Universal DDoS Mitigation Bypass

Tony T.N. Miu[1], Albert K.T. Hui[2], W.L. Lee[2], Daniel X.P. Luo[2], Alan K.L. Chung[2], and Judy W.S. Wong[2]

[1]Nexusguard Limited
tony.miu@nexusguard.com

[2]Network Threats Information Sharing and Analysis Center (NT-ISAC)
Bloodspear Labs
{albert,leng,daniel,alan,judy}@bloodspear.org

Abstract. Today's commercial distributed denial of service (DDoS) mitigation technologies employ many different techniques for identifying DDoS traffic and blocking these threats. Common techniques range from basic malformed traffic checks, to traffic profiling and rate limiting, to traffic source verification and so on, with captive redirection utilizing JavaScript- or CAPTCHA-based authentications being the most effective by far. However, in our research weaknesses were found in a majority of these sort of techniques.

We rolled all our exploits into a proof-of-concept attack tool, giving it near-perfect DDoS mitigation bypass capability against almost every existing commercial DDoS mitigation solutions. The ramifications are huge. For the vast majority of web sites, these mitigation solutions stand as the last line of defense. Breaching this defense can expose these web sites' backend to devastating damages.

We have extensively surveyed DDoS mitigation technologies available on the market today, uncovering the countermeasure techniques they employ, how they work, and how to defeat each of them. Essentially, bypass is achieved through emulating legitimate traffic characteristics. Afterwards, our attack tool is introduced to demonstrate how all these exploits can be brought together to execute a "combo attack" to bypass all layers of protection in order to gain access to the backend. The effectiveness of this tool is illustrated via testing results against specific DDoS mitigation products and popular web sites known to be protected by specific technologies. To conclude our research, a next-gen mitigation technique is also proposed as a countermeasure against our attack methodology.

Keywords: DDoS mitigation, DDoS, large-scale network attack

1 Introduction

DDoS attacks remain a major threat to internet security because they are relatively cheap yet highly effective in taking down otherwise well-protected networks. One need look no further than the attack on Spamhaus to realize the damage potential – bandwidth clog peaked at 300Gbps, all from a mere 750Mbps generated attack traffic [1]!

In the following sections, we first examine DDoS attacks observed in the wild and commercially available mitigation techniques against those attacks, with brief discussion on each technique's inherent weaknesses. Next, we introduce bypass mechanisms that exploit these weaknesses and, through illustrating our proof-of-concept (PoC) tool "Kill 'em All", show how bypass mechanisms can be combined to achieve total bypass, thereby defeating defense-in-depth design typically adopted in DDoS mitigation solutions.

To conclude, we substantiate our claim with testing results against specific mitigation solutions, and propose a next-generation mitigation methodology capable of defending against "Kill 'em All"-type attacks.

2 DDoS Attack Categories

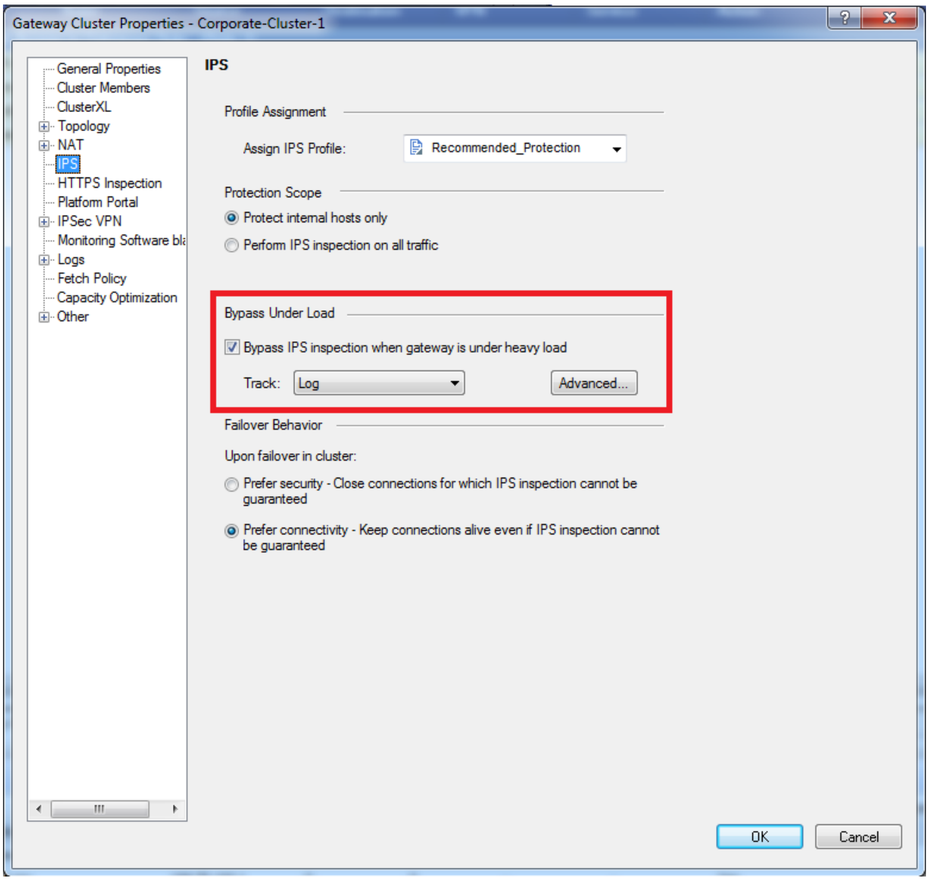
The crudest form of DDoS attack are *volumetric DDoS attacks*, whereby a huge volume of traffic pours into the victim in a brute-force manner, hogging all bandwidth otherwise available for legitimate purposes. Execution is expensive, as the attacker would have to send traffic whose volume is on par with the victim's spare capacity. This translates to a higher monetary cost associated with hiring botnets. The age-old ping flood is a prime example.

