Global Information Assurance Certification Paper

Leveraging the Load Balancer to Fight DDoS

GIAC (GCIA) Gold Certification

Author: Brough Davis, brough.davis@gmail.com
Advisor: Kristof Boeynaems

Accepted: December 13th 2009

Abstract

Can Load Balancing hardware traditionally used to help scale servers also help against DDoS attacks? Many DDoS mitigation techniques traditionally focus on either getting help from the upstream ISP or reconfiguring the end servers that are getting attacked. Many people forget that the load balancing hardware that typically lives in between the edge routers and end servers may have features that can help fight against a DDoS attack. In fact, some common load balancing features have unintended benefits that can help fight against DDoS attacks. This paper will explain some common DDoS attack methods currently seen today and describe different methods for dealing with them by leveraging the load balancing hardware commonly found in web-hosted environments.

1. Introduction

DDoS (Distributed Denial of Service) attacks have been an ever increasing concern in the Internet world. As technologies become less expensive and the Internet grows it is becoming easier and profitable for criminal organizations and the naive vandal to launch destructive attacks on organizations (Mikovic et al., 2005). DDoS attacks are also becoming common tools for governments or activist groups to help serve political agendas (Ristic, 2005). Security professionals will likely always be one step behind new attack methods. In order to understand how Load Balancing technologies can be used to help mitigate DDoS attacks a quick DDoS and Load Balancing primer is needed.

1.1 DDoS Primer

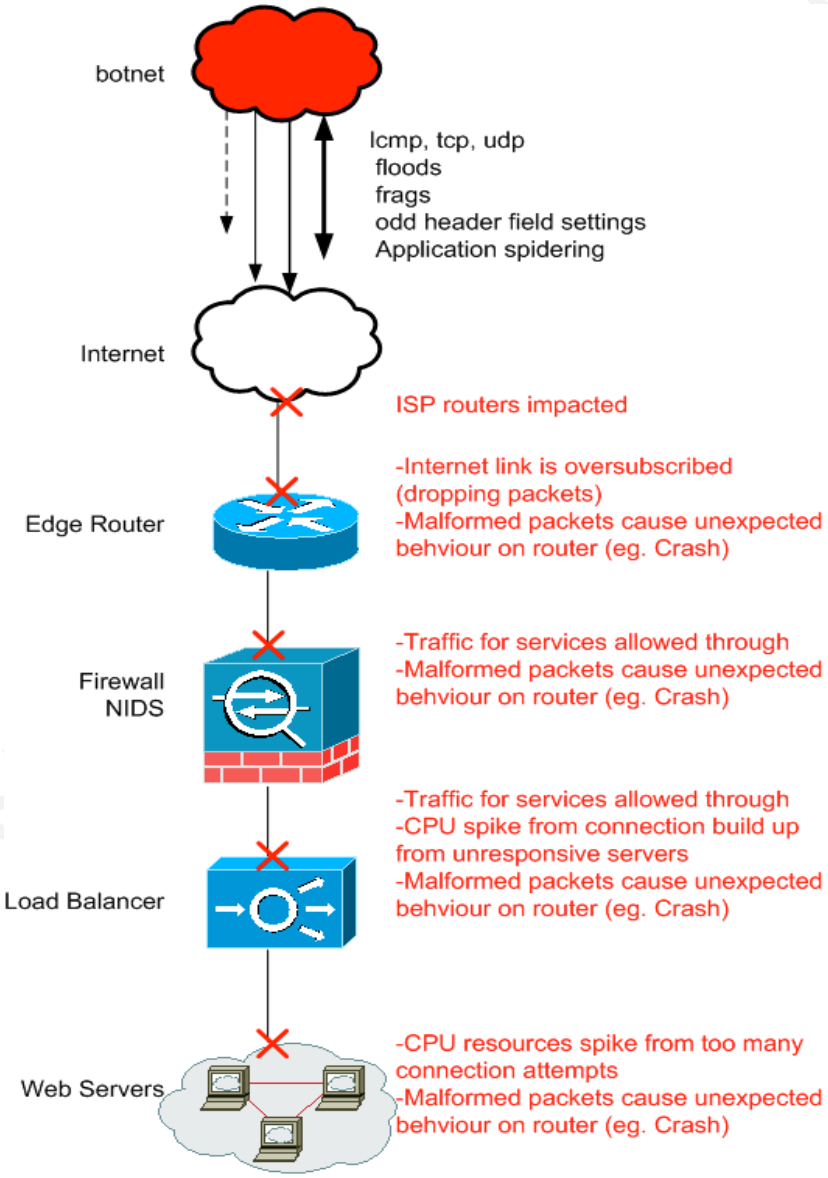
A Denial of Service (DoS) is a term commonly used to describe an intentional attack on a service, such as a web site, with the main goal of disabling the service or keeping other people (customers) from being able to connect to that service. Historically, a DoS attack came from one source location. DoS attacks evolved over time to become Distributed attacks with ever changing sources. Many DDoS attacks are sourced from bot networks or botnets. Botnet is an Internet term for a collection of software agents that were typically installed by trojans or worms unknowingly to the system users. Botnet's even have their own black market for selling time to anyone that wants to use the bot nets for nefarious reasons such as DDoS, spam, and credit card aggregation (Mikovic et al., 2005). The main goal of a DDoS attack is to overwhelm server resources, network resources, or both.

Server Overload - A DDoS attack will sometimes overload the maximum number of service connections a server can transact. If all the possible server connections are taken up by a DDoS attack a valid Internet client will not be able to access the service.

Traffic Overload - A DDoS attack may send enough traffic to saturate the Internet circuits that are servicing a server or service. If the Internet circuit is saturated by DDoS

traffic then valid customer traffic will be dropped by the upstream router of the saturated connection.

The below diagram, created by the author, provides an overview of the typical types of traffic sent from a bot net DDoS attack to a targeted service. Note that at each level in the network a possible issue can occur that will lead to denial of service. It should also be mentioned that at each level in the network different technologies and methods can be used to mitigate DDoS attacks.



1.2 Load Balancing Technology Primer

Load Balancing technologies on the Internet originated in order to help spread the processing requirements for incoming connections across multiple servers. The traditional goal of Load Balancing is to spread the processing requirements over multiple systems. This has the benefit of reducing the investment from one expensive server to many less expensive servers in addition to adding redundancy and increasing availability. In the Internet the two main services commonly load balanced are HTTP and HTTPS/SSL.

Vendors started using ASIC (Application Specific Integrated Circuits) that were designed for fast switching and routing that are not commonly found in common Server Hardware. Hardware Load balancing is very useful for quickly making load balancing decisions on a large amount of traffic.

Over the years Load balancing switches became better at inspecting traffic in order to make load balancing decisions. The term Load Balancing switch has become antiquated and is being steadily replaced with the term Application Delivery Controller (ADC). The ADC's have additional features not found in older load balancing technologies such as content manipulation, advanced routing strategies as well as highly configurable server health monitoring. ADC's tend to offer features like compression, cache, connection multiplexing, application layer security, SSL offload, content switching combined with basic server load balancing. These next generation Load Balancing switches offer additional features that may help fight against DDoS attacks (Fabbi & Skorupa, 2009).

2. DDoS Trends

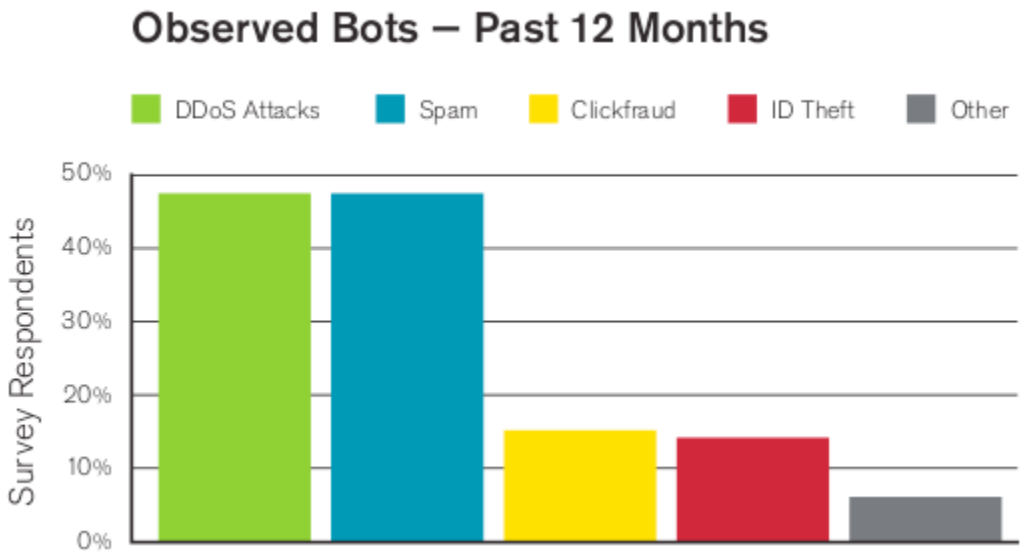
DoS and DDoS attacks having been around for quite some time and are changing as new technologies change. Unfortunately, there is very little data surrounding DDoS attacks that is publicly available. Arbor Networks is one of the few companies that performs yearly DDoS trending reports based on feedback from ISP's, web-hosting companies, Universities, etc. Below are some reported trends from the "World Wide

Infrastructure Security Report 2009" that Arbor Networks released in 2010 (McPherson & Rolands, 2010). Additional data from the Arbor Report is included in the Appendix. Trending DDoS historical behavior may help companies better focus where and how to focus DDoS mitigation techniques.

