Attacking DDoS at the Source*

Jelena Mirković Gregory Prier Peter Reiher
University of California Los Angeles
Computer Science Department
3564 Boelter Hall
Los Angeles, CA 90095, USA

Abstract

Distributed denial-of-service (DDoS) attacks present an Internet-wide threat. We propose D-WARD, a DDoS defense system deployed at source-end networks that autonomously detects and stops attacks originating from these networks. Attacks are detected by the constant monitoring of two-way traffic flows between the network and the rest of the Internet and periodic comparison with normal flow models. Mismatching flows are rate-limited in proportion to their aggressiveness. D-WARD offers good service to legitimate traffic even during an attack, while effectively reducing DDoS traffic to a negligible level. A prototype of the system has been built in a Linux router. We show its effectiveness in various attack scenarios, discuss motivations for deployment, and describe associated costs.

1. Introduction

Distributed denial-of-service attacks are comprised of packet streams from disparate sources. These streams converge on the victim, consuming some key resource and rendering it unavailable to legitimate clients. The cooperation of distributed machines that generate attack flows makes traceback and mitigation very challenging. Some defense mechanisms concentrate on detecting the attack close to the victim machine, characterizing it and filtering out the attack packets. While the detection accuracy of these mechanisms is high, the traffic is usually so aggregated that it is difficult to distinguish legitimate packets from attack packets. More importantly, the attack volume can be larger than the system can handle. Several distributed DDoS defense systems have been proposed that cooperate among core routers to suppress attack streams. These routers are augmented to monitor traffic and grant requests for rate-limiting or filtering of the streams they deliver to their peers. Even when partially deployed, these mechanisms can significantly reduce attack volume and lessen impact on the victim. Unfortunately, core routers can spare only limited resources for attack detection and response, and therefore legitimate flows are often marked as suspicious and suffer collateral damage. The required cooperation of routers is hard to achieve due to distributed Internet management, and securing and authenticating this communication incurs high cost.

Ideally, DDoS attacks should be stopped as close to the sources as possible. In this paper we propose a DDoS defense system called D-WARD that is deployed at the source-end networks (stub networks or ISP networks) and prevents the machines from participating in DDoS attacks. D-WARD is configured with a set of addresses whose outgoing traffic should be policed (its *police address set*), and monitors two-way traffic between the police address set and the rest of the Internet. Online traffic statistics are compared to predefined models of normal traffic, and non-complying flows are rate-limited. The imposed rate limit is dynamically adjusted as flow behavior changes, facilitating fast recovery of misclassified legitimate flows while severely limiting ill-behaved aggressive flows that are likely part of an attack. D-WARD strives to guarantee good service to legitimate traffic by profiling individual connections and serving those that are classified as good, regardless of the imposed rate limit.

The D-WARD approach requires that many routers at different network entry points each independently run the system, since each D-WARD router only polices data flows originating from its own network. However, incremental deployment brings incremental benefit, since each deployed D-WARD router reduces the number of effective attack machines available on the Internet. The major challenge to a D-WARD deployment is incentive, since the direct benefit of the system is felt by the victim, not by the deploying network.

Section 2 of this paper describes the proposed D-WARD system, specifically its monitoring, detection and response

*This work is funded by DARPA under contract number N66001-01-1-8937. The authors can be contacted at {sunshine, greg, reiher}@cs.ucla.edu.

strategy. Section 3 describes the implementation of the system in a Linux router, presents several experiments, and discusses the performance results and deployment cost. Section 4 investigates security issues, and Section 5 discusses motivation for deployment. Section 6 gives an overview of related work. Section 7 discusses future work, and Section 8 concludes the paper.

2. D-WARD

Placing DDoS defenses close to the sources of the attack has many advantages. The attack flows can be stopped before they enter the Internet core and blend with other flows, thereby creating possible congestion. Being close to the source can facilitate easier traceback and investigation of the attack. The low degree of flow aggregation allows the use of more complex detection strategies with higher accuracy. Also, routers closer to the sources are likely to relay less traffic than core routers and can dedicate more of their resources to DDoS defense.

The D-WARD system is installed at the *source router* that serves as a gateway between the deploying network (*source network*) and the rest of the Internet. We assume that D-WARD is able to identify the police address set, either through some protocol or through manual configuration. We further assume that all machines from the police address set use the source router as the "exit router" to reach a particular set of destination networks and to receive traffic from these destination networks. (Asymmetric routing is discussed in Section 7.)

D-WARD monitors the behavior of each peer with whom the source network communicates, looking for signs of communication difficulties, such as a reduction in the number of response packets or longer inter-arrival times. D-WARD periodically compares the observed values of the two-way traffic statistics for each peer against a predefined model of normal traffic. If the comparison reveals the possibility of a DDoS attack, D-WARD responds by imposing a rate limit on the suspicious outgoing flow for this peer. Subsequent observation either confirms or refutes this hypothesis. Upon confirmation, D-WARD restricts the allowed rate limit further. Refutation leads to a slow increase of the allowable traffic for the flow.

2.1. System architecture

D-WARD is a self-regulating reverse-feedback system. It consists of observation and throttling components that can be part of the source router itself, or can belong to a separate unit that interacts with the source router to obtain traffic statistics and install rate-limiting rules. Figure 1 depicts the architecture corresponding to the second approach.

