

Bandwidth Distributed Denial of Service: Attacks and Defenses

Moti Geva, Amir Herzberg, and Yehoshua Gev | Bar-Ilan University

The Internet is vulnerable to bandwidth distributed denial-of-service (BW-DDoS) attacks, wherein many hosts send a huge number of packets to cause congestion and disrupt legitimate traffic. So far, BW-DDoS attacks have employed relatively crude, inefficient, brute force mechanisms; future attacks might be significantly more effective and harmful. To meet the increasing threats, more advanced defenses are necessary.

Internet services are vulnerable to denial-of-service (DoS) attacks, especially distributed DoS (DDoS) attacks, in which many attacking agents cooperate to cause excessive load to a victim host, service, or network. These attacks have increased in number and strength¹—in a recent survey of network operators, DDoS was identified as the most common “significant threat” (76 percent of respondents).² Furthermore, researchers have found significant growth in attack size and sophistication.^{1,2}

Bandwidth DDoS (BW-DDoS) attacks disrupt network infrastructure operation by causing congestion, which is carried out by increasing the total amount of traffic (in bytes) or the total amount of packets (often a lower limit, using short packets such as TCP SYN or ACK carrying no payload). These attacks can cause loss or severe degradation of connectivity between the Internet and victim networks or even whole autonomous systems (ASs), possibly disconnecting entire regions of the Internet. Recent BW-DDoS attacks reached a volume of 300 Gbps³; according to a Prolexic attack report, 60 to 86.5 percent of BW-DDoS attacks targeted the network infrastructure, including the DDoS mitigation infrastructure itself.¹

BW-DDoS attackers use different techniques and different types of attacking agents. Strong attacking agents include *privileged zombies*—software agents with

high privileges and complete control over the machine on which they’re executed, with the ability to manipulate the protocol stack, for instance, sending spoofed IP packets. Weak agents include *puppets*—programs that are being downloaded automatically and run in sandboxes, such as JavaScript-based webpages. In addition, attackers might use simple types of bandwidth flooding or elaborate techniques that amplify bandwidth so uncompromised machines assist the attack.

In this article, we compare significant known BW-DDoS attacks and discuss results from the vast body of research on BW-DDoS defenses.

BW-DDoS Attacks

BW-DDoS attacks are usually generated from a large number of compromised computers (zombies or puppets). According to recent surveys, BW-DDoS attacks are the most frequently used DoS method.^{1,2} Most BW-DDoS attacks use a few simple ideas, mainly flooding (many agents sending packets at the maximal rate) and reflection (sending requests to an uncompromised server with a spoofed sender IP address, causing the server to send longer response packets to the victim). Table 1 summarizes the different attacks we discuss in this article.

Flooding attacks have created significant damage, because attackers were able to use a sufficient number of agents to cause massive bandwidth consumption

Table 1. Comparison of BW-DDoS attacks.

| Attack | Agent | Mechanism | Protocol manipulations | Target link |
|--|-------------|---|--|-------------|
| UDP flood | Zombie | Direct flooding | No manipulation | Last mile |
| MaxSYN ⁵ | Puppet | Direct flooding | No manipulation | Last mile |
| DNS reflection ⁷ | Root zombie | Reflection (amplification factor up to 100) | IP (spoof) | Last mile |
| Optimistic acknowledgment ⁸ | Root zombie | Amplification (amplification factor more than 1,600 ⁸) | TCP (congestion control) | Last mile |
| ACK Storm ⁹ | Root zombie | Amplification (amplification factor more than 40,000 ⁹) | IP (spoof), TCP (acknowledgment mechanism) | Last mile |
| Coremelt ¹⁰ | Zombie | Direct flooding | No manipulation | Backbone |

leading to packet loss. However, it seems that, gradually, attackers are adopting more complex and effective attacks. For example, the largest attacks reported in recent years consisted of 100 Gbps in 2010, 60 Gbps in 2011 and 2012, and 300 Gbps in 2013.^{2,3} The 2010, 2011, and 2013 attacks were DNS reflection and amplification attacks. In 2012, the largest attack targeted the DNS infrastructure. Researchers have discovered even more effective BW-DDoS techniques, for instance, with higher amplification factors.

Inducing a significant percentage of packet loss is no easy task. Generally, packet delivery probability is the ratio between the available bottleneck link bandwidth and the attack rate. However, as Figure 1 shows, congestion or (small) packet loss probability causes dramatic performance degradation in TCP connections. This performance degradation is due to TCP's congestion control mechanism, which drastically reduces TCP's sending rate upon packet loss. Thus, BW-DDoS damage might be worse than the mere consumed bandwidth.