2.1 Bots and DDoS

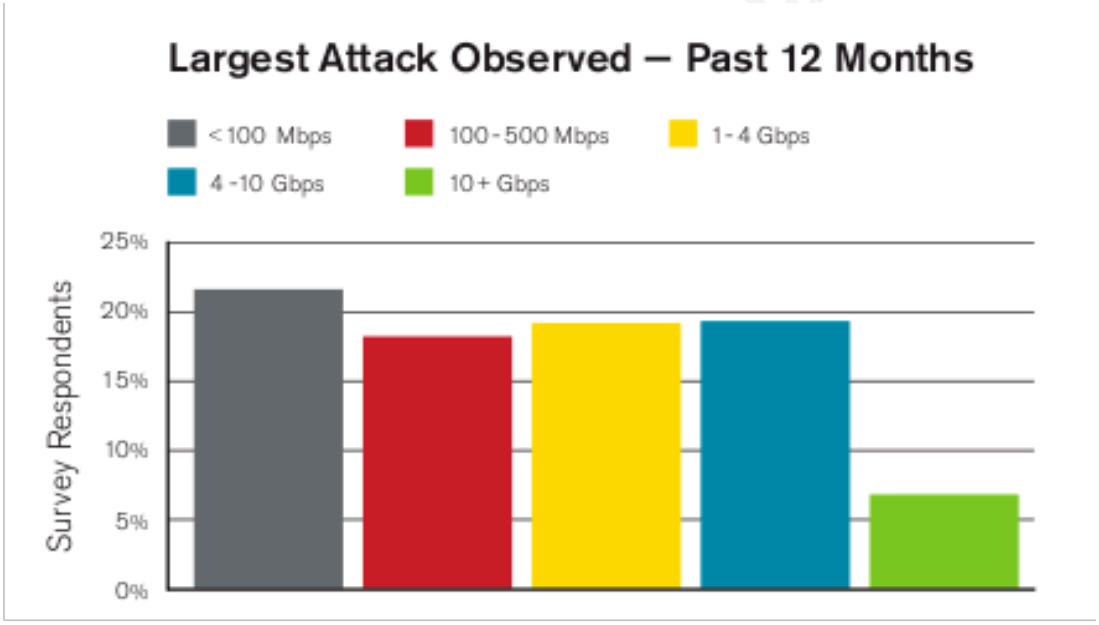
The below diagram depicts that nearly 50% of all bot activity during 2009 was related to DDoS Attacks taken from companies that participated in Arbor Networks Survey.

Note: The following diagrams are using data taken from the Arbor network surveyed companies. It is important to consider this data a small subset of the total population. This data could be skewed to favor Arbor Network DDoS mitigation products. This data is however the only publicly known data the author was able to find that can be used to analyze existing DDoS trends and patterns.



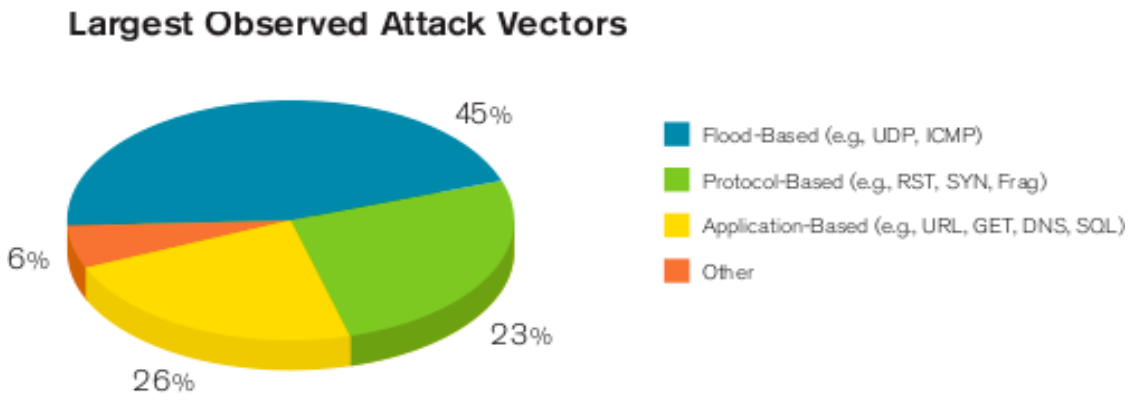
2.2 Build it and they will come

The following graph shows that approximately 60% of surveyed companies experienced attacks that generated over 1 Gbps of traffic. Since many companies have Internet connections of less than 1 Gbps this could indicate attacks hitting multiple links or sites for a company's services. This begs the question that no matter how big a site can be built and distributed, will it be enough to ward off a DDoS attack from 10,000 or more bots. If building larger networks is a part of the solution then what do smaller companies do that do not have the budget for increasing their infrastructure? Many companies are hoping to move their sites to cloud networks in which the cloud provider has a large and highly available network.



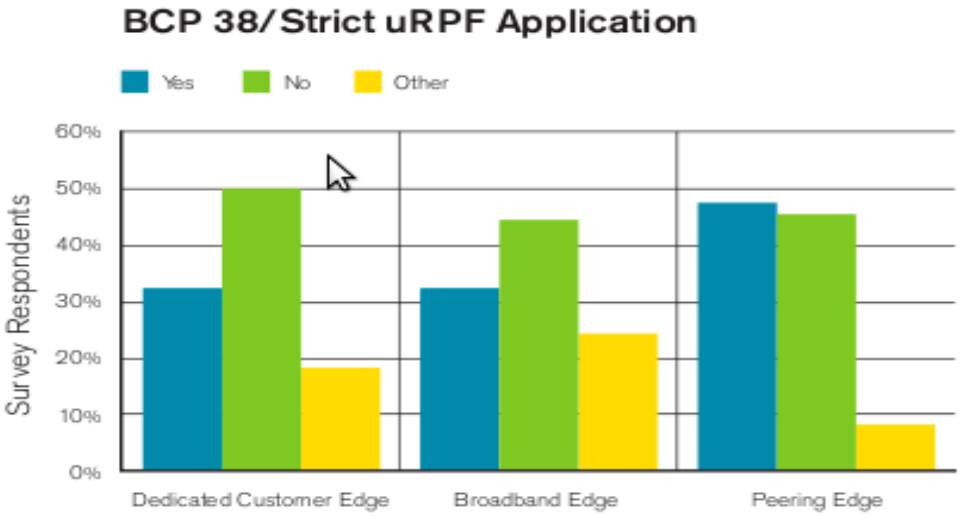
2.3 Narrowing

As more and more companies begin to utilize common DDoS mitigation techniques, attackers have to narrow the focus of the attacks on the specific applications they want impacted. With the ability to send from multiple bot hosts, an attacker can also decide to not spoof source address traffic. This allows the attacker the option of creating full interactive connections with applications. Full TCP 3-way handshakes to port 80 or 443 in rapid succession can easily overwhelm a service if done from 10,000 bots. The author has personally seen attacks on the Internet that have used large bot/zombie networks sending full TCP 3-way handshakes to web-hosted services. The attacker can even take the next step and dynamically interact with the application. Common bots such as agobot have had the ability for quite some time to do a **recursive HTTP flood** which means that the bots start from a given HTTP link and then follow all links on the provided website in a recursive way also called spidering. The below diagram depicts the largest observed attack vectors from surveyed companies. The diagram shows that flood-based and protocol-based attacks still represent about 68% of observed attacks while application-based attacks are around 26%. This could imply that if basic DDoS mitigation techniques are in place that the below graph would show a higher percentage of application-based attack vectors.



2.4 Spoofing Still Works

Many DDoS attacks on the Internet would seem to still leverage spoofing source IP address information. The below graph shows that approximately 50% of all surveyed companies do not use any Unicast Reverse Path Forwarding (uRPF) methods to prevent spoofing (Ferguson, 2000). uRPF is a method used on many edge routers to prevent any traffic into the network that has obvious invalid source address information. uRPF specifically prevents any source IP addresses that are known internally to the network. It would make sense for many DDoS attacks to utilize spoofed source addressing if many companies are not implementing uRPF or similar mechanisms to deny spoofed traffic into their network.



It should be noted that spoofing is becoming less and less common. One reason for the decrease is that more and more companies are implementing simple spoofing protection. Although spoofing works with the majority of attacks it is becoming less effective. Another disturbing reason why spoofing attacks are becoming less common is that attackers are gaining access to larger and larger bot-networks that do not need to spoof traffic in order to cause a denial of service. Richard Bejtlich writes in his book, The Tao of Network Security Monitoring, that even as early as 2003 he and other industry professionals started noticing intruders spoofing source IP's less and less (Bejtlich, 2005).

2.5 Bot DDoS Options

Trying to figure out how to protect against a DDoS attack can be very frustrating since it can be difficult to distinguish what traffic is being sent by a bot versus a live person (e.g. customer). In order to help clarify how attacks may be performed it's important to review the bot applications themselves to see what typical options are available to an attacker. Some common bots actually have public source licenses so it's very easy to get the code and review what DDoS options are available to an attacker. It should be noted that any attempt to get access to bot software can be fraught with danger and should be done in a sandbox or through a third party researcher that has cleaned any viruses or essentially neutered the code for research purposes only. The following are some DDoS options found in common Bots (McPherson, 2009).