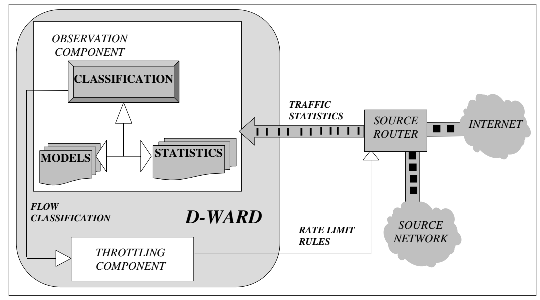

Figure 1. D-WARD architecture.

The observation component monitors all packets passing through the source router and gathers statistics on two-way communication between the police address set and the rest of the Internet. This monitoring can be performed, for example, by sniffing the traffic at the source router interfaces. Periodically, statistics are compared to models of normal traffic and results are passed to the throttling component which adjusts and transfers the new rate limit rules into the source router. The imposed rate limits modify associated traffic flows and thus affect future observations, closing the feedback loop.

2.2. Monitoring and attack detection

The observation component monitors two-way traffic at flow granularity in order to detect difficulties in communication that could be a sign of a DDoS attack. A flow is defined as the aggregate traffic between the police address set and a foreign host. Additionally, it monitors two-way traffic at the connection level, attempting to identify legitimate connections that should receive good service in case the associated flow becomes rate-limited. A connection is defined as the aggregate traffic between two IP addresses and port numbers, where one address belongs to the police address set, and the other is a foreign address.

Flow classification. Flow statistics are stored at the granularity of the IP address of the peer host. Keeping a record for each existing IP address is infeasible, so the records are kept in a limited-size *flow hash table*. The flow record is deleted from the flow hash table if: (i) no traffic is observed on the corresponding flow during a $FlowInactive$ interval, or (ii) the flow hash table is full. In the case of overflow, the least frequently used records are deleted until enough space has been reclaimed. The number of sent packets and the number of sent bytes determine a record's use frequency.

Each flow record contains statistics on three types of traffic: TCP, UDP, and ICMP. These statistics include the number of packets and bytes sent to and received from the peer during the *observation interval*, and the number of active connections. Additionally, for TCP and ICMP traffic, D-WARD keeps the smoothed ratio of the number of packets sent to the peer and the number of packets received from the peer.

D-WARD compares the statistics with normal flow models after every observation interval and classifies flows as *normal*, *suspicious* or *attack*. If all comparisons match their corresponding models, then the flow is compliant and will be examined further. Otherwise, the flow is classified as an attack flow. Compliant flows are further classified into suspicious and normal flows based on their past behavior. Suspicious flows are those flows that have recently been classified as attack. Even though the observations now indicate compliance with the normal flow model, the imposed rate limit must be carefully removed to prevent pulsing attacks that could cause periodic disturbance to the victim. Section 2.3 explains this policy in greater detail.

TCP normal traffic model. During a TCP session, the data flow from the source to destination host is controlled by the constant flow of acknowledgments in the reverse direction. D-WARD's TCP flow model defines TCP_{rto}—the maximum allowed ratio of the number of packets sent and received in the aggregate TCP flow to the peer. The flow is classified as an attack flow if its packet ratio is above the threshold; otherwise, it is considered a compliant flow.

ICMP normal traffic model. The ICMP protocol specifies many different message types. During normal operation the "timestamp," "information request," and "echo" messages should be paired with the corresponding reply. Using this observation, the normal ICMP flow model defines $ICMP_{rto}$—the maximum allowed ratio of the number of echo, time stamp, and information request and reply packets sent and received in the aggregate flow to the peer. The frequency of other ICMP messages, such as "destination unreachable," "source quench," "redirect," etc., is expected to be so small that a predefined rate limit can be used to control that portion of the traffic.

UDP normal traffic model. The UDP protocol is used for unreliable message delivery and in general does not require any reverse packets for its proper operation. Many applications that communicate through UDP packets generate a relatively constant packet rate, but the maximum rate depends heavily on the underlying application. On the other hand, UDP traffic usually occupies a small percentage of all network traffic and is conducted via a few connections. We use this observation to define the UDP flow model as a set of thresholds: n_{conn}—an upper bound on the number of allowed connections per destination, p_{conn}—a lower bound on the number of allowed packets per connection, and UDP_{rate}—a maximum allowed sending rate per connection. The model classifies a flow as an attack when at least one of these thresholds has been breached. The first two thresholds help identify a UDP attack through spoofed connections, while the third identifies a UDP attack through a few very aggressive, non-spoofed connections. An attacker can still get enough traffic past the thresholds to perpetrate an attack if he chooses to spoof a small number of addresses consistently and distributes the attack sufficiently so that each source network sees only a small portion of the traffic. We plan to address this issue in our future work.

Connection classification. Connection statistics are stored in a limited-size *connection hash table* and include the number of packets and bytes sent and received during the observation interval. Each connection is classified as *good* (complying to the model), *bad* (parameter values outside the model boundaries) or *transient* (not enough data to perform a classification) according to the traffic type of its protocol. The models defining good connections are similar to those defining normal flows. The TCP and ICMP connection models specify the maximum allowed packet ratio. In addition to this, TCP connections must be responsive to packet drops. The UDP connection model specifies the maximum allowed sending rate $UDP_{connrate}$. Good connections receive guaranteed good service during the rate limit phase, while transient connections have to compete with the attack traffic for the rate-limited bandwidth.

