BEST PRACTICES

F5 DDoS Protection: Recommended Practices

Contents

1 Concept

Distributed Denial-of-Service (DDoS) is a top concern for many organizations today, from high-profile financial industry brands to service providers. Experienced administrators know that F5 equipment is not only well-suited to mitigating DDoS attacks, but sometimes is the **only** equipment that can mitigate certain types of DDoS. What many administrators do not know is that a complete on-premises DDoS solution can be achieved with a complement of F5 products.

A DDoS attack can be a stressful engagement where parts of the network will be unresponsive and equipment may be failing all around. That is not the time to be planning a defense—preparing your network applications during "peacetime" will go a long way to helping you mitigate the attack in the future.

This guide assumes that you have a F5 networking solution and an optional F5 security solution.

All configurations, commands, and platforms are assumed to be TMOS 11.3.0 unless otherwise stated.

Even though much of the technical information is specific to F5 equipment, some of the strategies (such as the use of SNAT pools to avoid port exhaustion) may apply to other vendors' devices as well.

2 DDoS-Resistant Architecture

It is possible to build an application delivery network that is DDoS-resistant. This section discusses work that can be done prior to an attack to make the network and applications resilient.

2.1 F5's Recommended Architecture

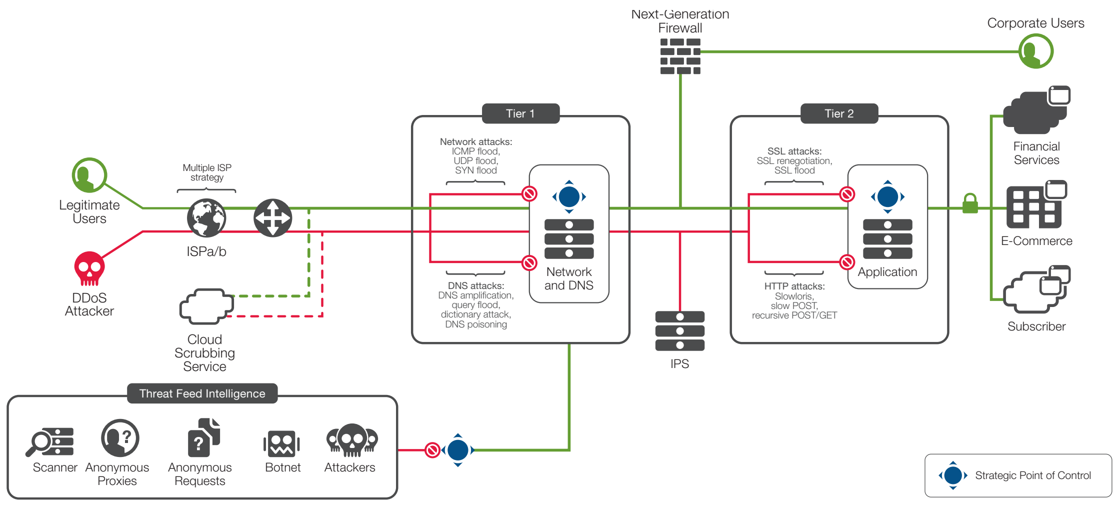


Figure 1: F5 recommends a two-tier DDoS approach

Many organizations are redesigning their architecture for DDoS resistance. For many customers, F5 recommends a **two-tier** DDoS solution, where the first (perimeter) tier is composed of layer 3 and 4 network firewalling and simple load-balancing to a second tier of more sophisticated (and also more CPU intensive) services including SSL termination and Web Application Firewalling.

There are several benefits of the two-tier approach:

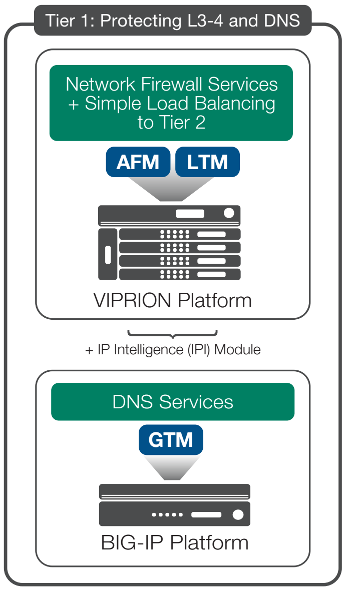
- Mitigation can be isolated so that layer 3 and 4 are mitigated at tier 1, with application protection at tier 2.
- The tiers can be scaled independently of one another. For example, if WAF usage grows, another appliance (or blade) can be added to the second tier without affecting the first tier.
- The tiers can have different platform types and even different software versions.
- When new policies are applied at the second tier, the first tier can direct just a portion of traffic to the new policies until they are fully validated.

	Tier 1	Tier 2	DMZ
F5 Components	AFM + LTM	LTM + ASM	GTM DNS Express
OSI Model	Layers 3 + 4	Layer 7+	DNS
Capabilities	Network Firewall First—Tier Load Balancing IP Reputation Blacklists	SSL Termination Web Application Firewall Secondary Load Balancing	DNS Resolution DNSSEC
Attacks Mitigated	SYN Floods ICMP Floods Malformed Packets TCP Flood Known Bad Actors	Slowloris Slow-post Apache Killer RUDY / Keep Dead SSL Renegotiation	UDP Floods DNS Floods NXDOMAIN Floods DNSSEC Attacks

2.2 Tier 1—Network Defense

The first tier is built around the network firewall. You almost certainly already have a network firewall (it may or may not be F5) and a network firewall team (or at least an administrator). At this tier you will prepare defenses around layer 3 and 4 (IP and TCP). This is where you will mitigate SYN floods, TCP floods, and block source addresses during a DDoS attack.

The following sections apply to the equipment at tier 1, whether that is the F5 AFM firewall module or an F5 LTM load-balancer in front of another vendor's network firewall.


Tier 1: Protecting L3-4 and DNS
Network Firewall Services + Simple Load Balancing to Tier 2
AFM
LTM
VIPRION Platform
+ IP Intelligence (IPI) Module
DNS Services
GTM
BIG-IP Platform

2.2.1 Choosing Virtual Server Types

Organizations using either the F5 firewall (AFM) or the F5 load-balancer (LTM) at tier 1 have a choice about how to structure their configuration. There are four options for defining a "listening" object. While all of these are valid ways to arrange the configuration, some have different strengths when dealing with DDoS.

- **Full-Proxy Virtual Servers** are the standard virtual servers in an F5 configuration. These listeners establish a real connection with each incoming client before they initiate a secondary connection to the server. The very act of terminating and validating the client connection provides a broad range of protection before the second tier is even invoked.
- **Forwarding Virtual Servers** perform faster and still protect against SYN floods, but do not provide the broader level as protection that full proxy virtual servers do.
- **Wildcard Virtual Servers** allow the **decoupling of firewall rules from the application virtual server**. This enables the creation of a rule that says "for any address supplying FTP services, apply this ruleset, this mirroring policy, and this source NAT policy."
- **Route Domains**, which isolate duplicate IP subnets into logical, separate routing tables, are common in service provider environments. While route domains provide little or no benefit regarding DDoS in and of themselves, they can be used as pegs on which to hang layer 4 security policies.

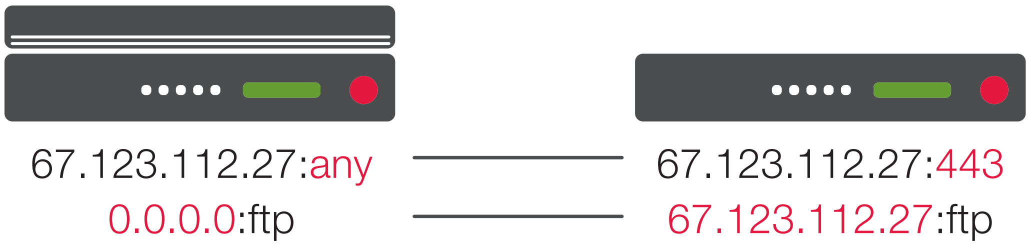


Figure 2: Wildcard servers are an option at tier 1

```
ltm virtual ws _ ftp {
      destination 0.0.0.0:ftp
      ip-protocol tcp
      profiles { ftp { } tcp { } }
      translate-address disabled
      }
```

In general, **F5 recommends the use of Full Proxy or Forwarding Virtual Servers** at Tier 1 when DDoS is a top concern.

2.2.2 Mitigate SYN floods at Tier 1

TCP SYN floods are always mitigated by F5. In version 11.5, F5 even migrates SYN floods against Direct server Return (DSR) virtual servers. To verify that your BIG-IP is managing SYN flood protection, you can view SYN flood statistics for each individual virtual server with the simple **show** command.

% tmsh show ltm virtual vip1

```
…
SYN Cookies
  Status                                       full-software
  Hardware SYN Cookie Instances                            0
  Software SYN Cookie Instances                            2
  Current SYN Cache                                        0
  SYN Cache Overflow                                       0
  Total Software                                      432.2K
  Total Software Accepted                                  0
  Total Software Rejected                                  0
  Total Hardware                                           0
  Total Hardware Accepted                                  0
```

Many F5 platforms can mitigate SYN floods in hardware, which allows the main traffic steering CPUs to perform other tasks.