Figure 2 depicts the results of an Internet-scale simulation we conducted, which emphasizes the potential damage of various sized BW-DDoS attacks. The simulated topology is based on CAIDA's autonomous system (AS) topology (www.caida.org/data/active/as-relationships), taking into consideration client-provider and sibling constraints. To derive the links' bandwidths, we took an approach similar to the one proposed by the PAWS (Parallel Worm Simulator) simulator: we categorized each AS as large, medium, or small based on the number of links it had with other ASs.⁴ We then used the AS sizes to derive each link's bandwidth capacity, as Table 2 shows. We simulated BW-DDoS attacks consisting of between 200 and 102,400 randomly distributed zombies; each zombie sent 1 Mbps of UDP traffic to the victim AS, resulting in attack rates between 200 Mbps and 100 Gbps. Based on the figure, large-scale attacks in the order of magnitude seen to date can cripple even large ASs as well as hosts and networks.

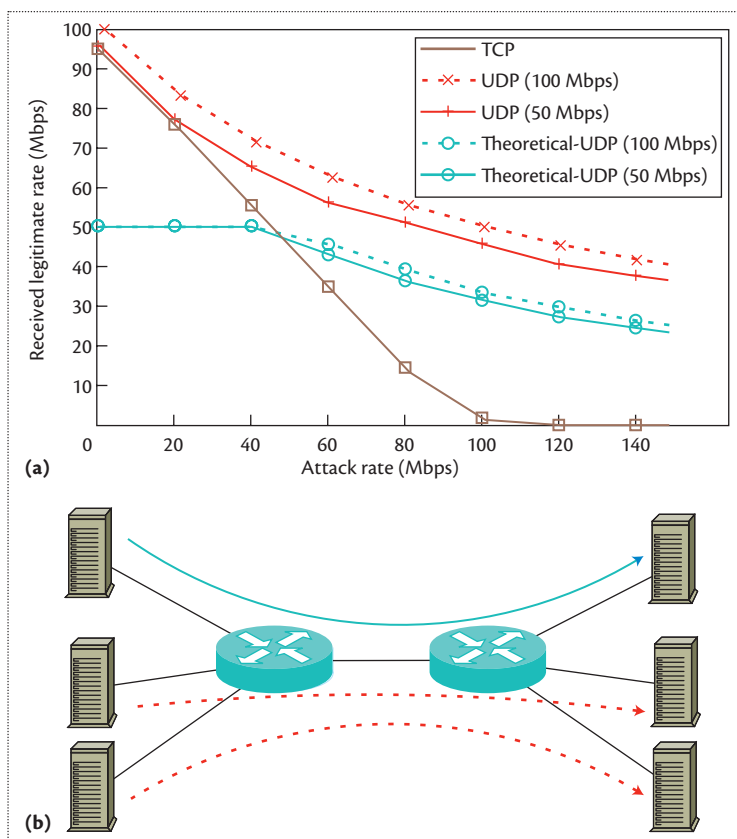


Figure 1. Experimental results of delivered rate versus congestion over a bottleneck link. The brown square line represents TCP, the blue and red lines represent constant-rate UDP, and the dashed lines are theoretical rates for constant-rate UDP flows. The top line in the topology (blue) represents the legitimate traffic, and the bottom lines (red dashed) represent the attacker. TCP reduces its rate as a function of available bandwidth, and UDP suffers from packet loss. All links are 100 Mbps.

BW-DDoS attacks utilize various mechanisms to induce excessive bandwidth consumption. We discuss the main features that differentiate attackers and the capabilities required to launch such attacks.

