AdvWAF-v14.x Updates

Lab Guide

Participant Hands-on Lab Guide



Last Updated: *7/16/15*

©2015 F5 Networks, Inc. All rights reserved. F5, F5 Networks, and the F5 logo are trademarks of F5 Networks, Inc. in the U.S. and in certain other countries. Other F5 trademarks are identified at f5.com.

Any other products, services, or company names referenced herein may be trademarks of their respective owners with no endorsement or affiliation, express or implied, claimed by F5.

**Table of Contents**

[Lab Network Overview 2](#_Toc21424523)

[Accessing the lab environment 4](#_Toc21424524)

[Exercise 1 – ASM Policy Creation and Logging 6](#_Toc21424525)

[Create ASM Policy 6](#_Toc21424526)

[Creating Application Security Policy 7](#_Toc21424527)

[Enable Application Security Logging 9](#_Toc21424528)

[Exercise 2 – Policy Staging 14](#_Toc21424529)

[Understanding Staging 14](#_Toc21424530)

[Perform a SQL Injection 14](#_Toc21424531)

[Perform a Cross Site Scripting attack 16](#_Toc21424532)

[Signature Staging 18](#_Toc21424533)

[Exercise 3 - Mitigating BoT Attacks 22](#_Toc21424534)

[Mitigating Bots using a Bot profile (v14.1) 22](#_Toc21424535)

[Set up the Bot profile 22](#_Toc21424536)

[Create a BoT Logging profile 23](#_Toc21424537)

[Add the BoT profile to a virtual server 24](#_Toc21424538)

[Attack with BoTs and observe the results 24](#_Toc21424539)

[Exercise 4 – Protecting Credentials with DataSafe 28](#_Toc21424540)

[Exercise 1 – Review and Attack the Login Page 28](#_Toc21424541)

[Task 1 – Review Form Fields with the Developer Tools 28](#_Toc21424542)

[Task 2 – Review Methods for Stealing Credentials 28](#_Toc21424543)

[Task 3 – Perform a Form Field “Web Inject” 29](#_Toc21424544)

[Exercise 2 – Review and Configure DataSafe Components 31](#_Toc21424545)