Table 1: SYN Flood Hardware Support Platform List

Platform	Hardware SYNs per second	Version
B4300 Blade	80M	11.3
B2100 Blade	40M	11.3
10200V	80M	11.3
10000S	40M	11.4
7200V	40M	11.4
7000S	20M	11.4
5200V	40M	11.4
5000S	20M	11.4

*Older platforms including 8800, 8400, 6800 and 6400 also include hardware SYN cookie support; however, those models are not supported by version 11.3, which is the basis for this document.

To enable hardware offload for SYN flood mitigation for a specific virtual server, create a tcp profile with a tighter security posture. This example sets two DDoS-related variables. It enables hardware **SYN cookies**. It also sets the **deferred-accept** variable that reduces the impact that "zero-window" TCP attacks can have on the virtual server.

```
% tmsh create ltm profile tcp tcp _ ddos { hardware-syn-cookie deferred-accept
enabled zero-window-timeout 10000 }
```

Then associate the new tcp profile with the virtual server by replacing the existing "tcp" profile.

```
% tmsh list ltm virtual vip1 profiles
% tmsh modify ltm virtual vip1 profiles replace-all-with { tcp _ ddos my _ ddos1
http }
```

2.2.3 Deny UDP and UDP Floods at Tier 1

UDP floods are a common DDoS vector, because they are easy to generate and can be hard to defend. In general, do not allow UDP traffic to a virtual server unless the application behind it is actively accepting it.

Even for applications that accept UDP, a UDP flood can overwhelm the system, and you may find it necessary to temporarily deny UDP traffic to the application's virtual server.

```
% tmsh create security firewall rule-list drop _ udp { rules add { drop _ udp _ rule
{ action drop ip-protocol udp place-after first } } }
% tmsh modify ltm virtual vip1 fw-rules { drop _ udp _ vip1 { rule-list drop _ udp }
} }
```

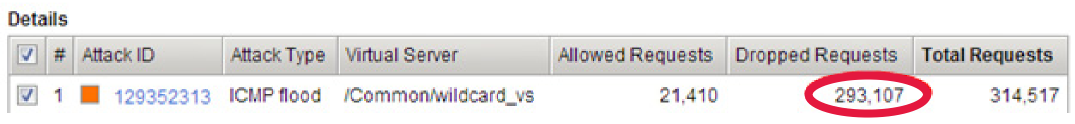
When the attack has ceased, you can remove the rule from the virtual server.

Version 11.5 can monitor and mitigate UDP floods with granular exceptions. This enables a baseline of UDP traffic to pass through a virtual server at tier 1. If the UDP traffic exceeds the thresholds, it is dropped, unless it matches one of eight user-defined port exceptions (for example, RTSP or DNS).

2.2.4 Deny ICMP Floods

ICMP is another common DDoS vector. ICMP fragments are easy to generate and easy to spoof, and can tie up resources on many different types of networking devices.

AFM can differentiate between a normal amount of ICMP and an ICMP flood based on traffic pattern analysis. When AFM's network firewall is enabled on a virtual server, it will monitor for an increase in several types of traffic. A normal amount will be allowed, with the rest of the flood prohibited.


Details

✓	#	Attack ID	Attack Type	Virtual Server	Allowed Requests	Dropped Requests	Total Requests
✓	1	🟧 129352313	ICMP flood	/Common/wildcard_vs	21,410	293,107	314,517

2.2.5 Use the DDoS Device Profile of AFM

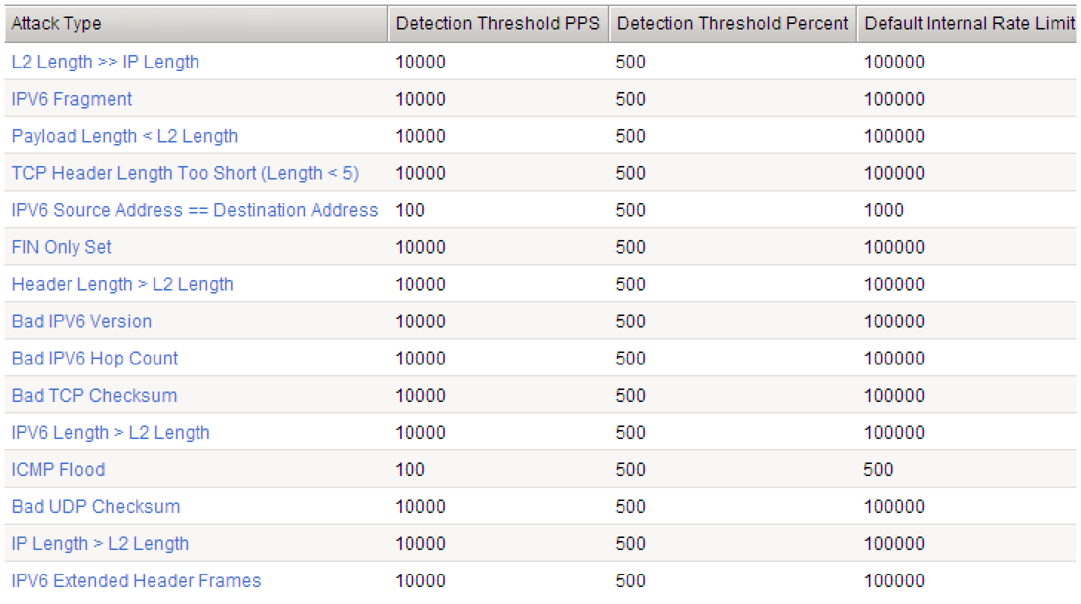
One way that attackers can consume firewall resources is by throwing floods of specially crafted invalid packets. The firewall will need to look at (and log) each packet. F5 has found that suspect combinations of flags (such as PSH+ACK with empty payloads) can be popular one month and then be abandoned in favor of a different combination later.

This shifting landscape makes it difficult to be predictive about what L3/L4 attacks are likely to happen. The security administrator (for other vendors' firewalls) should be aware of these attacks and be ready to insert rules to block them, taking care to avoid using more CPU than necessary.

F5's approach to this has problem has been to move much of the L3/L4 protocol validation into the custom hardware logic on the TMOS platforms that support it. By default, the AFM module is monitoring for dozens of layer 3 and layer 4 DDoS attack vectors such as floods of Christmas tree packets or LAND attack packets. Nearly all of these packets are discarded regardless of any BIG-IP setting. AFM can send a special log message when a flood of these packets is detected.

Table 1 (in section 2.2.2) shows which TMOS platforms have support for hardware-assisted L3/L4 protocol validation. These are the same platforms that have SYN flood hardware support.

All platforms (including the virtual edition) allow management of the parameters that track these L3/L4 suspect packet floods. The management screen is available from the Security tab of the user interface. Then select **DoS Protection** and **Device Configuration**.



Attack Type	Detection Threshold PPS	Detection Threshold Percent	Default Internal Rate Limit
L2 Length >> IP Length	10000	500	100000
IPV6 Fragment	10000	500	100000
Payload Length < L2 Length	10000	500	100000
TCP Header Length Too Short (Length < 5)	10000	500	100000
IPV6 Source Address == Destination Address	100	500	1000
FIN Only Set	10000	500	100000
Header Length > L2 Length	10000	500	100000
Bad IPV6 Version	10000	500	100000
Bad IPV6 Hop Count	10000	500	100000
Bad TCP Checksum	10000	500	100000
IPV6 Length > L2 Length	10000	500	100000
ICMP Flood	100	500	500
Bad UDP Checksum	10000	500	100000
IP Length > L2 Length	10000	500	100000
IPV6 Extended Header Frames	10000	500	100000

Figure 3: Network DDoS Configuration Settings

These settings are also available via the command-line with the **security dos device-config** command. Also note that these settings are per traffic management microkernel (tmm), not per platform. In the table, the columns map to these values.

- **Detection Threshold PPS**. This is the number of packets per second (of this attack type) that the BIG-IP system uses to determine if an attack is occurring. When the number of packets per second goes above the threshold amount, the BIG-IP system logs and reports the attack, and then continues to check every second, and marks the threshold as an attack as long as the threshold is exceeded.

- **Detection Threshold Percent**. This is the percentage increase value that specifies an attack is occurring. The BIG-IP system compares the current rate to an average rate from the last hour. For example, if the average rate for the last hour is 1000 packets per second, and you set the percentage increase threshold to 100, an attack is detected at 100 percent above the average, or 2000 packets per second. When the threshold is passed, an attack is logged and reported. The BIG-IP system then automatically institutes a rate limit equal to the average for the last hour, and all packets above that limit are dropped. The BIG-IP system continues to check every second until the incoming packet rate drops below the percentage increase threshold. Rate limiting continues until the rate drops below the specified limit again.

- **Default Internal Rate Limit**. This is the value, in packets per second, that cannot be exceeded by packets of this type. All packets of this type over the threshold are dropped. Rate limiting continues until the rate drops below the specified limit again.