Semantic DDoS attacks work smarter, amplifying firepower by exploiting semantic contexts such as protocol and application weaknesses [2]. This effectively tips the balance in the attacker's favor, making attacks much cheaper. Examples of semantic attacks include Slowloris [3] and Smurf [4] attacks, as well as attacks that make excessive database lookups in web applications.

The last one, effecting database lookups, exemplifies emerging *application level attacks*, whereby attacks target weaknesses in specific applications. As of the time of this paper, API attacks are on the rise, paving the way to *attack pivoting* with which attacks can be extended to other computing systems through the API of applications on the system being directly targeted.

A third category, *blended DDoS attacks*, aims to achieve stealthy attacks through blending into legitimate traffic, practically rendering ineffective most countermeasures designed to filter out abnormal, presumably malicious, traffic. HOIC [5] with "booster packs" (elements that add randomized headers and so on in order to make attack traffics look legit) is an example of an attack that employs blending techniques via randomized headers. Another flavor of *blended DDoS attacks* [6] mixes weaponized exploits with high volume DDoS attacks, in order to increase the exploit success rate and evade detection. This works because many security controls fail under DDoS, leaving affected systems wide-open to exploits (see Figure 1 for an example). Moreover, DDoS can also overwhelm audit trail and logging mechanisms, making incident response and forensics much more difficult.

Note that these categories are by no means mutually exclusive. For instance, blended attacks that also exploit application weaknesses are not at all uncommon in the wild.



Gateway Cluster Properties - Corporate-Cluster-1

IPS

Profile Assignment

Assign IPS Profile: Recommended_Protection

Protection Scope

● Protect internal hosts only

○ Perform IPS inspection on all traffic

Bypass Under Load

☑ Bypass IPS inspection when gateway is under heavy load

Track: Log

Failover Behavior

Upon failover in cluster:

○ Prefer security - Close connections for which IPS inspection cannot be guaranteed

● Prefer connectivity - Keep connections alive even if IPS inspection cannot be guaranteed

Figure 1. Example of security control failure under DDoS.

3 Commercial DDoS Mitigation Techniques and Their Weaknesses

Over the years, as DDoS attacks gain sophistication, so do countermeasures. DDoS countermeasures can be broadly classified into three elements: prevention, detection and mitigation. In this paper we shall limit our scope to DDoS mitigation, which concerns coping with ongoing attacks, reducing the impact and containing the damage. For immediate relevance we only consider currently available commercial solutions.

With reference to Figure 2, common commercial detection and mitigation methods are discussed below.