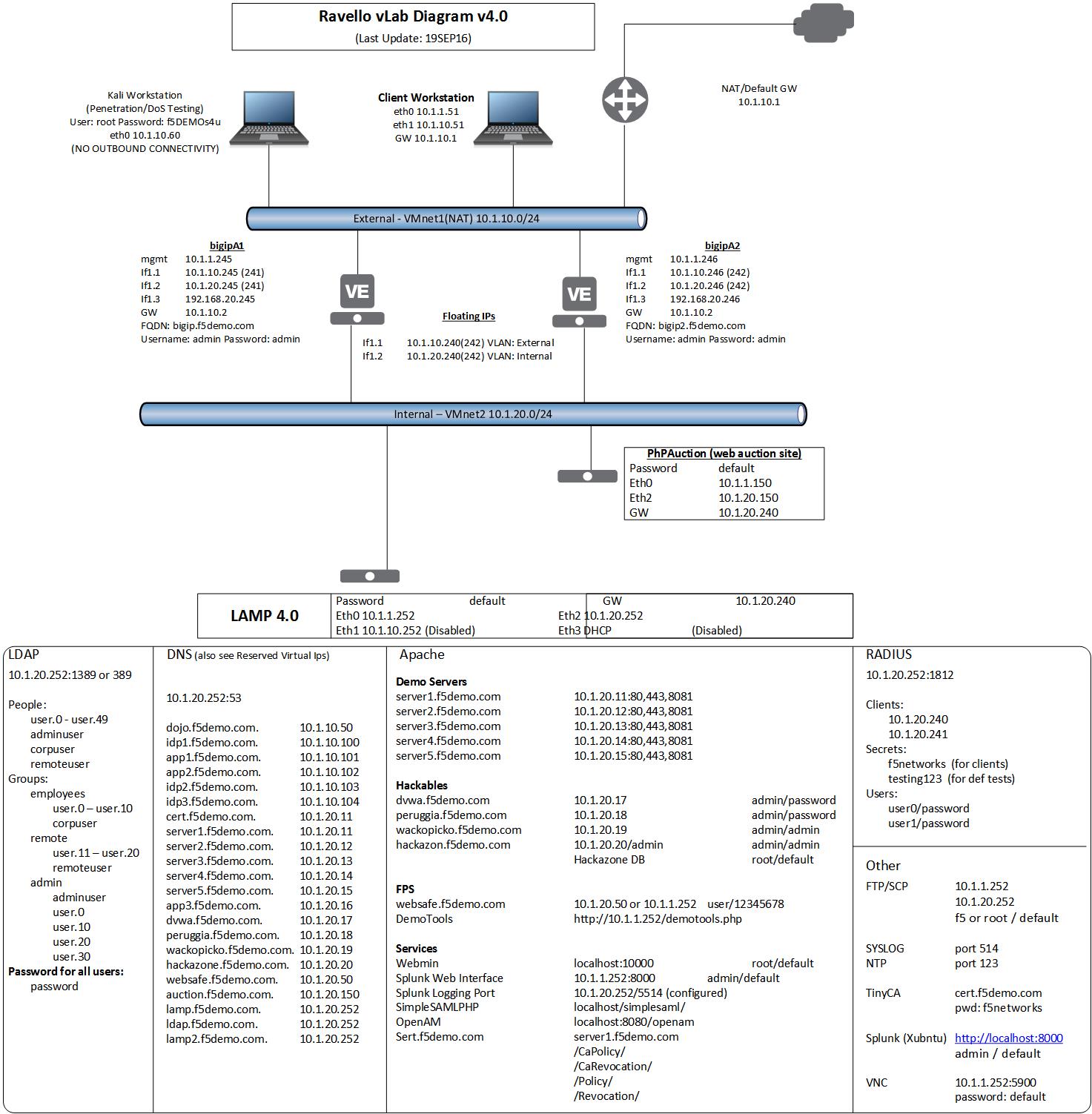
[Task 1 – DataSafe Licensing and Provisioning 31](#_Toc21424546)

[Exercise 3 – Testing DataSafe Protection 33](#_Toc21424547)

[Task 1 – Review the Protected Hackazon Login Page 33](#_Toc21424548)