2.2.6 Mitigate TCP Connection Floods

TCP Connection floods are a layer 4 anomaly and can affect any stateful device on the network, especially firewalls. Often these floods are empty of actual content. LTM or AFM at the first tier can mitigate these by absorbing the connections into high-capacity connection tables.

Table 2: Connection Table Sizes

Platform	TCP connection Table Size	SSL connection Table Size
VIPRION 4480 (4 X B4300)	144 Million	32 Million
VIPRION 4480 (1 X B4300)	36 Million	8 Million
VIPRION 4400 (4 X B4200)	48 Million	5 Million
VIPRION 4400 (1 x B4200)	12 Million	1 Million
VIPRION 2400 (4 x B2100)	48 Million	10 Million
VIPRION 2400 (1 x B2100)	12 Million	2.5 Million
11000 series	24-30 Million	2.64-3.9 Million
10200 series	36 Million	7 Million
8900 series	12 Million	2.64 Million

Platform	TCP connection Table Size	SSL connection Table Size
7000 series	24 Million	4 Million
6900 series	6 Million	660 Thousand
5000 series	24 Million	4 Million
4200V series	10 Million	2.4 Million
3900 series	6 Million	660 Thousand
Virtual Edition	3 Million	660 Thousand

2.2.7 Configure Adaptive Reaping

Even with high-capacity connection tables, there are still settings that can be adjusted to deepen the protection profile against flood attacks.

In the event that the BIG-IP connection table does become full, connections will be "reaped" according to the adaptive reaping low-water and high-water settings. These can be adjusted downward from the default values of 85 and 95 in order to begin mitigating a "spiky" DDoS faster, and thus reducing the window during which the initial attack will load the servers.

```
% tmsh modify ltm global-settings connection adaptive-reaper-lowater  75
```

2.2.8 Modify Idle Timeouts to Combat Empty Connection Floods

While layer 4 connection floods do not typically pose a high risk to F5 devices, they definitely impact other stateful devices such as other firewalls. Those devices will nearly always collapse long before the F5 state tables fill up (see Table 2 in section 2.2.6). If the connection flood consists primarily of empty connections you can instruct BIG-IP to be more aggressive about closing these empty connections.

There are three primary profiles associated with layer 4 on BIG-IP:

- fastL4—the hardware assisted, high-performance TCP profile
- tcp—the standard TCP profile used by the majority of virtual servers
- udp—the standard UDP profile

Note: You may see others, such as those associated with WAN optimization, which are based on the tcp or udp profiles.

Use the following attributes of these profiles to control how long a connection is idle before it is closed by BIG-IP. During a heavy attack, use smaller and smaller values.

For the fastL4 profile, override the **reset-on-timeout** and **idle-timeout** values. The default timeout is 300 seconds, which should be trimmed significantly during an attack.

```
% tmsh create ltm profile fastl4 fastl4_ddos { reset-on-timeout disabled idle-
timeout 15 }
```

For each fastL4 virtual server under attack, replace the fastL4 profile with your new one.

For the tcp profile, override the same two values for the same reasons. While you are there, you may also want to adjust the **hardware-syn-cookie** and **zero-window-timeout** values. See section 2.2.2.

For the udp profile, reduce only the **idle-timeout** value (the default is 60 seconds).

2.2.9 Control Rate-Shaping

Another defensive technique that can be deployed quickly is rate-shaping. Rate-shaping can limit the rate of ingress traffic at the BIG-IP and may be the easiest way to push back against a volumetric attack. While powerful, rate-shaping is a less-than-ideal technique for defending against DDoS. Because it does not differentiate between good and bad requests, rate-shaping can discard your good traffic as well, which is probably not what you want.

You configure rate-shaping profiles manually and then assign them to a virtual server.

In this example, the rate-shaping class named "protect_apache" guarantees that at least 1mbs of traffic will reach the target, but that no more than 10mbs will be allowed.

```
net rate-shaping class protect _ apache {
      rate 1mbps
      ceiling 10mbps
}
```

Then apply this rate-shaping class to each of your targeted virtual servers.

2.2.10 Set the Max ICMP Reject Rate

The **TM.MaxRejectRate** system variable can reduce the effects of a denial of service attack by allowing you to limit the number of TCP RSTs or ICMP unreachable packets that the BIG-IP system sends in response to incoming connections that cannot be matched with virtual server connections. The default value for the **TM.MaxRejectRate** system variable is 250 TCP RSTs or 250 ICMP unreachable packets per second.

Dropping the value to 100 can contribute to a reduction in outbound congestion without otherwise affecting network performance.

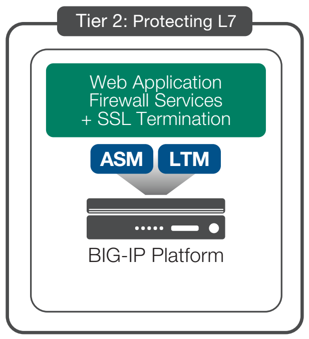
```
% tmsh modify sys db tm.maxrejectrate value 100
```

2.3 Tier 2—Application Defense

The second tier is where you deploy application-aware, CPU-intensive defense mechanisms like login-walls, web application firewall policy and LTM iRules. Tier 2 is also where SSL termination typically takes place. While some organizations terminate SSL at tier 1, it is less common there due to the sensitivity of SSL keys and policies against keeping them at the security perimeter.

2.3.1 Understand GET Floods

Recursive GETs and POSTs are among today's most pernicious attacks. They can be very hard to distinguish from legitimate traffic.



GET floods can overwhelm databases and servers. GET Floods can also cause a "reverse full pipe." F5 recorded one attacker that was sending 100Mbs of GET queries into a victim and bringing out 20Gbs of data.

If you have a signature-based anti-DDoS solution (from F5 or another vendor) leverage it to protect your application. With LTM and ASM, F5 provides many different ways to mitigate difficult application-layer attacks.

Mitigations strategies for GET floods include:

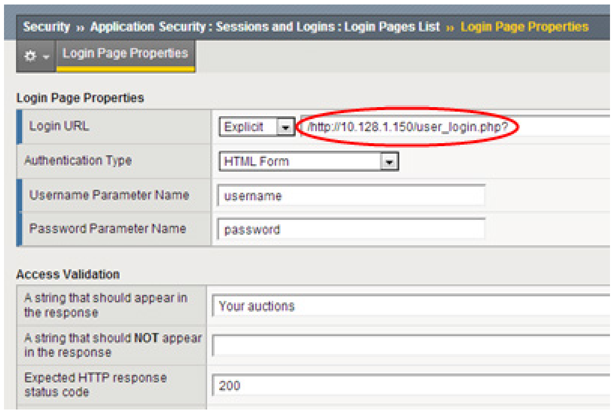
- The Login-Wall Defense
- DDoS Protection Profiles
- Real Browser Enforcement
- CAPTCHA
- Request-Throttling iRules
- Custom iRule

2.3.2 Reduce Threat Surface by Configuring a Login-Wall

The most powerful technique to foil application-level attacks is to allow only authenticated users to access the database portions of your application. Creating a login-wall can be delicate work that is much better done during peacetime and not during a hectic DDoS attack. Note that not all applications can rely on registered users and have to process anonymous traffic, but for those that can, login-walls are the defense.

2.3.2.1 Designate a Login-Wall with ASM

ASM offers facilities to do this within the ASM policy through the use of login pages and login enforcement. This feature will enforce that users may not interact with a set of URLs until they have successfully authenticated themselves at one of the login pages.


Security » Application Security : Sessions and Logins : Login Pages List » Login Page Properties
Login Page Properties

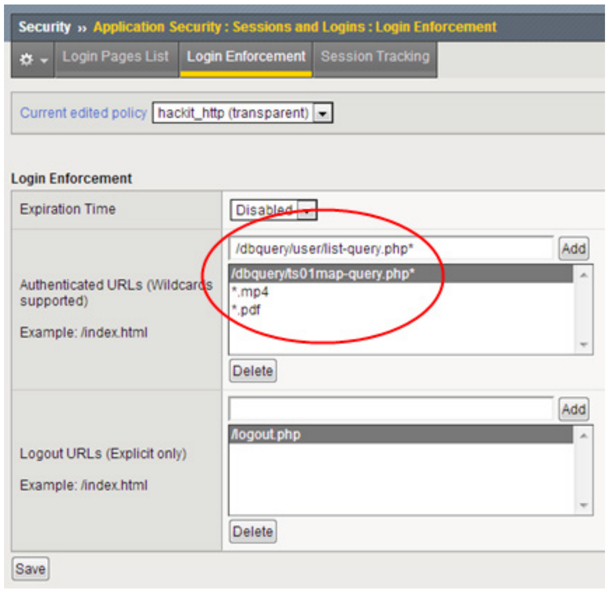
Login Page Properties

Login URL	Explicit /http://10.128.1.150/user_login.php?
Authentication Type	HTML Form
Username Parameter Name	username
Password Parameter Name	password

Access Validation

A string that should appear in the response	Your auctions
A string that should NOT appear in the response	
Expected HTTP response status code	200

First define the login pages from the Security -> Application Security -> Sessions and Logins screen.


