cross-AS attacks are particularly difficult to detect and handle.

At least five parties are involved in a DDoS attack: the attacker, a small number of masters, a large number of DDoS attack daemons or zombies, ISPs or transit ASes, and the attack target. Masters are also implanted attack programs, which receive attack instructions from the attacker, and forward them to their daemons. The ISPs or transit ISPs serve as the carriers for the messages communicated among the attacker, masters, and daemons and for the attack packets targeting at the victim.

Ingress filtering and egress filtering are effective ways to block DDoS attacks, which usually use spoofed source IP addresses. However, they are seldom used by ISPs, because the packet filters degrade ISPs' network performance significantly [9]. Moreover, some DDoS attack tools, such as TFN2K and Stacheldraht [7], can bypass ingress filters by spoofing the source addresses to valid neighbor IP addresses.

1.2 Contributions of This Work

Both the preventive and reactive solutions reviewed above are clearly unable to address the DDoS problem. It is infeasible to get rid of implanted DDoS daemons, and there has been no effective solutions to handle DDoS attacks once it is started. Therefore, a natural approach to the problem is to *detect-and-respond* to a DDoS attack quickly enough before a victim is overcome by the attack packets. Since the victim could be any host in the Internet, building and sharing a global defense infrastructure (GDI) is deemed to be most cost- and resource-effective.

Our primary contribution in this paper is to propose a deployable and practical GDI for DDoS attacks. Except for [12], we are not aware of any proposals based on a global infrastructure. In particular, we have identified all the necessary components based on the Common Intrusion Detection Framework (CIDF), and employed various standards to implement them. This infrastructure is very flexible and is not tied to a particular approach of detecting and responding to DDoS attacks.

Our another equally important contribution is to propose a traffic volume anomaly based detection method for DDoS attacks. That is, a set of distributed detection systems monitors traffic rate, and they would report any anomaly to each other when such is detected. Our key premise behind this approach is based on the fact that multiple detection systems, if properly placed in the Internet, should have detected an abnormally high volume of traffic of particular types when a DDoS attack is in

[1] The work described in this paper was partially supported by a grant from the Research Grant Council of the Hong Kong Special Administrative region, China (Project No. PolyU 5080/02E).

0-7803-7533-5/02/$17.00 ©2002 IEEE.

progress. Based on this approach, we have proposed simple algorithms to detect DDoS attacks, and performed simulation studies to demonstrate its effectiveness and responsiveness.

1.3 Related Work

Although DDoS attacks have already become a major threat to the stability of the Internet, there are very few published work that directly address this problem. The only work that employs a global infrastructure is given in [12]. The infrastructure there also consists of a number of distributed packet filters. However, their approach is to filter spoofed packets based on the routing information, i.e., the source and destinations addresses. However, inter-domain routing protocols today do not provide source IP addresses.

IP traceback is another active research area, which is closely related to DDoS attacks, e.g., [14-15]. The problem, roughly speaking, is to identify the source of any data sent across the Internet. Therefore, IP traceback can be used to trace the real sources of DDoS attacks in which spoofed source IP addresses are usually used. Effective and responsive IP traceback mechanisms are difficult to design. Therefore, IP traceback seems to be more useful for later law enforcement, rather than for curbing an ongoing attack. Nevertheless, it should be included in a GDI.

Another area of work concerns the design of distributed intrusion detection systems. For example, EMERALD is designed to address intrusion detection issues associated with large, loosely coupled enterprise networks [13]. It uses a hierarchical approach to provide three levels of analysis performed by a three-tiered system of monitors. NetSTAT aims at real-time network-based intrusion detection in complex networks composed of several sub-networks [16]. It uses a set of probes to identify intrusions, which involve separate sub-networks. Although these intrusion detection systems are not designed specifically for the DDoS problems, many of the issues considered there are actually very similar to those encountered in the engineering of a GDI for DDoS attacks.

We organize the rest of this paper as follows. In section 2, we give an overview of our proposed GDI, and discuss evaluation criteria for a GDI. In section 3, we describe our component-based approach to designing a LDS. In section 4, we detail the overall attack detection and response processes performed by LDSes, and the alert messages involved. In section 5, we mainly describe the traffic volume anomaly detection algorithms and how they are implemented by the LDSes' components. In section 6, we perform computer simulation to evaluate the effectiveness of the GDI. In section 7, we conclude this work with future work.

2 AN OVERVIEW OF THE GDI

2.1 A Network of LDSes

Our proposed GDI is a network of local detection systems (LDSes) placed at various strategic locations in the Internet. A fully configured LDS performs four main functions: detecting suspicious DDoS attacks, communicating detection information with other LDSes, deciding whether there is an ongoing DDoS attack, and responding to a confirmed attack.

When a suspicious attack is detected, the LDS concerned communicates this information with its peer LDSes at other locations. A LDS then determines whether a DDoS attack is in progress based on the local and remote detection information. The premise behind this decision is that multiple LDSes must have detected suspicious DDoS attacks when there is an ongoing DDoS attack.

If a LDS confirms a DDoS attack, it instructs local routers to block the attack traffic, and communicates with the upstream LDSes about the attack. The upstream LDSes then perform similar response actions. The process reiterates until the LDSes closest to the attack sources block the attack traffic.

2.1.1 Two types of local detection systems

Because the cost of continuously monitoring the Internet traffic for suspicious DDoS attacks is very high in terms of CPU, memory, and IO, we propose two types of LDSes: *minimally configured LDS* (MLDS), and *fully configured LDS* (FLDS). A FLDS is designed to perform all the four functions described earlier.

MLDSes, on the other hand, do not perform the resource-demanding operations of detecting for suspicious attacks. However, they are equipped to stop attack traffic and trace the attack sources, when they are informed of a DDoS attack (by the FLDSes). Moreover, they will be activated only when there is a confirmed attack, and they only process packets related to the attack target's address. As a result, their system resources requirements are much lower than that for FDLSes.

