
Introduction to ASM/AWAF Behavioral DoS Documentation

Release 1.0

Michael Everett

Aug 09, 2018

Contents:

1	Welcome	1
1.1	Getting Started	1
1.2	Base Configuration and Traffic Baseline	5
1.3	Application Security DoS Profiles	14
1.4	Stress-Based and Behavioral DoS Profile Settings	17
1.5	Request Signatures	21
1.6	Bad Actor Detection	24
1.7	Bad Actor Detection and Request Signatures	26

CHAPTER 1

Welcome

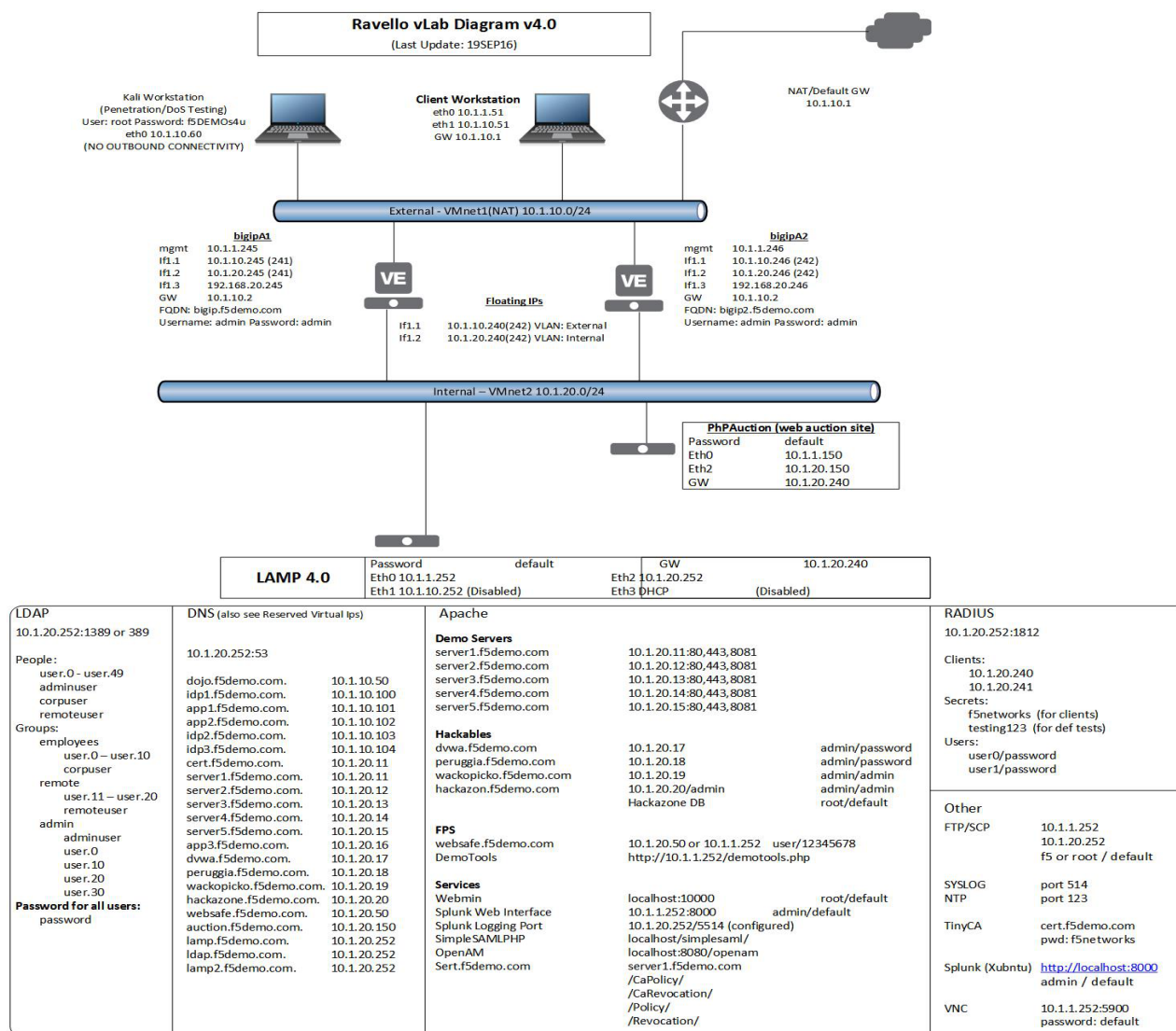
F5 Application Security Manager (ASM), Advanced WAF, and DDoS Hybrid Defender products all include advanced functionality for defending L7DoS attacks. In this self-paced lab, attendees will have an opportunity to explore one component of the overall feature set, L7 Behavioral DoS (BaDOS), leverage BaDOS to mitigate various L7DoS attacks, and examine the built-in reporting and monitoring functions provided by ASM. At the conclusion of the lab, the attendee will have comfort in the basics of BaDOS, how the feature is deployed, and the types of attacks it can be used to mitigate.

1.1 Getting Started

Please follow the instructions provided by the instructor to start your lab and access your jump host.

Note: All work for this lab will be performed exclusively from the Linux Workstation jumphost. No installation or interaction with your local system is required.

1.1.1 Lab Topology



1.1.2 Lab Components

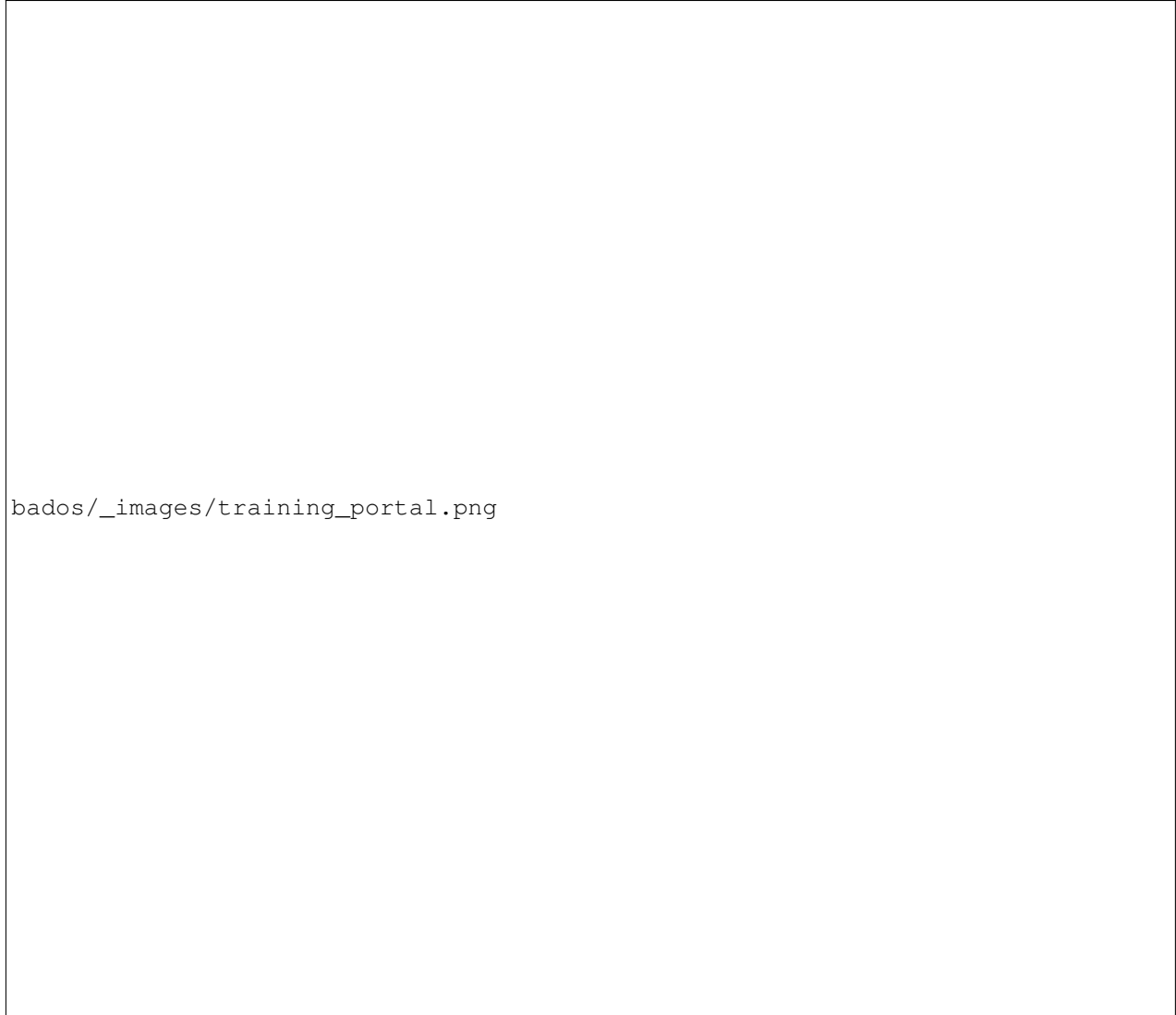
The following table lists VLANS, IP Addresses and Credentials for all components:

Component	VLAN/IP Address(es)	Credentials
bigip01	<ul style="list-style-type: none">• Management: 10.1.1.245• Internal: 10.1.20.245• External: 10.1.10.245	admin/admin
bigip02	<ul style="list-style-type: none">• Management: 10.1.1.246• Internal: 10.1.20.246• External: 10.1.10.246	admin/admin
Ubuntu Linux Workstation	<ul style="list-style-type: none">• eth0: 10.1.1.51• eth1: 10.1.10.51	f5student/f5DEMOS4u
Kali Linux Workstation	<ul style="list-style-type: none">• eth0: 10.1.10.60	root/f5DEMOS4u