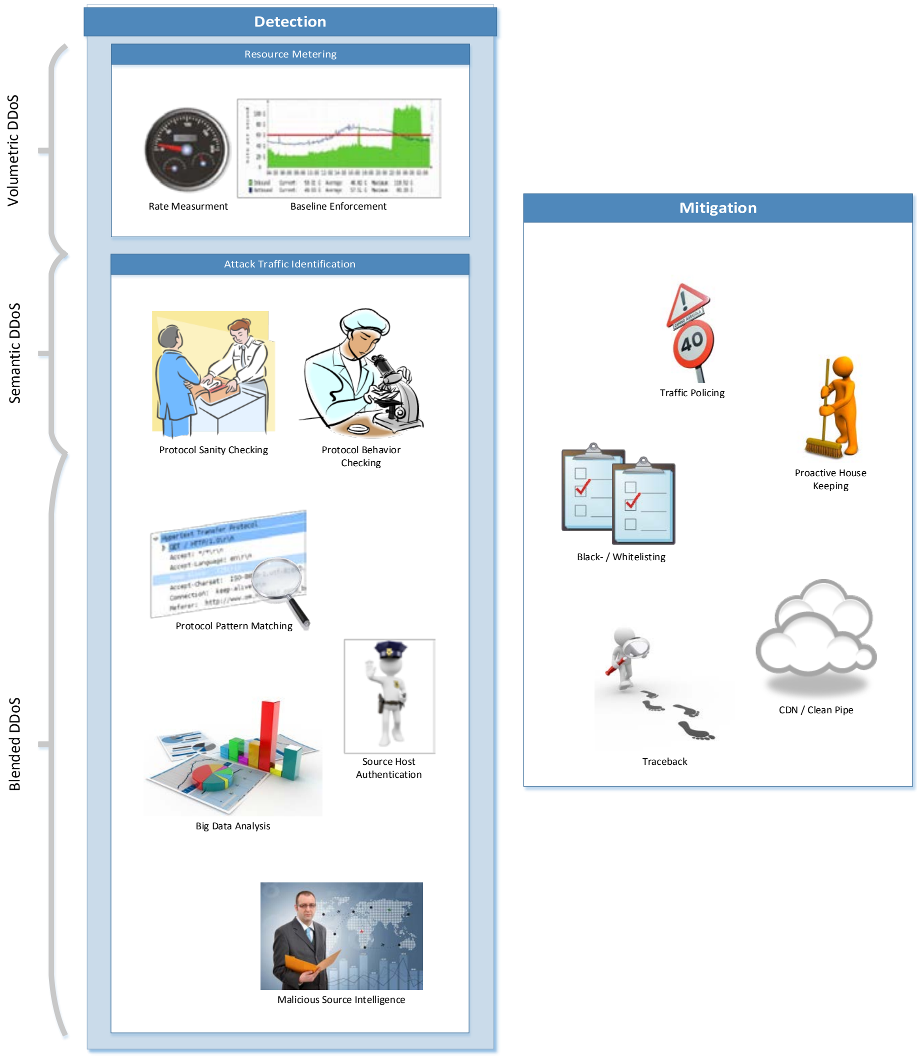


Figure 2. DDoS Mitigation Techniques

3.1 Techniques Primarily Dealing with Volumetric Attacks

A network system has multiple capacity limits, such as:

1. maximum inbound bandwidth (data link layer statistics),
2. maximum number of packet rate (network layer statistics),
3. maximum HTTP request rate (application layer protocol statistics),
4. maximum HTTP object return rate (server load statistics),
5. maximum concurrent TCP connections (system resource statistics), and so on.

Volumetric attacks attempt to exhaust these limits in order to render the system unavailable.

Rate Measurement, Baseline Enforcement and Traffic Policing

Against volumetric attacks, a direct mitigating tactic employs *traffic policing* to curb attack traffic. Common implementations typically involve *baseline enforcement* and *rate limiting*, whereby traffic that exceeds a capacity threshold or otherwise violates predetermined traffic conditions (baseline profile) are forcibly suppressed to ensure conformance with capacity rules. This is usually achieved through selective packet dropping (*traffic shaping*), or outright blacklisting of infringing traffic sources.

An inherent weakness of this approach is that an attacker can probe the target with test traffic to determine the thresholds at which policing will take place. Upon this discovery, the attacker can fire an attack that goes just below the radar, and multiply the firepower by using multiple attack sources.

Indeed, rate metering and baseline enforcement can be applied to specific source IP addresses or to address ranges such as entire subnets. But, a pure traffic policing approach cannot correlate across unrelated sources, because that would require visibility into traffic characteristics deeper than just capacity rule violations. Historically this inherent weakness has given rise to the proliferation of botnets, as they make possible the execution of coordinated attacks across massive unrelated sources which are deadly against these first generation DDoS mitigation techniques.

3.2 Techniques Primarily Dealing with Semantic Attacks

Semantic DDoS attacks exploit weaknesses in protocol, application or other design issues to cause resource starvation. Examples include:

1. Smurf Attack (exploit ICMP reply and IP broadcast behavior),
2. SYN Flood (exploit TCP half-open connection’s provision for waiting),
3. Slowloris Attack [3] (exploit HTTP request’s provision for waiting),
4. Teardrop Attack (crash OS with malformed IP packets),
5. CrashIIS Attack (crash IIS with malformed HTTP GET requests),
6. Apache Killer (exploits weakness in Apache’s range implementation),
7. database amplification attack, i.e. making cheap HTTP requests that involve expensive database queries in rapid succession (exploit request-response cost asymmetry), and so on.