The connection record is deleted from the connection hash table if: (i) The connection is classified as transient and has been inactive for $TransientConnInactivePeriod$, (ii) the connection is classified as good and has been inactive for $GoodConnInactivePeriod$, or (iii) the connection hash table has filled to capacity. In the case of overflow, records are deleted until enough space has been reclaimed. Bad connection records are deleted first, and next are the least frequently used transient connection records. Good connection records are never deleted if the table overflows.

Since the connection hash table has limited size, it might overflow if the attack is performed using spoofed packets. This suggests the possibility of poor service offered to legitimate connections that start during the attack, since they are likely to be expelled from the connection hash table before their goodness has been established. New legitimate connections may suffer packet losses even if they remain in the table. During the time they are classified as transient, they have to fight for the limited outgoing bandwidth with more aggressive attack traffic. Both of these problems exist because it is difficult to distinguish legitimate packets from attack packets based on the first packet in the connection.

2.3. Attack response

The throttling component defines the allowed sending rate for a particular flow based on the current flow characterization and its aggressiveness. The problem of regulating the sending rate of a one-way flow to the level manageable by the receiver (or the route to the receiver) has been recognized and addressed by the TCP congestion control mechanism. D-WARD strives to solve a similar problem at a more aggregated scale. It needs to control the total flow to the peer, or the portion of that flow that has been characterized as troublesome, and it infers the peer's state from its response packets. D-WARD's rate-limiting strategy applies modified TCP congestion control ideas to this problem: fast exponential decrease of the sending rate when the peer is not sufficiently responsive, slow recovery of rate-limited flows for a certain time period, and fast recovery once flows have proved that they will behave. In addition to this, the flow can be restrained more if it does not comply with the imposed rate limit, and tries to send more than it is allowed.

When the flow is classified as an attack flow for the first time after a long period of normal activity, its rate is limited to a fraction of the offending sending rate. The size of the fraction is specified by the configuration parameter f_{dec}. Subsequent classification of a flow as an attack restricts the rate limit further, according to the formula:

$$rl = min(rl, rate) * f_{dec} * \frac{B_{sent}}{B_{sent} + B_{dropped}} \quad (1)$$

where $rate$ is the realized sending rate for the flow during the previous observation interval, rl is the current rate limit, B_{sent} represents the number of bytes sent for this flow (after the rate limit is applied) during the interval, and B_{drop} is the number of dropped bytes because of the imposed rate limit. Thus, the last factor in the equation describes the degree of misbehavior of the flow, and defines the restrictiveness of the rate limit. Flows that have worse behavior are quickly restricted to very low rates, whereas this restriction is more gradual for better-behaving flows. The lowest rate limit that can be imposed is defined by the $MinRate$ configuration parameter so that at least some packets can reach the destination and trigger a recovery phase. When the flow becomes compliant it will be classified as suspicious, at which point the recovery mechanism is triggered. The recovery phase is divided into slow-recovery and fast-recovery. During the slow-recovery phase, the allowed flow rate is increased according to the formula:

$$rl = rl + rate_{inc} * \frac{B_{sent}}{B_{sent} + B_{dropped}} \quad (2)$$

The speed of the recovery is defined by the $rate_{inc}$ parameter, and the duration of the slow-recovery phase is defined by the $PenaltyPeriod$. Note that although the slow-recovery phase limits the effectiveness of repeated attacks, they can still take place, but with a pause duration larger than $PenaltyPeriod$.

After the flow has been rate-limited and classified as compliant for $PenaltyPeriod$ consecutive observation intervals, the fast-recovery phase is triggered. During the fast-recovery phase the rate is increased exponentially according to the formula:

$$rl = rl * (1 + f_{inc} * \frac{B_{sent}}{B_{sent} + B_{dropped}}) \quad (3)$$

The speed of the recovery is defined by the f_{inc} parameter, and the rate increase is limited by the $MaxRate$ configuration parameter. As soon as the rate limit becomes greater than $MaxRate$, the recovery phase is finished, and the rate limit is removed.

3. Test results and analysis

We implemented the D-WARD system in a Linux software router and tested it against several attack scenarios. D-WARD is implemented partly at the application level and partly as a kernel module. This dual implementation is necessary since D-WARD's memory requirements cannot be satisfied at the kernel level, and the speed of the application-level implementation cannot handle a large traffic volume and therefore cannot be tested using real attack scenarios. The application part acts as the monitoring and throttling component. It uses the libpcap facility to capture information about every packet and update the flow and connection statistics. A separate thread classifies connections and flows and determines the appropriate rate limit. Information about good connections and the desired rate limit is inserted into the kernel module through system calls. The module detects and forwards packets belonging to good connections from rate-limited flows and enforces the rate limit on the rest of the flow. At large packet rates, the libpcap monitoring facility cannot capture all packets. More accurate statistics are then obtained from the kernel module for rate-limited flows and good connections.

In order to test different attack scenarios we developed a customizable DDoS attack tool. It uses a master-slave architecture to coordinate attacks among multiple slaves. Attack traffic mixture (relative ratio of TCP SYN, ICMP_ECHO and UDP packets), packet size, attack rate, target ports, spoofing techniques and attack dynamics can be customized.