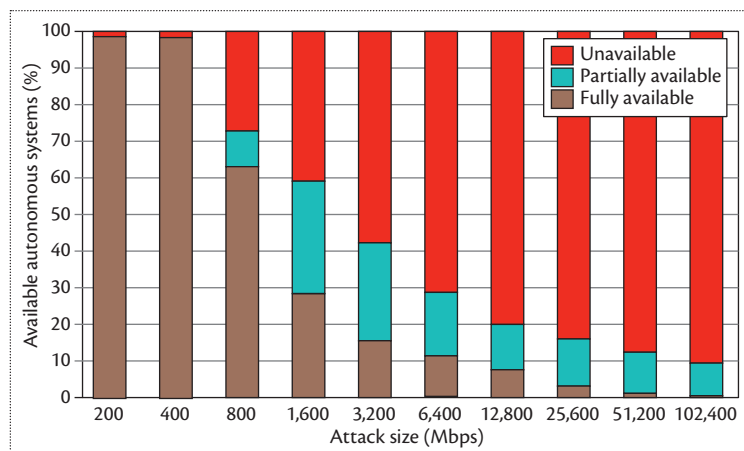


Figure 2. Percentage of available autonomous systems (ASs) versus attack size (Mbps). Brown bars are ASs with available bandwidth to support TCP connections from all incoming routes. Blue bars are ASs with available bandwidth for only part of the incoming links. Red bars are fully congested ASs that can't sustain TCP. Most of the Internet is prone to DDoS at attack scales seen to date.

Attacking Agent

We consider three types of attacking agents: puppets, zombies, and root zombies. Acquiring puppets is relatively easy and can be done by fooling users into browsing to an attacker's website. Zombies are more difficult; they require attackers to install malware on zombie machines by exploiting some vulnerability or tricking users into installing the malware. Root zombies require either zombies that were initially installed with high privileges or a privilege escalation exploit that can maliciously gain such privileges.

Let's first consider a naive BW-DDoS attack in which attackers send as many packets as possible directly to the victim or via attacker-controlled zombies, or *bots*. The simplest scenario is one in which attackers send multiple packets using a connectionless protocol such as UDP. In UDP flood attacks, attackers commonly have a user-mode executable on a zombie machine, which opens standard UDP sockets and sends many UDP packets to the victim.

For UDP floods and many other BW-DDoS attacks, attacking agents must have zombies, that is, hosts running adversary-controlled malware, allowing the malware to use the standard TCP/IP sockets. Other attacks require only puppets, that is, scripts, applets, and so forth, downloaded and run automatically by client agents such as Web browsers. Being untrusted, puppet operations are restricted by a sandbox; they can't send UDP packets, let alone spoof packets, and they're limited in establishing TCP connections. Nevertheless, even though puppets can't induce as much bandwidth as zombies, they can still induce significant amounts.

For example, the MaxSYN attack aims to maximize the number of SYN packets by setting the sources of several JavaScript image objects to nonexistent URLs repeatedly every 50 milliseconds.⁵ Every time the script writes into the image source URL variable, the browser stalls old connections and establishes new ones to fetch the newly set image URL, thereby inducing the transmission of additional SYN packets.

Other types of attacks require zombies to have administrative privileges for execution. We refer to privileged zombies as *root zombies*. To send packets with spoofed source IP addresses, zombies commonly need to open raw sockets, which is permitted for privileged users only. In addition, for some protocol manipulations to succeed, the network mustn't prevent or block the manipulation; for example, spoofing is commonly filtered by ingress filtering at the ISPs.⁶

Attack Mechanism

We consider three types of attack mechanisms: direct flooding, amplification, and reflection. In a naive attack, attack traffic is limited by the compromised machines' bandwidth capacity, and the victim's entire load is due to the direct flooding induced by packets the zombies send. Amplification attacks use the attacking agents' bandwidth more effectively, such that, on average, every packet a zombie sends causes noncompromised machines to transmit multiple or larger packets to the victim. Specifically, attackers choose a request r of size $|r|$ that results in longer response r' of size $|r'|$, achieving an amplification factor of

$$\frac{|r'|}{|r|}$$

Hence, direct-flooding BW-DDoS attacks have no amplification—amplification factor of 1—whereas sophisticated attacks, such as DNS amplification attacks, can have factors greater than 1.

DNS amplification attacks rely on the fact that DNS responses might be larger than DNS requests.⁷ DNS requests are short—for instance, 40 bytes—whereas responses tend to be much longer. Originally, DNS responses over UDP were limited to 512 bytes; however, DNS extensions (EDNS) allow longer responses, up to 4,000 bytes. Hence, the DNS amplification factor can be $512/40 = 12.8$, and with EDNS it can reach up to $4,000/40 = 100$. With the growing popularity of DNSSEC, which relies on EDNS's long packet capabilities, long responses will likely become more common, increasing the potential amplification factor.