Protocol Sanity and Behavior Checking

Semantic attacks usually follow specific patterns. For instance, Teardrop Attack’s telltale signature is its overlapping IP fragments. Checking for these signatures may not be trivial to implement but nevertheless provides definite criteria for filtering. It is for this reason that *protocol sanity and behavior checking* are mostly effective for catching known semantic attacks.

However, extending sanity checking to cover 0-day semantic attacks by checking for malformed protocol data units (packets, datagrams, segments, HTTP requests, etc.) in general is often met with mixed success. This is because RFCs are often ambiguous about less common conditions, and all networking stack implementations have their own interpretations of the standards and idiosyncrasies. There are also widespread usages that are actually non-compliant — this reality makes an aggressive filtering approach prone to breaking real-world applications.

Interplay among layers of networking protocols further complicates the issue, giving way to ample opportunities for exploitation. One such example is the TCPxHTTP Attack [7].

Proactive Resource Release

Another approach that is most effective against resource starvation attacks is proactive resource release whereby resources prone to starvation are forcibly freed up.

For compatibility and scalability reasons, commercial mitigation solutions are usually deployed externally to individual computer systems and networking devices, treating them as black boxes. This precludes resource release measures that require host-based mechanisms such as enlarging the TCP concurrent connection pool.

That said, resource freeing by means of TCP connection reset can be instrumented externally — sending a TCP RST packet to a server host is sufficient to close and free up a connection. For TCP-based DDoS attacks, forceful TCP connection reset is a very practical control mechanism.

However, proactive resource release can inadvertently disrupt legitimate uses. As such graceful recovery is a desirable compensatory feature to have.

Resource holding attacks like Slowloris [3] are best handled with proactive resource release. However, the detection of these attacks often requires matching predefined traffic behavior profiles. Even more troublesome for modified implementations, for which no predefined profile would work, detection would have to resort to spotting deviations from normal traffic.

Proactive resource release can by definition be circumvented by staying just below release threshold.

3.3 Techniques Primarily Dealing with Blended Attacks

In response to mitigation techniques that excel at filtering out malformed traffic, blended attacks gained popularity. They strive to evade filtering by mimicking legitimate traffic, such as for HTTP requests to bear believable real-world User-Agent string, and have variable lengths.

Traffic Statistics and Behavior Big Data Analysis

Traffic statistics and behavior big data analysis aims at building a baseline profile of traffic such that significant deviation at runtime can trigger a red flag. Generally datamining can work on the following three aspects:

Protocol Parameter Profiling — Historical implementations have given individual protocols certain common choices for parameter values in normal traffic, for instance, a normal TCP SYN packet (created via `connect()`) is 48 to 60-byte long, has a TTL value of 64 and has the DF bit set, whereas SYN packets commonly found in DDoS attacks are usually much shorter and have different values for TTL and DF, mainly due to the use of raw packet crafting and for bandwidth economy. Another example is that a majority of legitimate ICMP Pings have a TTL value of either 128 (for Windows) or 255 (for Linux). Likewise, frequency distribution of common values can be drawn for upper layer attributes such as HTTP methods and User-Agent strings.

Traffic Behavior Profiling — Certain behavior features can be mined from traffic to individual sites. The most prominent aspect is that of temporal activity patterns. For instance, web games traffic generally picks up from 6am in the morning, gradually ramping up until 9am at which point traffic plummets, only to pick up briefly again during lunch hours, with 7pm to 3am being the most heated gaming time period. Other useful features to be mined include proportions of individual protocols, average session lengths and frequency distribution of TCP flags.

Demographic Profiling — Visitors to a website exhibit a certain demographic profile, such as where they come from and what browsers they use. Likewise, other network destinations tend to cater mainly to a specific group of similar clients. Detection of these correlations will facilitate red-flagging of abnormal traffic. For instance, a surge of visitor traffic from Russia to a web site written only in German is almost always indicative of an ongoing DDoS attack.

Protocol Pattern Matching

The technology behind protocol pattern matching can be as simple as old-school attack signature matching, yet highly effective. This is because many widespread DDoS tools generate traffic with idiosyncratic packet patterns that can be easily identified. For instance, HOIC [5] version 2.1 makes an "`HTTP/1.0`" GET request with a "`Host: `" header which is also strangely listed last, and before header payloads telltale double-spaces can be seen.