Security » Application Security : Sessions and Logins : Login Enforcement
Login Pages List
Login Enforcement
Session Tracking
Add
Delete
Add
Delete
Save

Current edited policy: hackit_http (transparent)

Login Enforcement

Expiration Time	Disabled
Authenticated URLs (Wildcards supported) Example: /index.html	/dbquery/user/list-query.php* /dbquery/ts01map-query.php* *.mp4 *.pdf
Logout URLs (Explicit only) Example: /index.html	/logout.php

Then use the **Login Enforcement** tab to specify which pages need to be protected. Ideally these will be large objects such as .MP4s and .PDFs, and any database queries that could be used against you in an asymmetric attack.

For a full explanation of the Login Enforcement feature, see the section "Creating Login Pages" in the ASM configuration guide.

Note: If you are not sure what resources to protect, you can reconnoiter your own applications—see section 3.2.2.

2.3.2.2 Script a Login-Wall

You can make a login-wall with just an LTM iRule by setting a specific cookie at the login page, and then checking that cookie on every other page. Create this iRule, attach it and test it. Then detach it and keep it in your library to be activated as necessary.

Here's a link to a login-wall iRule on DevCentral.

2.3.2.3 Protect Applications with DoS Protection Profiles

The F5 Web Application Firewall, ASM, includes application-specific "DoS profiles." These powerful profiles detect DoS conditions by monitoring **server latency or http request rates**. ASM can then trigger an optional iRule event as the attack is mitigated.

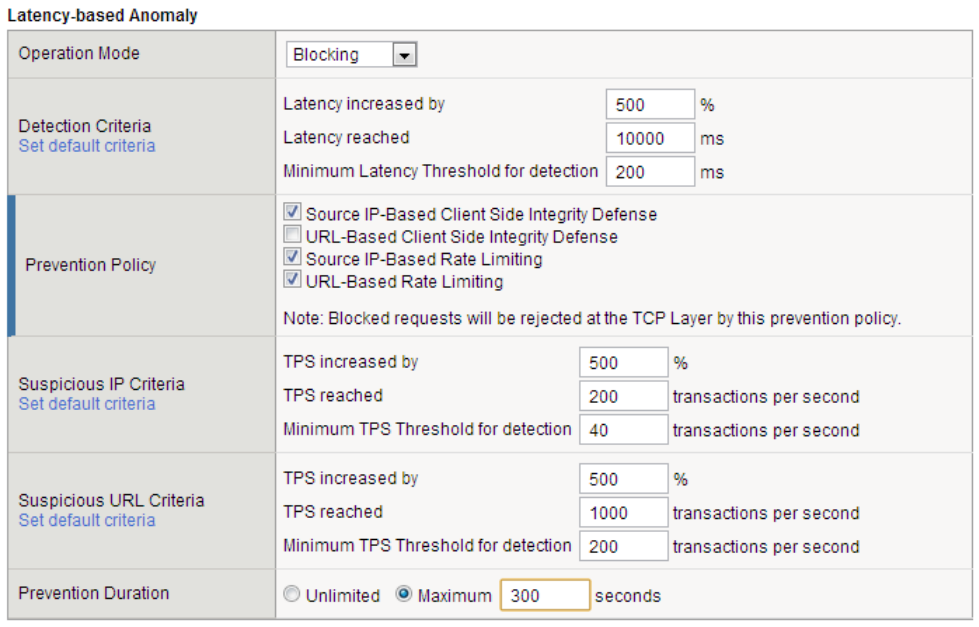
The mitigation options are:

- Drop the suspicious connections.
- Return a JavaScript redirect to the client to enforce that a browser is being used.
- Rate-limit by client address or URI.

Use the following commands to create a DoS profile and attach it to the application:

```
% tmsh create security dos profile my_dos_prof { application add { Lrule1 {
latency-based { url-rate-limiting enabled mode blocking } } } }
% tmsh modify ltm virtual my_vip1 profiles add { my_dos_prof }
```

You can access this DoS profile from the Security tab. Then select DoS Protection. From that screen check Application Security and then configure the L7DOS protection parameters.



Latency-based Anomaly

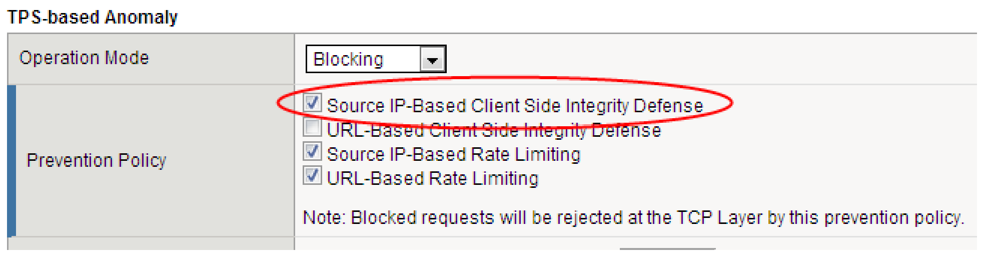
Operation Mode	Blocking
Detection Criteria Set default criteria	Latency increased by 500 % Latency reached 10000 ms Minimum Latency Threshold for detection 200 ms
Prevention Policy	✓ Source IP-Based Client Side Integrity Defense ☐ URL-Based Client Side Integrity Defense ✓ Source IP-Based Rate Limiting ✓ URL-Based Rate Limiting Note: Blocked requests will be rejected at the TCP Layer by this prevention policy.
Suspicious IP Criteria Set default criteria	TPS increased by 500 % TPS reached 200 transactions per second Minimum TPS Threshold for detection 40 transactions per second
Suspicious URL Criteria Set default criteria	TPS increased by 500 % TPS reached 1000 transactions per second Minimum TPS Threshold for detection 200 transactions per second
Prevention Duration	Unlimited [✓ Maximum] 300 seconds

Figure 4. Configuration for the ASM Module's comprehensive L7DOS protection

2.3.2.4 Enforce Real Browsers

Besides authentication and tps-based detection (section 2.3.2.3), there are additional ways that F5 devices can separate real web browsers from probable bots.

The easiest way, with ASM, is to create a DoS protection profile and turn on the “Source IP-Based Client Side Integrity Defense” option. This will inject a JavaScript redirect into the client stream and verify each connection the first time that source IP address is seen.


TPS-based Anomaly

Operation Mode	Blocking
Prevention Policy	[✓] Source IP-Based Client Side Integrity Defense [ ] URL-Based Client Side Integrity Defense [✓] Source IP-Based Rate Limiting [✓] URL-Based Rate Limiting Note: Blocked requests will be rejected at the TCP Layer by this prevention policy.

Figure 5. Insert a Javascript Redirect to verify a real browser

From the command line:

```
% modify security dos profile my_ddos1 application modify { Lrule1 { tps-based {
ip-client-side-defense enabled } } }
```

2.3.2.5 Throttle GET Request Floods via Script

The F5 DevCentral community has developed several powerful iRules that automatically throttle GET requests. Customers are continually refining these to keep up with current attack techniques.

Here is one of the iRules that is simple enough to be represented in this document. The live version can be found at this DevCentral page: HTTP-Request-Throttle

```
when RULE_INIT {
    # Life timer of the subtable object. Defines how long this object exist in the
subtable
    set static::maxRate 10
    # This defines how long is the sliding window to count the requests.
    # This example allows 10 requests in 3 seconds
    set static::windowSecs 3
    set static::timeout 30
}

when HTTP_REQUEST {
    if { [HTTP::method] eq "GET" } {
        set getCount [table key -count -subtable [IP::client_addr]]
        if { $getCount < $static::maxRate } {
            incr getCount 1
            table set -subtable [IP::client_addr] $getCount "ignore"
$static::timeout $static::windowSecs
        } else {
            HTTP::respond 501 content "Request blockedExceeded requests/sec
limit."

            return
        }
    }
}
```

Another iRule, which is in fact descended from the above, is an advanced version that also includes a way to manage the banned IPs address from within the iRule itself:

- URI-Request Limiter iRule—Drops excessive HTTP requests to specific URIs or from an IP

2.3.2.6 Use CAPTCHAs to Eliminate Bots

Another way to mitigate GET floods is by verifying "humanness" by using a CAPTCHA mechanism. The CAPTCHA mechanism shows pictures of scrambled words to the user, who proves his or her humanity by typing the words into a web-form. CAPTCHAs are still one of the best ways to distinguish humans from computers even though hackers and researchers have been trying to "break" them for more than ten years. Advances in pattern-recognition algorithms seem to bring attackers close to automating the CAPTCHA system. It is F5's experience, though, that the computational work required to "break" a CAPTCHA vastly decreases the asymmetric advantage of a modern DDoS attacker, and this keeps these attacks theoretical for now. This means that CAPTCHAs are still an effective means of repelling botnets.