The test network consists of a source router deploying D-WARD, the attacker and the legitimate client who both belong to the source network and are part of the police address set, and a foreign host playing the role of the victim. Since D-WARD operates autonomously and analyzes only its incoming and outgoing traffic, multiple attacking domains would only affect the detection of the attack by making the victim feel the attack sooner. The effect of deploying multiple attack and legitimate client machines in the source network is mimicked by using only two machines that generate high traffic loads. Since D-WARD analyzes all incoming and outgoing traffic seen by the router, the number

$TCP_{rto} = 3$	$ICMP_{rto} = 1.1$
Observation interval = 1 sec	$PenaltyPeriod = 20\text{sec}$
$MinRate = 2\text{KBps}$	$MaxRate = 1\text{MBps}$
$f_{dec} = 0.5$	$f_{inc} = 1$
$n_{conn} = 100$	$p_{conn} = 1$
$UDP_{rate} = 10\text{MBps}$	$UDP_{connrate} = 100\text{KBps}$
$FlowInactivePeriod = 360\text{sec}$	$rate_{inc} = 2\text{KBps}$
$TransientConnInactivePeriod = 120\text{sec}$	
$GoodConnInactivePeriod = 360\text{sec}$	

Table 1. Test parameters.

of machines generating the traffic is transparent to the system. The parameters for the test were estimated from traffic traces gathered from our network and are listed in Table 1.

3.1. Attack and legitimate bandwidth

In these tests we evaluate the ability of D-WARD to detect and restrain the attack, while offering good service to legitimate traffic. Each test run lasts for 12 minutes. We generate several TCP connections between legitimate clients and the victim and interleave them with the attack traffic. The attack is started at 25 seconds and lasts until 625 seconds. Legitimate connections are started at 0, 5, 125, 626, 627, 628 and 629 seconds. In the measurements we vary the attack parameters and note the system's behavior.

Attack dynamics. The goal of this test is to illustrate the behavior of the system under various attacks. We generate TCP SYN flood attacks with a maximum rate of 500 KBps, and test with four attack rate dynamics:

- **Constant rate attack.** The maximum rate is achieved immediately and maintained until the attack is stopped.
- **Pulsing attack.** The attack rate oscillates between the maximum rate and zero. The duration of the active and inactive period is the same: 100 seconds.
- **Increasing rate attack.** The maximum rate is achieved gradually over 300 seconds and is maintained until the attack is stopped.
- **Gradual pulse attack.** The maximum rate is achieved gradually over 300 seconds, maintained for 20 seconds and then gradually decreased to zero over 10 seconds. The inactive period lasts for 40 seconds, and then the attack begins again.

Figure 2 gives the results of these tests. The solid line represents the attack bandwidth passed to the victim, and the dotted line represents the actual attack bandwidth offered to D-WARD. In all four cases, the attack is detected and severely rate-limited (less than 1% of the maximum attack rate is allowed to pass) within several seconds. Note that gradually increasing attacks take a longer time to be detected than attacks that start off at the maximum rate. That is to be expected since gradual attacks create less disturbance in the observed statistics. Also note that since the inactivity period is much larger than the $PenaltyPeriod$, periodic attacks appear to the system as new attack instances.

In all tests legitimate traffic experienced around 1% of drops. Good packets were only dropped at times when a new connection attempted to start during or immediately after the attack, and only during a few seconds until the goodness of the connection could be established.

Maximum attack rate. We next tested the relationship between the effectiveness of the system and the maximum attack rate. We generated TCP SYN, ICMP_ECHO, and UDP attacks with continuous and gradually increasing rates, varying the maximum rate from 100 KBps to 2 MBps, and measuring the cumulative attack and good traffic that was delivered to the victim during the test.

Figure 3 gives results of these tests, with 95% confidence interval. A solid line represents the traffic passed to the victim in the case of a UDP attack, a dotted line represents the case of a TCP attack, and a dash-dotted line represents the case of an ICMP attack. UDP and TCP attacks use a fixed packet size of 1KB whereas ICMP attacks use a packet size of 100B. Since all attacks are generated from a single machine, the maximum rate that can be generated in the ICMP attack case is lower than the maximum rate generated in the TCP and UDP case. This is reflected in the dash-dotted line reaching only to half of the rate axis.

Constant rate attacks pass similar amounts of the attack traffic for both UDP and TCP cases. This is due to the sudden onset of the attack, which creates a sufficient disturbance in the network to be quickly detected and controlled. ICMP attacks can pass undiscovered if the maximum attack rate is small (100KBps). At higher attack rates, they are detected with an efficiency similar to UDP and TCP attacks, and quickly constrained. The total attack traffic that the victim receives increases linearly with the maximum attack rate. Most of the attack traffic is passed in the first few seconds, while the smoothed statistics have not sufficiently changed enough to affect attack detection. If the attack rate is higher during this interval, this will be reflected in the total attack traffic that reaches the victim.