Whereas matching can be applied to payloads just as well as headers, implementations are not as common due to the high cost associated with payload matching.

A high-confidence match would require multiple matching criteria to all be satisfied. For this reason, regular expression algorithms are usually employed for efficient execution. Due to the high cost associated with matching after request reassembly, a common implementation shortcoming is the inability to match across individual packets, making it possible to evade matching by fragmenting requests into multiple packets.

Source Host Verification

Source host verification aims at identifying illegitimate sources (mainly spoofed addresses and zombie computers running specialized DDoS traffic generators) and blocking them. A step up from passively inspecting traffic to look for red flags, this approach actively probes the sources for verification, usually via checking for features normally only found in full-fledged browsers and TCP/IP stacks.

TCP SYN Authentication — With this method, the authenticity of the client's TCP stack is validated through testing for correct response to exceptional conditions. Common tactics include sending back a RST packet on the first SYN expecting the client to retry, as well as deliberately sending back a SYN-ACK with wrong sequence number expecting the client to send back as RST and then retry.

The best approach to defeating this method is to have the OS networking stack handle such tests.

HTTP Redirect Authentication — The basic idea is that a legitimate browser will honor HTTP 302 redirects. As such, by inserting artificial redirects, it would be safe to block non-compliant clients.

Clearly, it is not particularly difficult to implement just enough support for HTTP redirects to fool HTTP Redirect Authentication.

HTTP Cookie Authentication — This method works like, and is usually used together with, HTTP Redirect Authentication. Essentially, browser's cookie handling is tested. Clients that do not carry cookies in subsequent HTTP requests are clearly suspect and can be safely blocked.

As in adding support for HTTP Redirect Authentication, cookie support does add additional complexity and reduces raw firepower in DDoS attacks.

JavaScript Authentication — With JavaScript Authentication, a piece of JavaScript code embedded in the HTML is sent to clients as a challenge. Obviously, only clients equipped with a full-fledged JavaScript engine can perform the computation. It would not be economical for DDoS attack tools to hijack or otherwise make use of a real heavyweight browser to carry out attacks.

An extended implementation would make use of UI elements such as JavaScript dialog boxes or detecting mouse movements in order to solicit human inputs. Going this far would impede otherwise legitimate automated queries, making this mechanism only suitable for a subset of web sites designed for human usages, but not those web APIs such as REST web services.

Attack tools however, can incorporate standalone JavaScript engines such as Spidermonkey[1] or V8[2] which are relatively lightweight and would not bog down attacks too much. As of this writing, the major challenge with this bypass method lies with adequate DOM implementations.

CAPTCHA Authentication — A very heavy-handed approach that involves human intervention whereby CAPTCHA challenges are inserted into suspicious traffic. If the client end is successful in solving the CAPTCHA, it will be whitelisted for a certain

[1] https://developer.mozilla.org/en-US/docs/SpiderMonkey

[2] https://code.google.com/p/v8/

period of time or for certain amount of subsequent traffic, after which it will need to authenticate itself again.

This method is, in itself, rather intrusive and in practice used only sparingly. While far from easy, automated means to solve CAPTCHA do exist and is a topic of ongoing research.

3.4 Generally Applicable Detection Methods

Source Isolation

Source Isolation mechanisms aim to figure out where DDoS attack traffic comes from and stop it at the sources. If an attacker is able to bypass attack identification (and detection in general), such as with detection techniques discussed in this paper, no mitigation including Source Isolation will be triggered. In practice, the effectiveness of source isolation is questionable due to the extensive use of botnets. An example of source isolation implementation is given in [11] Source Path Isolation Engine (SPIE).

Malicious Source Intelligence

Much like source isolation, blocking decisions can also be based on attack traffic identified elsewhere (in this case, primarily through thfiird-party lter lists), saving identification burden and reducing delays in mitigation. Trust placed on third parties must be carefully managed however.

3.5 Generally Applicable Mitigation Methods

Blacklisting

Blacklisting is essentially a short circuit mechanism aimed at cutting down the tedious work of having to classify individual flows by outright dropping traffic from entire IP addresses for a certain period of time or for a certain amount of traffic volume immediately upon identification of one attack from those sources. Blacklisting cannot be permanent, as IP addresses can be dynamically assigned and zombied computers can be repaired. Mitigation bypass should strive to avoid triggering blacklisting.

Whitelisting

In contrast to blacklisting, whitelisting preapproves traffic from entire IP addresses for a certain period of time or for a certain amount of volume upon determining those sources are well behaving.