2.1.2 Placement of the local detection systems

Because FLDSes require much more system resources, they should be located at strategic locations in the Internet, where most cross-domain traffic will pass through.

Based on the current Internet topology, network access points (NAPs), Internet exchanges (IXs), and backbone ISPs are the prime locations to place the FLDSes. To further validate our conjecture, we have performed *traceroute* from 18 *traceroute* servers to 14 very popular web sites (details of the traces are documented in [17]). For each of the 252 (18 x 14) source-destination pairs, 3 traces were obtained at different times and different days to cater for route instability.

There were altogether 96 ASes appeared in the traces. Backbone ISPs, NAPs, and IXs accounted for 57% of the transit ASes. Moreover, 20% of the traces included at least one NAP or IX. More importantly, only 3 out of 252 traces did not include any backbone ISPs, NAPs and IXs. These results support that most cross-domain Internet traffic can be monitored by installing FLDSes at backbone ISPs, NAPs, and IXs.

Moreover, the five major backbone ISPs appeared in 65% of all the traces. This observation is very important, because it shows that the GDI can be very effective even if only a small number of major backbone ISPs participate in this infrastructure by installing FLDSes.

2.2 Evaluation Criteria for a GDI

We consider two sets of metrics for measuring a GDI's performance of detecting and responding to DDoS attacks: false positive and negative ratios, and normal packet survival ratio. The *false positive ratio* and *false negative ratio*, denoted by Ψ^+ and Ψ^-, respectively, can be measured on the attack level and packet level. The attack-level Ψ^+ measures the % of occurrences of false alarms, and the attack level Ψ^- measures the % of occurrences of undetected attacks.

The packet-level Ψ^+ measures the % of normal packets dropped by a GDI, while the packet-level *false negative ratio* measures the % of attack packets go undetected by the GDI. Denote Ω_i as the set of packets processed by *i*th LDS in a GDI over a period of time. And $\Omega_i = X_{d,i} \cup X_{p,i}$, where $X_{d,i}$ is the set of packets, which are processed and dropped by *i*th LDS, and $X_{p,i}$ is the set of packets, which are processed and passed by *i*th LDS. Moreover, $\Omega_i = Y_{a,i} \cup Y_{n,i}$, where $Y_{a,i}$ is the set of attack packets processed by *i*th LDS, and $Y_{n,i}$ is the set of normal packets processed by *i*th LDS. Therefore,

$$\Psi^+ = \Sigma_{\forall i} |Y_{n,i} \cap X_{d,i}| / \Sigma_{\forall i} |Y_{n,i}| * 100\%, \text{ and}$$

$$\Psi^- = \Sigma_{\forall i} |Y_{a,i} \cap X_{p,i}| / \Sigma_{\forall i} |Y_{a,i}| * 100\%,$$

where $|S|$ gives the number of elements in set S.

Another important performance metric is the *normal packet survival ratio*, denoted by Γ, which measures the % of normal packets, which make their way to an attack target in the midst of a DDoS attack. In addition to the packet dropping at LDSes, normal packets may also be dropped by routers because of the congestion created by attack packets. Denote Z as the total number of normal packets sent to an attack target during a period of time, and Z_d as the number of normal packets, which are dropped (by LDSes and routers) before reaching the target. Therefore,

$$\Gamma = (Z - Z_d)/Z * 100\%.$$

3 A COMPONENT-BASED DESIGN OF LDS

As depicted in Fig. 1, our LDS (FLDS, to be exact) performs DDoS attack detection, analyses, and responses in two main stages. The first stage, which is relatively resource-inexpensive, is to confirm DDoS attacks by analyzing local traffic and suspicious attack alerts received from other LDSes. As soon as an attack is confirmed, the LDS will then perform a more expensive task of detecting the attack interfaces and subsequently installing traffic rate-limiting filters for the attack packets. The LDS will also send attack alerts to the upstream LDSes, so that filtering could be performed more effectively there.

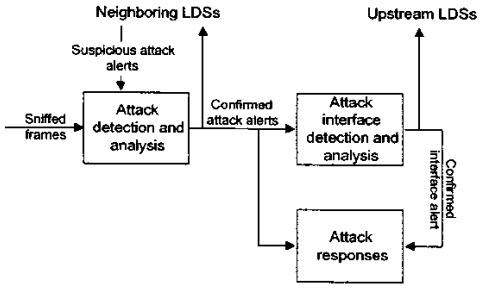

Fig. 1. A two-staged process of DDoS attack detection, analyses, and responses.

3.1 LDS Design Considerations

3.1.1 Attack detection design

Since there are a number of well-known DoS and DDoS attacks, misuse detection (pattern or signature recognition) can be used to detect these attacks effectively [3]. However, as new DDoS tools are still evolving, only misuse detection is not enough to handle new types of attacks. Therefore, anomaly detection is required even though it may be more difficult to design, and is also less effective than misuse detection against well-known attacks. As a result, we employ both misuse detection and anomaly detection for DDoS attacks. For the latter, we use traffic volume anomaly to determine whether the rate of traffic (of a certain packet type) destined for a particular IP host is normal.

3.1.2 Alert analysis design

Suspicious DDoS attack alerts may be generated locally by a FLDS, or by a remote FLDS, as a result of performing misuse detection and anomaly detection. In order to confirm whether a DDoS attack is really in progress, a LDS needs to analyze and consolidate these alerts for the same IP host and for the same attack packet type. For this purpose, a *confidence level* is attached to each alert, which quantifies the amount of evidence supporting the suspected attack. If the confidence level of the consolidated alert is higher than a certain threshold, an attack is confirmed. Moreover, local alerts generated for suspicious attack interfaces are also analyzed and consolidated in a similar way as for the attack alerts.