In the case of attacks with gradually increasing rates, UDP attacks get detected more quickly than TCP attacks at lower attack rates. Since D-WARD detects the occurrence of spoofed UDP packets, it can detect the UDP attack as soon as the attacker sends enough spoofed packets. In the TCP attack case, however, D-WARD detects the disturbance at the victim through a decrease in the response packet rate. This occurs with a larger delay for small rate attacks and thus slows down the detection. ICMP attacks experience an even larger detection delay than TCP attacks

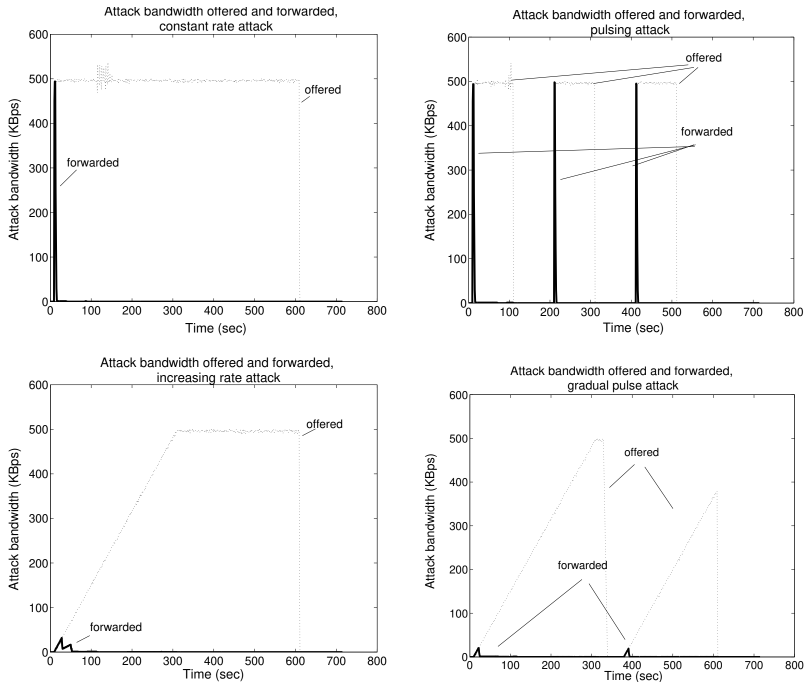

Figure 2. Attack bandwidth passed to the victim.

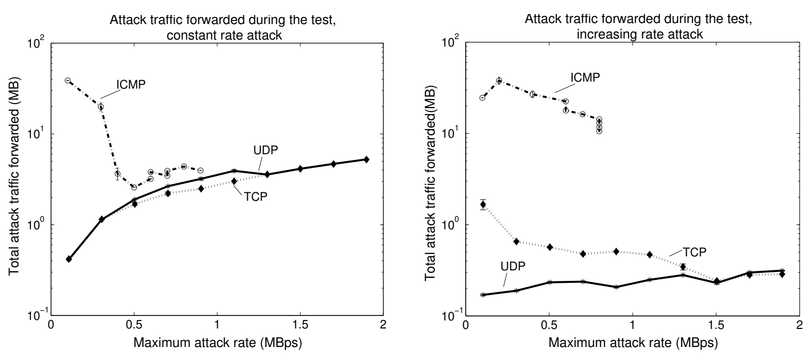

Figure 3. Relationship of the total attack traffic and legitimate traffic passed to the maximum attack rate.

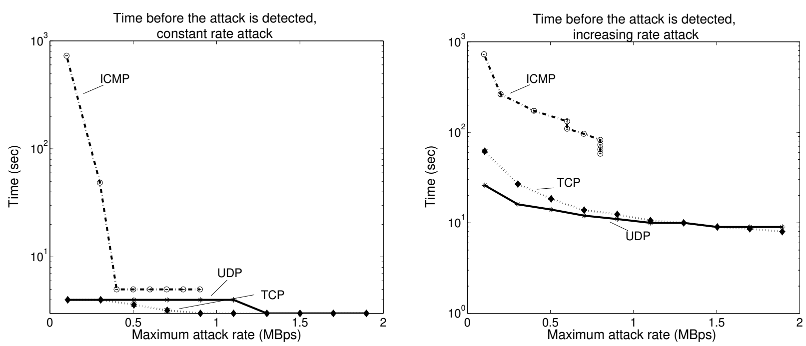

Figure 4. Relationship of the attack detection time to the maximum attack rate.

Trace Number	Flow Misclassifications	Connection Misclassifications
1	0.43%	0.085%
2	0.06%	0.003%
3	0.11%	0.016%
4	0.10%	0.011%
5	0.36%	0.010%
6	0.14%	0.004%
7	0.11%	0.003%
8	0.18%	0.013%
9	0.13%	0.010%

Table 2. Percentage of false positives.

since the victim can handle a larger volume of ICMP packets than of TCP packets. The differences in detection delay indicate that the detection of all attacks could be improved by adding a spoofing verification step to the classification. This will be investigated in our future work.

Figure 4 gives the time needed for detection of the attack. This time is measured from the start of the attack. As we expected, the gradually increasing rate attacks are detected later than the continuous rate attacks. The TCP attacks experience a larger detection delay than UDP attacks, and the ICMP attacks are detected much later than both of those attack types. The detection delay is reduced as the maximum attack rate increases.

In all experiments, the percentage of good traffic dropped was similar—between 1% and 1.5%. As we expected, this amount did not depend on the maximum attack rate.

3.2. False positives

In order to test D-WARD's performance with realistic traffic, we modified the system to read packet header data from a tcpdump-generated trace file instead of sniffing it from the network. We used packet traces gathered from our university network during August 2001. The network has approximately 800 machines and experiences an average of 5.5 Mbps (peak 20 Mbps) of outgoing traffic and 5.8 Mbps (peak 23 Mbps) of incoming traffic. We assume that no attack has occurred during the trace-gathering process.