2.5.1 SDBot

SDBot has some very simple but possibly destructive DDoS option including TCP syn, UDP, and ICMP floods.

```
syn [ip] [port] [seconds|amount] [sip] [sport] [rand] (sdbot 05b pure
version)
udp [host] [num] [size] [delay] [[port]]size (sdbot 05b ago version)
ping [host] [num] [size] [delay]num
```

2.5.2 UrXbot

Urxbot DDoS options go a little further than SDBot by adding additional TCP and ICMP flood options.

```
ddos.(syn|ack|random) [ip] [port] [length]
(syn|synflood) [ip] [port] [length]
(udp|udpflood|u) [host] [num][ [size] [delay] [[port]]
(tcp|tcpflood) (syn|ack|random) [ip] [port] [time]
(ping|pingflood|p) [host] [num][ [size] [delay]
(icmpflood|icmp) [ip] [time]
```

2.5.3 Agobot

Agobot was written in C++ and assembly and was released under GNU General Public License (GPL). It has been noted to have been written very well and its popularity has spawned many different variants such as Phatbot and Forbot. The below options have common DDoS feature that would be expected such as SYN flood, ICMP flood, and UDP flood, as well as a targa attack which sends malformed/unexpected packets (fragmented, IP options, etc.). The alarming option is the **httpflood** option which not only sends full TCP HTTP connections but also spiders the website under attack. This option has been around a while and the other reason why it may not be used as much is 1) no one knows when it is used 2) the other common DDoS options work with the majority of targets 3) there is a bot restriction such as not enough bots to complete 3-way TCP handshakes (Drupal, 2008).

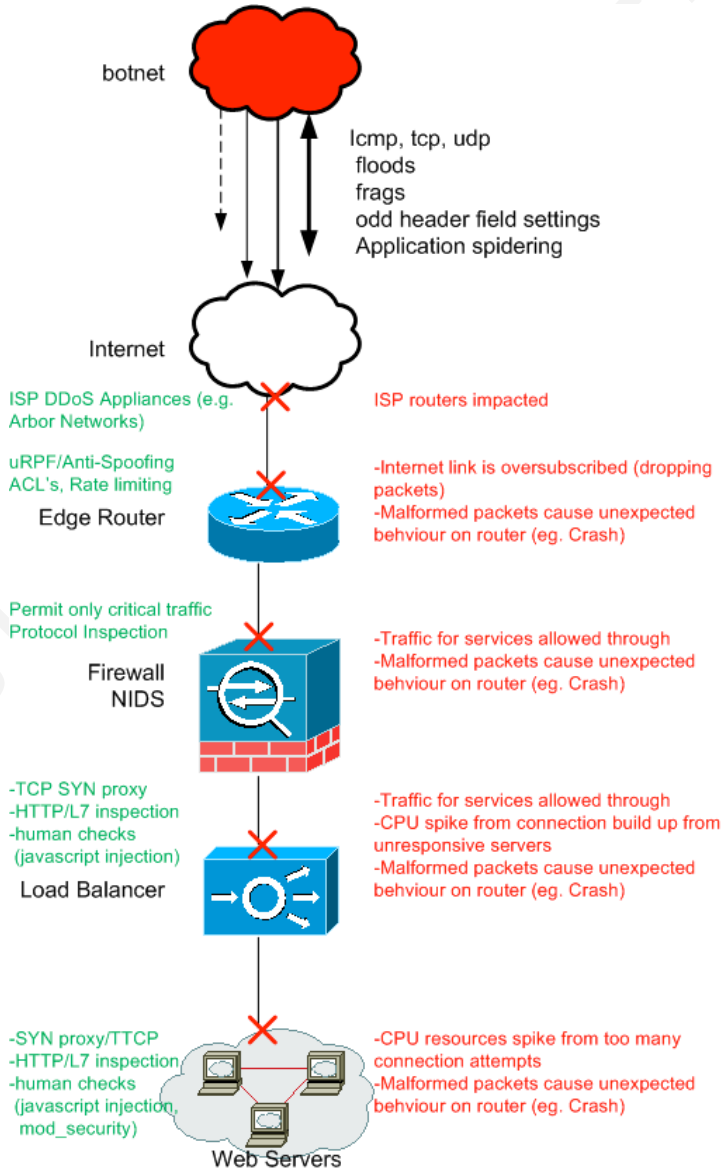
```
ddos.phatwonk [host] [time] [delay]
    starts leet flood

      Starts a SYN-flood on ports 21,22,23,25,53,80,81,88,
                  110,113,119,135,137,139,143,443,445,1024,1025,1433,
                  1500,1720,3306,3389,5000,6667,8000,8080

ddos.phatsyn [host] [time] [delay] [port]
    starts syn flood

ddos.phaticmp [host] [time] [delay]
        starts icmp flood

ddos.synflood [host] [time] [delay] [port]
        starts an SYN flood

ddos.updflood [host] [port] [time] [delay]
        start a UDP flood

ddos.targa3 [host] [time]
    start a targa3 flood

        Implements the well known DDoS attack Mixter authored in 1999.
        /*
        * targa3 - 1999 (c) Mixter <mixter@newyorkoffice.com>
        *
        * IP stack penetration tool / 'exploit generator'
        * Sends combinations of uncommon IP packets to hosts
        * to generate attacks using invalid fragmentation, protocol,
        * packet size, header values, options, offsets, tcp segments,
        * routing flags, and other unknown/unexpected packet values.
        * Useful for testing IP stacks, routers, firewalls, NIDS,
        * etc. for stability and reactions to unexpected packets.
```

```
        * Some of these packets might not pass through routers with
        * filtering enabled - tests with source and destination host
        * on the same ethernet segment gives best effects.
        */
        taken from
        http://packetstormsecurity.org/DoS/targa3.c
        </mixter@newyorkoffice.com>

ddos.httpflood [url] [number] [referrer] [recursive = true||false]
       starts a HTTP flood
```

The **ddos.httpflood** option is particularly dangerous since it fetches websites from a web server. If "recursive" is set, the bot parses the replies and follows links recursively.

3. DDOS Mitigation

DDoS mitigation can be performed at almost every layer in the network. The following section is a brief review of common DDoS mitigation methods across the Edge routing layer and the Server/Application layers. By understanding how other DDoS defense methods work within a network a better idea of how Load Balancing/ADC devices can help supplement an overall DDoS defense strategy. The following diagram, created by the author, depicts the same DDoS attack vector diagram previously shown with the addition of possible mitigation methods.



3.1 ISP/Edge

Many companies forget that performing simple spoofing access lists on their edge routers can protect many of the DoS attacks that try to send spoofed traffic. Simple access-lists on the edge routers can prohibit RFC1918 address space along with internal address space that should be routed on the Internet and is likely illegitimate traffic.

Only Allow Needed Traffic

Many DoS attack methods use spoofing and malformed protocol traffic to confuse networking equipment and saturate bandwidth. Much of this type of traffic can be filtered at the edge by only allowing traffic that is necessary. For example if the only public services are running on TCP port 80 and 443 then why should anything else be allowed at the edge. Simple access-lists can be created for edge equipment in order to reduce the possible impact on downstream network equipment (e.g. firewall, IDS, Load Balancing device, servers, and applications). It is important that the edge access lists do not become so complicated that management becomes difficult. When management is too complicated access-lists mistakes are more likely which will make the access-lists ineffective.

Global Server Load Balancing (GSLB)

If the attack is targeting a host name instead of an IP address then Global Server Load Balancing may be used. Even if a simple round robin DNS is used with multiple A records the attack would be effectively split across how many A records are listed for the host name which may lessen the overall impact. With that said an attacker may want to focus on a specific IP address since forgoing a DNS query would reduce the resources on the attackers system in addition to speed the attack up.

The Cloud

Some companies may want to move some of their infrastructure to a cloud service like Amazon EC2. Some people may argue that moving services to a highly available

cloud will make it more difficult for a DDoS attack to be successful. Depending on how the cloud services are configured will dictate how successful an attack will be. Unfortunately, since cloud networks are not open architectures, it is difficult to understand what mechanisms are in place to prevent DDoS from impacting services. It is known those cloud networks are still vulnerable. For example, in October of 2009, a service hosted by Amazon EC2 was DDoS'd (Amazon Web Services, 2009) which impacted the service over a few days. Unfortunately because there is a fundamental separation between application and infrastructure management it was difficult for the company and Amazon to deduce a DDoS was actually occurring until many hours into the attack. Amazon advertises a "proprietary DDoS protection" method in the EC2 Service Agreement[1] but not knowing how they achieve DDoS protection will always be a concern to how effective it actually is.

Rate Limiting

Successfully fighting a DDoS attack means allowing good traffic and denying bad traffic. Rate limiting still allows all the 'bad traffic' to use up all available service connections while 'good' traffic struggles to connect. Rate limiting should only be considered as one of the very last if not very last DDoS mitigation methods. Rate limiting is more about the protection against overloading network and server equipment and less about fighting the DDoS. Rate limiting can be enabled to maintain reasonable management of servers and network devices while continuing to look for other methods to mitigate the attack.

Remote Triggered Black Hole Filtering (RTBH)

Remote Triggered Black Hole Filtering (RTBH) is a method commonly used by ISP's to fight Denial of Service Attacks. RTBH is process for injecting a route into an ISP's network routing protocol that specifically triggers a routing rule to black hole traffic destined to a host or network that is under a DoS attack (Turk, 2004). It is also possible to combine the RTBH method with uRPF (Kumari, 2009) by injecting a route that

1 Amazon EC2 Web Services Agreement can be found at http://aws.amazon.com/agreement/

specifies the attacker's source in order to trigger uRPF to reject any traffic coming from that particular attacker. Both methods have possible issues. RTBH has the negative impact of taking the destination service totally off-line while uRPF has the possibility of rejecting valid source traffic.