1.1.3 Accessing Lab Environment

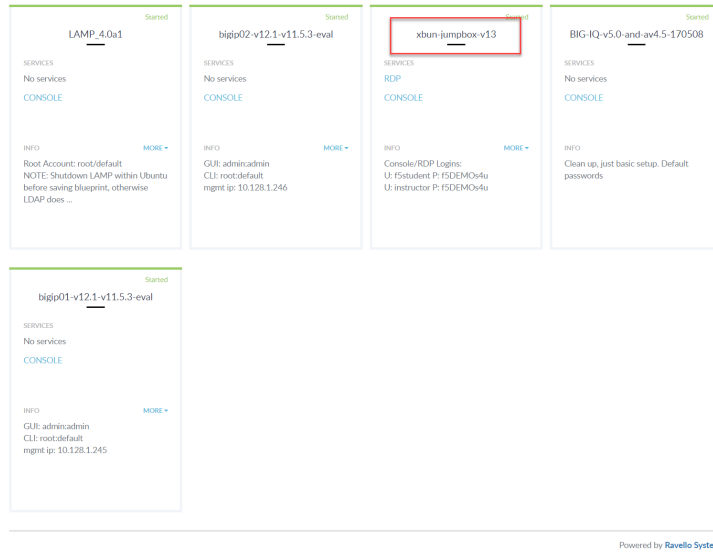
Please follow the instructions below to access the lab environment.

1. Open a browser and go to <http://training.f5agility.com/>

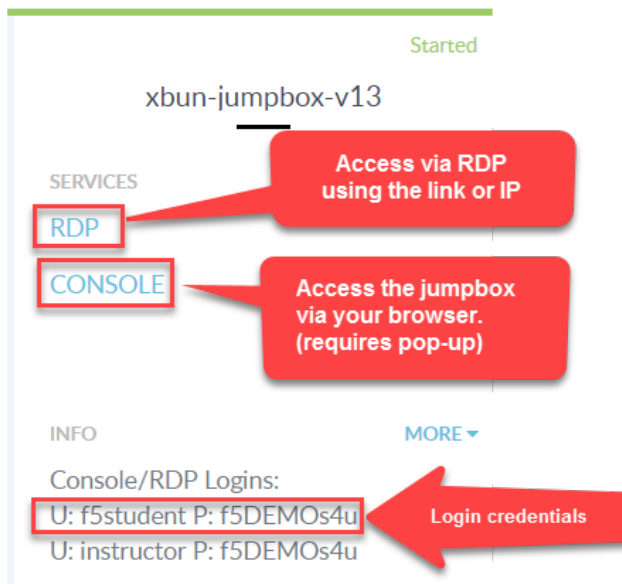


bados/_images/training_portal.png

2. Use the class number and student number included on the class survey to login to the training portal. Once logged in:
 - (a) Look for the **xubuntu-jumpbox-vxx**. You will use the Xubuntu Jumpbox for all the labs. (see below)



- (b) You can click on **RDP** to RDP to the Xubuntu Jumpbox, or you can select the **CONSOLE** link and access the jumpbox via your browser. **The CONSOLE link requires you turn off pop-up blockers.**



1.2 Base Configuration and Traffic Baseline

In this module, we will configure the base DoS profile and Local Traffic Manager objects used in the remaining modules. Additionally, you will generate traffic needed for Advanced Web Application Firewall Behavioral DoS engine to build a learning baseline.

Objectives:

- Create DoS Profile
- Create Logging Profile and attach to virtual server
- Create iRule for inserting X-Forwarded-For headers and attach to virtual server
- Generate good traffic to establish BaDOS baseline

- Verify BaDOS learning status

Attention: In this lab, you will configure a number of options to get the lab started. In modules 3 and 4 we will spend time examining the configuration options in more detail. For now, just configure the options as outlined, and we will examine further in later modules.

1.2.1 Set up the DoS profile

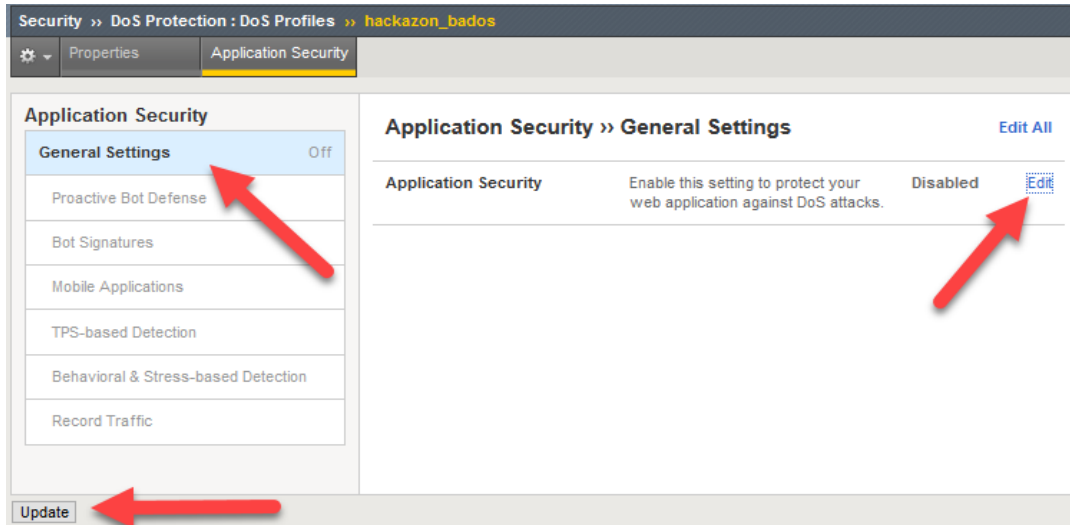
In the section you will create a DoS profile with **Behavioral Detection and Analysis** enabled, and attach the DoS profile to the virtual server.

1. Using Chromium Browser on the Xubuntu Jumpbox, open a tab to the GUI on bigip01 (<https://10.1.1.245>).
2. Navigate to **Security » DoS Protection : DoS Profiles**
3. Select **Create**. Name your profile **hackazon_bados** and select **Finished**. Open your **hackazon_bados** DoS profile.
4. Select the **Application Security** tab from DoS Profile navigation bar.

The screenshot displays the BIG-IP configuration interface for the 'hackazon_bados' DoS profile. The breadcrumb navigation at the top reads 'Security » DoS Protection : DoS Profiles » hackazon_bados'. Below this, there are two tabs: 'Properties' and 'Application Security'. A red arrow points to the 'Application Security' tab. The 'Properties' section contains the following fields: 'Name' (hackazon_bados), 'Partition' (Common), 'Description' (empty), and 'Threshold Sensitivity' (Medium with a warning icon). The 'Whitelists' section includes 'Default Whitelist' (Address List) and 'HTTP Whitelist' (Use Default). At the bottom of the whitelists section are 'Update' and 'Delete' buttons. On the right side, there is a 'Shared Objects' panel with a 'Filter Shared Objects' input field and a section for 'Address Lists (0)' with a plus icon to add more.

5. Click **General Settings**, select **Edit** to the right of **Application Security** in the rightmost panel, and check the **Enable** box.

This will activate the other sections of the DoS profile.



Tip: At any point you can save your changes by hitting the **Update** button in the lower left-hand corner

6. Select the **Bot Signatures** section, then select the **Edit** link to the right of **Bot Signature Check**, and check the **Enabled** box.

Select **Edit** next to **Bot Signature Categories** then change both the **Malicious Categories** and **Benign Categories** to **Report**. This step is necessary because the tools used to generate baseline and attack traffic in this lab will both be categorized as bots.

Security » DoS Protection : DoS Profiles » **hackazon_bados**

⚙ Properties Application Security

Application Security

- General Settings ✓
- Proactive Bot Defense Off
- Bot Signatures** ✓
- Mobile Applications Off
- TPS-based Detection ✓
- Behavioral & Stress-based Detection Off
- Record Traffic Off

Application Security » Bot Signatures Edit All

This feature automatically detects well known bots according to their HTTP characteristics. Malicious bots can be configured to be blocked, while benign bots can be configured to pass through the anti-bot defense mechanisms.

Bot Signature Check	When enabled, bot signatures are checked. This allows well-known bots to be detected.	<input checked="" type="checkbox"/> Enabled Close
Bot Signature Categories	Specifies the action for each bot signature category.	<p>Malicious Categories: Report Close</p> <p>/Common</p> <p>DOS Tool Report</p> <p>E-Mail Collector Report</p> <p>Exploit Tool Report</p> <p>Network Scanner Report</p> <p>Spam Bot Report</p> <p>Spyware Report</p> <p>Vulnerability Scanner Report</p> <p>Web Spider Report</p> <p>Webserver Stress Tool Report</p> <p>Benign Categories: Report</p> <p>/Common</p>

This feature will not be fully operational since the DNS Resolver List is empty. To add one, navigate to Network » DNS Resolvers : DNS Resolver List.

Attention: The message in red below the **Enabled** box indicates a DNS Resolver has not been set up. The DNS resolver is used to perform DNS reverse lookups as part of bot identity validation, but is not relevant for this lab exercise.

7. Select **TPS-base DoS Detection** and change **Operation Mode** to **Off**.

Application Security

- General Settings ✓
- Proactive Bot Defense Off
- Bot Signatures ✓
- Mobile Applications Off
- TPS-based Detection** Off
- Behavioral & Stress-based Detection Off

Application Security » TPS-based DoS Detection Edit All

This section configures the detection of DoS attacks based on high volume of incoming traffic.