We determine the level of false positives by measuring the number of flow and connection misclassifications (the number of times that any flow was misclassified as attack or suspicious, and the number of times that any connection was misclassified as bad). We report this measure relative to the total number of flow and connection classifications performed during the trace. Table 2 presents the results for several traces. All measurements yield a low level of false positives, less than 0.5%. This means that if D-WARD were deployed in a real network, its operation would not have any noticeable impact on legitimate traffic.

3.3. Deployment cost

The cost of deploying D-WARD consists of the delay introduced by passing packets through the rate-limiting module, and the storage dedicated to the flow hash table and the connection hash table. The kernel module delay stays stable regardless of the imposed load and is between 1 and 10 μs. The application layer delay increases as the hash tables fill up, since some time is spent keeping them reasonably empty so that new records can be inserted. This delay is 83 μs during normal operation and around 1ms under heavy load.

The maximum size of the connection hash table and flow hash table in our tests is set to 1,000,000 records and 10,000 records, respectively. When hash tables are full they use 44MB and 1.08MB respectively. Under normal load (during trace replay) they use a maximum of 132 KB and 324 KB.

The system had no problem coping with high packet loads and did not introduce any considerable overhead to the packet forwarding mechanism on the deploying machine.

4. Security considerations

Compromising a D-WARD router would allow an attacker to apply rate limits on any packets flowing out of the domain. However, the attacker can do much more damage with full control of the router, so adding D-WARD functionality makes the situation no worse. D-WARD nodes are specifically designed to operate autonomously from other D-WARD nodes. The autonomous operation removes the problems associated with securing communication sessions among a large number of participants and relying on potentially subverted hosts.

Clever attackers will try to disguise attack traffic as normal traffic so that D-WARD will not filter it. To do so, the attack must mimic the congestion control mechanisms found in the protocols. Since the attackers want to degrade performance through congestion, obeying congestion-avoidance mechanisms is contradictory. The attacker would be forced to employ many more machines to get the same effect at the victim.

UDP traffic poses a special problem for D-WARD. TCP and ICMP traffic both elicit responses from the server. D-WARD cannot assume that UDP traffic will do the same thing. D-WARD recognizes UDP attacks that use spoofed connections. An attacker could choose to consistently spoof a few addresses and thus avoid detection while still getting a sufficiently high volume of packets to the victim. We plan to address this in our future work.

When classifying TCP and ICMP traffic, D-WARD relies on return packets from the destination to determine whether the host is under attack. If an attacker could spoof these reply packets, then D-WARD would believe that it was seeing legitimate traffic. This type of attack is possible, but it would require the attacker to gather twice the number of slave machines, half within the D-WARD network for the actual attack and half on the outside for spoofed reply generation. If such an attack did occur, having D-WARD in place would make the situation no worse. One possible solution would be to allow communication between the victim and D-WARD. Since the victim would issue explicit notifications of the attack, fake replies could not delay detection.

D-WARD is a DDoS defense system, and it would be highly undesirable if attackers were able to leverage it to deny service to legitimate traffic. Since D-WARD examines traffic on a connection granularity, an attacker who can spoof a currently active connection or soon-to-be-active connection can: (i) "smuggle" its packets among legitimate packets, if he is not sending aggressively, or (ii) deny service to legitimate packets if his aggressive traffic leads to rate-limiting and classification of a given connection as bad. If the attacker attempts to smuggle an excess amount of traffic, the attack would be detected and the connection would be classified as bad. On the other hand, the attacker will succeed if his aim is to deny service to legitimate hosts from the source network, since the legitimate traffic to the victim on this connection would be subject to rate-limiting and some legitimate packets would be dropped. Hijacking connections is possible in networks today, and D-WARD does not offer any feature to make this easier for the attacker. Additionally, the hijacked connection is likely to lead to interrupted communication between the legitimate client and the attacker, since inserted bogus packets will confuse the end hosts and lead one side to close the connection. The possible legitimate packet drops due to rate-limiting only speed up this process.

An attacker could perform a denial-of-service attack on the source network, preventing the response packets from reaching the D-WARD system. Seeing the reduced number of response packets, D-WARD could reach the conclusion that the source network is generating a DDoS attack and place the rate limit on outgoing flows. Thus the attacker denies the outgoing bandwidth to legitimate clients from the source network. There are two aspects to be noted here: (i) well-behaved flows will back-off themselves in the absence of response packets that lead to their classification as "good," so they will not be affected by the rate limit, and (ii) in the case where most reverse traffic does not reach the source network, legitimate communication from the source network is difficult or impossible, anyway. The lack of reverse traffic will also confuse most protocols in today's networks, and they will probably reduce their sending rates or shut down the connections entirely, so adding D-WARD preserves the status quo.

5. Motivation for deployment

Cooperation formed the basis of the original ARPANET, and this is still seen in the Internet protocols of today. Different sessions of TCP, for instance, will fairly share bandwidth on a link, and new protocols are often judged on whether they are "TCP friendly." D-WARD operates in much the same way by protecting against DDoS attacks originating from a D-WARD protected network. The site deploying D-WARD benefits by not losing bandwidth to outgoing attacks (for sites who pay by usage this could represent a significant benefit) and not having to deal with the social implications of hosting DDoS slaves.

Currently, the only ramifications of unknowingly hosting a DDoS attack are annoying calls to system administrators. In the future it is possible that contracted or legislative action will hold those who do not take reasonable steps to secure their system liable for damages inflicted by attackers misusing their machines. In this case, a corporation deploying D-WARD could argue that it followed current security practice and therefore cannot be held liable if an attack originates on its networks.