3.2 Network IDS

Intrusion Detection Systems such as Snort have some methods for detecting DoS attacks. Most of the signatures however deal with specific vulnerabilities that are unique like the emerging threats (previously bleeding-edge) rule below. The below rule shows a signature that matches NTP traffic with a specific content and rate. Snort and other IDS systems have a much more difficult time detecting and preventing bot traffic that is sending traffic that is almost identical to normal traffic patterns such as HTTP requests.

```
alert udp $HOME_NET 123 -> $EXTERNAL_NET 123 (msg:"ET DOS Potential Inbound NTP
denial-of-service attempt (repeated mode 7 request)"; dsize:1; content:"|17|";
threshold:type limit, count 1, seconds 60, track by_src;
reference:url,www.kb.cert.org/vuls/id/568372; reference:cve,2009-3563;
classtype:attempted-dos; reference:url,doc.emergingthreats.net/2010488;
reference:url,www.emergingthreats.net/cgi-bin/cvsweb.cgi/sigs/DOS/DOS_Ntp;
sid:2010488; rev:2;)
```

It should be noted that since Snort is open source, any custom rule could be written for a specific DDoS concern. This can be very powerful in circumstances where a custom application may be under a DoS attack with specific payload information.

3.3 DDoS Appliances

There are commercial appliances that are sold in the marketplace that claim to help mitigate against DDoS attacks. One such product is from Arbor Networks called the Peakflow SP TMS (Threat Mitigation System). This appliance is commonly used by ISP's in order to use a combination of netflow and BGP routing statistics to build a traffic profile that will help identify large traffic spikes that may be DDoS attacks. These spikes can be mitigated by black-holing the traffic using BGP routing methods. While black-holing the traffic will help other resources that may have been impacted by the DDoS, the

end resource that is black-holed is no longer available which may ultimately serve the attackers end goal. There is a chance that Arbor also has another product call Peakflow X which is made to inspect internal traffic for any systems that may have trojan/bot software and used to launch DDoS attacks. This is similar to the snort rule mentioned earlier.

3.4 Server Side

Server side DoS prevention has mostly revolved around protecting TCP services as well as how the Applications themselves are written. The TCP protocol has evolved slowly as many of the reserved header options are replaced with useful options for security and scalability. Unfortunately in order to use many of these new TCP options both the server and client side have to support them. A statement within the TCP RFC 793 sums up the philosophy of how to adapt to this progression.

[RFC 793] "general principle of robustness: be conservative in what you do, be liberal in what you accept from others."

TCP SYN Cookie

TCP SYN Cookies are the key element of a technique used to guard against SYN flood attacks. Daniel J. Bernstein, the technique's primary inventor, defines SYN Cookies as "particular choices of initial TCP sequence numbers by TCP servers" (Bernstein, 2010). In particular, the use of SYN Cookies allows a server to avoid dropping connections when the SYN queue fills up. The server sends back the appropriate SYN+ACK response to the client but discards the SYN queue entry. If the server receives a subsequent ACK response from the client, the server is able to reconstruct the SYN queue entry using information encoded in the TCP sequence number. The Server creates a TCP sequence number (32 bits) based on a time stamp (5 bits), the MSS value (3 bits), and a cryptographic secret based on source and destination IP address and port as well as time (24 bits). TCP SYN Cookie implementations have been around since 1997. Many Load Balancing device vendors attempt to implement TCP Syn cookie techniques, although their implementations and marketing names may differ slightly. The following

diagram depicts the basic TCP 3-way handshake with the cookie information (Zuquete, 2002).

	client socket state	segment	server socket state
1	CLOSED		LISTEN
2	SYN-SENT	→ ⟨SEQ=x⟩⟨CTL=SYN⟩⟨**tentative TCP options**⟩ →	
3	ESTABL.	← ⟨SEQ=**cookie**⟩⟨ACK=$x+1$⟩⟨CTL=SYN,ACK⟩ ←	
4		→ ⟨SEQ=$x+1$⟩⟨ACK=**cookie**+1⟩⟨CTL=ACK⟩ →	ESTABL.

TCP SYN Cookie Drawbacks

There are, however, a few caveats that take effect when SYN Cookies are in use. First, the SYN Cookie can only use 3 bits to encode available MSS values in the calculated sequence number. This limits MSS values to only 8 possible options (2^3) which restricts certain MSS values and could make it easier for attackers to guess sequence numbers. Second, the server must reject all TCP options (such as large windows), because the server discards the SYN queue entry where that information would otherwise be stored. Third, a connection may freeze when the final ACK of the three-way handshake is lost and the client first awaits data from the server (Bernstein, 2010). For example, if a client completes the three-way handshake and the server does not receive the client's ACK then the connection is never fully opened.

One additional drawback to TCP SYN Cookies is bot-nets using full TCP 3-way handshakes. If an attacker has a large bot-net at their finger tips then they can effectively circumvent TCP SYN Cookies by the sheer volume of clients.

Some attack tools focus more on after a TCP connection is made rather than a SYN Flood. For example, the Naptha attack leaves established TCP connections idle but respond to keep-alive packets so that the connections do not time out (Mikovic et al., 2005). This prevents server kernel resources to be freed up. The attacker could also slowly build up the amount of open connections in order to go undetected by SYN Flood detection mechanisms.

TCPCT

The newer **T**CP **C**ookie **T**ransactions (TCPCT) standard is designed to overcome these shortcomings of SYN cookies and improve it on a couple of aspects. Unlike SYN cookies, TCPCT is a TCP extension and requires support from both endpoints. TCPCT avoids resource exhaustion on server-side by not allocating any resources until the completion of the three-way handshake. Additionally, TCPCT allows the server to release memory immediately after the connection closes, while it persists in the TIME-WAIT state. The immediate reason for the TCPCT extension is deployment of the DNSSEC protocol. TCPCT support was partly merged into the Linux kernel in December 2009, and is included in the 2.6.33 release. The largest drawback to TCPCT is that it requires TCPCT support in the client (initiator) as well as the server (responder) TCP stack (Metzger et al., 2009).

Application "Human Checks"

How can we check if the user at the other end is in fact a human and not software? This is sometimes referred to as a Reverse Turing test. The Turing test is when software tries to convince a human that they are communicating to another human. A Reverse Turing test, or "human check", is the human trying to verify if they are communicating with software or a human (Mikovic et al., 2005). Some web servers are looking at using "human checks" to validate if a bot or an actual human is interacting with the server. One example of a 'human check' is a JavaScript server request that checks for typical human behavior such as mouse clicks and form focus. If these 'human' behaviors are not found then the traffic is not allowed. Bots and scripts have a difficult time recreating these 'human' behaviors although they are not impossible to recreate. For specific code examples refer to the appendix (Hunt, 2009).

Apache and **ModSecurity** software have additional tools to help with DDoS attacks. Apache is a common web server application that is typically found on many Linux distributions. It is important to note that the effectiveness of a DDoS attack is directly

related to how well the Apache servers are configured. Simple configuration options can be enabled to allow the server to work more efficiently so an attacker will have to use more resources to do the same amount of damage. Some examples of common Apache configurations are setting the "MaxClient" connection variable and rejecting referrer pages from outside the domain (Ristic, 2005). ModSecurity is an open source, free web application firewall (WAF) Apache module.[2] ModSecurity also has the ability to perform 'human checks' by performing content injection on HTTP responses to clients. ModSecurity can inject HTTP JavaScript tokens or additional JavaScript code into server responses to clients. The current use of this method is for protection against cross site request forgery (CSRF/XSRF). Content injection could also be used to inject additional JavaScript in order to test if the client is a human or zombie/bot. This additional content may be too sophisticated to be seen by bot/zombies. If the bot/zombies do not reply with valid content that was unique to the injected content then the bot/zombie request could be rejected.

The biggest benefit to this approach is if the remote bot is an automated program that is not able to perform common browser functions. Some Java script logic may try to look for mouse focus or activity that could indicate a human versus a bot. One possible problem is if the bots are zombie systems that could be using common web browsers that could emulate human behavior. It maybe that an attacker can reverse engineer all the actions needed to programmatically automate 'human' behavior into a bot. The only hope is making it difficult enough to stop the majority of attackers.

CAPTCHA[3] (**C**ompletely **A**utomated **P**ublic **T**uring test to tell **C**omputers and **H**umans **A**part) is another type of 'human' test used. The process usually involves one computer (a server) asking a user to complete a simple test which the computer is able to generate and grade. A common type of CAPTCHA requires that the user types letters or digits from a distorted image that appears on the screen. This is typically only used on key transaction pages (purchasing a product) since the test is very disruptive to a users experience to quickly navigate web pages. Current bots are supposedly known to use OCR and cheap

2 More information about ModSecurity can be found at http://www.modsecurity.org
3 More information about CAPTCHA can be found at http://www.captcha.net

labor outsourcing (foreign country contractors or porn site social engineering) to automate the CAPTCHA tests. Below is an example CAPTHA image.

Anomaly Detection

Some open source applications such as scrutinizer ("Denial of service", 2007) claim to use statistical anomaly detection by using old application log files (e.g. apache access logs) to create a profile of what 'normal' traffic is supposed to be in order to identify 'bad' traffic. Once the 'bad' traffic is identified the application will dynamically add local server firewall (e.g. iptables) access rules to deny the unwanted traffic. While this technology would seem very powerful it remains to be seen how effective it is.