Operation Mode	Specifies how the system reacts when it detects an attack.	Off Close
Thresholds Mode	Specifies what type of thresholds to use.	Manual Edit

8. Select **Behavioral & Stress-based Detection** and change **Operation Mode** to **Blocking**.
 - (a) Set the **Thresholds Mode** to **Automatic**.
 - (b) Under **Stress-based Detection and Mitigation** edit **By SourceIP** and uncheck **Request Blocking**. Under **By URL** uncheck **Heavy URL Protection** and **Request Blocking**.
 - (c) Under **Behavioral Detection and Mitigation** check the **Request signatures detection** and set the **Mitigation** to **Standard**. For now, please leave **bad actors detection** unchecked.
 - (d) Click **Update** in the lower left-hand corner. Collapse all the sections, and **Behavioral & Stress-based Detection** should match the figure below.

Application Security » Behavioral & Stress-based (D)DoS Detection Edit All			
This section configures the detection of DoS attacks based on server stress. The system automatically detects an increase in server stress and mitigates DoS attacks causing it.			
Operation Mode	Specifies how the system reacts when it detects an attack.	Blocking	Edit
Thresholds Mode	Specifies what type of thresholds to use.	Automatic	Edit
Stress-based Detection and Mitigation	By Source IP	No mitigation	Edit
	By Device ID	No mitigation	Edit
	By Geolocation	No mitigation	Edit
	By URL	No mitigation	Edit
	Site Wide	No mitigation	Edit
Behavioral Detection and Mitigation	By Bad Actors Behavior / Signatures	Bad Actors & Signatures: Standard protection *	Edit
Prevention Duration	Specifies the time spent in each mitigation step until it is stopped, and the next one is started.	Escalation Period: 120 seconds De-escalation Period: 7200 seconds	Edit

1.2.2 Create a DoS Logging Profile

Logging profiles are required to enable local and remote logging for Application DoS and Bot events. In this lab, we will use local logging to review events. Below are the steps to configure the logging profile and attach to your test virtual server.

1. Go to **Security >> Event Logs : Logging Profiles** and click **Create** on right-hand side of the configuration screen. Name your profile **i7_dos_bot_logger** then check the **DoS Protection** and **Bot Defense** enable boxes.
2. From the **DoS Protection** tab enable the **Local Publisher**.
3. From the **Bot Defense** tab check ALL the boxes.
4. Click **Finished**.

Security >> Event Logs : Logging Profiles >> Create New Logging Profile...

Logging Profile Properties Cancel Finished

Profile Name	i7_dos_bot_logger
Description	
Application Security	<input type="checkbox"/> Enabled
Protocol Security	<input type="checkbox"/> Enabled
Network Firewall	<input type="checkbox"/> Enabled
DoS Protection	<input checked="" type="checkbox"/> Enabled
Bot Defense	<input checked="" type="checkbox"/> Enabled

DoS Protection **Bot Defense**

Request Log

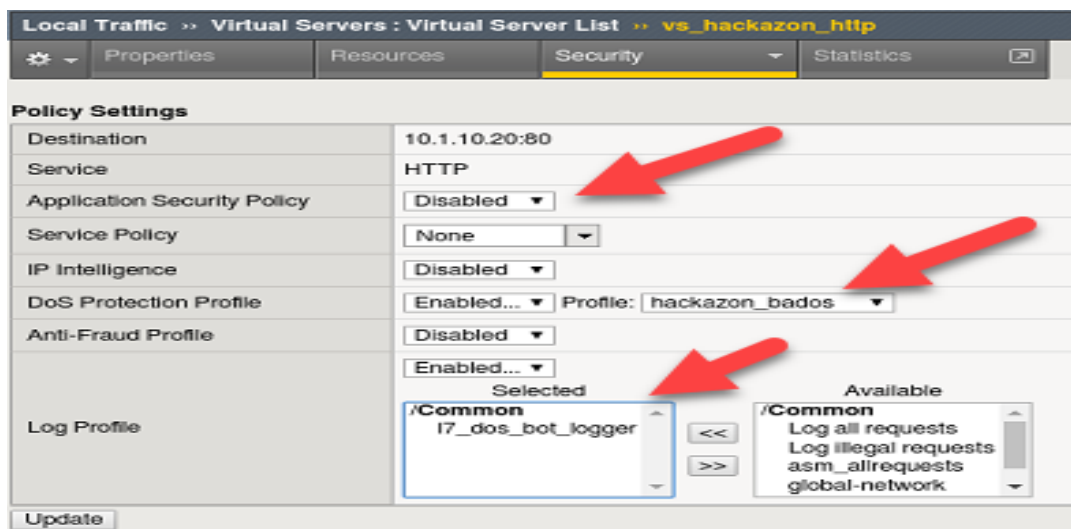
Local Publisher	<input checked="" type="checkbox"/> Enabled
Remote Publisher	none
Log Illegal Requests	<input checked="" type="checkbox"/> Enabled
Log Captcha Challenged Requests	<input checked="" type="checkbox"/> Enabled
Log Challenged Requests	<input checked="" type="checkbox"/> Enabled
Log Bot Signature Matched Requests	<input checked="" type="checkbox"/> Enabled
Log Legal Requests	<input checked="" type="checkbox"/> Enabled

Cancel Finished

1.2.3 Add the DoS profile to a virtual server

Below are the steps to associate this profile with the Local Traffic Manager virtual server processing the application traffic in this lab.

1. Navigate to **Local Traffic > Virtual Servers > Virtual Server List** and select **vs_hackazon_http**. Under the **Security** tab on the top bar select **Policies**.
2. Enable the **DoS Protection Profile** and select the **hackazon_bados** profile.
3. Add **i7_dos_bot_logger** to the **Log Profile** and **Update**
4. For purposes of this lab, **Disable** the **Application Security Policy** and remove **asm_allrequests** from the **Log Profile**.



1.2.4 Create XFF-Mixed_Attacker iRule

Because we do not have dozens of good and bad source IPs available for clients and attackers in this environment, we simulate them by adding an iRule to the virtual server. The iRule adds a randomized X-Forwarded-For (XFF) header to each request.

1. Navigate to **Local Traffic >> iRules : iRule List** and select **Create**. Name a new iRule named **XFF_mixed_Attacker_Good_iRule**. Copy and paste the iRule below.

```

1  when HTTP_REQUEST {
2      # Good traffic
3      if { [IP::addr [IP::client_addr] equals 10.1.10.52] } {
4          set xff 153.172.223.[expr int(rand()*100)]
5          HTTP::header insert X-Forwarded-For $xff
6      }
7
8      # Attack traffic
9      if { [IP::addr [IP::client_addr] equals 10.1.10.53] } {
10         set xff 132.173.99.[expr int(rand()*25)]
11         HTTP::header insert X-Forwarded-For $xff
12     }
13 }

```

Advanced Web Application Firewall/Application Security Manager will honor the X-Forwarded-For header by enabling this in the http profile.

1.2.5 Create HTTP Profile to Accept X-Forwarded-For HTTP Header

1. Navigate to **Local Traffic >> Profiles : Services : HTTP** and click **Create**. Name the new http profile **xff_http**, and click the rightmost checkbox in the row **Accept XFF** to enable a custom setting, then click the checkbox to the immediate right of **Accept XFF** to enable processing of an inbound X-Forwarded-For header.
2. Click **Finished** button at bottom of configuration page.

Tip: Due to a large number of service profiles, occasionally part of the Services menu will get stuck under the browser menu. If that happens, click on **Profiles** on the side-bar, then click **Services** in the top navigation bar to get to

the HTTP profile.

1.2.6 Attach iRule and HTTP Profile to Local Traffic Manager Virtual Server

1. Navigate to the **vs_hackazon_http** virtual server. In the **Properties** tab, under **Configuration** section, select **xff_http** for the **HTTP Profile**.
2. Click the **Resources** tab in the virtual server navigation bar, in the **iRules** section select the **Manage** button, and move the **XFF_mixed_Attacker_Good_iRule** from the **Available** to the **Enabled** box.
3. Click **Finished** button at bottom of the Resource Management page.

1.2.7 Generate Traffic to Establish Baseline

Advanced Web Application Firewall's Behavioral DoS feature is based on learning and analyzing all traffic to the web application, building baselines, and then identifying anomalies when server stress is detected. As a result, in this lab, we need to generate normal traffic allowing Advanced Web Application Firewall to build a baseline.

You will use the Xubuntu Jumpbox to generate legitimate traffic and bad traffic, eth1 has 10.1.10.51-55 configured and 10.1.10.52 will be the source-IP used for the good traffic script. The source IP will match XFF_mixed_Attacker_Good_iRule created above, and an X-Forwarded-For header will be placed in the HTTP request in the 153.172.223.0/24 IP address range.

In the home directory (/home/f5student) on the Xubuntu Jumpbox, you will find the two scripts used for this lab:

- **baseline_menu.sh** - is used to create baseline traffic
- **AB_DOS.sh** - is used to launch L7 DOS attacks

1. Start baseline traffic, using Xubuntu Jumpbox Terminal application, navigate to the home directory, then type:

```
f5student@xjumpbox~$ ./baseline_menu.sh  
- Select option 2 **alternate** and keep it running in the window
```

Tip: This is your valid traffic, and the number of requests will change over time. The requests also change as the script continuously alters the User-Agent header and the requested URI. Both values are randomly taken from files in the "source" directory in the home directory.