Many people have concluded that stopping attacks completely is impossible, since there is a vast number of machines whose owners are unaware of security holes or are unwilling to fix them. For example, despite the best efforts to eradicate worms like Code Red [3] and Nimda[4], these still control a massive number of machines on the Internet. D-WARD brings this problem to a level of ISP or stub networks. Their administrators are likely to be security conscious, and a single D-WARD system installed at the exit router would prevent DDoS attacks originating from their network.

6. Related work

There are many approaches to solving the serious problem of DDoS. Space permits only a brief review of those problems, with detailed descriptions of only those most related to D-WARD. In almost all cases, the systems are compatible with D-WARD, and combined use could offer synergistic protection from DDoS attacks.

Most systems for combating DDoS attacks work on the victim side. Intrusion detection systems such as NetRanger [6], NID [2], SecureNet PRO [15], RealSecure [20], [17] and NFR-NID [19] could be used to detect anomalies and attack signatures in outgoing traffic. Most of these systems do not take automated action to stop the attack. Intrusion detection system methods could be used to signal D-WARD that certain traffic requires more analysis.

Many commercial routers have built-in features such as logging and IP accounting that can be used for characterizing and tracking common attacks [5]. These features usually gather statistical data and offer no automated analysis or response. However, current routers could easily be extended with D-WARD components, thus adding an analysis and response layer to an existing monitoring capability.

In [8] Floyd et al. have proposed to augment routers with the ability to detect and control flows that create congestion (frequently a sign of DDoS attacks). Flows are detected by monitoring the packets in the router queue and identifying high-bandwidth aggregates that are responsible for the majority of drops. The rate limit is then imposed on the aggregate. Pushback can be used to install rate limits at upstream routers if the congested router cannot control the aggregate itself. This approach faces challenges related to augmentation of large numbers of routers, handling legacy routers, and cooperation among different administrative domains. Since the rate limit is imposed at the aggregate traffic close to the victim, legitimate flows suffer collateral damage [11].

Secure Overlay Services (SOS) [12] prevent denial-of-service attacks on critical servers by routing requests from previously authenticated clients to those servers via an overlay network. All other requests are filtered by the overlay. SOS is a distributed system that offers excellent protection to the specified target at the cost of modifying client systems to make them aware of the overlay and use it to access the target. Additionally, large numbers of overlay nodes are required to make the system resilient to DoS attacks.

MULTOPS [9] proposes a heuristic and a data-structure that network devices can use to detect DDoS attacks. Each network device maintains a multi-level tree, monitoring certain traffic characteristics and storing data in nodes corresponding to subnet prefixes. The tree expands and contracts within a fixed memory budget. The attack is detected by abnormal packet ratio values, and offending flows are rate-limited. The system is designed so that it can operate as either a source-end or victim-end DDoS defense system. While the high-level design of this system has much in common with D-WARD, the details are different. MULTOPS uses only the aggregate packet ratio to model normal flows. Non-TCP flows in a system using MULTOPS can either be misclassified as attack flows, or recognized as special and rate-limited to a fixed value. In the first approach, harm is done to a legitimate flow, while in the second approach, sufficiently distributed attacks can successfully make use of the allowed transfer rate. [9] offers too few details on the rate-limiting mechanism and methods for removing rate limits to allow a full comparison with D-WARD.

Several systems combat DDoS attacks by using traceback mechanisms to locate attacking nodes ([1], [18]). These systems provide information about the identity of attacking machines, but do not stop DDoS attacks. The complexity of a traceback mechanism is large if the attack is distributed, and the mechanism must be resilient to attacks.

Several filtering mechanisms have been proposed to prevent spoofing source addresses in IP packets ([7], [13], [16]). While IP spoofing is not necessary in DDoS attacks, it helps attackers hide the identity of attacking machines so they can reuse them for future attacks. D-WARD and many other DDoS prevention mechanisms would benefit from more reliable packet source addresses.

Protocol and application scrubbing [14] (typically applied at the entry point to a victim network) have been proposed to remove ambiguities from transport and application protocols. Scrubbing can eliminate many vulnerability attacks that use protocol ambiguities to bypass intrusion detection systems. A protocol scrubber could be installed at the exit point of the source network and thus prevent vulnerability-based attacks originating from this network.

7. Future work

D-WARD can successfully recognize and constrain many attacks, and offers continued good service to legitimate connections that originate before and after the attack. It has several shortcomings, however, that have not been addressed in the current design. We briefly discuss them here and plan to investigate them in our future work.

Repeated attacks. D-WARD retains no memory of previous attacks and thus will recognize flooding periods of pulsing attacks as new attack instances. Repeated attacks are a common problem for all DDoS defense systems. We plan to investigate the introduction of past-attack memory into the classification process. We expect that this measure will be effective against repeated attacks that are similar to past attack instances.

Detection of UDP attacks. D-WARD currently detects only those UDP attacks that use IP source address spoofing or generate high-rate floods. In our future work, we will investigate possibilities for communication between D-WARD and the victim, and between several D-WARD systems with the goal of detecting more subtle UDP attacks. Proper authentication will be performed to prevent the attackers from misusing this communication to deny service to any host.

Asymmetric routes. If the source router deploying D-WARD is not the only border router of the source network, it might not see both directions of the flow for certain peers, and thus will not classify these flows properly. This is a very likely situation, for many of today's networks use several border routers for resiliency and load-balancing purposes. We will investigate techniques to detect asymmetric flows and to obtain correct statistics from D-WARD systems installed at other border routers in the domain.