Theoretically, based on an amplification factor of 100, attackers require roughly 100 zombies, each sending DNS requests at 100 Kbps, to achieve a 1-Gbps attack. For a 10-Gbps attack, 1,000 zombies are

Table 2. Bandwidth estimation based on AS size.*

| Source/Destination | Small | Medium | Large |
|--------------------|-------------------------|-------------|-------------|
| Small | OC (Optical Carrier) -3 | OC-12 | OC-24 |
| | 155.52 Mbps | 622.08 Mbps | 1.244 Gbps |
| Medium | OC-12 | OC-48 | OC-192 |
| | 622.08 Mbps | 2.488 Gbps | 9.953 Gbps |
| Large | OC-24 | OC-92 | OC-68 |
| | 1.244 Gbps | 9.953 Gbps | 39.813 Gbps |

* We estimated each AS's size based on the number of links it had to other ASs. Small AS had up to four links, medium AS five to 300 links, and large AS had more than 300 links.

required, and so on. Alarming, significantly larger botnets, consisting of hundreds of thousands of zombies, have already been discovered.

Reflection attacks fool legitimate hosts into sending unsolicited responses to victim hosts. For example, DNS reflection attacks are based on the DNS protocol, which is a UDP-based request-response protocol. A DNS resolver (server) will return responses to clients that issue DNS requests. A client's return address is determined using the source IP address appearing in the request. DNS reflection attacks exploit this behavior: attackers send a spoofed DNS request to a DNS server, making the server issue a response packet to the spoofed address. Commonly, attackers use the victim's IP address as the spoofed source address. Furthermore, attackers might amplify their attacks, lengthening the reflected response.

Protocol Manipulations

We discuss two types of protocol manipulations that attackers use. The first attempts to avoid detection, and the second tries to exploit legitimate protocol behavior and cause legitimate clients and servers to excessively misuse their bandwidth against the victim. Typically, protocol manipulation for bandwidth attacks requires a strong zombie with administrative privileges, because the manipulation is commonly done at the low protocol layers handled by the OS, usually IP and TCP. Note that not all BW-DDoS attacks use manipulation.

For naive attacks such as UDP floods, attack sources are visible—that is, victim hosts can see the zombies' source addresses, making it relatively easy to block packets and take technical or legal measures against the attacking machines and their owners. Therefore, attackers might try to avoid detection by manipulating the source IP address, commonly called *spoofing*. Thus, an attacker can send multiple packets, each containing a different spoofed source address, making it harder for a victim to identify and block the attacker source.

The second type of attack exploits legitimate protocol behavior by misusing packet fields or acting

dishonestly, causing legitimate hosts to send unwanted or excessive packets to victims. For example, using IP spoofing, attackers can induce a DNS reflection attack.

Some attacks manipulate other mechanisms. For example, optimistic acknowledgment (opt-ack) manipulates TCP's congestion control.⁸ Generally, the congestion control mechanism adapts the TCP transmission rate based on available bandwidth. The basic assumption behind the congestion control mechanism is that the main reason for packet loss is congestion. Hence, whenever TCP packets are received successfully, the rate increases, and whenever packets are lost, the rate decreases. TCP knows whether packets have been received based on acknowledgment (ACK) packets sent by the destination. In an opt-ack attack, a malicious client sends a request to a server, then as the server sends the response packets, the client optimistically acknowledges receiving them by sending ACK packets, without actually having received the packets. Thus, very low bandwidth is required to cause servers to send a lot of traffic, limited mainly by the servers' bandwidth.

ACK Storm attacks TCP's ACK mechanism.⁹ First, attackers eavesdrop on an existing TCP connection. Next, they spoof ACK packets with a higher sequence number than what was actually sent; this induces a response ACK packet containing the real sequence number, that is, the sequence number that was actually sent. Sending such packets to both ends of the connection simultaneously induces a repeated back-and-forth exchange of ACK packets until either end terminates the connection.

Attack Target

We discuss two types of attack targets: last-mile links and backbone links. Whereas most BW-DDoS attacks target last-mile links, new types of attacks target backbone links. For example, Coremelt uses a peer-to-peer model in which zombies communicate directly with one another.¹⁰ Among N zombies, there are $O(N^2)$ routes, some of which use the victim backbone link. Hence, attackers can create excessive traffic on the victim link

Table 3. Comparison of BW-DDoS defense mechanisms.