Google offers the reCAPTCHA service, which performs this function while also decoding ancient texts. There is a Google ReCAPTCHA iRule on DevCentral that can be used to provide verification that a human is at the other end of the connection. Download the iRule (approximately 150 lines) and edit it to provide some of the basic information (such as your Google reCAPTCHA key and your DNS server). Make it available on your BIG-IP. Attach it to a virtual server and test it, and then keep it ready for deployment.

2.3.3 Script a Custom Mitigation

If all other techniques must be ruled out, you may find it necessary to write a custom iRule to defend your application from an application layer attack. These custom iRules typically fall into one of two categories: filtering and indiscriminate blocking.

While this is perhaps the most "manual" of all the techniques in this document, it is also the most powerful and the most used among agile F5 customers. The extreme programmability of F5 iRules gives an administrator the ability to block nearly any kind of attack provided that he or she can script well enough. Security-related iRules protect many organizations today and are one of F5's real differentiators for application-layer DDoS attacks.

If the attack leaves you with some outbound Internet access, search devcentral.f5.com for some of the keywords that might match your attack. You may find that an iRule has already been written for you!

To write your own iRule, first dissect the attack traffic and find a feature about the incoming attack traffic that you can use to distinguish bad traffic from good. Then write an iRule that detects that traffic and drops it. If you are not an iRule author, there are iRules scattered throughout this document (and all over DevCentral) that you can use as examples. Attach your new iRule to the application's virtual server.

An example of a simple security iRule is this early **Dirt Jumper** iRule, which keys in on the fact that the malware does not include a // in its referrer field (see also section 4.9).

```
when HTTP _ REQUEST {
 if { [HTTP::header exists "Referer"] } {
   if { not ([HTTP::header "Referer"] contains "\x2F\x2F") } {
     drop
   }
 }
}
```

If you are not able to easily distinguish good traffic from bad, you can write an iRule that discards traffic based on the object being requested. For example, if the attackers are requesting a particular large PDF or MP4 file, you can use an iRule to drop all requests to that object.

```
ltm data-group internal block _ uris {
    records {
        /faqs/faq.mp4 { }
        /locator/locations.pdf { }
        /cgi-bin { }
    }
    type string
}
```

You can also use external data groups that are hosted outside the BIG-IP.

Then use a simple scrubber iRule to drop requests that request URIs that match the data class.

```
when HTTP _ REQUEST {
  set origUri ""
  if {[HTTP::query] eq ""}{
    set origUri "[URI::path [HTTP::path]][URI::basename [HTTP::path]]"
  } else {
    set origUri "[URI::path [HTTP::path]][URI::basename [HTTP::path]]?[HTTP::query]"
  }
  if { [class match -- [ string tolower $origUri ] contains block _ uris] } {
    drop
  }
}
```

This is definitely not the best solution, because it will turn away good traffic as well as bad. It may keep your servers alive, but if you have the time and ability to write a rule like the one above, you can usually find something to distinguish good traffic from bad.

2.3.4 Mitigate SSL DDoS at Tier 2

While it is possible and sometimes preferable to terminate SSL at either tier, F5 recommends a physical (non-virtual) appliance for terminating SSL at tier 2. Many SSL DDoS attacks will be mitigated by the very presence of the SSL acceleration hardware used in F5 physical devices. These include:

- SSL protocol attacks
- SSL replay attacks
- SSL connection floods

Whether hardware is used or not, F5 will also mitigate SSL connection floods with adaptive reaping (see section 2.2.7) and a high-capacity connection table (section 2.2.6).

The SSL renegotiation attack can be mitigated in one of two ways. In most cases, you can simply temporarily disable the SSL renegotiation feature at the virtual server's SSL clientssl profile. However, very long-lived connections (such as automatic teller machines or database connections) will still require the ability to renegotiate. For those cases, see the iRule in section 4.8.

2.3.5 Understand Connection Multiplexing and Port Exhaustion

In general, do not perform functions like connection multiplexing and SNAT at tier 1. These functions and associated extras like the insertion of the X-Forwarded-For header should be processed at tier 2.

2.3.5.1 Connection Multiplexing

A layer 7 DDoS can exhaust back-end resources such as connection tables. One way to combat this effect is by multiplexing the connections through the load balancer. On LTM this feature is called OneConnect and can decrease the number of TCP connections used by an order of magnitude while still maintaining (or even improving) overall requests-per second.

The OneConnect feature should be tested with each application prior to being used as DDoS defense. Some applications may rely on separate connections per user.

2.3.5.2 Port Exhaustion

A SNAT supports approximately 64,000 concurrent connections per destination IP. A high volume of requests can exceed the 64,000 connection limit and result in TCP port exhaustion. You can use a SNAT pool to overcome this limitation. Configure and set the appropriate IP address within the SNAT pool to mitigate the exhaustion.

For example, if your virtual server **vip1** is using simple automap source address translation, you can change it to use a pool of IP addresses with the following commands. This example uses just three addresses to increase the available ports from 64,000 to 192,000.

```
% tmsh create ltm snatpool  ddos _ snatpool members add { 10.1.20.161 10.1.20.162
10.1.20.163 }
% tmsh modify ltm virtual vip1 source-address-translation { pool ddos _ snatpool }
```

For each address added to the SNAT pool, you may want to assign a discrete timeout value (the default is indefinite). With an idle timeout, BIG-IP can close idle connections and help protect upstream stateful firewalls.

```
% tmsh modify ltm snat-translation 10.128.20.161 { ip-idle-timeout 60 }
```

Repeat this command for each of the addresses in your snatpool (10.1.20.162 and 10.1.20.163 in the example above).

3 More DDoS Recommended Practices

3.1 Mitigate DNS DDoS

DNS is the second-most-targeted service after HTTP. When DNS is disrupted, all external data center services (not just a single application) are affected. This single point of total failure, along with the historically under-provisioned DNS infrastructure, makes DNS a very tempting target for attackers. Even when attackers are not specifically targeting DNS, they often inadvertently do: if the attack clients are all querying for the IP of the target host before launching their floods, the result is an indirect attack against DNS.

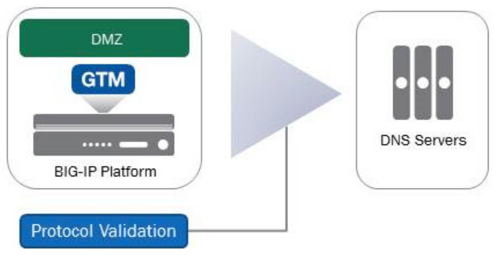


Figure 6: Mitigating DNS DDoS

Because of the relatively simple, UDP-based DNS protocol, a DNS attack has two main characteristics:

- DNS attacks are easy to generate.
- DNS attacks are difficult to defend against.

There are four on-premises strategies for mitigating DNS DDoS attacks:

- Use Protocol Validation.
- Detect and prevent DNS floods.
- Overprovision DNS Services against NXDOMAIN query floods.
- Blacklist as a last resort.

3.1.1 Consider the Placement of DNS services

You may notice that in Figure 1 the DNS service exists as its own set of devices behind the security perimeter. Often DNS is served from this so-called DMZ in between security tiers. This is done to keep DNS independent of the applications that it serves—for example, if that part of the data center goes dark, DNS can redirect requests to a secondary data center (or the cloud). **F5 recommends this strategy of keeping DNS separate** from the security and application tiers for maximum flexibility and availability.

Some large enterprises with multiple data centers will go one step further and serve DNS outside the main security perimeter using a combination of F5's GTM DNS Express and the AFM firewall module. The main benefit of this approach is that the DNS services remain available even in the event that tier 1 firewalls go offline due to DDoS.

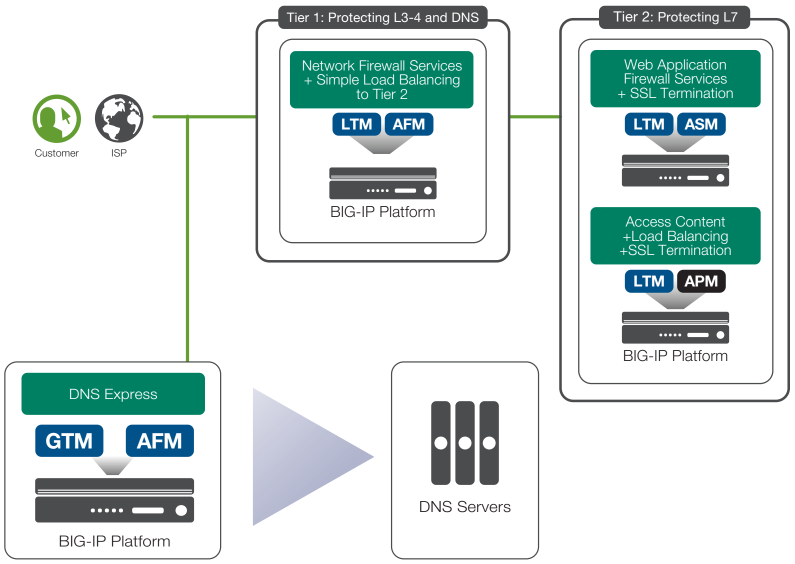

Tier 1: Protecting L3-4 and DNS
Network Firewall Services + Simple Load Balancing to Tier 2
LTM
AFM
BIG-IP Platform
Tier 2: Protecting L7
Web Application Firewall Services + SSL Termination
LTM
ASM
Access Content +Load Balancing +SSL Termination
LTM
APM
BIG-IP Platform
Customer
ISP
DNS Express
GTM
AFM
BIG-IP Platform
DNS Servers