3.5 Leveraging the Load Balancer

Now that a handful of common DDoS mitigation methods have been reviewed the following section will discuss how the Load Balancing or ADC device can help against DDoS attacks. Load Balancing switches have evolved over the last decade with not only faster hardware but also the ability to perform more advanced features such as SSL termination, Transparent Cache Switching, and Layer 7 content switching to name a few. The term Load Balancing device has become somewhat antiquated and is now being replaced with the term Application Delivery Controller or ADC (Fabbi, 2009).

Before looking at specific vendor appliance configuration options it is useful to point out how some companies may find themselves relying on the Load Balancing/ADC appliance to help mitigate DDoS attacks.

Cost

Most companies cannot afford a commercial DDoS solution and are forced to make do with what they have such as edge routers, server hooks, and of course Load Balancing switches.

Scale/Focus

Most companies will rely on upstream ISP's to peel off bandwidth flooding attacks which leaves the more insidious application specific attacks that can only be mitigated against with deeper packet inspection that a Load Balancing/ADC device may be able to provide.

Assumption

It is important to note that using a Load Balancing/ADC device to mitigate a DDoS makes the large assumption that a DDoS attack has not saturated the upstream ISP links. If the upstream ISP links are saturated then using a Load Balancing/ADC device becomes ineffective.

Vendor Exposure

The below sections will review configuration options from the following common commercial Load Balancing/ADC vendors.

- **Brocade ServerIron**
- **Citrix Netscaler**
- **Cisco ACE**
- **F5 BIGIP**

The cost of purchasing every vendor appliance prohibited the ability to test every appliance. The configuration examples are a mix of hands-on experience and vendor references. Brocade and Netscaler devices were tested in lab environment. Cisco ACE and F5 BigIP configuration options were not able to be tested in a lab environment.

3.5.1 Brocade ServerIron

Foundry Networks was one of the first companies to manufacture L4-7 switches in the late 90's. The L4-7 switch models were called ServerIrons. Brocade acquired Foundry Networks in 2008. Brocade continues to develop the ServerIron platform. The below features cover options in the ServerIron GT 10/11 code versions as well as the newer ServerIron ADX 12 code versions. The following discusses two features that may help mitigate against DDoS attacks ("Network Security", 2008).

- TCP SYN Proxy
- Content Switching

TCP SYN Proxy

Brocade has a feature called TCP SYN Proxy which is an implementation of TCP SYN Cookies. Starting with Release 09.0.00S, SYN-proxy functions like this: When the connecting client sends a TCP SYN to a server, the ServerIron responds with a SYN ACK, but does not create an internal session. Instead, the SYN ACK sent by the ServerIron contains a special sequence number that can be used to identify the SYN sent by the client. When and if the client returns an ACK, the ServerIron determines from the acknowledgement number which SYN the ACK refers to. If the time between the SYN and the ACK is within the allotted interval (1 minute), the ServerIron then establishes a session with the destination server. Since the ServerIron does not create a session entry for each SYN received from a client, connections are processed faster and resources are conserved.

Granular Syn-proxy feature

This feature prevents ServerIron from responding with TCP SYN-ACK to TCP SYN for ports not defined under a virtual IP address (VIP). This was found to be a very useful implementation option in lab and production environments. Without this option the Brocade will perform a TCP SYN-ACK on any traffic sent to the subnet of the interface that the feature is turned on. This can cause false positives when performing vulnerability scans or monitoring health checks.

Syn-Proxy Configuration

Configuring 'syn-proxy' is fairly straightforward. The feature has to be enabled globally as well as enabled on the specific interface that is facing possible attacker traffic. The 'server syn-cookie-check-vport' command enables the granular syn-proxy feature previously mentioned.

```
! Configure syn-proxy in the global mode.
ServerIron(config)# ip tcp syn-proxy
! Enable syn-proxy on each interface inbound SYN requests
ServerIron(config)#interface e 3/1
ServerIron(config-if-3/1)# ip tcp syn-proxy in
! Configure Syn cookie only on virtual IP address
ServerIron(config)# server syn-cookie-check-vport
```

The ServerIron can show a few different statistics in order to see if the syn-proxy feature is blocking invalid TCP SYN-ACK responses. The 'show server traffic' command below shows the current and max TCP connection rates while the 'show server syn-cookie' command shows how many invalid SYN-ACK responses have been received.

```
#Show server traffic
last conn rate       =         27  max conn rate        =        192
last TCP attack rate =         21  max TCP attack rate  =         71
```

```
#show server syn-cookie
    SYNs rcvd :            14530          SYN-ACKs sent :     14530
    Valid ACKs rcvd :      13958          Invalid ACKs rcvd : 5411
    ACL passed :           0              ACL failed :        0
    Frags allowed :        0              Frags dropped :     0
    ACK without data dro : 0
```

The 'show server tcp-attack' command shows a combination of the two previous commands by showing connection rate information and the number of invalid SYN-ACK responses.

```
#show ser tcp-attack
Connection counters:
    Current conn rate =         23             Max conn rate =105
Attack counters:
  Current attack rate =         32           Max attack rate =71
Client-side counters:
```

Brough Davis, brough.davis@gmail.com

```
             SYN rcvd =      15908              SYN-ACK sent =15908
      Valid ACKs rcvd =      15261         Invalid ACKs rcvd =5616
      Client pkt rcvd =     206280           Data pkt stored =6165
ACK without data dropp =          0
Server-side counters:
             SYN sent =      15261              SYN-ACK rcvd =8631
   Duplicate SYN sent =          0    Duplicate SYN-ACK rcvd =0
      Server pkt rcvd =     305956           Stored pkt sent =0
```

Content Switching

Foundry/Brocade ServerIron GT/C Chassis models offer the configuration option of load balancing based on layer 7/application criteria. This can be useful in both application/load optimization and security issues. Below is an example of load balancing HTTP 1.0 and 1.1 requests. Any other traffic being sent over TCP port 80 will be dropped. This is useful is DDoS attacks that are sending full TCP connections to the load balancing device that do not contain valid HTTP header information. Content switching will increase the CPU on the WSM. Implement with care. Most CSW options for security purposes are now migrated to the application firewall/Deep Scan features which will also look at payload information (size/regex,etc)....but there is also a CPU trade-off. For more info see the ServerIron GT/C Security Guide.

```
! ---- Define the rules
csw-rule "r1" version eq "1.0"
csw-rule "r2" version eq "1.1"
csw-rule "r3" nested-rule "r1 || r2"
!
! - Define Policy - defines what to do with matched rule csw-policy p1
 match r3 forward 1
! above forwards valid HTTP traffic to group 1 defined on real servers
 default forward 0
! Above - All non-http traffic to TCP 80 is dropped (group 0)
!
! ---- add the policy to the virtual server
server virtual-name VIP1 1.1.1.1
 port http csw-policy p1
 port http csw
 bind http RS1 http RS2 http
!
server real RS1 2.2.2.1
 port http
 port http url "HEAD /"
 port http group-id 1 1
!
server real RS2 2.2.2.2
 port http
```

```
port http url "HEAD /"
port http group-id 1 1
```

To see policy information and matched traffic hitting the policy the following command can be used. Note the hit counts for the default group ('bad traffic') at the end of the output.

```
#show csw-policy

Policy Count: 1   Policies Allocated: 1   Policies Deleted: 0

Policy Name         :p1
Policy Type         :Content Switching
Policy index        :1
Reference Count     :1
total received packe:0
created session     :0                    total scanned packet:0
no session drop     :0                    no session frag drop:0
send mirror ip packe:0                    send mirror packet  :0
send redirect packet:0                    case-insensitive    :FALSE

Action code description:
fwd: forward      rst: reset-client per: persist
rdr: redirect     err: reply-error  got: goto
rwt: rewrite      mir: mirror       log: log
con: count  drp: drop   rec: vir-reset
red: cont-red     mip: mirror-ip    unk: unknown

Flag description:
A: insert-cookie        B: delete-cookie        C: destroy-cookie
D: req-ins-hdr          E: req-ins-client-ip    F: resp-ins-hdr
G: delete-content H: insert-content I: modify-content
L: log

Rule Name   |Act|Data1     |Data2     |Data3     |Flags     |Hit Cnt
-------------------------------------------------------------
r3          |     |      |               |     |423846544  |
r3          |fwd|1         |          |N/A       |_________ |42384654
-------------------------------------------------------------
default     |     |      |               |     |5230       |
default     |fwd|0         |          |N/A       |_________ |5230
-------------------------------------------------------------
```

3.5.2 Citrix Netscaler 9010/9500

According the Citrix product documentation the Citrix NetScaler is an all-in-one web application delivery controller that makes applications run five times better.

Available as a separate hardware network device or as a virtualized appliance, NetScaler single-handedly offloads servers, accelerates performance, and integrates application security. The following features require using the code version 9.1 and later ("Protection Features", 2009).

- TCP SYN DoS Protection
- HTTP DoS Protection

TCP SYN DoS Protection

The NetScaler appliance also defends against SYN flood attacks by using SYN cookies instead of maintaining half-open connections on the system memory stack. This prevents SYN attacks and allows normal TCP communications with legitimate clients to continue uninterrupted. SYN DoS protection on the NetScaler appliance ensures the memory of the NetScaler is not wasted on false SYN packets; instead, memory is used to serve legitimate clients.

In addition, because the NetScaler appliance allocates memory for connection state only after it receives an HTTP request, it also prevents the protected Web sites from experiencing idle connection attacks. Configuring SYN DoS protection on your NetScaler appliance requires no external configuration; **it is enabled by default**.