3.1.3 Attack response design

Packet filtering is the primary reaction taken by LDSes in response to confirmed DDoS attacks. However, packet filtering also degrades routers' performance significantly, especially during an ongoing DDoS attack. Therefore, our approach is to install packet filters on demand and for a short duration. As soon as a LDS confirms an attack, packet filtering is performed at *all* switch interfaces. However, the LDS immediately starts looking for attack interfaces, where a significant amount of attack packets comes from. Subsequently, packet filtering is performed only at the attack interfaces. Moreover, upstream LDSes are informed about the attack, so that attack packets can be filtered more effectively there.

3.1.4 Communication design

A suspicious attack alert, which is generated by any FLDS, is propagated to all other LDSes in the GDI through a reliable flooding mechanism. The LDSes, therefore, must always be connected, and each LDS is configured to peer with one or more neighboring LDSes. Neighboring LDSes send hello messages to each other periodically to maintain their neighboring relationship. There are also other issues to consider, such as re-connecting a partitioned network of LDSes, but they will not be discussed further here due to the lack of space.

We adopt the Intrusion Detection Message Exchange Format (IDMEF) as the common language for communicating intrusion detection information for both intra-LDS and inter-LDS communications [8]. Moreover, all messages must be authenticated and encrypted. For this purpose, we adopt the Intrusion Detection Exchange Protocol (IDXP) for transporting the alert messages [10].

3.2 The LDS components

We adopt the Common Intrusion Detection Framework (CIDF) to design the components for our LDSes [6]. The current CIDF has proposed four components for general intrusion detection purposes: E-, A-, D-, and R-boxes. Our LDS also

includes the four components, but their functions are not exactly the same as that in CIDF, because the four boxes are not designed specifically for DDoS attacks. Furthermore, we have proposed another four new components: S-, P-, M-, and C-boxes, which are needed for detecting DDoS attacks.

Attack detection components include traffic distributors (S-boxes), event generators (E-boxes), and packet capturers (P-boxes). A-box is an alert analysis component, and R-box is a response component. M-box is an intra-LDS communication component. Finally, system management components include D-boxes (databases), and C-boxes (consoles).

In Fig. 1, we show our proposed network structure for a FLDS. The switch mirrors ingress traffic received from each switch interface to S-box, which is connected to one or more traffic analysis networks, depending on the volume of traffic to be handled. Each traffic analysis network hosts a set of E- and P-boxes for attack detection. Moreover, all the E- and P-boxes are connected to other boxes on an intra-box communication network. Note that R-box has another interface connecting to the switch for dynamic packet filter installations in response to confirmed DDoS attacks. Moreover, M-box is the only component that has access to the Internet. The M-box, however, will not forward packets other than those received from the neighboring LDSes, so that the internal FLDS network is safely guarded from attacks.

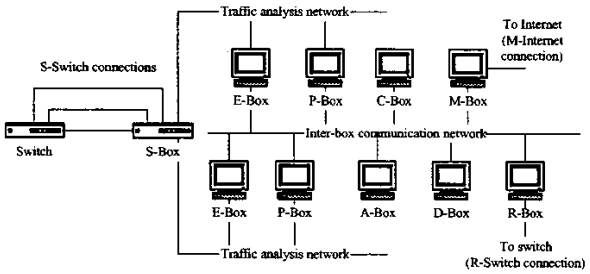

Fig. 2. A FLDS with two traffic analysis networks.

4 ALERT TYPES AND DETECTION PROCESSES

4.1.1 Attack detection, analysis, and response

Alert messages are used to communicate either suspicious or confirmed DDoS attacks, and they are also used to trigger responses to curb DDoS attacks.

Suspicious attack alert and *confirmed attack alert* messages are used for DDoS attacks detection and analysis, and for triggering attack responses. Here we concentrate mainly on the alerts generated by P-boxes (volume anomalies). Each suspicious attack alert or confirmed attack alert is triggered for an attack target and an attack packet type. Therefore, each alert message specifies the attack target's IP address, the attack packet type, and a confidence level of this alert. The confidence level ranges from 0 to 100. The special value of 100 represents a full confidence, and it is reserved to identify the *confirmed* attack alerts and *confirmed* interface alerts. The suspicious attack alert also specifies the measured traffic rate of the attack packets, so that R-box could limit the attack traffic rate.

Fig. 4 shows how a FLDS arrives at the conclusion of a DDoS attack.

1. S-box receives mirrored frames passing through the switch of a network node (IX or backbone ISP node) and forwards them to E- and P-boxes for traffic analysis.
2. If the E-box finds suspicious DDoS attack control packets, it generates local suspicious attack alerts and sends them to A-box. Likewise, if the P-box finds suspicious traffic volume anomalies, it performs the same actions as the E-box.
3. The A-box consolidates the local suspicious attack alerts (indicated by a shaded node) and sends a copy of the consolidated alert to the neighboring LDSes via the M-box.
4. The M-box, on the other hand, may receive suspicious attack alerts from other LDSes, which are forwarded to the A-box. The M-box also further propagates the alerts to the neighboring LDSes.
5. Based on the consolidated local suspicious attack alert and remote suspicious attack alerts, the A-box may conclude that there is really an on-going DDoS attack. In this case, it generates a confirmed attack alert and sends a copy each to the R- and P-boxes.
6. Recipient of a upstream interface alert from a downstream LDS also triggers the same event as in item 5 (the upstream interface alert will be discussed in section 4.1.2).
7. Upon receiving a confirmed attack alert, the R-box installs traffic rate-limiting filters for the target on the switch or routers, which are connected to the switch. The P-box, after receiving the confirmed attack alert, begins to monitor for suspicious attack interface.

The corresponding diagram for an MLDS is a subset of Fig. 3. That is, it only consists of S-, P-, M-, and R-boxes, and items (1), (6), and (7).