Figure 7: Alternate External DNS Architecture

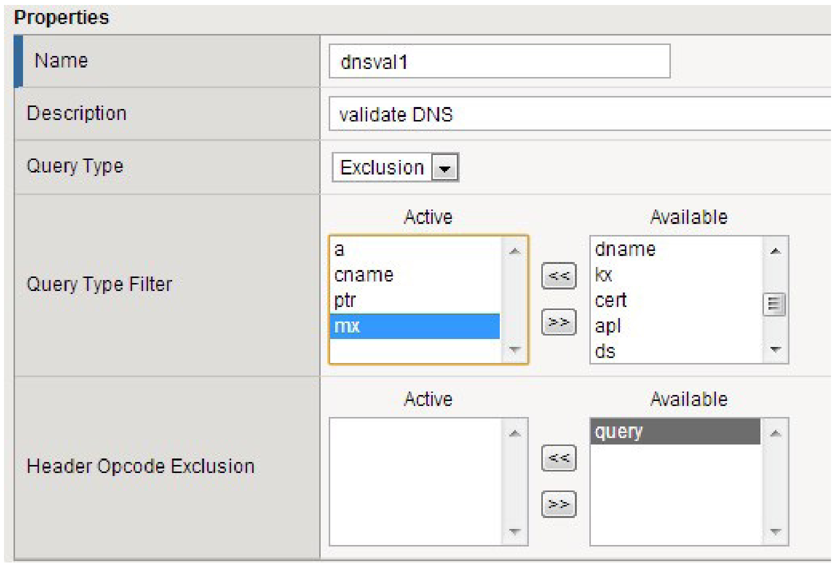
3.1.2 Use Protocol Validation to Protect DNS Services

Regardless of whether you serve DNS inside or outside the DMZ, you can use either GTM or AFM to validate the DNS requests before they hit the DNS server.

If you have GTM performing global server load balancing, it will likely be blocking many DNS DDoS attacks already. You can view the DNS query/response performance from the main dashboard in the GTM GUI. Because GTM is full-proxy for DNS, it will automatically validate every request and discard the invalid ones.

However, you may find that your servers are still overwhelmed by valid-looking requests. If you have the F5 firewall module AFM, you can use a **Protocol Security Profile** to further filter only specific types of DNS requests.

From the **Security** tab, select **Protocol Security** and then **Security Profiles**. Select **DNS** and press the **Create** button. At this screen you can build a protocol security profile to filter or block different types of requests.

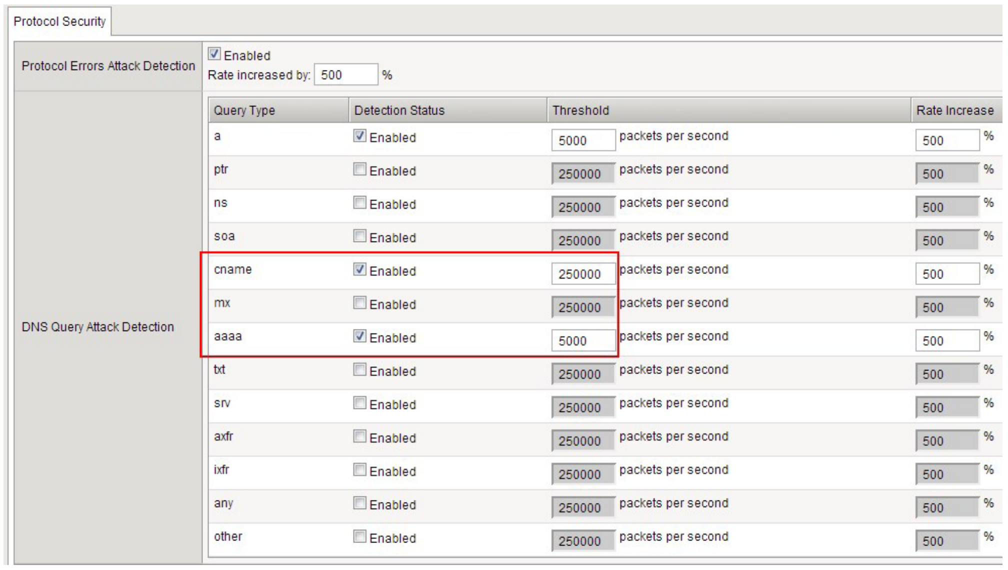

Properties

Name	dnsval1
Description	validate DNS
Query Type	Exclusion
Query Type Filter	Active: a, cname, ptr, mx — Available: dname, kx, cert, apl, ds
Header Opcode Exclusion	Active: — Available: query

Figure 8: DNS Protocol Validation

3.1.3 Detect DNS Floods

The F5 firewall module AFM has a powerful DNS DDoS function—it can detect DNS floods by record type. From the **Security** tab, select **DoS Protection**, then **DoS Profiles** and finally **create**. From the creation screen click the checkbox for DNS and set and accept the threshold parameters.



Protocol Security

Protocol Errors Attack Detection: ✓ Enabled
Rate increased by: 500 %

DNS Query Attack Detection

Query Type	Detection Status	Threshold		Rate Increase	
a	✓ Enabled	5000	packets per second	500	%
ptr	☐ Enabled	250000	packets per second	500	%
ns	☐ Enabled	250000	packets per second	500	%
soa	☐ Enabled	250000	packets per second	500	%
cname	✓ Enabled	250000	packets per second	500	%
mx	☐ Enabled	250000	packets per second	500	%
aaaa	✓ Enabled	5000	packets per second	500	%
txt	☐ Enabled	250000	packets per second	500	%
srv	☐ Enabled	250000	packets per second	500	%
axfr	☐ Enabled	250000	packets per second	500	%
ixfr	☐ Enabled	250000	packets per second	500	%
any	☐ Enabled	250000	packets per second	500	%
other	☐ Enabled	250000	packets per second	500	%

Figure 9: DNS Flood Detection

The Protocol Errors checkbox means that the system detects malicious or malformed DNS queries, and displays, in percentage, how much of an increase in DNS query traffic is legal before the system tracks malformed and malicious DNS queries.

Note: At this time, this firewall feature detects floods but does not drop packets to mitigate them.

3.1.4 Overprovision DNS Services Against Query Floods

DNS services have been historically under-provisioned. Part of the reason for this is that for many organizations the ownership of DNS has not been a positive development for any particular team. Whatever the real reason, a significant percentage of DNS deployments are under-provisioned to the point where they are unable to withstand even small- to medium-size DDoS attacks.

DNS Caches have become popular as they can boost the perceived performance of a DNS cache and they have provided some resilience against standard DNS query attacks. Attackers have switched to what is called the “no such domain” (or NXDOMAIN) attacks, which quickly drain the performance benefits provided by the cache.

F5’s recommended way to remedy this is to front the DNS service with the special, high-performance DNS proxy module called DNS Express. DNS Express acts as an absolute resolver in front of the existing DNS servers. It loads the zone information from the servers then resolves every single request or returns NXDOMAIN. It is not a cache and cannot be emptied via NXDOMAIN query floods.

In GTM or DNS Services, DNS Express can serve 250,000 requests per second per CPU and is therefore resistant to all but the most virulent DNS attacks. The DNS servers remain in place to manage the zone data.

3.1.5 Blacklist as a Last Resort

DNS traffic is traditionally UDP, which is easy to generate and easy to spoof. Conventional layer 3 and 4 defenses, such as blacklisting by source IP, are usually ineffective against a DNS flood. In fact, blocking DNS requests by source IP can be downright dangerous. For example, if you unknowingly block requests from a major ISP, you may deny service to many legitimate users without realizing it.

See "BIG-IP Systems: DOS Protection and Protocol Firewall Implementations" (chapter 3—Detecting and Preventing DNS DDoS attacks).

3.1.6 Beware DDoS Attack Participation

3.1.6.1 Unused Query Types

An attacker can trick a DNS service into bombing a third-party target by sending queries for unused services. Use the AFM screens (see Figure 9 above) to disable query types that you aren't using. Then, when queries come in for these types they will be dropped. No response will be provided, thereby helping to avoid participation in a DDoS attack.

This is especially true for MX (mail services) and zone transfers. If your organization does one transfers at known, specific times, keep the IXFR, AXFR and ZXFR types disabled at all other times.

3.1.6.2 DNSSEC

DNSSEC is an important evolution in the global domain name service. Ultimately it will cut down on deceptive practices such as phishing. The picture is more complicated for DNS DDoS. DNSSEC responses are sometimes 10–20 times larger than traditional DNS UDP responses. This means that DNSSEC servers are actually tricked into attacking other computers by inadvertently bombarding them with invalid responses.