| Mechanism | Response | Location | Infrastructure adaptation | Cooperation |
|---|-----------------------|---|---------------------------|--|
| Ingress filter | Filter | Router | Configuration | Stand-alone |
| Access control lists | Filter | Router | Configuration | Stand-alone |
| Remote-Triggered Blackhole | Filter | Router | Configuration | Inter-AS (Border Gateway Protocol [BGP]) |
| Capabilities | Rate limiting | Destination (DST), router, source (SRC) | Router software | Inter-AS |
| PSP (Proactive Surge Protection) | Rate limiting | Router | Router software/IP fields | Intra-AS |
| Pushback | Filter | Router | Router software | Inter-AS |
| BTT (Backward Traffic Throttling) | Rate limiting | Router | Configuration | Inter-AS |
| RON (Resilient Overlay Networks) | Detour | SRC, cloud | End hosts software, cloud | End host and overlay |
| SOS (Secure Overlay Services) | Absorb, filter | SRC, cloud | End hosts software, cloud | End host and overlay |
| Scrubbing | Absorb, filter | Router, cloud | Configuration, cloud | Inter-AS (BGP), cloud |
| QoSDoS (QoS over DoS-Prone Networks) | Breakthrough | SRC, DST | End hosts | End hosts |
| LOT (Lightweight Opportunistic Tunneling) | Filter, rate limiting | Router | Router software | Inter-LOT routers |

using only interzombie communication. Coremelt requires regular zombies—that is, zombies without high privileges—because it can use the standard TCP/IP stack without any special protocol manipulation. Coremelt attacks are mainly theoretical because Internet backbone links are highly provisioned and would require huge peer-to-peer networks to clog. Moreover, based on CAIDA datasets, there are more than 100,000 links between more than 35,000 different ASs, making it very hard to take down a specific backbone link. Nevertheless, assuming enough zombies can be obtained, Coremelt might prove difficult to detect and filter, because each connection uses a small amount of bandwidth.

Current attacks use relatively crude methods, whereas future attacks are likely to be significantly more effective, with higher amplification factors. It might be very difficult for attackers to actually launch such advanced attacks on the Internet, but the basic know-how is there. Finally, existing attacks might challenge currently deployed defense mechanisms, motivating investigation of new mechanisms.

Network-Level Defense Mechanisms

BW-DDoS defense mechanisms focus on several types of schemes, including detecting, filtering, absorbing, and cooperating. We surveyed defense schemes of both

deployed and academically proposed mechanisms. Here, we discuss different defense mechanisms, their deployment location in the network, and the infrastructure adaptation and type of cooperation they require, if any. Note that many defense mechanisms rely on the ability to differentiate between attacks and legitimate flows; however, in this article, we don't discuss differentiation techniques as they have been surveyed before.¹¹ Table 3 summarizes the defense mechanisms.

Response Mechanism

We consider four types of defense mechanisms: filtering, rate limiting, detouring and absorbing, and breakthrough.

Filtering. Assuming the offending flows are identified, they can be filtered out. Filtering can take place in various network locations: close to the destination, at the core (that is, in routers), or close to the source. Usually, to be effective in BW-DDoS mitigation, filtering must occur before the congested link, because the victim usually isn't in a position to hold back the attack.

One example of filtering is preventing source IP spoofing. RFCs 2827 and 3704 recommend that ISPs employ ingress filtering and filter packets with IP addresses external to that network. Many ISPs do

this; however, approximately 15 percent of Internet addresses can still send spoofed packets.^{2,6} LOT (Light-weight Opportunistic Tunneling) is another solution to mitigate spoofing by opportunistically establishing tunnels between gateways and adding a random tag to tunneled packets, making it difficult for attackers to guess the correct tag value.¹² Packets not carrying the correct tag are discarded, preventing the spoofing of packets that originate from incorrect networks.

Additional filtering mechanisms include access control lists (ACLs), Remote-Triggered Blackhole (RTBH), and firewalls. ACLs are router mechanisms that allow or deny matching flows. They're often configured manually; however, some intrusion prevention systems can configure ACLs automatically. Each ACL entry takes a significant amount of memory and some time to process, so routers should limit ACL rules in both number and processing time. Memory and CPU use increase as more ACL entries are used, which might be an additional target for DDoS—not necessarily bandwidth based.

RTBH (RFC 5635) uses the router's forwarding tables such that all traffic to the victim or from attacking sources is forwarded to a "blackhole," completely denying access to the target. RTBH uses a small amount of memory and its processing is faster than ACL. However, RTBH filtering is significantly more aggressive and might help an attacker disconnect its victim from its sources and/or destinations, thereby potentially achieving the goal with little resources.