Legitimate flows that start during the attack. As discussed in Section 2.2, the limited size of the connection hash table offers the possibility of poor service to legitimate connections that begin during the attack. We are investigating techniques that will enable us to make an early distinction between transient and bad connections, and thus address this problem in a timely manner.

Porting to a hardware router. Our implementation of D-WARD in a Linux router gives us maximum flexibility for designing, developing, and testing. We plan to implement D-WARD in the Intel IXP, a programmable hardware router ([10]). This will help us assess the level of difficulty in porting D-WARD to real routers and enable us test whether D-WARD can handle traffic at high speeds.

8. Conclusion

D-WARD offers an effective defense against distributed denial-of-service attacks. By applying that defense at the point where DDoS traffic enters the network, D-WARD is able to spread the deployment cost among many systems and remove the useless load of DDoS traffic from the Internet as a whole. The early version of D-WARD can effectively detect individual flows contributing to a DDoS attack and apply reasonable rate limits to bring them under control. In a few seconds, D-WARD can detect many common forms of DDoS attacks, and can dramatically reduce their effect almost immediately. More sophisticated attacks, such as those that slowly increase their sending rates, take longer to detect, but D-WARD controls them almost as soon as they have sufficient volume to affect the victim. At the same time, legitimate flows from the source network proceed unharmed. The performance results shown here were obtained using real traffic, real attacks, real attack and victim machines, and a real software router. The traffic patterns used for ordinary background traffic were derived from real traces of our network, and in some cases were even replays of that traced data. Thus, we expect that these results are likely to apply to real deployments.

The great complexity of the DDoS problem suggests that its solution will require the use of multiple defenses, such as filtering, traceback, and pushback systems. This paper demonstrates that D-WARD's approach is likely to be an important component in such an integrated defense system.

References

[1] S. Bellovin, M. Leech, and T. Taylor. ICMP Traceback Messages. *Internet draft, work in progress*, October 2001. http://search.ietf.org/internet-drafts/draft-ietf-itrace-01.txt.

[2] C. I. A. Capability. Network Intrusion Detector Overview. http://ciac.llnl.gov/cstc/nid/intro.html.

[3] C. C. Center. CERT Advisory CA-2001-19 "Code Red" Worm Exploiting Buffer Overflow In IIS Indexing Service DLL. http://www.cert.org/advisories/CA-2001-19.html.

[4] C. C. Center. CERT Advisory CA-2001-26 "Nimda Worm". http://www.cert.org/advisories/CA-2001-26.html.

[5] Cisco. Characterizing and Tracing Packet Floods Using Cisco Routers. http://www.cisco.com/warp/public/707/22.html#2b.

[6] Cisco. NetRanger Overview. http://www.cisco.com/univercd/cc/td/doc/product/iaabu/csids/csids1/csidsug/overview.htm.

[7] P. Ferguson and D.Senie. Network Ingress Filtering: Defeating Denial of Service Attacks which Employ IP Source Address Spoofing. *RFC 2827*.

[8] S. Floyd, S. M. Bellovin, J. Ioannidis, K. Kompella, R. Mahajan, and V. Paxson. Pushback Messages for Controlling Aggregates in the Network. *Internet draft, work in progress*, July 2001. http://search.ietf.org/internet-drafts/draft-floyd-pushback-messages-00.txt.

[9] T. M. Gil and M. Poletto. MULTOPS: a data-structure for bandwidth attack detection. In *Proceedings of 10th Usenix Security Symposium*, August 2001.

[10] Intel. Intel IXP1200 Network Processor Family. http://www.intel.com/design/network/products/npfamily/ixp1200.htm.

[11] J. Ioannidis and S. M. Bellovin. Pushback: Router-Based Defense Against DDoS Attacks. In *Proceedings of NDSS*, February 2002.

[12] A. D. Keromytis, V. Misra, and D. Rubenstein. SOS: Secure Overlay Services. In *Proceedings of SIGCOMM 2002*, 2002.

[13] J. Li, J. Mirkovic, M. Wang, P. Reiher, and L. Zhang. SAVE: Source Address Validity Enforcement Protocol. In *Proceedings of INFOCOM 2002*, June 2002. to appear.

[14] G. R. Malan, D. Watson, F. Jahanian, and P. Howell. Transport and application protocol scrubbing. In *Proceedings of INFOCOM 2000*, pages 1381–1390, 2000.

[15] MimeStar.com. SecureNet PRO Feature List. http://www.mimestar.com/products/.

[16] K. Park and H. Lee. On the Effectiveness of Route-Based Packet Filtering for Distributed DoS Attack Prevention in Power-Law Internets. In *Proceedings of ACM SIGCOMM 2001*, August 2001.

[17] P. Porras and P. Neumann. EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances. In *Proceedings of the Nineteenth National Computer Security Conference*, October 1997.

[18] S. Savage, D. Wetherall, A. Karlin, and T. Anderson. Practical Network Support for IP Traceback. In *Proceedings of ACM SIGCOMM 2000*, August 2000.

[19] N. Security. NFR Network Intrusion Detection. http://www.nfr.com/products/NID/.

[20] I. S. Systems. Intrusion Detection Security Products. http://www.iss.net/securing_e-business/security_products/intrusion_detection/index.php.