2. Next, validate you are seeing the traffic, and Advanced Web Application Firewall is actively building learning baselines. From a separate Terminal window type:

```
f5student@xjumpbox$~ ssh root@10.1.1.245
```

Then, run the following command:

```
[root@bigipo01:Active:Standalone] config # admd -s vs./Common/vs_hackazon_  
↳http+/Common/hackazon_bados.info.learning  
  
- /Common/vs_hackazon_http - is the name of the virtual server  
- /Common/hackazon_bados - is the name of the DoS profile.  
**It may take several minutes for baseline numbers to be generated**
```


Screenshot of sample output below:



Tip: If you aren't getting any output, or seeing no signs of accumulated signals, verify the name of the virtual server and profile in the `admd` command are accurate.

1. **baseline_learning_confidence:**

- **Description:** in % how confident the system is in the baseline learning.
- **Desired Value:** > 90%

2. **learned_bins_count:**

- **Description:** number of learned bins
- **Desired Value:** > 0

3. **good_table_size:**

- **Description:** number of learned requests
- **Desired Value:** > 2000

4. **good_table_confidence:**

- **Description:** how confident, as %, the system is in the good table
- **Desired Value:** Must be 100 for signatures

Note: It may take 5 or more minutes before you begin to get learned baseline numbers. Also, the desired values are the minimum values we would like to see prior to triggering attacks as part of this lab exercise. You can, however, move onto module 3 and 4 in this lab while baselines are being established. **Do not stop baseline traffic script**

To see all of the values available and wide range of interesting statistics, enter the following command from BIG-IP console:

```
admd -s vs./Common/vs_hackazon_http
```

To view Advanced Web Application Firewall layer 7 DoS log, enter the following command from BIG-IP console:

```
tail -f /var/log/dosl7/dosl7d.log
```

Note: The goal of this module is to explain DoS profile configuration options. The module does not contain any exercises. If you are already familiar with the settings in an Application Security DoS profile you can skip to module 4.

1.3 Application Security DoS Profiles

In this module, we will review the various settings and options that make up a layer 7 DoS profile. We will not review each and every setting, leaving that exercise up to the reader, but instead will focus on key settings which will most likely require attention during a production deployment. More detail on each individual setting can be found by viewing the **Help** on left side of the BIG-IP Configuration Utility (GUI).

1.3.1 Review DoS Profile General Settings

Navigate to **Security >> DoS Protection >> DoS Profiles** and click the DoS profile **hackazon_bados** created earlier for this module.

Settings in this screen are profile wide, and can affect all aspects of the dos configuration.

Application Security

- General Settings** ✓
- Proactive Bot Defense During Attacks ✓
- Bot Signatures ✓
- Mobile Applications Off
- TPS-based Detection ✓
- Behavioral & Stress-based Detection Off
- Record Traffic Off

Application Security » General Settings [Edit All](#)

Application Security	Enable this setting to protect your web application against DoS attacks.	Enabled	Edit
Heavy URL Protection	Configure Heavy URL include list, automatic detection, and exclude list	Automatic Detection: Enabled (Threshold: 1000 ms) Heavy URLs: Not configured Ignored URLs: Not configured	Edit
Geolocations	Overrides the DoS profile's Geolocation Detection Criteria threshold settings by selecting countries from which to allow or block traffic during a DoS attack.	Not configured	Edit
CAPTCHA Response	Customize the CAPTCHA page users see during DoS events.	First Response Type: Default Failure Response Type: Default	Edit
Trigger iRule	Enable this setting if you have an iRule that manages DoS events in a customized manner.	Enabled	Edit
Single Page Application	Enable this setting if your website is a Single Page Application.	Enabled	Edit
URL Patterns Example: /product/*.php	Configure URL patterns to be used. Each URL pattern defines a set of URLs which are logically the same URL with the varying part of the pattern acting as a parameter.	Not configured	Edit
Traffic Scrubbing	Specifies whether to enable Traffic Scrubbing during attacks by advertising BGP routes. This requires configuration of the Scrubber Profile, and will function even when the Operation Mode is set to Transparent.	Disabled	Edit
RTBH	Specifies whether to enable Remote Triggered Black Hole (RTBH) of attacking IPs by advertising BGP routes. This requires configuration of the Blacklist Publisher, and will function even when the Operation Mode is set to Transparent.	Disabled	Edit
Performance acceleration	Configure TCP fastL4 profile to be used as fast-path for acceleration	Disabled	Edit

1. Application Security

This setting enables or disables the DoS profile.

2. Heavy URL Protection

Heavy URL's are application resources which may consume more backend resources with each client request. Additionally, URLs which are not generally considered heavy may become heavy under significant load or attack. As a result, low rate requests targeting these URLs can cause significant DoS attacks, and be difficult to differentiate from legitimate requirements based on rate alone. Advanced Web Application Firewall automatically detects heavy URLs by measuring the latency tail ratio, which is the number of transactions whose latency is consistently greater than the latency threshold defined in this configuration option. A URL is considered heavy if its latency is more than two times the site global average over a 24 hour (default) period.

Heavy URL Protection Close

Configure Heavy URL include list, automatic detection, and exclude list. ☒ A URL is considered heavy if its portion of transactions with latency above this threshold is higher than usual for this site. ms

1.

Configure a list of Heavy URLs to protect, in addition to the automatically detected ones.

URL (Example: /query.html)

Threshold (requests per sec)

Add

2.

Delete

3.

Configure a list of URLs which are excluded from being automatically detected as Heavy URLs (Wildcard supported).

URL (Example: /product/*)

Add

Delete

1. Checkbox, enables or disables, automatic detection of the heavy URLs profile-wide. The text box allows for configuration of the baseline threshold that URLs must exceed before being considered for heavy URL determination.
2. This section of the DoS Profile Heavy URL configuration allows an administrator to explicitly configure a URL(s) as heavy, whether it is detected as heavy by Advanced Web Application Firewall or not. Use this section to define application resources which are known to be heavier in terms of resource consumption, or known to be less resilient to higher volumes of traffic than the rest of the application.
3. This section of the Dos Profile Heavy URL configuration allows an administrator to explicitly configure URL(s) and wildcard URL patterns to be excluded from automatic heavy URL detection. Use this section, to identify URL's which you know may perform slower than average under normal conditions, or URLs you do not wish to have Advanced Web Application Firewall offering heavy URL protection.

Note: To provide mitigation for heavy URLs, you must enable at least one of the URL-based prevention policy methods in the TPS or Stress-based Anomaly sections of the DoS profile.

3. **Geolocations** Geolocations provides options to override the dos profile geolocation detection criteria by explicitly whitelisting or blacklisting specific geolocations.

Geolocations Close

Overrides the DoS profile's Geolocation Detection Criteria threshold settings by selecting countries from which to allow or block traffic during a DoS attack.

Geolocation Blacklist:

Available Geolocations:

- Alghanistan
- Aland Islands
- Albania
- Algeria
- American Samoa
- Andorra
- Angola
- Anguilla
- Antarctica
- Antigua and Barbuda

Geolocation Whitelist:

4. **Single Page Application** Single Page Applications (SPA) represent a change in application architecture that moves much of the content rendering and routing to client-side code. Application requests which require server-side processing are sent as AJAX requests towards server, and the response is typically JSON/XML; this is different from traditional web applications that send HTTP requests, and generally leverage HTML as the predominant response content type. As a result, Advanced Web Application Firewall needs to modify the way it challenges clients for features like Proactive Bot Defense and capturing Device ID in the TPS/Stress based anomaly detections. Enabling this option modifies Advanced Web Application Firewall's challenge and challenge validation mechanisms. When deploying L7 DoS protections it is important to understand the application architecture, and if protecting a SPA, enabling this option is critical for proper operation.

Note: The goal of this module is to explain the Stress-Based and Behavioral DoS configuration options. The module does not contain any exercises. If you are already familiar with the settings you can skip to module 5.

1.4 Stress-Based and Behavioral DoS Profile Settings

In this module, we will review the various settings for configuring Stress-based and Behavioral DoS protections in more detail. We will not review each and every setting, leaving that exercise up to the reader, but instead will focus on key settings which will most likely require attention during a production deployment. More detail on each individual setting can be found by viewing the **Help** on left side of the BIG-IP Configuration Utility (GUI).

1.4.1 Review Stress-Based Dos Profile Settings

To appreciate the powerful nature of Advanced Web Application Firewall's Behavioral DoS feature, it first makes sense to analyze one of the other L7 DoS protection mechanisms. For this exercise, we will examine the options and behaviors of the Stress-based DoS protections available in an Application Security DoS profile.

To review the settings below, navigate to **Security >> DoS Protection >> DoS Profiles**, click the DoS profile **hack-azon_bados** created earlier for this module, then click **Behavioral & Stress-based Detection** in the **Application Security** navigation menu, and set the **Operation Mode** to **Transparent**.

Application Security » Behavioral & Stress-based (D)DoS Detection

[Edit All](#)

This section configures the detection of DoS attacks based on server stress.
The system automatically detects an increase in server stress and mitigates DoS attacks causing it.