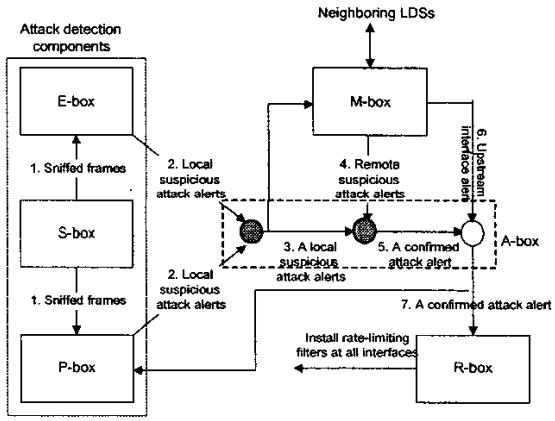

Fig. 3. The process of confirming a DDoS attack by a FLDS.

4.1.2 Attack interface detection, analysis, and response

Another four alert messages are used for attack interfaces detection and analysis, and for triggering attack responses: (suspicious, confirmed, upstream, and cancelled) interface alerts. We again concentrate mainly on the alerts generated by P-boxes (volume anomalies). Each suspicious interface alert or confirmed interface alert is triggered for an attack target, an attack packet type, and a switch interface. Therefore, each alert message

contains the attack target's IP address, the attack packet type, the identity of the switch interface, traffic rate of the attack packets, and a confidence level of this alert.

Fig. 4 shows the process of detecting attack interfaces.

1. P-box's interface monitoring process is triggered by the receipt of a confirmed attack alert.
2. If it finds traffic volume anomalies for the target and for the attack packet type from some interfaces, it sends suspicious interface alerts to A-box.
3. After consolidating all the suspicious interface alerts, the A-box may generate a confirmed interface alert and send it to R-box.
4. The R-box installs traffic rate-limiting filters at the confirmed interfaces for the attack packets.
5. The R-box, in response to a confirmed interface alert, also sends upstream interface alerts to upstream LDSes, so that packet filters could be activated there too.
6. If the A-box does not confirm a suspicious attack interface, it then sends a cancelled response alert to the P-boxes concerned for corrective actions.

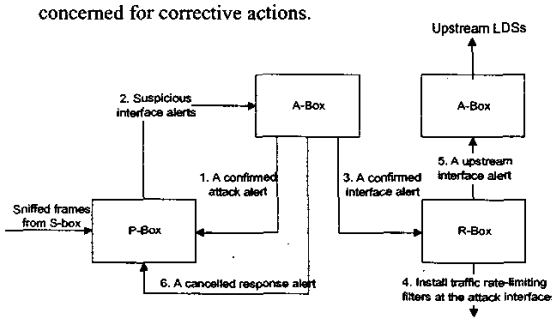

Fig. 4. The process of confirming attack interfaces by a LDS.

For the purpose of clarity, the above discussion leaves out D-box, and Hello messages. All alert messages, in fact, must be sent to D-box for logging. Each component must send regular Hellos to A-box for keepalive purposes and to D-box for status logging. Moreover, A-box sends Hello messages periodically to the neighboring LDSes via the M-box to ensure the connectedness of the LDS network.

5 LDS COMPONENT DESIGN

Owing to the limited space, we only discuss the design of P- and A-boxes in this section, which are key components in a LDS. Interested readers may refer to [17] for the detail design of other components.

5.1 Traffic Volume Anomaly Detection (P-Box Design)

5.1.1 Detecting suspicious DDoS attacks

To detect traffic volume anomalies for suspicious DDoS attacks, P-box counts the number of packets destined to IP addresses for a period of T_A, an *attack detection interval*. The length of a T_A should be long enough to avoid false alarms caused by bursty traffic. However, it cannot be too long; otherwise, the system would not be responsive enough. After taking into these considerations and others, we propose the length of a T_A to be one minute.

To implement this efficiently, a P-box may maintain a very large hash table, consisting of, say, 1 million slots, and the hashing function takes the destination IP addresses as inputs. To reduce the memory requirement, multiple IP addresses may be hashed to the same slot. Therefore, the packet count is an aggregated count for a number of IP destinations.

For each slot, P-box counts five types of packets destined to the IP addresses in the slot during each T_A: ICMP, UDP, TCP data, TCP SYN segments, and TCP RST segments. The TCP RST segments are included in the analysis to cater for the case that DDoS daemons may attack a target using TCP packets, and spoofing source addresses to valid neighbor IP addresses (or not using spoofed addresses at all). Under such attacks, the target will send back an abnormally high volume of TCP RST segments.

At the end of ith T_A, or $T_A(i)$, P-box updates two counters for slot j for each packet type k: $cur\_count(i, j, k)$ and $max\_count(i, j, k)$. The former counts the total number of packets of type k received for slot j during $T_A(i)$, while the latter records a maximal number of packets of type k received for slot j from the beginning and up to $T_A(i)$ inclusive. To keep the notations simple, we ignore the indices j and k in *cur_count* and *max_count* with the understanding that the counts concern a particular packet type and destined to a particular slot in the hash table.

Based on *cur_count*, *max_count*, and a few other parameters to be explained shortly, we design an algorithm to detect traffic volume anomalies, which is detailed in Algorithm 1. The algorithm is based on the following simple idea. At the end of $T_A(i)$, $cur\_count(i)$ and $max\_count(i-1)$ are compared. If $cur\_count(i) < max\_count(i-1)$, traffic volume anomalies do not exist. Otherwise, if $cur\_count(i)$ is larger than $max\_count(i-1)$, but not so significantly, update $max\_count(i)$ by setting it equal to $cur\_count(i)$. As a final case, if $cur\_count(i)$ is *significantly* larger than $max\_count(i-1)$, a suspicious attack alert may be generated, and in this case P-box needs to identify the attack target's IP address (item (a) in the else-statement) and send the alert to A- and D-boxes (item (b) in the else-statement).