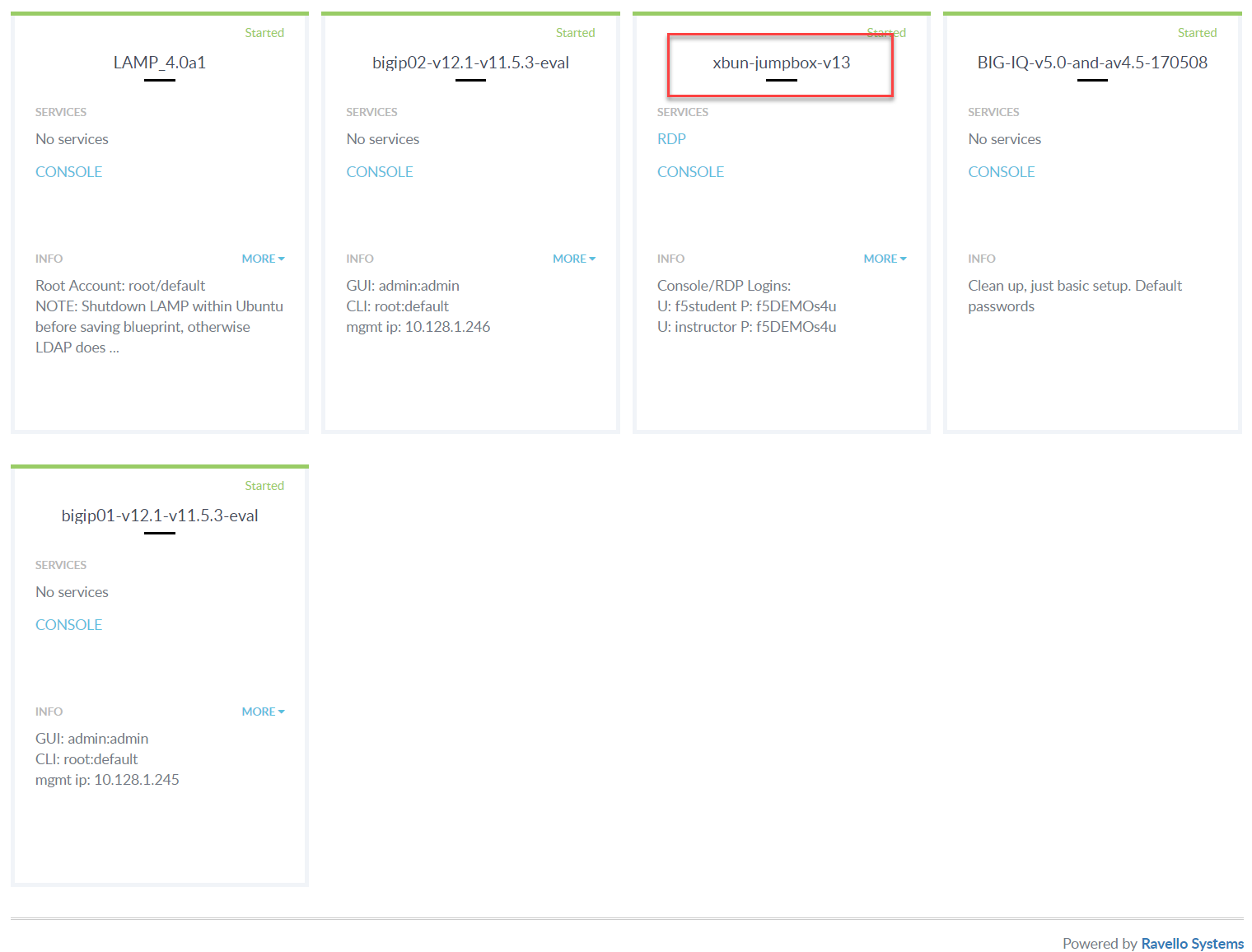
Lab Network Overview

Each student will have a BIG-IP VE environment with IP addressing as below:

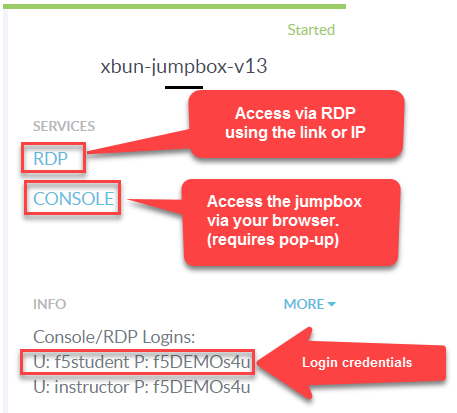


Accessing the lab environment

* 1. Open a browser and go to the link assign to you b (where **X** is your student number)
  2. Look for the **xubuntu-jumpbox-vxx**. You will use the xubuntu jumpbox for all the labs. (see below)



* 1. 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. Open the Chrome browser and log into the BIG-IP GUI to verify the BIG-IP is up.
   1. Go to **https://10.1.1.245**
      1. User: **admin**
      2. Password: **admin**
2. Now you will perform an initial configuration via command line.
   1. Open a terminal window from the taskbar at the bottom.
      1. Log in to the BIG-IP using the command: **ssh root@10.1.1.245**
      2. The password is **default.**
      3. At the BIG-IP prompt, enter **tmsh**
         1. This will place you in the BIG-IP command line mode.
   2. In your browser, open then the **Lab Guides** link on the bookmarks bar in a new tab/window.
   3. Open the **AdvWAF Base Setup.txt** file and review the commands.
   4. Copy all the commands between **# BEGIN COPY - Lab prep** and **# END COPY - Lab prep**
   5. Paste the commands into the terminal window at the **tmsh** prompt.
   6. **The BIG-IP will take several minutes to come back online.**
      1. Good time for a bathroom break. Smoke ‘em if you got ‘em.
3. Verify the virtual server and web site are up and running.
   1. Go to **Local Traffic >> Network Map**. There should be two virtual servers and all should be available (green).
   2. Open up the Firefox browser. Go to <http://hackazon.f5demo.com> and <https://hackazon.f5demo.com>