With GTM, F5 has the highest-performing DNSSEC solution on the market. The capacity of GTM could make for a potent weapon if it were to be used as an attack vector. Therefore, GTM allows you to rate-limit the number of responses to prevent itself from participating in an attack.

```
% tmsh modify sys db dnssec.maxnsec3persec value 10
```

The **dnssec.maxnsec3persec** variable controls the upper limit of NSEC3 authoritative NXDOMAIN messages that GTM will send per second. 0 is unlimited and the default. A more restrictive value, such as between 10 and 100 per second, may prevent GTM itself from being used during an attack.

```
% tmsh modify sys db dnssec. signaturecachensec3 value true
```

Setting the **dnssec.signaturecachensec3** variable to false prevents NXDOMAIN messages from using the GTM cache at all, thus preventing an attacker from filling GTM's cache with "no such domain" responses.

3.2 Additional DDoS Best Practices Preparation Procedures

The time spent preparing for a DDoS attack will increase the effectiveness of your defense. Here are a few more ways that you can prepare your organization for a DDoS attack.

3.2.1 Configure and Verify Logging

During an attack there is a good chance that you will be sending diagnostics and logging anomalies and traffic spikes. High performance is critical when dealing with a large DDoS attack. Instrumentation is important as well, meaning that you will want to use the High-Speed Logging facility of BIG-IP to send this information to a third-party logging device such as Splunk or a SIEM such as ArcSight.

Note: You **MUST** use the High-Speed-Logging facilities of BIG-IP at tier 1 when mitigating a DDoS attack. Do not use the local logging; an intense DDoS attack can overwhelm the local disk-based logging.

3.2.1.1 Set Up High-Speed Logging

1. Create a pool to map to your external log servers (in this case they are syslog). Rewrite as necessary for ArcSight, TrustWave or whichever SIEM solution your environment supports. Then create the log config objects to format and forward the strings properly.

```
% tmsh create ltm pool hsl _ pool members add { 10.128.10.250:514 }
% tmsh create sys log-config destination remote-high-speed-log log _ dest _ HSL {
pool-name hsl _ pool }
% tmsh create sys log-config destination remote-syslog log _ dest _ format { format
rfc5424 remote-high-speed-log log _ dest _ HSL }
% tmsh create sys log-config publisher log _ pub _ ddos { destinations { log _
dest _ HSL log _ dest _ format } }
```

2. Create a log profile object using the GUI.
   Access the **Security > Event Logs > Logging Profiles** page. Create a log profile using the following:

Profile Name	ddos_log_profile
Network Firewall	Enabled
Network Firewall: Publisher	log_pub_ddos
Log Rule Matches	Accept, Drop, and Reject
Log IP Errors	Enabled
Log TCP Errors	Enabled
Log TCP Events	Enabled
Storage Format	field-list Select all Available Items and move them to the Selected Items list

3. Associate that log profile object with the virtual servers protecting your application.

```
% tmsh modify /ltm virtual vip1 { security-log-profiles add { ddos _
log _ profile }  }
```

3.2.2 Reconnoiter Your Own Applications

Modern DDoS attackers will reconnoiter an application days or weeks before launching their DDoS attack. They will spider your website and retrieve the **load-time** and **data-size** for each valid URI. By sorting the resulting dataset, they will quickly isolate your most CPU- or database-intensive queries and your largest objects (such as PDFs and MP4s). During the DDoS attack they will repeatedly query for these objects, overwhelming your infrastructure.

Though section 3.8 will help you mitigate that attack when it happens, you can help yourself beforehand by reconnoitering your own applications. This will give you advanced visibility into what URIs and subsystems will be likely targets, allowing you to make more informed triage decisions later.

Ideally you have a tool like LoadRunner or another performance-monitoring tool that can provide you with the metrics you need. If you lack this capability, perhaps the simplest way to retrieve the basic table of URL, load-time and data-size is to run the wget utility, which is available on most Linux distributions. Run it with the following syntax:

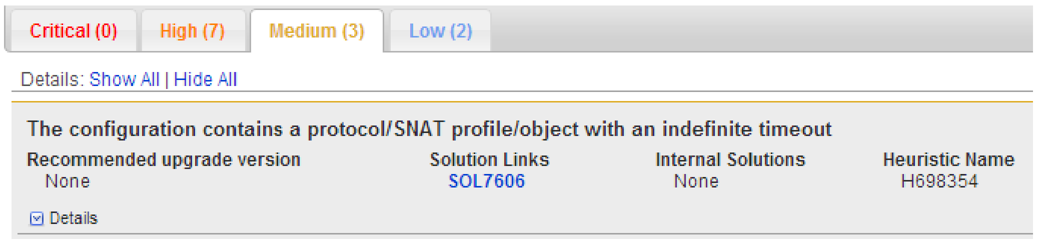
```
% wget -r --spider http://10.128.10.150 2>&1 | grep saved
2013-08-25 15:44:29 (2.48 MB/s) - `10.128.1.150/index.html' saved [22304]
2013-08-25 15:44:29 (5.53 MB/s) - `10.128.1.150/index.php' saved [22304]
2013-08-25 15:44:29 (7.06 MB/s) - `10.128.1.150/sell.php' saved [41695]
```

The last number (in square brackets) is the data-size of the request. You will have to get the load-time by subtracting the times (second field) from each other.

3.2.3 Validate the Health of Existing BIG-IP Devices with iHealth

F5 provides a cloud-based diagnostics and heuristics service called iHealth. iHealth will examine an F5 device's configuration and make recommendations to keep BIG-IP fast, secure and available. While the majority of the settings may be more applicable to the first two, some of these settings can apply to availability and, by extension, to DDoS-resilience.

In this example, iHealth is showing that a SNAT pool has been configured without timeout values. This can be a reminder to a resource-conscious administrator to ensure that the SNAT pool used for their hardened virtual servers should include idle timeouts to keep the number of connections down and prevent an upstream firewall from tipping.



The configuration contains a protocol/SNAT profile/object with an indefinite timeout

Recommended upgrade version	Solution Links	Internal Solutions	Heuristic Name
None	SOL7606	None	H698354

See the iHealth web site for more information about iHealth.

3.2.4 Prepare a DDoS Playbook

A DDoS Playbook or Runbook is a procedural manual to assist your IT employees in combatting a DDoS attack. A good playbook will help new (and existing) administrators combat a DDoS attack. The Playbook should be kept current with updated whitelists and contact information.

A few organizations will perform periodic DDoS drills (or even tests) against themselves to keep current and to test the playbook. Try to have your staff practice the procedures from the playbook when key people are not present—attacks do not always happen at the most convenient schedule.

If you do not have a playbook, one is available from F5.

3.2.5 Review Defensive Tactics in the Two-Tier Architecture

Some of the defensive tactics described in the previous sections are worth review, especially for administrators using a non-F5 network firewall.

Remember:

- SNAT pools mitigate port exhaustion at tier 1.
- Shape traffic at tier 1.
- Aggressively reap TCP connections.
- Blacklist DNS only as a last resort.
- Implement login-walls and CAPTCHAs at tier 2.
- Disable optional CPU-intensive features at tier 2.
- Always use high-speed remote logging.

By implementing the proposed suggestions in this Best Practices guide you will have done a lot to prepare your applications for a DDoS attack.

4 Conclusion

Understanding the modern threat spectrum of denial-of-service attacks is important. Understanding how to make use of the defensive equipment that you already have is even more important.

Depending on your resources and need, you may already have redesigned your network for DDoS resilience. If this is something that you are considering, then pay close attention to the recommended multi-tier architecture described in section 2.1. Even if your network security is not completely built from F5 technology, it still makes sense to tackle layer 4 and layer 7 independently for DDoS purposes.

Architecture
Recommended Practices
Process

By following the recommended practices, you will be preparing your network, applications and people to be attack-resistant.

The final step in F5's recommended practices for DDoS mitigation is to prepare a DDoS playbook. Such a playbook is a real-time procedural guide for mitigating an attack that includes worksheets and logs. F5 can provide you with a template to get started.

Experts predict that DDoS will be an issue on the Internet for a long time to come. Soon, being prepared will not just be an option but a requirement.

Appendix

Application Attack Taxonomy and Countermeasures

The following section recommends mitigations for specific layer 7 attack vectors. Many of these are the “slow-and-low” type of attacks that can be particularly pernicious.