Operation Mode	Specifies how the system reacts when it detects an attack.	Blocking	1	Close
Thresholds Mode	Specifies what type of thresholds to use.	Manual	2	Edit
Stress-based Detection and Mitigation	By Source IP	Consider an IP as an attacking entity if either of the following conditions occur: Relative Threshold: TPS increased by: 500 % and reached at least 40 transactions per second OR Absolute Threshold: TPS reached: 200 transactions per second Set default criteria Select mitigation methods to use on the attacking IP's: <input type="checkbox"/> Client Side Integrity Defense <input type="checkbox"/> CAPTCHA Challenge <input checked="" type="checkbox"/> Request Blocking / Rate Limit	3 4 5	Close
	By Device ID	No mitigation		Edit
	By Geolocation	No mitigation		Edit
	By URL	Mitigation: Request Blocking (Rate Limit)		Edit
	Site Wide	No mitigation		Edit
Behavioral Detection and Mitigation	By Bad Actors Behavior / Signatures	Disabled		Edit
Prevention Duration	Specifies the time spent in each mitigation step until it is stopped, and the next one is started.	Escalation Period: 120 seconds De-escalation Period: 7200 seconds	6	Edit

- Operation Mode** Defines the operational mode for the stress-based dos protection feature. Available options include: Blocking, Transparent, Off. Blocking means feature will detect, report, and mitigate. Transparent means feature will detect, report, but will not mitigate. Off means the feature is disabled.
- Threshold Mode** Defines how Advanced Web Application Firewall derives thresholds to be used in detecting the TPS component of a stress-based attack. Options include:
 - Manual:** Administrator explicitly configures TPS and percentage thresholds based on their knowledge of the environment or specific requirements.
 - Automatic:** Advanced Web Application Firewall monitors traffic rates automatically and calculates the thresholds based on normal traffic volume to the application.
- Stress-based Detection Options** Advanced Web Application Firewall can trigger an attack if any/all of the following detection methods exceed the thresholds defined or calculated for the detection method:
 - By Source IP:** A specific source IP has exceeded the thresholds defined in the detection thresholds.
 - By Device ID:** A specific device has exceeded the thresholds defined in the detection thresholds. Device ID is ASM calculating a fingerprint for a given device. The feature requires Javascript injection for proper operation. However, the feature offers the benefit of detecting a specific device, even if the attack varies its source IP address.

- **By Geolocation:** A country/geolocation has exceeded the thresholds defined in the detection thresholds.
- **By URL:** Request traffic to a specific (or set of URL's identified in URL patterns section of the DoS Profile General Properties) has exceeded the thresholds defined in the detection thresholds.
- **Site Wide:** Request traffic to the entire web site has exceeded the thresholds defined in the detection thresholds, **and** an attack has not been detected using any of the other detection criteria. Site-wide is considered last resort.

Note: It is important to understand that while stress-based protections are monitoring server latency, and tracking application request volume in short and long term intervals, the detection methods listed above are the only ways to identify when an attack is on-going. This, as you will see, is quite a bit different than they way Advanced Web Application Firewall Behavioral DoS feature identifies attacks and attackers!

1.4.2 Review Behavioral DoS Settings

Having reviewed the options for configuring Stress-based dos mitigation, now let's examine the options required for configuring Advanced Web Application Firewall's Behavioral DOS mitigations.

Behavioral & Stress-based Detection	
Record Traffic	Off

Stress-based Detection and Mitigation	
By Source IP	Mitigation: Request Blocking (Rate Limit) Edit
By Device ID	No mitigation Edit
By Geolocation	No mitigation Edit
By URL	Mitigation: Request Blocking (Rate Limit) Edit
Site Wide	No mitigation Edit

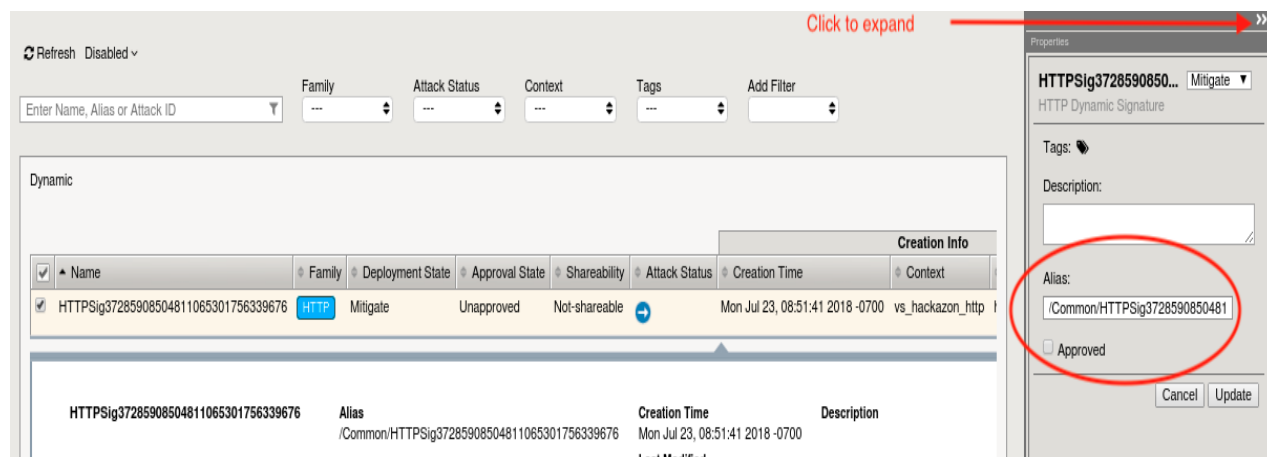
Behavioral Detection and Mitigation	
By Bad Actors Behavior / Signatures	<input checked="" type="checkbox"/> Bad actors behavior detection Close <small>Enables traffic behavior, server's capacity learning, and anomaly detection.</small>
	<input checked="" type="checkbox"/> Request signatures detection <small>Enables signatures detection</small>
	<input checked="" type="checkbox"/> Use approved signatures only
	Mitigation <div>No mitigation</div> <small>Learns and monitors traffic behavior, but no action is taken.</small>

1. **Bad Actors Behavior Detection** Determines whether Behavioral DoS engine tracks and attempts to identify the bad actors contributing to a given set of malicious traffic. When Bad Actor Behavior Detection is enabled, once Advanced Web Application Firewall detects server stress and identifies a set of malicious traffic contributing to the server stress, the Behavioral DoS engine then attempts to identify what source IP addresses are generating the malicious traffic, and what percentage of malicious traffic a given bad actor is contributing. Bad actors, are mitigated at transport layer via slowdown mitigation techniques, and the rate at which they are mitigated is directly related to their percentage of contribution to the malicious traffic set, and the mitigation mode selected.
2. **Request Signature Detection** Determines whether Behavioral DoS engine will attempt to generate a traffic signature to block anomalous traffic. Advanced Web Application Firewall Behavioral DoS feature is in a permanent learning state, always tracking application requests, and the construction of these requests, and then comparing to an evolving baseline. When Request Signatures Detection is enabled, once Advanced Web Application Firewall detects server stress, it looks to identify traffic characteristics which have deviated from the baseline. If

there are deviating characteristics, the Behavioral DoS engine, then dynamically generates a signature based on these deviating characteristics to block anomalous traffic.

Note: In addition to generating signatures the Behavioral DoS Engine also continually evaluates the signature for efficacy, minimizing the risk of signature becoming false positive and blocking known good traffic.

3. **Use Approved Signatures Only** By default, when Request Signatures Detection is enabled, Advanced Web Application Firewall will generate and use dynamically generated attack signatures as defined by the mitigation mode selection. By enabling this option, the administrator overrides this behavior, and forces a manual step to review and approve the signature prior to any mitigations taking effect. Signatures can be reviewed from Advanced Web Application Firewall GUI via **Security -> DoS Protection -> Signatures**.



Once a signature has been approved, the Signature Approval State for the signature will change to “Manually-approved”. When approved signatures only is selected, only signatures which have been approved will be active.

4. **Mitigation** Defines the mitigation mode for Advanced Web Application Firewall Behavioral DoS. Options include:
 - **No Mitigation:**
 - Monitors traffic, generates signatures, and identifies bad actors, but does not perform any mitigation.
 - **Conservative Protection:**
 - If Bad Actors Behavior Detection is enabled, slows down bad identified bad actors.
 - If Request Signatures Detection is enabled, blocks requests that match attack signatures
 - **Standard Protection:**
 - If Bad Actors Behavior Detection is enabled, slows down bad identified bad actors.
 - If Request Signatures Detection is enabled, blocks requests that match attack signatures
 - Rate limits all requests based on server health
 - Limits the number of concurrent connections from bad actor IP addresses
 - If necessary, limits the number of all concurrent connections based on server health
 - **Aggressive Protection:**
 - If Bad Actors Behavior Detection is enabled, slows down bad identified bad actors.

- If Request Signatures Detection is enabled, blocks requests that match attack signatures
- Rate limits all requests based on server health
- Limits the number of concurrent connections from bad actor IP addresses
- If necessary, limits the number of all concurrent connections based on server health
- **Proactively** performs all protection actions, even before attack detection, increasing impact of protection techniques.

Advanced Web Application Firewall mitigates DoS with the most effective and efficient method available, and as quickly as possible to restore server health. Meaning, the mitigation method will often change over time as more data is learned and analyzed. For example, at the onset of an attack, Advanced Web Application Firewall may apply global rate limiting in an attempt to mitigate an onslaught of traffic. Then, as the signature engine has observed enough traffic to identify malicious traffic and generate a signature, the Behavioral DoS engine will begin mitigating with request signatures and discontinue global rate limiting. Finally, as bad actors are identified, traffic from those sources is mitigated using layer four slowdown mechanisms, and request signatures are only used for traffic matching the signature and not in the bad actor list. This approach allows Advanced Web Application Firewall to perform better under attack, and minimizes the risk of blocking good traffic while mitigating DoS.