A common exploit against whitelisting mechanisms is to have traffic sources send legitimate traffic long enough, and to pass authentication if required, for those sources to trigger whitelisting, and then start DDoS attacks under the protection of being whitelisted.

3.6 Other Mitigation Solutions And Tools

Clean Pipes

So-called clean pipes work by redirecting all incoming traffic to a scrubbing center which applies DDoS defense mechanisms including all other mitigation techniques documented in this paper, in order to scrub them clean—taking out attack traffic leaving only clean traffic to the backend.

A significant drawback to this asymmetric approach is that only traffic inbound to backends ever gets to be inspected by the scrubbing center (return traffic goes directly from the backends to the clients). This limited visibility precludes stateful inspection that requires looking at traffic in both directions. For instance, clean pipes can be oblivious to TCP Half-Open Attacks by following SYN packets with an appropriate ACK, unless information about return traffic is somehow fed back from peer networks to complete the picture.

Secure CDNs

While not initially designed as a DDoS mitigation mechanism, CDNs nevertheless are sometimes (mis)used as a preemptive defense to alleviate DDoS damages.

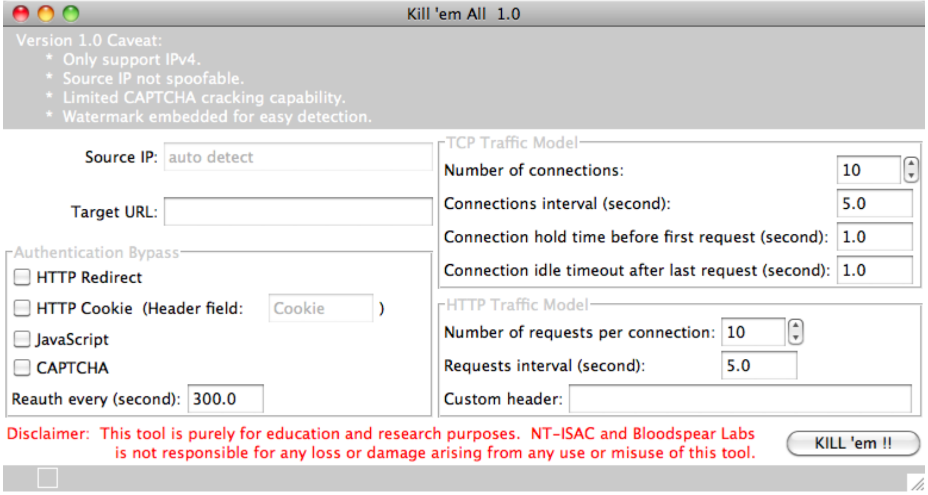
The problem with this approach is that backends typically trust the CDN unconditionally, making them susceptible to attacks spoofing as traffic from the CDN. Ironically, the presence of CDN can inadvertently worsen a DDoS attack by adding its own headers, occupying even more bandwidth.

Firewalls and IPS Systems

Traditional protection devices such as firewalls and IPS systems [8] generally have many of the mitigation techniques dealing with volumetric and semantic attacks implemented. It is against blended attacks where they fall short.

4 Performance Testing

Through extensive testing we have developed a sure-fire methodology capable of bypassing most commercial mitigation solutions. The key idea is to satisfy source host verification (authentication) so as to be cleared of further scrutiny, and then send attack traffic staying just below traffic threshold. A proof-of-concept tool “Kill ’em All” developed to demonstrate the effectiveness of this approach, is shown in Figure 3.



Kill 'em All 1.0

Version 1.0 Caveat:
- Only support IPv4.
- Source IP not spoofable.
- Limited CAPTCHA cracking capability.
- Watermark embedded for easy detection.

Source IP: auto detect

Target URL:

Authentication Bypass

☐ HTTP Redirect

☐ HTTP Cookie (Header field: Cookie)

☐ JavaScript

☐ CAPTCHA

Reauth every (second): 300.0

TCP Traffic Model

Number of connections: 10

Connections interval (second): 5.0

Connection hold time before first request (second): 1.0

Connection idle timeout after last request (second): 1.0

HTTP Traffic Model

Number of requests per connection: 10

Requests interval (second): 5.0

Custom header:

Disclaimer: This tool is purely for education and research purposes. NT-ISAC and Bloodspear Labs is not responsible for any loss or damage arising from any use or misuse of this tool.