Contents

Slowloris

Slowloris is a common HTTP vector where an attacker will send (very slowly) small HTTP headers to keep the HTTP session alive (e.g., "**X-a: b**" every 299 seconds). If the virtual server is currently load-balancing at layer 4, consider switching it to layer 7. This will add some native protection when the HTTP profile is added.

```
% tmsh list ltm virtual vip1
ltm virtual vip1 {
      destination 10.128.10.141:http
      profiles {
              fastL4 { }
     }
}
% tmsh modify ltm virtual vip1 profiles replace-all-with { tcp http }
```

This will cause BIG-IP to absorb the Slowloris connections. If you become concerned that too many are building up and causing problems for other devices (such as a firewall), use the following Slowloris iRule to drop any connection that has not completed after 10 seconds (feel free to adjust this number).

```
# Slowloris iRule
when CLIENT_ACCEPTED {
      set hsl [HSL::open -proto UDP -pool hsl_pool]
      set rtimer 0
      after 10000 {
              if { not $rtimer } {
                      drop
                      HSL::send $hsl "Dropped [IP::client_addr] - connection too
slow"
              }
      }
}

when HTTP_REQUEST {
      set rtimer 1
}
```

Keep Dead

This attack is based on consuming CPU and RAM. By using Keep-Alive and the HTTP HEAD method, it can create a flood of requests without triggering a firewall defense that is based on the number of connections opened to the server.

The ASM module can disallow HEAD requests (which are not typically used by browsers). You can reject HEAD requests by configuring the "Allowed Methods" in the application security policy in question.

See solution 12312 for more information.

Low Orbit Ion Cannon (LOIC)

The Low Orbit Ion Cannon is a voluntary botnet tool closely associated with the hacktivist group Anonymous. While the tool uses SYN floods and UDP floods, it is most famous for its layer 7 HTTP floods. Assuming that the SYN and UDP floods have been mitigated (see sections 2.2.2 and 2.2.3), the last step is to mitigate the LOIC GET floods.

Often the fastest way to do this is to filter it on the attack "protest message" included with each LOIC HTTP request. Use Wireshark or tcpdump or another tool to isolate the message, then add that message to a datagroup. Use %20 to represent spaces. The message may change over time and you may need to monitor it for the duration of the attack.

```
ltm data-group anonmsgs {
    records {
        Somos%20legi { }
        U%20dun%20goofed { }
    }
    type string
}
```

Note that you can use external data classes that are hosted outside the BIG-IP—see "**help search data-group**" in the tmsh command shell.

Then use a simple scrubber iRule to drop requests that contain any payloads in that data class.

```
ltm rule loic _ defense _ rule {
    when CLIENT _ ACCEPTED {
              set hsl [HSL::open -proto UDP -pool hsl _ pool]
    }
    when HTTP _ REQUEST {
    if { [class match [HTTP::uri] contains anonmsgs] } {
      drop
      HSL::send $hsl "Dropped [IP::client _ addr] - suspected Low Orbit Ion Cannon"
    }
  }
}
```

Slow-POSTs

The heart of the Slow-POST attack relies on sending a POST request with given "content-length," which is typically a large number, and then very slowly sending the message body to the server, while keeping the idle time long. The server will leave the connection open as it continues to receive data. If a large number of these requests are executed against a server, there is potential for exhausting the connection table, which would leave the server unable to respond to further requests.

If you have the ASM module, you can mitigate slow posts with two of the variables found in the ASM system variables screen—**Navigate to Security : Options : Application Security : Advanced Configuration : System Variables**—and modify the following variables.

slow_transaction_timeout (defaults to 10 seconds). Lower this value as needed.

max_slow_transactions (defaults to 25 transactions). Lower this value to 5 or less as needed.

If you do not have ASM, then see this LTM iRule to mitigate Slow-POST. It can be used with the Slow-read iRule in the next section (just attach them as two separate iRules) because the Slow-read iRule is server-based and the Slow-POST iRule is client-based.

Zero Window Attacks

The Zero Window attack is a difficult-to-detect layer 4 attack. It works by establishing a TCP connection to the target, requesting some data, and then setting the TCP window size to zero. This stalls the connection at the server, cache or middleware.

If the attacker is setting a TCP zero window length against a BIG-IP you can use the zero-window-timeout tcp profile value mentioned in section 2.2.2 to mitigate.

Slow-Read Attack

The Slow-read attack works by sending legitimate HTTP requests and then reading the HTTP responses very slowly from the buffer, aiming to keep as many connections as possible in an active state on the victim.

In version 11.3.0, the ASM module's low-and-slow prevention works on inbound requests such as a Slow POST. For Slow-read, use the following LTM iRule mitigation:

```
when SERVER _ CONNECTED {
    TCP::collect
}

when SERVER _ DATA {
    set rtimer 0

# Time in milliseconds before HTTP response read is considered slow:
    after 5000 {
        if { not $rtimer} {
            set hsl [HSL::open -proto UDP -pool hsl _ pool]

# Slow read detected for this server response. Increment the count by adding a
table entry:
# Add the client source IP::port to the subtable with a timeout
            table set -subtable "MyApp" "[IP::client _ addr]:[TCP::client _ port]"
"ignored" 180

# If we are over the concurrency limit then reject
            if { [table keys -subtable "MyApp" -count] > 5} {
                clientside {reject}
                table delete -subtable "MyApp" "[IP::client _ addr]:[TCP::client _
port]"
                HSL::send $hsl "Dropped [IP::client _ addr] - reading too slow"
            }
        }
    }

    TCP::notify response
    TCP::release
    TCP::collect
}

when USER _ RESPONSE {
    set rtimer 1
}

when CLIENT _ CLOSED {
    table delete -subtable "MyApp" "[IP::client _ addr]:[TCP::client _ port]"
}
```

RUDY

R-U-Dead-Yet (RUDY for short) uses Slow-POST and a generic HTTP DoS attack via long-form field submissions. See the section 4.4 for the mitigation for the Slow-POST attack.

Apache Killer

The Apache Killer is also known as a Range Attack. When a client browser (such as a mobile handset browser) needs just part of document, it can request a "range" of the data with an HTTP range header. If the client wants just the first 100 bytes, it could say:

```
Range:bytes=0-100
```

The Apache Killer attack works by requesting multiple, overlapping ranges that confuse web servers like Apache:

```
Range:bytes=0-,5-1,5-2,5-3,…
```

There are three ways to mitigate Apache Killer. You can modify the HTTP profile to simply remove the Range header. For example, if your http profile was named "http_ddos2," you would run this command:

```
  % tmsh modify ltm profile http http _ ddos2 { header-erase range }
```

A more surgical way to mitigate Apache Killer is with the following iRule, which only removes range requests when more than five ranges are requested.

```
  when CLIENT _ ACCEPTED {
               set hsl [HSL::open -proto UDP -pool hsl _ pool]
  }
  when HTTP _ REQUEST {
      # remove Range requests for CVE-2011-3192 if more than five ranges are
  requested
      if { [HTTP::header "Range"] matches _ regex {bytes=(([0-9\- ])+,){5,}} } {
          HTTP::header remove Range
          HSL::send $hsl "Client [IP::client _ addr] sent more than 5 ranges. Erasing
  range header."
      }
  }
```

The third method of mitigation using BIG-IP solutions is to use the following ASM attack signature to detect and act upon an attack using this technique.

```
pcre:"/Range:[\t ]*bytes=(([0-9\- ])+,){5,}/Hi";
```

SSL Renegotiation

If you are seeing a lot of renegotiations happening from specific SSL clients, you might be suffering an SSL renegotiation attack. The easiest way to mitigate it is to disable SSL renegotiation from the virtual server's associated clientssl profile. However, if you need to support renegotiation for legitimate clients (such as old "step-up" or server-gated-cryptography browsers) while still mitigating the attack, you can use this iRule, or others like it. This rule closes any connection that attempts more than five renegotiations in a minute:

```
when RULE _ INIT {
    set static::maxquery 5
    set static::mseconds 60000
}
when CLIENT _ ACCEPTED {
    set ssl _ hs _ reqs 0
    set hsl [HSL::open -proto UDP -pool hsl _ pool]
}
when CLIENTSSL _ HANDSHAKE {
    incr ssl _ hs _ reqs
    after $static::mseconds { if {$ssl _ hs _ reqs > 0} {incr ssl _ hs _ reqs -1} }
    if { $ssl _ hs _ reqs > $static::maxquery } {
        after 5000
        drop
        HSL::send $hsl "Dropped [IP::client _ addr] - too many SSL renegotiations"
    }
}
```

Dirt Jumper iRule

Certain versions of the Dirt Jumper tool do not include a // in their referrer field. Here is a simple iRule to detect and drop Dirt Jumper connections.

```
when CLIENT _ ACCEPTED {
    set hsl [HSL::open -proto UDP -pool hsl _ pool]
}
when HTTP _ REQUEST {
 if { [HTTP::header exists "Referer"] } {
   if { not ([HTTP::header "Referer"] contains "\x2F\x2F") } {
     HSL::send $hsl "DDoS Dirt-Jumper HTTP Header Structure missing x2f x2f
Referer protocol identifier from [IP::client _ addr]"
     drop
   }
 }

}
```

F5 Networks, Inc. 401 Elliott Avenue West, Seattle, WA 98119 888-882-4447 www.f5.com

F5 Networks, Inc. Corporate Headquarters info@f5.com	F5 Networks Asia-Pacific apacinfo@f5.com	F5 Networks Ltd. Europe/Middle-East/Africa emeainfo@f5.com	F5 Networks Japan K.K. f5j-info@f5.com

Solutions for an application world.