Rate limiting. In contrast to completely blocking the attacking flows, rate-limiting schemes let the offending flows transmit their typical rate or obey some other limit. Researchers proposed rate limiting at routers in several forms, including capabilities, packet tagging, and scheduling based. Capabilities are tokens issued by the destination (server) to the source (client). Capabilities inform the source, and more importantly the routers en route, that the destination is willing to accept traffic from this source. The issued capabilities are attached to packets the source sends, allowing routers en route to identify and prioritize approved flows. Note that packets without capabilities aren't filtered; instead, they get lower delivery probability, which effectively limits their rate during attack periods.

SIFF (Stateless Internet Flow Filter) proposed *stateless capabilities* wherein capabilities are calculated using (keyed) hash.¹³ Routers check and prioritize flows carrying verified capabilities. TVA (traffic validation architecture) keeps a (small) state in routers and lets servers request specific restrictions per flow.¹⁶ Capabilities-based solutions assume that victims will authorize only legitimate sources and won't cooperate with attackers.

Deployment of capabilities-based solutions requires changes to both end hosts and routers.

PSP (Proactive Surge Protection) collects network statistics at the provider level and infers typical traffic rates between origin–destination pairs.¹⁵ Upon arrival to the provider, packets are tagged as either *normal* or *excessive*. Whenever a router gets congested, packets tagged as excessive are discarded first, effectively prioritizing packets tagged as normal. PSP is deployed only within provider boundaries and requires changing routers' software for packet tagging and prioritizing. Otherwise, it uses existing IP packet fields, which might be used by different applications and hence potentially damage some flows.

BTT (Backward Traffic Throttling) is another type of scheduling-based filtering in routers.¹⁶ Similar to PSP, BTT collects network traffic statistics; however, it doesn't manipulate packets. Instead, BTT employs a weighted fair queuing scheduling scheme that prioritizes flows transmitting at their typical rate over flows using excessive traffic. Next, BTT requests upstream BTT nodes to shape their traffic going through the congested link, for example, by using token buckets. BTT's main advantage is that it merely configures the dataplane routers using existing router mechanisms without requiring router software or firmware change. However, despite the fact that BTT can be deployed gradually, it requires cooperation between ASs and extensive deployment to become really effective. Both PSP and BTT assume that a typical rate exists and that it is measurable.

Detouring and Absorbing. Additional schemes use overlay networks and cloud computing. Overlays mitigating BW-DDoS attacks can be divided into two general types: detouring and absorption. Detouring overlays bypass networks' default routing, thereby overcoming Border Gateway Protocol's (BGP's) shortcomings, such as update speed, route selection under different matrixes, and utilizing special network features such as multihoming.¹⁷ Detouring overlays can implicitly mitigate BW-DDoS attacks only when some routes are congested while other aren't, as the blue bars in Figure 2 depict.

Absorption overlays are overprovisioned with bandwidth and can absorb BW-DDoS attacks. They construct a perimeter around the victim server that only selected nodes can penetrate; unauthorized traffic is filtered. Cloud (practical) or overlay (academic) solutions route traffic via the cloud or overlay, which "scrubs" the attack flows. Absorption clouds and overlays were designed specifically to mitigate BW-DDoS and were investigated in several works, such as SOS (Secure Overlay Services).¹⁸ Note that overlay solutions usually introduce new protocols and hence typically require updating host software. Other solutions, mainly those

deployed, make no protocol changes and instead rely on configuring BGP or DNS records to divert traffic to a cloud-based scrubbing service.

Breakthrough. The final category of BW-DDoS mechanisms are those that use aggressive clients to break through the congestion. Aggressive clients use TCP-friendly protocols as long as they can sustain enough goodput. When TCP's goodput drops below some threshold, aggressive clients commence using protocols without congestion control, such as UDP, thereby exploiting the real network delivery probability, as Figure 1 depicts. An important design goal of aggressive clients is to avoid self-generated BW-DDoS attacks.

QoSoDoS (QoS over DoS-Prone Networks) mechanisms assume that even under strong BW-DDoS attacks, a nonnegligible packet delivery probability remains.¹⁹ Hence, whenever TCP's goodput drops, QoSoDoS retransmits packets using UDP. Assuming the attacker's rate can be bound, it's possible to ensure modest QoS with high probability while limiting the number of retransmissions. By controlling the number of QoSoDoS clients and their transmission rate, QoSoDoS can avoid self-created BW-DDoS attacks. QoSoDoS deployment requires changes to end hosts.