1.4.3 Summarizing Key Points

After reviewing several options for both Stress-based and Behavioral DoS features, the goal of this section is to call out some key points which might be overlooked when reviewing configuration options:

- All DoS features are complementary to Advanced Web Application Firewall web application firewall (WAF) and bot protection features. DoS features mitigate traffic that exceeds a certain rate or induces server-side stress. This traffic is, many times, completely legitimate traffic which will not trigger a WAF block.
- Heavy URL, TPS-based DoS, Stress-based DoS, and Behavioral DoS features can all be configured concurrently, complementing one another, or separate and independent of one another.
- Both Stress-based and Behavioral DoS protection features continually monitor application server performance for signs of server stress. Both features will consider server stress as a key component in detecting an attack, and neither will trigger a mitigation if the server is perceived to be healthy.
- Stress-based and TPS based DoS features can detect DoS attacks across a pre-defined set of detection criteria (source IP, URL, device ID, geolocation, site). Behavioral DoS is not constrained to a pre-defined set of detection criteria, but instead is a self adjusting dynamic DoS defense system which can detect DoS across hundreds of traffic predicates. As a result, Behavioral DoS, is much more effective in mitigating multi-vector layer seven DoS attacks which mutate over time. Conversely, TPS and/or Stress-based DoS features are much better at defining specific rate limits for traffic entering your application.

1.5 Request Signatures

In this module you will be initiating a L7 DDoS attack on the hackazon virtual server, from eth1, using 10.1.10.53 as the source IP address. This source IP will match XFF_mixed_Attacker_Good_iRule, and an X-Forward-For header will be inserted in the HTTP request in the 132.173.99.0/24 IP address range.

Once the attack begins the BIG-IP WAF (ASM) will immediately switch into attack mode due to the server health deteriorating almost immediately. As the server gets totally overwhelmed, you may at first notice the good script dropping requests. *That's why BaDoS first mitigates with a global rate limit just to protect the server.* In a short time, the good script will go back to all 200 OK responses. During this time Behavioral DoS identifies anomalous traffic

and generates **Dynamic Signatures** matching only the malicious traffic. Once mitigation is in effect, the server health will rapidly improve and application performance will return to normal.

1. Using Chromium Browser on the Xubuntu Jumpbox, open tab to the GUI on bigip01 (<https://10.1.1.245>)
2. Navigate to **Security » DoS Protection:Signatures** and click on the **Dynamic** box, then set the **Refresh** value to **20 secs**.
3. Open another tab/window in Chromium Browser, and go to **Security »Reporting : DoS : Dashboard**. The dashboard is NOT real time in may take up to 10 minutes for traffic to display.
4. Revisit the Terminal window you opened earlier which is monitoring behavioral DoS learning signals. Verify the first number (baseline_learning_confidence) is at or above 80%. Normally, above 90% would be ideal, but for the purposes of this lab over 80% will suffice.
5. Revisit the Terminal window you opened earlier which is still running the baseline traffic generation script. Make note of the normal, pre-attack, response time for each request.
6. From Xubuntu Jumpbox open a NEW Terminal window. From your home directory enter:

```
f5student@xjumpbox~$ ./AB_DOS.sh
- Select **2** - Attack start - score
```

7. Using Chromium Browser on the Xubuntu Jumpbox, open another tab to the GUI on bigip01, and navigate to **Security » Event Logs » DoS » Application Events**
8. Almost immediately you should see an attack has started, and Advanced Web Application Firewall has assigned an Attack ID to the event. You will see something similar to the screenshot below:

Time	Virtual Server	Profile Name	Event	Detection Mode	Mitigation	TPS	Detection Threshold	Mitigate To Threshold	Threshold Condition	Attack ID	Entity Type	Entity	De
2018-05-31 06:37:54	/Common/vs_hackazon_http	/Common/hackazon_bados	Attack started	Behavioral detection	Behavioral	6 tps				1783458958			0

9. Review the **Dyanmic Signatures** UI page opened in step #2. It might take a few moments for a dynamic signature(s) to generate, but shortly after the attack has been detected a signature should be created. Once a signature(s) is generated, if you click on the signature (NOT on the blue link, but somewhere on the signature bar), you will get the details about the signature in Wireshark format. Also, you can examine the current status of the signature (mitigating or not), and statistics on recent attacks which used the signature.

Security >> DoS Protection : Signatures

Refresh Disabled

Enter Name, Alias or Attack ID

Family: --- Attack Status: --- Context: --- Tags: --- Add Filter: ---

Dynamic

Name	Family	Deployment State	Approval State	Shareability	Attack Status	Creation Time	Context	Profile
TPSig37285908504811065301783458959	HTTP	Mitigate	Unapproved	Not-shareable	↓	Thu May 31, 07:11:13 2018 -0700	vs_hackazon_http	hackazon_bados

Creation Info

Creation Time: Thu May 31, 07:11:13 2018 -0700

Last Modified: Thu May 31, 07:11:13 2018 -0700

Predicates String

http.x_forwarded_for_header_exists eq true) and (http.accept_encoding_header_exists eq true) and (http.pragma_header_exists eq true) and (http.host_header_exists eq true) and (http.unknown_header_exists eq true) and (http.uri_len between 0-15) and (http.accept_header_exists eq true) and (http.uri_parameters eq no-query) and (http.headers_count eq 11) and (http.referer_header_exists eq true) and (http.cache_control hashes-to 14) and (http.cache_control_header_exists eq true) and (http.referer hashes-like http://10.0.2.1/none.html) and (http.user_agent hashes-like http://10.0.2.1/none.html)

Most Recent Attacks

Attack ID	Context	Profile	Attack Time	Accuracy	Detection Threshold EPS
1783458959	vs_hackazon_http	hackazon_bados	Thu May 31, 07:12:41 2018 -0700	100%	0

- **Signature ID:** Signature ID generated for this signature. You can use the signature ID in DoS Analysis/Dashboard views (explored in module 6) to get more details on actions taken by this signature.
- **Deployment State:** current state of the signature. Options include:
 - **Mitigate** - Collect stats, learn, alert, and mitigate. All thresholds and threshold actions are applied, and rate limiting occurs if the device is under high stress.
 - **Detect Only** - Collects stats, learn, and alert. Develops dynamic signatures without enforcing any thresholds or limits.
 - **Learn Only** - Collect stats and learn. Develops dynamic signatures without enforcing any thresholds or limits
 - **Disabled** - No stat collection or mitigation, totally disables the signature.
- **Attack Status** - the state of the signature with respect to ongoing attacks. Specifically, defines whether this particular signature is being used to mitigate an on-going attack.
- **Attack ID** - the attack ID for the attack that generated this signature. Clicking the attack ID will take you to the DoS Analysis views filtered on this attack ID.
- **Predicates List** - the conditions for the request to be associated with this signature. Includes one or more match expressions, joined by logical operators, which the system uses to match traffic causing a DoS attack.
- **Attack History** - provides an account of all attacks in which this signature has been used to mitigate.

Note: Dynamic Attack signatures generated will remain in the list up to the max number of signatures supported, and will be re-used whenever an attack is detected, and traffic matches the conditions

defined in the signature

10. With the attack script still running, examine the output of the baseline script. You should be getting HTTP 200 OK responses, and the response time should be inline with pre-attack response times. Also, verify you can use browse to <http://hackazon.f5demo.com> without issue.
11. In the window where you are running the attack script, enter **CTRL-C**, then type **4** to kill the attack script cleanly.
12. Using Chromium Browser, navigate to **Security » DoS Protection:Signatures** and click on the **Dynamic** box. Then click the check box next to the Name column to select all signatures, and click delete to remove all attack signatures created during this module.
13. Leave **baseline_menu.sh** script running.

1.6 Bad Actor Detection

In the last module, you used request signature detection to mitigate an application layer DoS attack. You also saw the Behavioral DoS engine deploy global rate limiting to bring the servers back to health while signatures were being generated, then mitigate targeted attack traffic with the newly generated signature. In this module, we will leverage Bad Actor Detection to throttle known bad actors.

1. Navigate to **Security » DoS Protection : DoS Profiles** and click the **hackazon_bados** profile we created earlier.
2. Click the **Application Security** tab, and then click **Behavioral & Stress-based Detection** button in the Application Security panel.
3. Click the **Edit** link to the right of the **Behavioral Detection and Mitigation** section, then check the checkbox for **Bad actors behavior detection**, and uncheck the box next to **request signatures detection**
4. Scroll down, and click the **Update** button.
5. From the Xubuntu Jumpbox open another Terminal window. Then:

```
f5student@xjumpbox$~ ssh root@10.1.1.245
```

6. From the SSH session, run the following command:

```
[root@bigipo01:Active:Standalone] config # watch ipidr -l /Common/vs_
↪hackazon_http+/Common/hackazon_bados
```

Initially, because no attack is active, the IP list will be empty. Keep this command running in one of the Terminal windows. Things are about to change!

7. Using the Terminal window on the Xubuntu Jumpbox from the previous module, or a new one, re-run the attack script using the following command:

```
f5student@xjumpbox~$ ./AB_DOS.sh

- Select **2** - Attack start - score
```

8. Using Chromium Browser on the Xubuntu Jumpbox, open another tab to the GUI on bigip01, and navigate to **Security » Event Logs » DoS » Application Events**
9. Almost immediately you should see an attack has started, and Advanced Web Application Firewall has assigned an Attack ID to the event. You will see something similar to the screenshot below:

Security » Event Logs : DoS : Application Events

Application

Protocol

Network

DoS

Bot Defense

Logging Profiles

Last Hour

Search

Custom Search...