HTTP DoS Protection

When the NetScaler appliance detects an attack, it responds to incoming requests based on the value of the Client Detect Rate parameter with a Java or HTML script containing a simple refresh and cookie. Standard web browser clients can parse the request and return the request with the cookie. Scripts and bots are commonly not able to interpret and send cookie information. When a POST request is received, it is first checked for a valid cookie. If the request has a valid cookie, the request goes through, but if the request does not have a valid cookie, the NetScaler appliance sends a Javascript to the client asking it to resend the information with a new cookie. If the client sends a new cookie, this cookie becomes invalid after four minutes, and every response to the client is sent with the new cookie. Any cookies sent before the attack will become invalid after the attack thresholds are hit. The cookie also becomes invalid when the 'think time' of the client exceeds four

minutes. Both of these scenarios are rare, but not impossible. All new connections during an attack are dropped and an error page with a cookie is sent. New connections as well as connections that cannot provide valid cookie data are not dropped but put into a low priority queue. Once Layer-7 protection is enabled the content switching policies will be enabled. Both the priority queuing and content switching policies create additional overhead on the device. The Netscaler CPU level should be carefully monitored after implementing Layer-7 protection. When using Layer-7 DoS protection, there is minimal effect on throughput, since the test JavaScript is sent to the client at a slow rate (default: 1% of the server's HTTP response rate).

Layer-7 DoS protection can be enabled by following the below steps in the configuration utility web tool interface.

1. In the navigation pane, expand **System**, and click **Settings**. The System Settings Overview page appears in the right pane.
2. Click **Advanced Features**. The **Configure Advanced Features** dialog box appears.
3. Select **HTTP DoS Protection** check box, click OK, and click Yes on the Enable/Disable Feature(s) dialog box.

The status bar displays a message indicating that the selected feature is enabled.

The below command show Layer-7 DoS related counters such as the rate and total number of times the Netscaler enabled sending cookies once Layer-7 DoS threshold conditions were met as well as the number of received valid cookies and clients.

```
[nsroot@nsr1]> show dos stats

DOS global statistics

Name                                    Rate (/s)        Total
DOS condition triggered                         0        4
Valid DOS clients                               0        0
DOS priority clients                            0        0
```

There is also a wealth of information in the TCP statistics command such as the surge queue, client/server connection rates, and TCP errors. The following example abbreviated output show some of these values.

```
[nsroot@nsr1]> stat tcp -detail
TCP and Connections Statistics

Connections                          Client   Server
All client connections                 1484      592
Opening client connections                0        0
Established client connections         1167      130
Closing client connections              317      462
Opened client connections             91908707  277713k
Opened client connections(Rate)           6     12

Surge queue                               0
...
TCP statistics
                                        Rate (/s)     Total
SYN packets received                    6             93922129
Server probes                           0               865383
FIN packets from server                 11            265163792
FIN packets from client                 6              78832992
Time wait to SYN                        0                 42501
Data in TIME_WAIT                       0                 14159
SYN packets held                        0               1398791
SYN packets flushed                     0                362781
TIME_WAIT connections closed            0               5458952

TCP errors
                                        Rate (/s)     Total
Bad TCP checksum                        0                     0
Data after FIN                          0                     0
SYN in SYN_RCVD state                   0                     0
SYN in ESTABLISHED state                0                396344
SYN_SENT incorrect ACK packet           0                     0
RST packets received                    1              18343541
RST on not ESTABLISHED                  1               9156269
RST out of window                       0                 21458
RST in TIME_WAIT                        0                   448
Server out of order packets             0                   812
Client out of order packets             0                    27
TCP hole on client connection           0                    13
TCP hole on server connection           0                    69
Seq number SYN cookie reject            0                   361
Signature SYN cookie reject             0                205312
Seq number SYN cookie drop              0                   413
MSS SYN cookie reject                   0                     0
Any IP port allocation failure          0                     0
IP port allocation failure              0                     0
Stray packets                           0              19156135
RST packets sent                        0              37389350
Bad state connections                   0                     1
RST threshold dropped                   0                   498
Packets out of window                   0                     3
SYNs dropped (Congestion)               0                     0
```

3.5.3 Cisco ACE 4710

Cisco has come a long way since the local-director and CSS days. The ACE appliance and ACE module for the 6500/7600 catalyst switch chassis are now more intuitive to configure with additional feature sets and have better performance.

The ANM (Application Network Management) software allows managing multiple ACE devices from a central location. This could be very useful when trying to quickly react to a DDoS across many systems.

TCP SYN Cookie

The Cisco ACE also uses a SYN Cookie option to protect against Syn Flood attacks. Cisco likes to use the term "embryonic connection" to describe a TCP connection that is in the process of a completing a full 3-way connection. The ACE device triggers the SYN Cookie feature when the number of embryonic connections goes over a certain rate threshold. This threshold is configurable.

If the SYN Cookie feature is not enabled then the ACE device will start dropping SYN requests when the SYN queue fills up. The SYN Cookie feature works as per the standard in which the ACE device calculates the sequence number in the SYN-ACK response to the client and releases any state information, thus freeing up resources. ("Configuring TCP/IP Normalization", 2008). Because the ACE does not track state, if the server drops the SYN that is sent by the ACE the ACE will not try to retransmit and the connection will timeout based on the ACE embryonic timers. Another possible issue is that due to SYN Cookie implementation issues the ACE will no longer support TCP options besides MSS after SYN Cookie is enabled.

SYN Cookie Configuration:

The SYN Cookie feature must be enabled on the ACE interface that is facing the direction that the attacks may come from. The command used is "syn-cookie *number*". The *number* is the threshold value of concurrent embryonic (half-open) connections that

is needed in order for SYN Cookies to be triggered. For example, to configure SYN-cookie DoS protection for servers in a data center connected to VLAN 100, enter:

```
host1/C1(config)# interface vlan 100
host1/C1(config-if)# syn-cookie 4096
```

The following command shows SYN-cookie related statistics on two different interfaces (vlan23, vlan101).

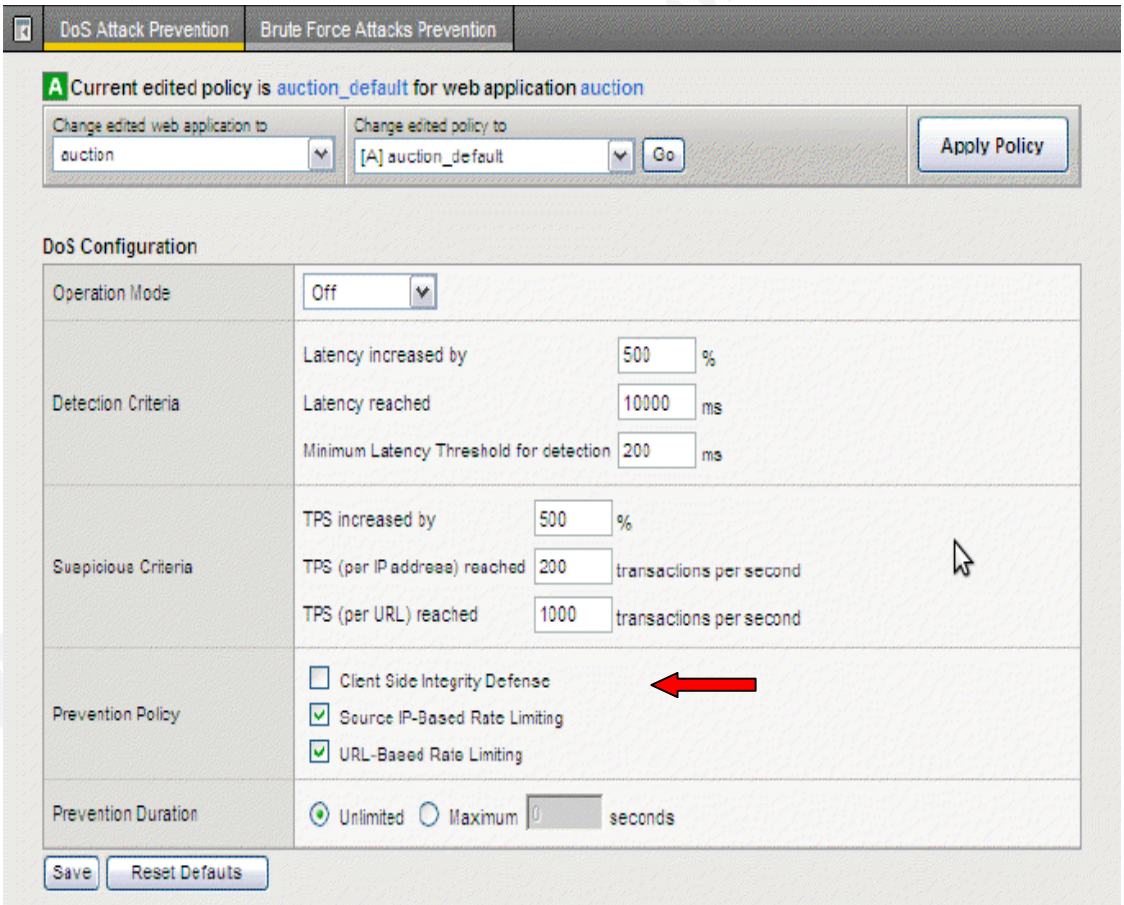
```
ACE30001/Admin# show syn-cookie

Interface vlan23
        Configured TCP Embryonic Connection Limit: 0
        Current number of Embryonic Connections: 0
        Number of TCP Syns Intercepted by SYN COOKIE: 0
        Number of TCP Acks Successfully Processed by SYN COOKIE: 0
        Failed Number of TCP Acks Processed by SYN COOKIE: 0
Interface vlan101
        Configured TCP Embryonic Connection Limit: 0
        Current number of Embryonic Connections: 0
        Number of TCP Syns Intercepted by SYN COOKIE: 0
        Number of TCP Acks Successfully Processed by SYN COOKIE: 0
        Failed Number of TCP Acks Processed by SYN COOKIE: 0
        Failed Number of TCP Acks Processed by SYN COOKIE: 0
```

3.5.4 F5 BigIP / ASM

F5 is a company that offers a network appliance named BigIP. The appliances are sold with the option of different features or product modules. One of these product modules is the Application Security Manager (ASM). The ASM claims to have a layer 7 DoS prevention mechanism. The layer 7 DoS prevention feature injects a small piece of JavaScript code in a HTTP server response. The code needs to be evaluated by the browser and then the code assigns a specific value to the BigIP. This method verifies that the request is valid, coming from a real browser instead of a bot script. This method accurately distinguishes malicious requests from the legitimate ones. Because many attacks are script-based and browsers were not designed efficiently to send several

requests per second this method provides a high degree of accuracy in defending against layer 7 DoS attacks (Koyfman, 2009). It should be noted that this method is only effective against HTTP based application traffic.