Defense Mechanism Location

The various defense mechanisms can be deployed at different network locations. Some are deployed close to the destination, that is, near the victim. Note that defense mechanisms close to the destination might get a good idea about some of the attack's properties, but they might not be well-positioned to mitigate BW-DDoS attacks because many packets are discarded near the victim due to the exhausted resources. Hence, many defense mechanisms try to mitigate attacks closer to the source.

Router- or backbone-based defense mechanisms are usually located near an overprovisioned link and try to ensure that traffic reaching the victim originates mostly from legitimate sources. Similarly, source-based defense mechanisms try to prevent attackers from sending excessive traffic, especially during BW-DDoS attacks.

Additional deployment locations are "in the cloud" and overlay networks. In such solutions, traffic is routed via an overprovisioned cloud service that scrubs the attacking flows and forwards only legitimate traffic to the victim.

Infrastructure Adaptations

A concern that might affect BW-DDoS solutions' deployment is the amount of changes that the infrastructure must undergo. For example, some solutions require installing new software at end hosts, some require software updates to routers, and others require

reconfiguration of networking equipment. Additional changes might take place by utilizing overlay networks or in the cloud.

Deploying BW-DDoS mitigation solutions also raises concerns regarding ISPs. Usually, ISPs aren't directly impacted by the problem—DoS attacks disable their customers. Estimating the impact of deploying different solutions is very difficult. Also, some changes are easier to make than others. For example, configuration changes are relatively easy to make, whereas software changes at end hosts are usually more difficult. And, we assume that any change to routers—that is, software, firmware, and especially hardware—are very difficult to make and deploy.

Cooperation Schemes

Pushback schemes focus on pushing the attack away from the victim and closer to its source. This is done by sending requests to upstream routers, asking them to filter the identified offending flows. The cooperation might be intra-AS, inter-AS, between end hosts, or with an overlay network or cloud service. Other solutions might be stand-alone and require no cooperation.

In Pushback, the victim identifies the attacking flows' profile, pushes the attack back, and frees the victim's resources to handle legitimate traffic.²⁰ FlowSpec (RFC 5575) describes an operational implementation similar to Pushback. Basically, Pushback and FlowSpec are ACL-like filtering schemes, but instead of employing the ACL entries within a single AS, they're distributed and pushed back upstream.

Pushback-based solutions let underprovisioned nodes filter offensive traffic away from victims. However, victim nodes might not always be able to identify the attack profile. Furthermore, similar to other ACL schemes, Pushback requests often require many filtering rules and ACL entries and might result in a DoS attack on routers' processing capabilities. This decoy attack could exhaust filtering rules, then attack the real target. Alternatively, this type of cooperation might let attackers issue Pushback requests, disconnecting the victim.

BTT is also a Pushback scheme; however, it's less prone to attack than Pushback because it requests only traffic shaping from upstream BTT nodes. The downstream BTT node requests its upstream nodes to limit their traffic that flows through the congested link. Any shaping request from an upstream node should never ask to limit its traffic to less than its typical rate. Hence, BTT QoS might be degraded, but not completely denied.

Cooperation-based schemes such as Pushback and BTT assume that the cooperating nodes are honest with each other and will propagate upstream requests only upon a BW-DDoS attack, which is debatable. Moreover, the signaling plane between cooperating nodes might be an attack target.

So far, BW-DDoS attacks employed relatively crude, inefficient, brute force mechanisms. However, several known attacks, which aren't commonly used, let attackers launch sophisticated attacks, which are difficult to detect and might considerably amplify attackers' strength.

Deployed and proposed defenses might struggle to meet these increasing threats; therefore, we need to deploy more advanced defenses. This might involve proposed mechanisms as well as new approaches. Some proposed defenses raise operational and political issues; these are beyond the scope of our article but should be considered carefully. Finally, for a defense mechanism to be practical, it must be easy to deploy and require minor changes, if any, especially to the Internet's core routers. ■

Acknowledgments

This research was supported by the Israeli Science Ministry and the Israeli Science Foundation.