At the end of $T_A(i)$, update $cur\_count(i)$ and perform:
if ($cur\_count(i) < max\_count(i-1)$)
  $max\_count(i) = max\_count(i-1) - 1$;
else if ($max\_count(i-1) \le cur\_count(i) < M_{sa} * max\_count(i-1)$
    OR $cur\_count(i) < H_{sa}$)
  $max\_count(i) = cur\_count(i)$;
else
  a) Identify the suspicious attack target's IP address corresponding to this hash slot by the highest traffic rate measured in the coming short period (e.g., 5 seconds).
  b) Send suspicious attack alerts to A- and D-boxes, with the target's IP address, the attack packet type, and a confidence level given by $F(cur\_count(i), max\_count(i-1))$. For example, $F = \min\{cur\_count(i)/max\_count(i-1)/M_{sa} * 10, 99\}$.
  c) If a cancelled interface alert is received from A-box within the last interface detection interval, i.e., no interface traffic volume anomalies have been detected, $max\_count(i) = 7/8 * max\_count(i-1) + 1/8 * cur\_count(i)$.
endif

Algorithm 1. A traffic volume anomaly detection algorithm.

To determine whether the difference between *cur_count(i)* and *max_count(i–1)* is significant, we use two other parameters, *suspicious attack multiplier* (M_{sa}) and *suspicious attack threshold* (H_{sa}). Our approach is to use a relative alert threshold as well as an absolute alert threshold to determine whether P-box should generate an alert. M_a is a dimensionless real number greater than 1, and M_{sa} * *max_count* serves as a relative alert threshold. H_{sa}, in terms of the number of packets, serves as an absolute alert threshold. Note that the alert thresholds are computed based on the traffic measurement for a particular packet type and for a particular hash table slot. Moreover, they should be set to balance between Ψ^+ and Ψ^-.

The volume anomaly detection algorithm is simple and feasible for processing a huge traffic volume. Moreover, it is adaptive to different traffic volumes destined to different IP addresses, because the peak traffic rates are recorded periodically for computing the alert thresholds. Using the past peak traffic rate to compute the alert thresholds aligns with our objective to minimize service interruption of the attack target, because the past peak traffic is a good indicator for the maximum capacity of the target.

Besides updating *max_count* to *cur_count* (under the else-if statement), *max_count* is decremented (under the if-statement) in order to age the state. Other types of aging functions may be used. Moreover, if there are no interface traffic volume anomalies found for a confirmed attack (under (b) in the else-statement), *max_count* will be revised upward. This is to cater for web sites, which are not high-volume sites on a continuing basis, but have to handle unprecedented load levels for certain periods of time (e.g., NASA site during the Mars landing).

5.1.2 Detecting suspicious attack interfaces for confirmed attacks

When P-box receives a confirmed attack alert against a particular target from A-box, it starts to find out suspicious switch interfaces that have received an abnormally high traffic volume destined at or originated from the target's IP address. For each attack target and each switch interface, P-box maintains four packet counters for each packet type: ICMP, UDP, TCP data, and TCP SYN (TCP RST will be counted in all cases). Over a period of T_I, *an interface detection interval*, which is set to 5 seconds,

1. $n_d(d, k)$ records the total number of ingress packets of type k, which are destined for IP address d, and egress TCP RST packets originated from d.
2. $n_i(f, k)$ records the total number of ingress packets of type k received from interface f and egress TCP RST packets delivered to interface f.
3. $n_{d,i}(d, f, k)$ records the total number of ingress packets of type k received from interface f, which are destined to IP address d, and exgress TCP RST packets delivered to interface f, which are originated from IP address d.
4. $n_t(k)$ records the total number of ingress packets of type k, and egress TCP RST packets received by the switch.

When S-box distributes traffic onto different traffic analysis networks, it preserves the data-link header information. Therefore, the source and destination data-link addresses can be used to determine the egress and ingress switch interfaces, respectively.

Based on the four parameters measured over a period of T_I, P-box computes for the target with IP address d a *victim-skew ratio* for switch interface f and packet type k, denoted by $r_k(d, f, k)$.

$$r_k(d,f,k) = \left[\frac{n_{d,i}(d,f,k)}{n_d(d,k)}\right] \Big/ \left[\frac{n_i(f,k) - n_{d,i}(d,f,k)}{n_t(k) - n_d(d,k)}\right].$$

From the above definition, a large value of $r_k(d, f, k)$ indicates that the switch interface concerned has received an abnormally high volume of traffic destined to the target and/or TCP RST sent from the target relative to the rest of the traffic. The interface, therefore, is suspected to be a source of the DDoS attack traffic.

At the end of a T_I, P-box will generate a suspicious interface alert for (d, f, k) if $r_k(d, f, k) \geq H_{si}$, a *suspicious interface threshold*, which depends on the attack packet types, and the line speed of the switch. This alert message also includes a confidence level and the corresponding traffic rate. The confidence level is computed by a function $F(r_k)$, which should be an increasing function of r_k. For example, $F(r_k) = r_k/H_{si}$ * 10. A more accurate formulation should be based on a statistical analysis on the traffic in the IX and backbone ISP networks. The traffic rate for packet type k is given by $n_{d,i}(d, f, k)/T_I$.

The confirmed attack alerts received from A-box are valid for a duration of two T_As. Thus, if no more alerts are received from A-box for this period, P-box stops the interface monitoring activities described in this section.

5.2 Attack Alert Analysis (the A-Box Design)

5.2.1 Alert analysis for suspicious DDoS attacks

A-box determines whether a DDoS attack is in process based on the suspicious attack alerts received from E- and P-boxes at the end of each T_A, and suspicious attack alerts from other FLDSes. We again consider only the traffic volume anomaly detection, for which A-box performs the following tasks:

1. At the end of each T_A, consolidate local alerts, if any, received from P-boxes during the interval into a single alert, which is then sent to D-box for logging and to other LDSes via M-box. This consolidated alert, unlike the local alerts, does not specify the attack packet type.
2. If remote suspicious attack alerts have also been received from other LDSes via M-box during a T_A, consolidate the already consolidated local alert, if any, with the remote alerts into a single alert at the end of this T_A to determine whether there is a DDoS attack.
3. After confirming a DDoS attack, inform D- and R-boxes for the purposes of logging and taking appropriate response actions.