Figure 3. Proof-of-Concept Tool "Kill 'em All"

Tests were conducted against products:

1. Arbor Peakflow SP Threat Management System (TMS) version 5.7, and
2. NSFocus Anti-DDoS System (ADS) version 4.5.88.2.026

as well as secure CDN services:

3. Clouflare Business, and
4. Akamai.

We are convinced TMS and NSFocus ADS represent a majority of the market, with the former most prevalent among Fortune 500 enterprises and the latter deployed in most every publicly listed company in mainland China.

4.1 Testing Methodology

Tests were conducted against products and cloud services. For product testing an attack workstation was connected to a web site through the DDoS mitigation device under test. For cloud service testing a web site was placed under the protection of the service under test, and then subjected to attacks from a workstation directing attacks towards it through the internet.

In order to simulate normal short-term browsing conditions, in all tests a single TCP connection was used to carry a multitude of HTTP requests and responses. Under this vigorous arrangement not a single attack identification mechanism can be triggered lest the entire connection gets blocked.

During testing, attack traffic was sent to the backend at which point received traffic was compared against the original generated traffic. Bypass was considered successful if all attack traffic passed through intact.

4.2 Testing Results

Attacks with bypass capability were applied against individual detection techniques as implemented on the aforementioned products and services. During the attack, effectiveness of the attacks was evaluated and observations were recorded as shown in Table 1 below. A "✓" means the bypass was successful with no mitigation activity observed.

Detection Techniques	Arbor Peakflow SP TMS	NSFocus ADS	Cloudflare	Akamai
Rate Measurement / Baseline Enforcement	✓ (Zombie Removal, Baseline Enforcement, Traffic Shaping, Rate Limiting)	✓	N/A	N/A
Protocol Sanity & Behavior Checking	✓ (HTTP Countermeasures)	✓	N/A	N/A
Proactive Resource Release	✓ (TCP Connection Reset)	✓	N/A	N/A
Big Data Analysis	✓ (GeoIP Policing)	— (Not implemented in ADS)	N/A	N/A
Malicious Source Intelligence	✓ (Black White List, IP Address Filter List, Global Exception List, GeoIP Filter List)	— (Not implemented in ADS)	N/A	N/A
Protocol Pattern Matching	✓ (URL/DNS Filter List, Payload Regex)	✓	N/A	N/A
Source Host Verification				
TCP SYN Authentication	✓	✓	N/A	N/A
HTTP Redirect Authentication	✓	✓	✓	N/A
HTTP Cookie Authentication	✓	✓	✓	N/A
JavaScript Authentication	— (Not implemented) in TMS)	✓	✓	N/A
CAPTCHA Authentication	— (Not implemented in TMS)	✓	✘	N/A

Table 1. Mitigation bypass testing results.

With reference to Arbor Network's A Guide for Peakflow® SP TMS Deployment[3], against TMS we were able to defeat all documented or otherwise active detection techniques relevant to HTTP DDoS attacks, passing through the TMS unscathed.

Attacks against NSFocus ADS[4] were met with remarkable success despite the presence of heavy-handed defenses including CAPTCHA Authentication — we were able to achieve a remarkable 50% success rate solving ADS's CAPTCHA implementation with our OCR algorithms. Due to the shotgun approach to attack, and that getting whitelisted is a big win for the attacker, a 50% success rate for solving CAPTCHA is much more impressive than it may appear at first glance.

Cloudflare essentially employs JavaScript that implements all JavaScript, Cookie and Redirect Authentications in one. We were successful in defeating them all and pushing attack traffic to the backend. Even though Cloudflare does support CAPTCHA Authentication, we observed that its use is not particularly prevalent in the wild, and for the purpose of our PoC since we have already demonstrated a workable solution against CAPTCHA for ADS, we have opted not to repeat this for Cloudflare.

Akamai has implemented source host verification techniques in its security solutions for a few months now, with which according to marketing brochure [8] visitors will be redirected to a JavaScript confirmation page when traffic is identified as potentially malicious. However, despite our best effort sending big traffic to our testing site bearing random HTTP query strings (in order to thwart caching) we have been unable to trigger that feature. Whereas we cannot rule out the remote possibility that our test traffic was way below detection threshold, a much more plausible reason might be that our traffic was indistinguishable from that generated by a real browser.

5 Discussions and Next-Gen Mitigation

In this era of blended attacks, detection methods designed to pick out bad traffics are rendered fundamentally ineffective. The reason why today to a certain extent they still work is mainly due to implementation immaturity (e.g. the lack of ready-to-use JavaScript engine with a workable DOM). Obviously these hurdles can be easily overcome given a little more time and development resources, as our research demonstrated.