Time	Virtual Server	Profile Name	Event	Detection Mode	Mitigation	TPS	Detection Threshold	Mitigate To Threshold	Threshold Condition	Attack ID	Entity Type	Entity	D
2018-05-31 06:37:54	/Common/vs_hackazon_http	/Common/hackazon_bados	Attack started	Behavioral detection	Behavioral	6 tps				1783458958		0	

10. From the Terminal window started in step #6, monitor the output of the `ipidr` command, and the status of the IP greylist. You should see something similar to the image below:

```
[root@bigip01:Active:Standalone] config # ipidr -l /Common/vs_hackazon_http+/Common/hackazon_bados
Trying to allocate key 1514 sz 82968 flags 186
Mapped memory @ 0x2b819a8ac000
Trying to allocate key 1513 sz 82968 flags 186
Mapped memory @ 0x2b819a8ac1000
autobl_dump_metadata() autobl_root [0x2b819a8ac000] Magic [B16B00DA] #Tmm [64] cat [0x2b819a8ac018] q [0x2b819a8b0418]
autobl_dump_metadata() autobl_root_d [0x2b819a8ac1000] Magic [B16B00DA] #Daemon [3] cat [0x2b819a8c1018] q [0x2b819a8c5418]
IP | Rate | Prod | Tout
-----|-----|-----|-----
132.173.99.18 | 80% | 1 | 1363
132.173.99.3 | 80% | 1 | 1363
132.173.99.14 | 80% | 1 | 1363
132.173.99.8 | 80% | 1 | 1363
132.173.99.5 | 80% | 1 | 1363
132.173.99.20 | 80% | 1 | 1363
132.173.99.21 | 80% | 1 | 1363
132.173.99.4 | 80% | 1 | 1363
132.173.99.9 | 80% | 1 | 1363
132.173.99.24 | 80% | 1 | 1363
132.173.99.15 | 80% | 1 | 1363
132.173.99.2 | 80% | 1 | 1363
132.173.99.19 | 80% | 1 | 1363
132.173.99.13 | 80% | 1 | 1363
132.173.99.17 | 80% | 1 | 1363
132.173.99.0 | 80% | 1 | 1363
132.173.99.6 | 80% | 1 | 1363
132.173.99.23 | 80% | 1 | 1363
132.173.99.11 | 80% | 1 | 1363
132.173.99.10 | 80% | 1 | 1363
132.173.99.22 | 80% | 1 | 1363
132.173.99.7 | 80% | 1 | 1363
132.173.99.1 | 80% | 1 | 1363
132.173.99.16 | 80% | 1 | 1363
132.173.99.12 | 80% | 1 | 1363
```

1. **IP:** IP address that is member of the greylist
 2. **Rate:** Probability of drop for an ingress packet. Higher number equals higher drop rate at the TCP layer. As drop rate goes up, retransmit rates increase, and subsequently TCP window sizes adjust closer to zero. Also, note this behavior will be different if the client IP is learned through a layer 7 header. If so, the behavior will be an HTTP rate limit versus TCP based mitigations.
 3. **Prod:** Number of stat producers. In this environment, this should always be 1.
 4. **Tout:** Time-out/TTL. Prior to releasing an IP address from the greylist, Advanced Web Application Firewall will quarantine the IP address for a period of time. During this time, TCP slowdown methods will discontinue, and HTTP rate limiting will take over. If during the quarantine period, the IP address triggers more attack traffic, the IP will be removed from quarantine and placed back in greylist. Quarantined IP addresses are visible in the DoS Dashboard/Analytics views in the Mitigation panel.
11. With the attack script still running, examine the output of the baseline script. You should be getting HTTP 200 OK responses, and the response time should be inline with pre-attack response times. Also, verify you can use browse to <http://hackazon.f5demo.com> without issue.

12. In the window where you are running the attack script, enter **CTRL-C**, then type **4** to kill the attack script cleanly.
13. Leave `baseline_menu.sh` script running.

1.7 Bad Actor Detection and Request Signatures

In the previous modules, we examined both request signature detection and bad actor detection mitigations individually. In this module, we will enable both mitigations together, and explore how they operate in tandem to mitigate a DoS attack. Additionally, we will use Advanced Web Application Firewall's DoS Reporting tools to further inspect the details of each attack.

1. Using Chromium Browser on the Xubuntu Jumpbox, open another tab to the GUI on `bigip01`
2. Navigate to **Security » DoS Protection : DoS Profiles** and click the **hackazon_bados** profile we created earlier.
3. Click the **Application Security** tab, and then click the **Behavioral & Stress-based Detection** button in the Application Security panel.
4. Click the **Edit** link to the right of the **Behavioral Detection and Mitigation** section, then uncheck the checkbox next to **Bad actors behavior detection**, and check the box next to **Request signatures detection**
5. Scroll down, and click **Update** button.
6. Navigate to **Security » Reporting » DoS » Dashboard**
7. From the **DoS Dashboard** select the refresh drop down and set value to 1 min, and grab the slider bar at the top and drag it as far right as possible.
8. On the right side of the **DoS Dashboard**, grab the handle just to the right of the HTTP and Network filter labels, and pull left to the midway point of the screen.
9. Using the inner-most vertical scroller on the right-hand side of the screen, scroll down until you see the **Transaction Outcomes** dynamic panel. Click the panel to expand, then click the three vertical lines to the left of the Transaction Outcomes label. Click on **Columns**, and click the green icon to remove all row labels except the following:
 - Transactions
 - Attacks
 - Valid Transactions
 - Mitigated Transactions
 - Blocked Transactions
 - Incomplete Transactions
10. Repeat the same process to filter the **Behavioral Signatures** dynamic panel.
11. With the baseline traffic still running, examine both the **Transaction Outcomes** and **Behavior Signatures** panels. You should see all transactions have an outcome of **Passthrough**. Also, the center column of the main dashboard view should show no current attacks in progress. Keep this window open.
12. From the Xubuntu Jumpbox open another Terminal window, or return to a previously opened window. Then:

```
f5student@xjumpbox$~ ssh root@10.1.1.245
```
13. From the SSH session, run the following command:

```
[root@bigipo01:Active:Standalone] config # watch ipidr -l /Common/vs_
↪hackazon_http+/Common/hackazon_bados
```

14. From the Xubuntu Jumpbox open another Terminal window, or return to a previously opened window. Then, re-run the attack script using the following command:

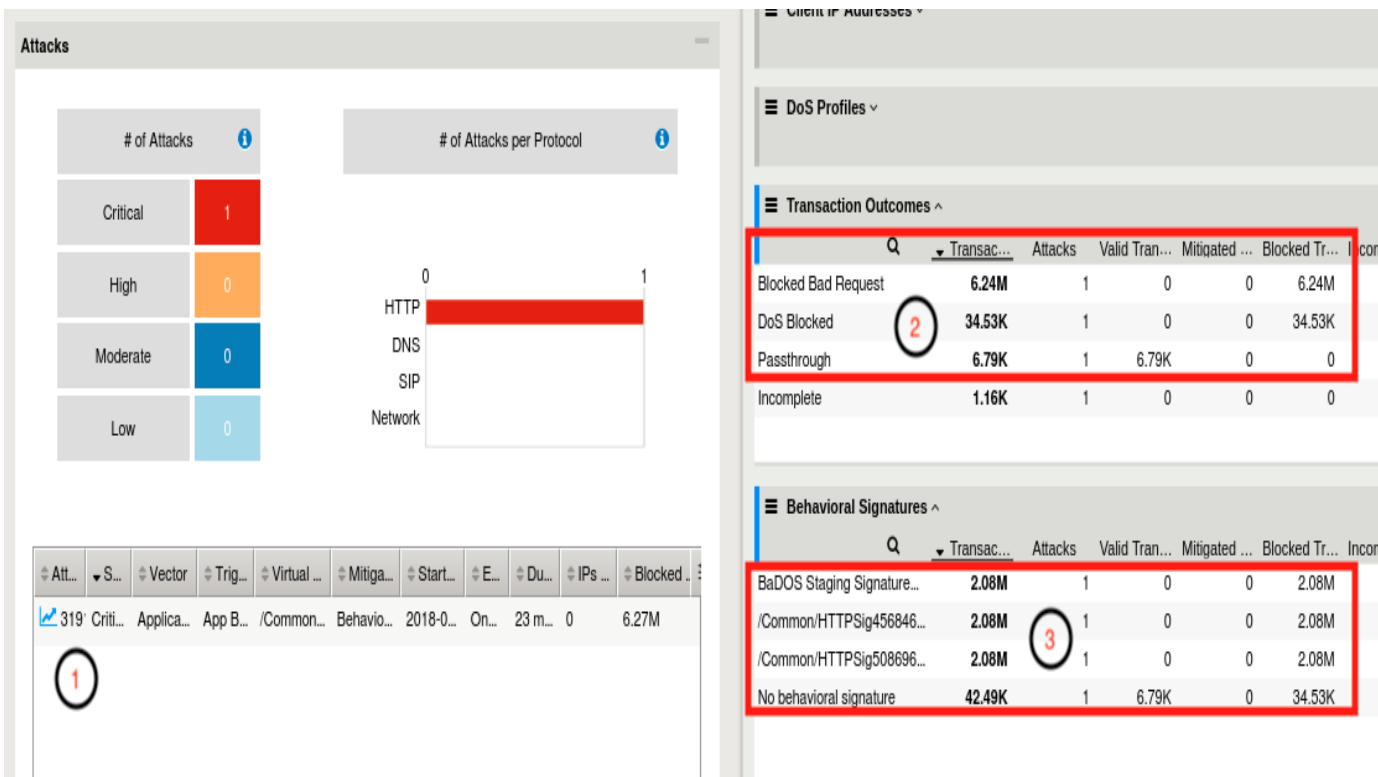
```
f5student@xjumpbox~$ ./AB_DOS.sh

- Select **2** - Attack start - score
```