The below picture depicts the configuration screen for Layer 7 DoS and Brute Force Protection on BIG-IP Application Security Manager. The prevention policy is where traffic is blocked. The prevention policy has three blocking mechanisms; javascript injection check, source IP address rate limiting, and URL rate limiting. Each of these can be enabled by themselves or all together. The Layer 7 features only get triggered if the detection and suspicious criteria are matched. For example, in the below diagram, if server response increases to 10,000 ms and the transactions per second to a specific URL increase to 1,000 per second then the JavaScript injection check can be enabled to start blocking traffic on any source that does not respond with the correct JavaScript response.



Advanced statistical information can be seen from the Local Traffic Manager module (LTM) as seen in the below graphical user interface. Note the connection rate and bandwidth statistics that can be especially useful when sensing DDoS attacks.

3.5.5 Load Balancing Feature Summary

The following table summarizes common security features that can help mitigate against DDoS attacks

	Cisco	**Brocade**	**F5**	**Netscaler**
TCP SYN Cookie	YES	YES	YES	YES
HTTP inspection	**YES**	**YES**	**YES**	**YES**
HTTP Cookie Injection	YES	YES	YES	YES
'human' JS check	NO	NO	YES	YES

It is more apparent after looking at the feature summary that most Load Balancing devices can perform basic DDoS mitigation techniques through basic TCP SYN Cookie options and basic HTTP inspection and HTTP Cookie injection. Unfortunately due to the nature of Load Balancing hardware not having the ability to keep up with newer scripting technologies it is difficult for a Load Balancer to be involved in more sophisticated

'human check' scripts. The only vendors that do perform 'human check' JavaScript injection and inspection, F5 and Netscaler, are limited in their configurations. The details of what is actually being performed behind the GUI interfaces also make it more difficult to verify the effectiveness of these features.

Matching Trends with Features

Previously in the paper it was noted that trends in DDoS attacks seem to imply the majority of attack vectors focus on basic TCP flood attacks. These attacks could be SYN Floods or bot-nets making full 3-way connections. This is where the Load Balancing switches can be the greatest benefit. By using the TCP SYN Cookie and HTTP Inspection features many of the DDoS attacks that are common on the internet today can be effectively mitigated. What many people don't realize is that the basic HTTP Inspection configurations typically used for load balancing decision making can play a key part in preventing DDoS attacks. Unfortunately, we also see from the trends that the DDoS attacks are becoming more sophisticated in which the client side requests are getting better at mimicking human requests. These application level attacks will bypass the simple HTTP inspection and TCP SYN Cookie features mentioned. Only time will tell if the Load Balancing/ADC vendors will continue to add features that will help with these newer threats or if we should focus mitigation on other devices such as Web Application Firewalls or in the application code itself.

4. Short Falls

Overworking the Load Balancing device

Many Load Balancing devices require more CPU resources to perform for deeper packet inspections in order to find distinguish legitimate traffic from the malicious DDoS traffic. This is probably one of the more serious downfalls since enabling some of the security features during a DDoS could possible overload the Load Balancing device to the point where no traffic was load balanced which would serve the goals of the attacker.

Legitimate Traffic Matching

Some types of customer traffic may be very difficult to distinguish from a DDoS attack. How would you know the difference between a bot doing a web crawl through a website versus a standard customer browsing your site? Finding the difference between these two traffic conditions goes beyond the standard TCP SYN-flood or raw TCP handshake floods. As Load Balancing switches and Application Director Controllers (ADC's) evolve with faster CPU's, more memory, and additional inspection features this may improve. Unfortunately it will be likely that appliance based solutions will have a difficult time keeping up with sophisticated attacks. Only time will tell.

5. Planning for the Future

As mentioned earlier in the paper, sophisticated DDoS attacks have been around for quite some time. It is likely that attackers will use sophisticated attacks (e.g. web crawl attacks) more frequently as the older methods for DDoS (SYN-Flood, large UDP/ICMP, Frag attacks) become less effective. Most attackers typically attack easier targets. Since many companies have little if no DDoS mitigation methods the attackers will continue to exploit those. The larger concern is the attackers that are financially motivated. These attackers are typically more sophisticated and will use more complex techniques in a DDoS attack. Below are some methods that may help a company fight against future DDoS attacks.

Know the traffic trends! By trending the traffic characteristics of a network the important traffic can be separated from the unnecessary traffic. This can help in formulating access-lists of what should be allowed into the network as well as better understanding the application traffic that is important. For example, if the only important traffic allowed into the network is HTTP over TCP port 80 then any TCP port 80 traffic that does not have an HTTP header should not be allowed into the network. This can be taken further by only allowing certain URL names and regular expressions that match important traffic.

Involve the developers! Once DDoS attacks start imitating client application traffic the difficulty in separating what is a human client from a script/bot becomes magnified. One method that is used with HTTP applications is to inject a hidden javascript 'human check' in order to only allow human clients. This 'human check' becomes more difficult with non-HTTP applications that may be proprietary.

Use everything! Every available resource within the network path of a DDoS attack should be considered a possible tool in either helping detect or mitigate DDoS attacks. Whether using simpler methods such as anti-spoofing access-lists on edge routers or using more complicated methods such as application 'human checks', every method should be considered a tool that can be applied in layers. DDoS attacks can come in many different forms and new types can be manufactured quite easily. It's important to have as many tools to pull from as possible resources.

6. Conclusion

There are many different attack methods that a DDoS attack can utilize. Over the last decade the common methods have been simple protocol flood and spoofing attacks. Unfortunately the trends show that these types of attacks are still very successful with many companies that have not yet implemented basic defense strategies such as anti-spoofing access lists on the edge networks. This paper has shown how to use common Load Balancing/ADC device TCP SYN Cookie and HTTP Inspection configurations in order to help combat many DDoS attacks currently seen. As companies start implementing common defense strategies such as anti-spoofing access-lists and server TCP cookies, attackers will more likely use application specific DDoS attacks (e.g. Full TCP connections, Web Crawl Attacks, etc). These attacks will likely need to be mitigated by application specific Reverse Turing tests (“human checks”). Unfortunately, many Load Balancing/ADC devices currently offer limited if any capability to perform custom Reverse Turing tests. The only way a company can attempt to successfully ward off DDoS attacks is by leveraging every possible resource. A tiered "defense in depth" strategy is needed by utilizing the edge routing and access-list configurations, firewall access-list configurations, IPS/IDS signature matching and dropping, server side OS and

application options, as well as Load Balancing/ADC devices. One technology or appliance cannot be relied upon to defend against all DDoS attacks. By looking at the Load Balancing/ADC appliance options another tool can be used in the fight against DDoS attacks. Many of the common Load Balancing/ADC appliances on the market today have some security feature that can help in mitigating against DDoS attacks. Hopefully this paper has given the reader some useful options and insight to their uses.

7. References

Amazon Web Services: Overview of Security Processes (November 2009). Retrieved February 2, 2010 from http://awsmedia.s3.amazonaws.com/pdf/AWS_Security_Whitepaper.pdf

Bejtlich, R. (2005). *The Tao of network security monitoring*. Boston, MA: Pearson Education, Inc..