References


1. "Prolexic Attack Report, Q3 2011–Q4 2012," P.T. Inc., 2012; www.prolexic.com/attackreports.
2. "Worldwide Infrastructure Security Reports Series (2005-2012)," Arbor Networks, 2013; www.arbor-networks.com/report.
3. M. Prince, "The DDoS that Almost Broke the Internet," CloudFlare, 27 Mar. 2013; <http://blog.cloudflare.com/the-ddos-that-almost-broke-the-internet>.
4. S. Wei, J. Mirkovic, and M. Swamy, "Distributed Worm Simulation with a Realistic Internet Model," Workshop Principles of Advanced and Distributed Simulation (PADS 05), IEEE CS, 2005, pp. 71–79.
5. S. Antonatos et al., "Puppetnets: Misusing Web Browsers as a Distributed Attack Infrastructure," *ACM Trans. Information and System Security*, vol. 12, no. 2, 2008, pp. 12:1–12:15.
6. "ANA Spoofer Project," Advanced Network Architecture Group, 2012; <http://spoofer.csail.mit.edu/summary.php>.
7. R. Vaughn and G. Evron, "DNS Amplification Attacks," 2006; <http://packetstorm.foofus.com/papers/attack/DNS-Amplification-Attacks.pdf>.
8. R. Sherwood, B. Bhattacharjee, and R. Braud, "Misbehaving TCP Receivers Can Cause Internet-Wide Congestion Collapse," *Proc. 12th ACM Conf. Computer and Communications Security*, ACM, 2005, pp. 383–392.
9. R. Abramov and A. Herzberg, "TCP Ack Storm DoS Attacks," *Computers and Security*, vol. 33, Mar. 2013, pp. 12–27.
10. A. Studer and A. Perrig, "The CoreMelt Attack," *ESORICS, LNCS 5789*, Springer, 2009, pp. 37–52.
11. G. Carl et al., "Denial-of-Service Attack-Detection Techniques," *IEEE Internet Computing*, vol. 10, no. 1, 2006, pp. 82–89.

12. Y. Gilad and A. Herzberg, "Lot: A defense Against IP Spoofing and Flooding Attacks," *ACM Trans. Information and System Security*, vol. 15, no. 2, 2012, pp. 6:1–6:30; <http://doi.acm.org/10.1145/2240276.2240277>.
13. A. Yaar, A. Perrig, and D.X. Song, "SIFF: A Stateless Internet Flow Filter to Mitigate DDoS Flooding Attacks," *Proc. Symp. IEEE Security and Privacy*, IEEE CS, 2004, pp. 130–146.
14. X. Yang, D. Wetherall, and T.E. Anderson, "A DoS-Limiting Network Architecture," *Proc. ACM SIGCOMM*, 2005, ACM, pp. 241–252.
15. J.C.Y. Chou et al., "Proactive Surge Protection: A Defense Mechanism for Bandwidth-Based Attacks," *IEEE/ACM Trans. Networking*, vol. 17, no. 6, 2009, pp. 1711–1723.
16. Y. Gev, M. Geva, and A. Herzberg, "Backward Traffic Throttling to Mitigate Bandwidth Floods," *Global Comm. Conf. (GLOBECOM 12)*, IEEE, 2012, pp. 904–910.
17. D.G. Andersen et al., "Resilient Overlay Networks," *Proc. 18th ACM Symp. Operating Systems Principles (SOSP 01)*, ACM, 2001, pp. 131–145.
18. A.D. Keromytis, V. Misra, and D. Rubenstein, "SOS: An Architecture For Mitigating DDoS Attacks," *IEEE J. Selected Areas in Communications*, vol. 22, no. 1, 2004, pp. 176–188.
19. M. Geva and A. Herzberg, "QoSoDoS: If You Can't Beat Them, Join Them!," *Proc. INFOCOM*, IEEE, 2011, pp. 1278–286.
20. J. Ioannidis and S.M. Bellovin, "Implementing Pushback: Router-Based Defense Against DDoS Attacks," *Network and Distributed System Security Symp. (NDSS 02)*, Internet Society, 2002; <https://www.cs.columbia.edu/~smb/papers/pushback-impl.pdf>.

Moti Geva is a PhD candidate in Bar-Ilan University's computer science department. His research interests include cybersecurity, networking and network protocols, operating systems, cloud computing, and distributed systems. Geva received his MS in computer science from Bar-Ilan University. Contact him at moti.geva@gmail.com.

Amir Herzberg is an associate professor in Bar-Ilan University's computer science department. His research interests include network security, applied cryptography, cybersecurity, and usable security. Herzberg received a DS in computer science from the Technion, Israel. Contact him at amir.herzberg@gmail.com.

Yehoshua Gev received an MS from Bar Ilan University. His research interests include security of IP networks and defenses against denial-of-service attacks. Contact him at yoshigev@gmail.com.

 Selected CS articles and columns are also available for free at <http://ComputingNow.computer.org>.