The approaches for consolidating multiple alerts into one alert for cases (1) and (2) above are very similar. For example, if A-box receives two local alerts for the same destination IP address but with different packet types locally, it generates a new alert for the same destination IP address with no packet type, and with a new confidence level computed as

$$\eta_C(d) = \min\{\Theta(\eta_1(d), \eta_2(d)), 99\},$$

where

- $\eta_C(d)$ is the confidence level for the consolidated alert for IP address d.
- $\eta_1(d)$ and $\eta_2(d)$ are the confidence levels specified in the two local alerts.
- $\Theta()$ is a function for computing the confidence level for the combined alert.

The main reason for upper bounding the confidence level for the consolidated local alert is to ensure that alerts from only one

or a few FLDSes are not conclusive enough to confirm an attack, even though they may be of very high confidence levels.

As for the case of combining a local alert and a remote alert for the same IP address, A-box generates a new alert with the same destination IP address and a new confidence level computed as

$$\eta_C(d) = \min\{\Theta(\eta_L(d), \eta_R(d)\},$$

where $\eta_L(d)$ and $\eta_R(d)$ are the confidence levels specified in the local alert and the remote alert, respectively. Unlike the previous case, the confidence level is not upper bounded, because the resulting confidence level will be used to determine the existence of a DDoS attack.

There is no doubt that the choice of $\Theta()$ is crucial for determining the GDI's responsiveness to DDoS attacks. However, our focus here is on the overall infrastructure design; therefore, we simply use an unweighted sum for $\Theta()$:

$$\Theta(\eta_1(d), \eta_2(d)) = \eta_1(d) + \eta_2(d).$$

Given a suspicious attack alert consolidated after each T_A, A-box confirms that there is a DDoS attack against an IP address d if

$$\eta_C(d) \geq H_{ac} * N_{FLDS} * H_{acp},$$

where

- *Attack confidence threshold* (H_{ac}): A consolidated local alert is considered high when its confidence level is at least H_{ac}.
- N_{FLDS} is the total number of FLDSes in the GDI.
- *Attack coverage percentage threshold* (H_{acp}): When there are at least $N_{FLDS} * H_{acp}$ FLDSes generating suspicious attacks against the same target with high confidence levels, a DDoS is confirmed.

Immediately after the confirmation, A-box sends a confirmed attack alert to R-box for response action. The confirmed attack alert contains the target's IP address, a confidence level of 100, and the traffic rates of the attack packets. Besides, A-box sends a confirmed attack alert to R-box for response action if it receives an upstream interface alert from a downstream LDS.

5.2.2 Alert analysis for suspicious interfaces

When an attack is confirmed, A-box also sends confirmed attack alerts to all local P-boxes, which in turn trigger the attack interface detection process, which has already been discussed in section 5.1.2. At the end of the process, P-box may send a suspicious interface alert to A-box if the target's victim-skew ratio is high enough for some switch interfaces. A-box then consolidates these alerts to find out whether these interfaces are really the attack interfaces by computing a confidence level for the consolidated alert for each interface f as

$$\sigma_C(d, f, i) = 0.5 * \sum_{\forall \text{ P-boxes}} \sigma(d, f, i) + 0.5 * \sigma_C(d, f, i-1),$$

where

- $\sigma_C(d, f, i)$: the confidence level for a consolidated interface alert for interface f received during ith T_I.
- $\sigma(d, f, i)$: the confidence level for a consolidated interface alert for interface f received from a P-box during ith T_I.

There are two reasons as why the confidence levels for different alerts need to be combined to confirm attack interfaces. First, a P-box may generate multiple suspicious interface alerts for the same target and attack interface, because the attack involves different types of attack packets. Second, TCP RST segments generated from the target and delivered to the same interface are generally processed by different P-boxes.

Moreover, by including the confidence level in the last T_I, the computed confidence level avoids false alarms caused by a sudden burst of normal traffic.

A-box generates a confirmed interface alert if $\sigma_C(d, f, i) \geq H_{ic}$, an *interface confidence threshold*, for a target and a switch interface, and it is sent to D-box for recording and R-box for response. In addition to the target's IP address and switch interface, the message also includes the packet type, for which the corresponding traffic rate contributes at least 25% of the aggregated confidence level, and its traffic rate.

On the other hand, A-box sends cancelled response alerts to P-boxes if $\sigma_C(d, f, i) < H_{ic}$. The P-boxes then adjust their *max_count*s upward, if necessary, as shown in Algorithm 1.

6 PERFORMANCE EVALUATION

6.1 Internet Topology and Traffic Generation

6.1.1 Generating an Internet-like topology

We use the Network Simulator version 2 (ns-2) [11] to simulate an Internet-like network, which consists of 320 network nodes on average. Each network node represents any one of the followings: local ISPs, POPs of backbone ISPs, and gateway nodes of backbone ISPs. Each local ISP node or backbone node may be an end point of IP traffic, and it may consist of normal hosts, sites, and DDoS daemons.

We use the GT-ITM Topology Generator to generate the Internet topology. GT-ITM can generate a random transit-stub graph based on a number of input parameters. The transit-stub graph model of GT-ITM closely resembles the topology that we want to generate. Each graph consists of stub domains (local ISPs) connected to transit domains (backbone ISPs), and different transit domains are also connected by transit-to-transit links.

Our simulated network size is approximately one sixth of the number of regions and one sixth of the number of local ISPs per region in the Internet. Simulation of this size and complexity should be able to evaluate the GDI's effectiveness, while the processing and memory resource requirements of the simulation are acceptable. In Table 1, we summarize the parameters used for the simulated network and those for the Internet.