Bernstein, DJ. SYN Cookies. Retrieved March 9, 2010 from http://cr.yp.to/syncookies.html

Bot-net Commands - Which commands the bots understand. Retrieved March 6, 2010 from http://www.honeynet.org/book/export/html/55

Configuring TCP/IP Normalization and IP Reassembly Parameters (2008). Cisco 4700 Series Application Control Engine Appliance Security Configuration Guide. Retrieved April 15, 2010 from http://www.cisco.com/en/US/docs/app_ntwk_services/data_center_app_services/ace_appliances/vA3_1_0/configuration/security/guide/tcpipnrm.html#wp1135715

Denial of Service Tools. (2007) Retrieved January 10, 2010 from http://www.packetstormsecurity.com/distributed/

Drupal. (2008, August 10). Uses of Botnets. Retrieved January 20, 2010 from http://www.honeynet.org/node/52

Fabbi, Mark (2009 February 2). Load Balancers Are Dead: Time to Focus on Application Delivery. Retrieved January 5, 2010 from http://www.gartner.com/technology/media-roducts/reprints/f5networks/164098.html

Fabbi, M., Skorupa, J., (2009 September 24). Magic Quadrant for Application Delivery Controllers. September 2009. Retrieved March 20th from

http://www.gartner.com/technology/media-products/reprints/f5networks/vol6/article1/article1.html

Ferguson, P. (2000, May). Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing. (2000). *IETF*: Networking Working Group RFC7304

Hunt, Ben (2009). Invisible Human Check for Web Form Validation. Retrieved May 15, 2010 from http://www.webdesignfromscratch.com/javascript/human-form-validation-check-trick/

Koyfman, Michael (2009). Intelligent Layer 7 DoS and Brute Force Protection for Web Applications. Retrieved April 5, 2010 from http://www.f5.com/pdf/white-papers/intelligent-layer7-protection-wp.pdf

Kumari, W., McPhereson, D. (2009, August). Remote Triggered Black Hole Filtering with Unicast Reverse Path Forwarding (2009). *IETF*: Networking Working Group RFC5635

McPherson, Danny. (2009, May 4). 60 Days of Attack Scale & Duration. Retrieved April 20, 2010 from http://asert.arbornetworks.com/2009/05/60-days-of-attack-scale-duration/

McPherson, D, Rolands, D. (2010, January). 2009 Worldwide Infrastructure Security Report. Retrieved April 15, 2010 from http://www.arbornetworks.com/dmdocuments/ISR2009_EN.pdf

Mikovic, J, Dietrich, S., Dittrich, D., & Reiher, P. (2005).*Internet denial of service*. Upper Saddle Riber, NJ: Pearson Education, Inc..

Network Security (2008). *ServerIron Security Guide Release 11.0.00*. Retrieved January 10, 2010 from http://www.foundrynet.com/services/documentation/SISecurity/11000/network_security.pdf

Perry E. Metzger, William Allen Simpson, Paul Vixie (December 2009). Improving TCP Security With Robust Cookies. ; LOGIN

Protection Features (2009). Citrix NetScaler Application Security Guide 9.1. Retrieved February 20, 2010 from http://support.citrix.com/servlet/KbServlet/download/20678-102-342827/NS-AppSecurity-Guide.pdf

Ristic, I. (2005). *Apache Security*. Sebastopol, CA: O'Reilly Media, Inc..

Turk, D. (2004, September). Configuring BGP to block Denial-of-Service Attacks. (2004). *IETF*: Networking Working Group RFC3882

Zuquete, A (2002). Improving the Functionality of SYN Cookies. IST / INESC-ID Lisboa

8. Appendix

8.1 Javascript Human Check Example

The following JavaScript was taken from www.webdesignfromscratch.com as an example method for detecting if the client is a human or bot/zombie script (Hunt, 2009). The script sets up an event detection which should be called by an event handler, and that event handler should be attached using an external JavaScript file, not written into the HTML code.

```
//Set up
addEvent(window, "load", setUpHumanTest, false);
function setUpHumanTest() {
  var myforms = document.getElementsByTagName("form") ;
  for (var i=0; i<myforms.length; i++) {
    addEvent(myforms[i], "focus", markAsHuman, false);
    addEvent(myforms[i], "click", markAsHuman, false);
  }
}
//Identify a human
function markAsHuman() {
  document.getElementById("imahuman").value = "1";
}
```

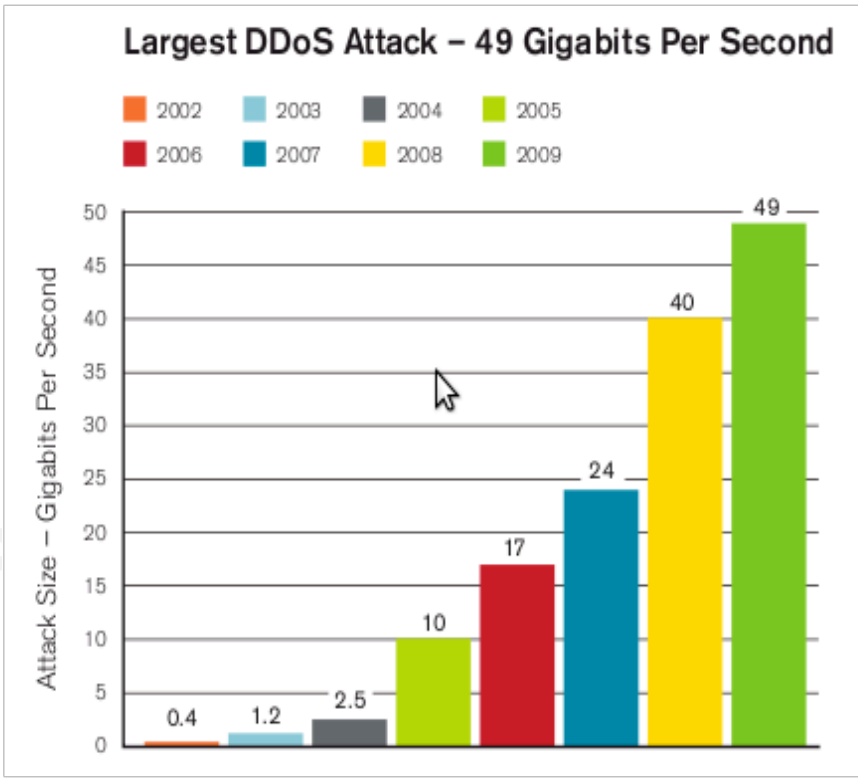
The above code depicts two key functions that help verify the client is a human or a script. The **setUpHumanTest** function checks all forms on the current page and verifies if there are any actions that would typically indicate a human such as mouse click and form focus. If the setUpHumanTest function passes the human check the **markAsHuman** function is called and the imahuman variable is set to 1. Any client request that does not pass this check will be rejected based on the imahuman variable. Both events are supposed to happen only in a browser window, so a script should have great difficulties reproducing them.

8.2 Arbor Networks Additional Report Information

The following information is additional data supplied from the Arbor Networks World Wide Infrastructure Security Report 2009 and their interpretations

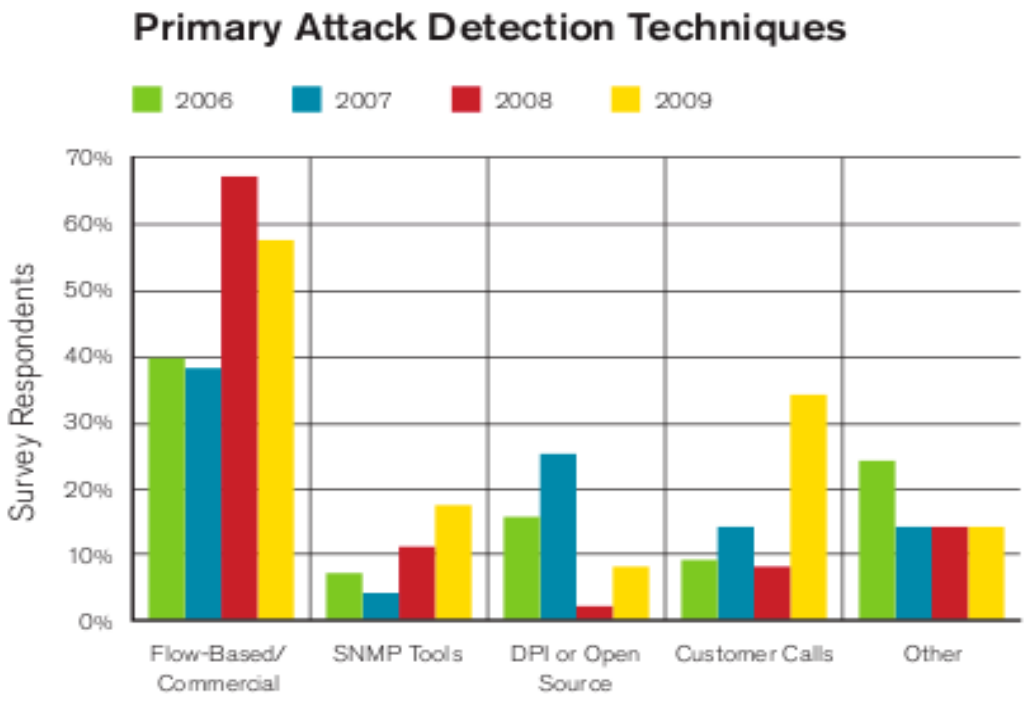
8.2.1 Growing

The below diagram shows that the largest DDoS attacks recorded each year are continually increasing larger bandwidth capabilities. This increase is most likely due to bot nets getting larger and Internet capacity growth. What is also interesting about this graph is that the growth rate has decreased from 66% in 2008 to 15% in 2009. The decrease in growth rate could be from bot nets hitting the ISP infrastructure bandwidth ceilings and the narrowing in application DDoS attacks. While the data could suggest that companies are getting better at fighting DDoS attacks it is not likely the case.



8.2.2 Detection Usefulness

Many companies that battle DDoS attacks decide to use a flow-based or vendor appliance approach to detect DDoS attacks. The usefulness of detection methods of a DDoS attack is somewhat ambiguous. While a sudden spike in specific traffic may be a part of a DDoS attack it is very likely that other standard monitoring systems have flagged a problem or even worse customers calling in to complain about degraded service. If services are not being impacted and there is a traffic spike how would it be classified as a DDoS attack? With that said if the flow-based/commercial methods also hook into a mitigation process then having this type of detection method would be very useful.



8.2.3 Fear of DDoS

The below graph depicts that out of the companies surveyed that approximately 35% are concerned about DDoS attacks above most other threats. This is understandable since DDoS attacks can come in different forms and impact a large part of a company's resources. This fear could also be from a company having to put faith into their upstream providers to help or could also be from having no plan in place to deal with DDoS attacks.