A notable exception is the use of CAPTCHA. Despite the fact that we have also demonstrated defeating certain CAPTCHA implementations in use on security products, and that there have been promising results from fellow researches [9] as well, admittedly CAPTCHA still represent the pinnacle of source host verification technique. However, CAPTCHA is necessarily a heavy-handed approach that materially diminishes the usability and accessibility of protected web sites. Specifically, automated queries and Web 2.0 mashing are made impossible. This shortcoming significantly reduces the scope of its application. It is therefore not surprising that CAPTCHA is often default off in security service offerings.

[3] http://www.arbornetworks.com/component/docman/doc_download/301-threat-management-system-a-technical-overview?Itemid=442

[4] http://www.nsfocus.com/jp/uploadfile/Product/ADS/White%20Paper/NSFOCUS%20ADS%20White% 20Paper.pdf

5.1 Next-Generation Mitigation

Seeing as that the underlying issue with a majority of DDoS attacks these days is their amplification property, which tips the cost-effectiveness balance to the attackers' favor, we are convinced that a control mechanism based on asymmetric client puzzle is the solution, as it presents a general approach that attacks directly this imbalance property, making it a lot more expensive to execute DDoS attacks. Prior researches include the seminal Princeton-RSA paper [10] and [11].

6 Acknowledgement

This research was made possible only with data and testing resources graciously sponsored by Nexusguard Limited[5] for the advancement of the art.

References

[1] M. Prince, "The DDoS that Knocked Spamhaus Offline (And How We Mitigated it)," 20 March 2013. [Online]. Available: http://blog.cloudflare.com/the-ddos-that-knocked-spamhaus-offline-and-ho.

[2] C. Weinschenk, "Attacks Go Low and Slow," IT Business Edge, 3 August 2007. [Online]. Available: http://www.itbusinessedge.com/cm/community/features/interviews/blog/attacks-go-low-and-slow/?cs=22594.

[3] R. Hansen, "Slowloris HTTP DoS," 7 June 2009. [Online]. Available: http://ckers.org/slowloris/.

[4] Carnegie Mellon University, "CERT® Advisory CA-1998-01 Smurf IP Denial-of-Service Attacks," 5 January 1988. [Online]. Available: http://www.cert.org/advisories/CA-1998-01.html.

[5] J. Breeden II, "Hackers' New Super Weapon Adds Firepower to DDOS," GCN, 24 October 2012. [Online]. Available: http://gcn.com/articles/2012/10/24/hackers-new-super-weapon-adds-firepower-to-ddos.aspx.

[6] E. Chien and P. Szor, "Blended Attacks Exploits, Vulnerabilities and Buffer-Overflow Techniques in Computer Viruses," 2003. [Online]. Available: http://www.symantec.com/avcenter/reference/blended.attacks.pdf.

[7] T. Miu, A. Lai, A. Chung and K. Wong, "DDoS Black and White "Kungfu" Revealed," in *DEF CON 20*, Las Vegas, 2012.

[8] Akamai, "Akamai Raises the Bar for Web Security with Enhancements to Kona Site Defender," 25 February 2013. [Online]. Available:

[5] http://www.nexusguard.com/

http://www.akamai.com/html/about/press/releases/2013/press_022513.html.

[9] DC949, "Stiltwalker: Nucaptcha, Paypal, SecurImage, Slashdot, Davids Summer Communication," 26 July 2012. [Online]. Available: http://www.dc949.org/projects/stiltwalker/.

[10 B. Waters, A. Juels, J. A. Halderman and W. F. Edward, "New Client Puzzle Outsourcing Techniques for DoS Resistance," in *ACM Conference on Computer and Communications Security (CCS)*, 2004, 2004.

[11 D. Stebila, L. Kuppusamy, J. Rangasamy and C. Boyd, "Stronger Difficulty Notions for Client Puzzles and Denial-of-Service-Resistent Protocols," in *RSA Conference*, 2011.

[12 R. Kenig, "How Much Can a DDoS Attack Cost Your Business?," 14 May 2013. [Online]. Available: http://blog.radware.com/security/2013/05/how-much-can-a-ddos-attack-cost-your-business/.

[13 H. Aljifri, "IP Traceback: A New Denial-of-Service Deterrent?," 11 June 2003. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1203219.

[14 K. S. Chao Gong, "A More Practical Approach for Single-Packet IP Traceback using Packet Logging and Marking," 29 August 2008. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4408575.

[15 W. Z. M. G. Yang Xiang, "Flexible Deterministic Packet Marking: An IP Traceback System to Find the Real Source of Attacks," 1 August 2008. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4585371.