Furthermore, we assign a higher bandwidth value to links between transit nodes or backbone nodes (655 Mbps) and a lower bandwidth value to links between transit nodes and stub nodes (3 Mbps for normal stub nodes and 5 Mbps for popular stub nodes). Moreover, we select the backbone nodes, which have more interfaces to other backbone nodes, to be the gateway nodes, to which we attach the LDSes.

6.1.2 Normal traffic generators

All the network nodes in the simulation are assumed to be both IP traffic generators and receivers. We attach UDP agents to network nodes, such that each agent sends normal IP packets to another node with a probability of 0.04. Moreover, we select 8 special nodes to be popular network nodes (web sites), and other nodes send normal IP packets to each of these special nodes with a probability of 0.2.

The traffic source distribution is exponential on/off with burst time (on period) equal to 5 seconds and idle time (off period) equal to 40 seconds. The transmission rate is 50 kbps and the packet size is 500 bytes.

Topology generation parameters	Internet	Simulation
1. Average number of transit domains	≈ 50	8
2. Approximate number of regions	≈ 150	25
3. Average number of transit nodes per transit domain	≈ 30	8
4. Average number of local ISPs per region	≈ 60	10
5. Average number of backbone ISPs with POPs in any region	≈ 10	3
6. Average number of stub domains (local ISPs) per transit node	≈ 6	4
7. Average number of stub nodes per stub domain	-	1
8. Average number of extra transit-stub links	≈ 900	25
9. Average number of extra stub-stub links	-	0
10. Probability of having a link between two transit domains	-	0.3
11. Probability of having a link between two transit nodes in a domain	-	0.3
12. Probability of having a link between two stub nodes in a domain	-	Not applicable

Table 1. Internet topology generation parameters.

6.1.3 DDoS attack daemons

A number of network nodes are selected to be daemon nodes, which simulate ISPs with clients that have DDoS attack daemons implanted. Unlike the normal traffic generator, the IP traffic generated from all daemon agents has the same destination, the attack target. Moreover, its transmission rate is much higher than that of a normal traffic generator.

Each daemon generates attack traffic with a constant bit rate. The transmission rate is 1.5 Mbps and the packet size is 500 bytes. The total number of daemon nodes is between 8 and 80.

6.2 LDSes Modeling and Placement

In our simulation, we select 18 network nodes to be FLDS nodes and another 18 nodes to be MLDS nodes with the parameters specified in Table 2.

FLDS and MLDS parameters	Recommended values	Values used in simulations
1. T_A	60 seconds	2 seconds
2. T_I	5 seconds	0.2 seconds
3. M_{sa}	Depend on traffic analysis	4
4. H_{sa}	Depend on traffic analysis	200 packets
5. H_{ac}	30	35
6. H_{acp}	10%	20%
7. H_{ac}*N_{FLDS}*H_{acp}	Depend on N_{FLDS}	126 (35*18*20%)
8. H_{si}	2	2

Table 2. FLDS and MLDS parameters.

In the FLDS and MLDS modules, we have modeled the P-, A-, and R-boxes. Other boxes are not included, because they not directly related to the attack detection and response.

The detection intervals and response intervals used in the simulation are made shorter than that proposed earlier to cater for a smaller network size used in the simulation. As a result, the time taken to detect a DDoS attack should be measured in terms of the number of detection intervals required, rather than the absolute simulated time.

The attack confidence thresholds used by A-box in the simulation are also higher. Since the number of FLDS nodes and network nodes in the simulated Internet are fewer than that in the Internet, statistical deviations of the measured performance in the simulation are expected to be higher than in the actual case. Therefore, we pick a less sensitive threshold in the simulation.

6.3 Simulation Results

Altogether we have performed simulations for ten scenarios, each of which has a different number of daemon nodes. Each simulation lasts for 20 seconds of simulation time. Normal traffic begins to transmit at the start of the simulation, and DDoS daemons begin the attack randomly between 5 seconds and 6 seconds. Moreover, three independent simulations are run for each scenario with different locations of the attack target. The simulation results presented here, therefore, are the averages of three simulation results.

We first highlight several findings from the simulation results and then discuss some of them in details.

1. All DDoS attacks can be detected by the GDI, i.e., the attack-level Ψ^- is 0%.
2. No confirmed attack alerts are triggered against nonattack targets, i.e., the attack-level Ψ^+ is 0%.
3. Each LDS drops approximately 40% of the attack packets, i.e., the packet-level Ψ^- is approximately 60%.
4. Less than 1% of normal packets are dropped when there is a DDoS attack, i.e., the packet-level Ψ^+ is less than 1%.
5. Γ, the normal packet survival ratio, is increased by 60% when the GDI is in place and 5% to 12.5% of network nodes are implanted with DDoS daemons.
6. Over 85% of attack packets are dropped by the LDSes before they arrive at the attack target when there is a large scale DDoS attack.

6.3.1 Packet-level false negative and positive ratios

As shown in Table 3, the packet-level Ψ^+s are below 1% in all scenarios. Moreover, when there is no DDoS attack, no normal packets are dropped mistakenly. It further confirms that the GDI can effectively detect an attack and at the same time avoid false alarms.

% of nodes with daemons	Ψ^+	Ψ^-
0%	0.00%	0.00%
2.5%	0.32%	81.43%
5.0%	0.62%	69.19%
7.5%	0.51%	65.24%
10.0%	0.56%	62.53%
12.5%	0.52%	67.56%
15.0%	0.70%	62.12%
17.5%	0.62%	58.51%
20.0%	0.77%	56.76%
25.0%	0.60%	57.75%

Table 3. Simulation results for packet-level Ψ^+ and Ψ^-.

The Ψs, on the other hand, are relatively high. One main reason is due to the low number of transit-transit links per LDS node in the simulated Internet. Consequently, a LDS cannot precisely identify the attack interfaces and block them accordingly. We will further elaborate on this in the next section.