15. Open another tab to the GUI on bigip01, and navigate to **Security » Event Logs » DoS » Application Events**
16. Almost immediately you should see an attack has started, and Advanced Web Application Firewall has assigned an Attack ID to the event. You will see something similar to the screenshot below:

Time	Virtual Server	Profile Name	Event	Detection Mode	Mitigation	TPS	Detection Threshold	Mitigate To Threshold	Threshold Condition	Attack ID	Entity Type
2018-05-31 06:37:54	/Common/vs_hackazon_http	/Common/hackazon_bados	Attack started	Behavioral detection	Behavioral	6 tps				1783458958	

17. Open another tab to the GUI on bigip01, and navigate to **Security » DoS Protection : Signatures**, and click on the **Dynamic** box, then set the **Refresh** value to **20 secs**. In a few moments, you should see request signatures being generated.
18. Return to the browser tab opened to the DoS Reporting Dashboard. Monitor the **Transaction Outcomes** and **Behavioral Signatures** dynamic panels. After a few minutes, you will begin to see signature based mitigations, and your dashboard should look similar to the image below:



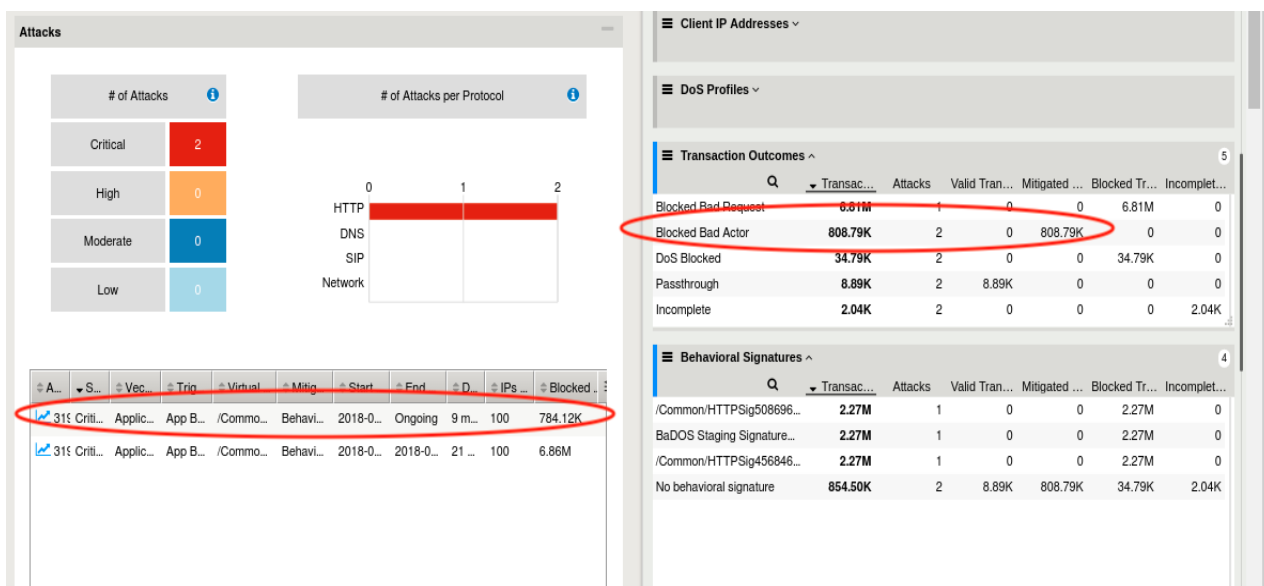
1. DoS Dashboard view shows an attack has been triggered. Select the attack, click the filter icon in upper right hand corner of Attacks table, and you can adjust the columns to view.
 2. This attack was initially mitigated with HTTP global rate limiting before a signature can be generated, accounted for in the **DoS Blocked** row. Then, as an attack signature is generated, all attack traffic should begin to be blocked with the request signature(s), evident by looking at the **Blocked Bad Request** row in transaction outcomes. At this point, if you refresh the dashboard, DoS Blocked counts should remain static, and Blocked Bad Request counters should be incrementing.
 3. Behavioral DoS will generate and adjust signatures as the traffic changes. This panel shows the signatures, referenced by signature name, that have been used to mitigate this attack.
19. Look back at the browser tab showing the Dynamic Request Signatures. You should now see that not only have signatures been generated, but they are active in mitigating a current attack. See below:

The screenshot displays the DoS Dashboard interface. At the top, a header bar shows attack details: ID 1303191789998, HTTP method, Mitigate status, Unapproved, Not-shareable, and a red status icon with a circled '1'. The main content area includes a table with columns: Alias (/Common/HTTPSig4568467065167231303191789998), Creation Time (Mon Jun 4, 11:13:24 2018 -0700), Last Modified (Mon Jun 4, 11:13:24 2018 -0700), and Description. Below this is the 'Predicates String' section, which contains a complex logical expression for attack detection. The 'Most Recent Attacks' table shows a single entry with Context vs_hackazon_http, Profile hackazon_bados, Attack Time Mon Jun 4, 01:24:48 2018 -0700, Accuracy 100%, Detection Threshold EPS 0, Mitigation Threshold EPS 0, and Current EPS 1528. A red status icon with a circled '2' is next to the Current EPS value. The bottom of the dashboard shows another attack entry with ID 233191789998, HTTP method, Mitigate status, Unapproved, Not-shareable, and a red status icon.

1. The Attack Status icon has changed to red, and shows “mitigated-with-attackid”.
 2. Most recent attacks should show an incrementing **Current EPS** (Events Per Second) counter.
20. Using a different browser tab, navigate to **Security >> DoS Protection : DoS Profiles** and click the hackazon_bados profile. As you did earlier, edit the **Behavioral Detection and Mitigation** section. This time, check the checkbox next to **Bad actors behavior detection**, then click **Update**.
21. Return to the browser tab monitoring the DoS event logs. Soon, you will see Advanced Web Application Firewall ends the current attack, and immediately triggers a new attack. Your DoS Application Events log should look similar to the below image:

Time	Virtual Server	Profile Name	Event	Detection Mode	Mitigation	TPS	Detection Threshold	Mitigate To Threshold	Threshold Condition	Attack ID	Entity Type	Entity
2018-06-04 13:11:51	/Common/vs_hackazon_http	/Common/hackazon_bados	Attack started	Behavioral detection	Behavioral	0 tps				3191790004		0
2018-06-04 13:11:50	/Common/vs_hackazon_http	/Common/hackazon_bados	Attack ended	Behavioral detection	Behavioral	5490 tps				3191790002		0
2018-06-04 12:50:34	/Common/vs_hackazon_http	/Common/hackazon_bados	Attack started	Behavioral detection	Behavioral	9 tps				3191790002		0
2018-06-04 12:29:23	/Common/vs_hackazon_http	/Common/hackazon_bados	Attack ended	Behavioral detection	Behavioral	0 tps				3191790001		0

22. Return to the browser tab opened to the DoS Reporting Dashboard. Monitor the **Transaction Outcomes** and **Behavioral Signatures** dynamic panels. After a few minutes, you will begin to see transactions being mitigated with **Blocked Bad Actor**. Shortly after you begin seeing transactions being mitigated via bad actor detection the Blocked Bad Request row should stop incrementing blocked transactions. Also, you should now see another attack has been triggered in the Attacks table. Your DoS dashboard should look similar to below image:



Note: Request Signatures **blocked** L7 requests that match the signature using a layer seven drop. Bad Actors are **mitigated** at layer three and four.

23. Return to the Terminal window from step #13 above. You should see the IP greylist again adding attacking IP addresses.
24. Return to the browser tab monitoring the Dynamic Request Signatures, and examine the attack status for the attack signatures and EPS counter. You should see the attack status as **Detected**, not mitigating, and EPS should be 0. This attack is now being mitigated exclusively by bad actors as in the previous module.

1.7.1 Bonus

The exercise above shows Request Signatures and Bad Actor Detection working in tandem to mitigate an attack. However, we have a relatively small set of attackers, so almost immediately Advanced Web Application Firewall will identify all the bad actors, and the attack will be 100% mitigated with bad actor detection. In the real world, it is highly likely the set of attackers will be very large and dynamic. So, it is quite possible, that as soon as bad actors are detected, the attacking sources will change. At that point, you will see an attack being mitigated by both request signatures and bad actors. Try the below steps to simulate this activity.

1. Return to the iRule configured in module 1 (*Create XFF-Mixed_Attacker iRule*)
2. Modify line #10 to match below and click **Update**

```
1  when HTTP_REQUEST {  
2      # Good traffic  
3      if { [IP::addr [IP::client_addr] equals 10.1.10.52] } {  
4          set xff 153.172.223.[expr int(rand()*100)]  
5          HTTP::header insert X-Forwarded-For $xff  
6      }  
7  
8      # Attack traffic  
9      if { [IP::addr [IP::client_addr] equals 10.1.10.53] } {  
10         set xff 112.173.99.[expr int(rand()*1000)]  
11         HTTP::header insert X-Forwarded-For $xff  
12     }  
13 }
```

3. Return to the browser tab monitoring the DoS Dashboard. Shortly, after the iRule change you should now see the **Blocked Bad Request** counter incrementing again. In time, Advanced Web Application Firewall will begin to learn all the new IP's as well, but you should have enough time to see both mitigations active concurrently.
4. Return to the browser tab monitoring the Dynamic Request Signatures. You should now see the attack signatures are again active and mitigating the attack until all new sources have been learned by bad actor detection.

This completes the Introduction to L7 Behavioral DoS Self Guided Lab. Thanks for attending the session, and have a great week at F5 Agility 2018!