6.3.2 Normal packet survival ratio

Even though most normal packets are not dropped by the LDSes, they may be dropped by routers, which are congested by the attack packets. However, packets normally will not be dropped at the backbone, because the transit-transit bandwidth is much larger than the transit-stub bandwidth. Therefore, most normal packets, which fail to reach the attack target, are likely to be dropped at the links connected to the destination network, where the target resides. The simulation results in Table 4 thus show that when the daemon coverage is between 5% and 12.5% inclusively, the LDSes drop a significant portion of the attack packets before they arrive at the attack target. As a result, the congestion at the links connected to the destination network can be relieved and the attack target can receive more normal packets, resulting in a higher value of Γ.

When there is a very small-scale DDoS attack, e.g., the daemon coverage is 2.5%, the detection results derived from traffic volume anomalies are not very conclusive. That is, the detection parameters used by the LDSes are not sensitive enough to block much attack packets under such circumstances. The packet-level Ψs are therefore very high, as shown in Table 3. Consequently, Γ does not increase significantly even when the GDI is in place.

When there is a very large-scale DDoS attack, e.g., the daemon coverage is 15% or above, there is also no significant increase in Γ when the GDI is used. Under this scenario, there are many daemons scattered at different nodes in the network, and attack packets actually come from all directions. As a result, the LDSes cannot effectively identify the attack interfaces; therefore, they can only block a certain percentage of all the traffic sent to the attack target. As a result, the number of both the normal and attack packets arrived at the target decreases drastically.

	Without the GDI		With the GDI	
% of nodes with daemons	$Z - Z_d$	Γ	$Z - Z_d$	Γ
0%	9096	100%	8895	100%
2.5%	3604	39.80%	3687	40.98%
5.0%	1478	16.32%	2564	28.51%
7.5%	1133	12.60%	1812	20.14%
10.0%	974	10.80%	1352	15.11%
12.5%	676	7.55%	1142	12.82%
15.0%	632	7.04%	801	8.91%
17.5%	462	5.17%	700	7.78%
20.0%	437	4.86%	473	5.28%
25.0%	345	3.84%	412	4.63%

Table 4. Simulation results for Γ.

7 CONCLUSIONS AND FUTURE WORKS

The GDI proposed in this paper serves as a good starting point towards a working system to effectively defend against DDoS attacks. Future work is required in many directions. The traffic volume anomalies based detection algorithms require further attention in terms of analysis, algorithm refinement, and parameter tuning. New scenarios of traffic volume patterns in the Internet need to be identified, analyzed, and handled. For example, online stock brokerage Web sites are required to accommodate a very sudden increase of requests, e.g., when the stock market opens. In this case, a confirmed attack alert may be mistakenly triggered. Although our attack detection algorithm will eventually adapt to this traffic pattern, a better approach would be to avoid the false alarms in the first place. This may be achieved by using a more sophisticated alert triggering method, rather than a simple threshold-based approach.

REFERENCES

[1] J. Balasubramaniyan, J. Garcia-Fernandez, D. Isacoff, E. Spafford, and D. Zamboni, "An Architecture for Intrusion Detection Using Autonomous Agents," *Proc. IEEE 14th Annual Computer Security Applications Conference*, Dec. 1998.

[2] CERT Coordination Center. Internet Denial of Service Attacks and the Federal Response, Feb. 2000. http://www.cert.org/congressional_ testimony/Fithen_testimony_Feb29.html.

[3] CERT Coordination Center. CERT Advisory CA-96.01 UDP Port Denial-of-Service Attack, Feb. 1996. http://www.cert.org/advisories/CA-1996-01.html.

[4] CERT Coordination Center. CERT Advisory CA-96.21 TCP SYN Flooding and IP Spoofing Attacks, Sept. 1996. http://www.cert.org/advisories/CA-1996-21.html.

[5] CERT Coordination Center. CERT Advisory CA-98.01 Smurf IP Denial-of-Service Attacks, Mar. 2000. http://www.cert.org/advisories/CA-1998-01.html.

[6] The Common Intrusion Detection Framework Architecture http://www.isi.edu/gost/cidf/drafts/architecture.txt.

[7] P. Criscuolo, "Distributed Denial of Service – Trin00, Tribe Flood Network, Tribe Flood Network 2000, and Stacheldraht CIAC-2319," Feb. 2000. http://ciac.llnl.gov/ciac/documents/CIAC-2319_Distributed_ Denial_of_ Service.pdf.

[8] D. Curry and H. Debar, "Intrusion Detection Message Exchange Format Data Model and Extensible Markup Language (XML) Document Type Definition," *IETF Internet Draft* draft-ietf-idwg-idmef-xml-06.txt, Dec 28, 2001.

[9] X. Geng and A. Whinston, "Defeating Distributed Denial of Service Attacks," *IT Professional*, Jul./Aug. 2000.

[10] B. Feinstein, G. Matthews, and J. White, "The Intrusion Detection *IETF Internet Draft* draft-ietf-idwg-beep-idxp-04, Jan. 2002.

[11] The Network Simulator-ns-2. http:// www.isi.edu/nsnam/ns/.

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

[13] P. Porras and P. Neumann, "EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances," *Proc. Natl. Information Systems Security Conference*, Oct. 1997.

[14] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical Network Support for IP Traceback," *Proc. ACM SIGCOMM*, Aug. 2000.

[15] A. Snoeren, C. Partridge, L. Sanchez, C. Jones, F. Tchakountio, S. Kent, and W. Strayer, "Hash-Based IP Traceback," *Proc. ACM SIGCOMM*, 2001.

[16] G. Vigna and R. Kemmerer, "NetSTAT: A Network-based Intrusion Detection Approach," *Proc. IEEE 14th Annual Computer Security Applications Conference*, Dec. 1998.

[17] K. Wan. *An Infrastructure to Defend Against Distributed Denial of Service Attack*, MSc Thesis, The Hong Kong Polytechnic University, June 2001. http://www4.comp.polyu.edu.hk/~csrchang/MSc/Kalman.pdf.