Exercise 3 - Mitigating BoT Attacks

Mitigating Bots using a Bot profile (v14.1)

Objectives:

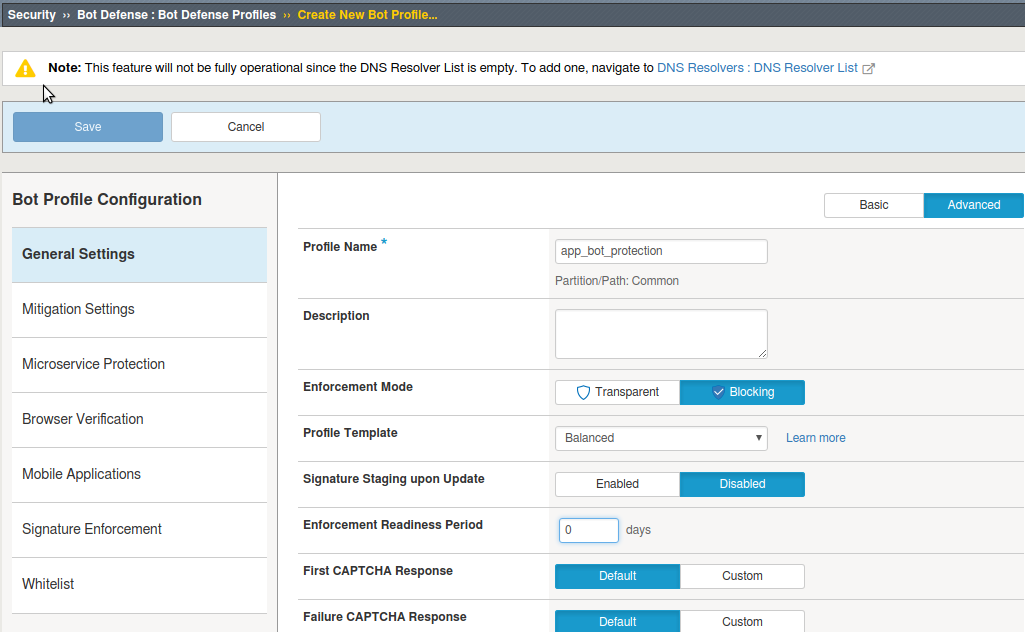
* Set up a Dos Profile using the “Balanced” profile as the profile template.
* Create a BoT logging profile.
* Attack the site using BoTs (curl, ab, OWASP Zap)
* Review the results in the BoT Dashboard and logs.
* Estimated time for completion: **15 minutes**

Set up the Bot profile

In version 14.1 Bot protection was separated from the DoS profile and give its own profile. In the section you will configure a BoT protection profile, create a logging profile and attach the Bot profile and logging profile to the virtual server.

You will need to create a new BoT profile before you can configure DoS mitigation.

1. Go to **Security ›› Bot Defense : Bot Defense Profiles** here you can see a number of pre-defined BoT profiles. Select **Create**. Move down the configuration settings on the **Bot Profile Configuration** sidebar.
   1. Note the **Note**. Because we haven’t set up DNS on the BIG-IP, BoT protection will not be able to determine if benign Bots, such as GoogleBots, are being impersonated.
      1. The BIG-IP will do DNS lookups to determine the appropriate IP addressing of certain BoTs.
   2. Under **General Settings** select the **Advanced** menu in the upper right corner.
      1. Profile Name: **app\_bot\_protection**
      2. Enforcement Mode: **Blocking**
         1. For our purposes we will go straight to Blocking mode
      3. Profile Template: Balanced
         1. Select the **Learn more** link to see the difference in the default profiles.
      4. Enforcement Readiness Period: 0 days
         1. Again to speed up the process



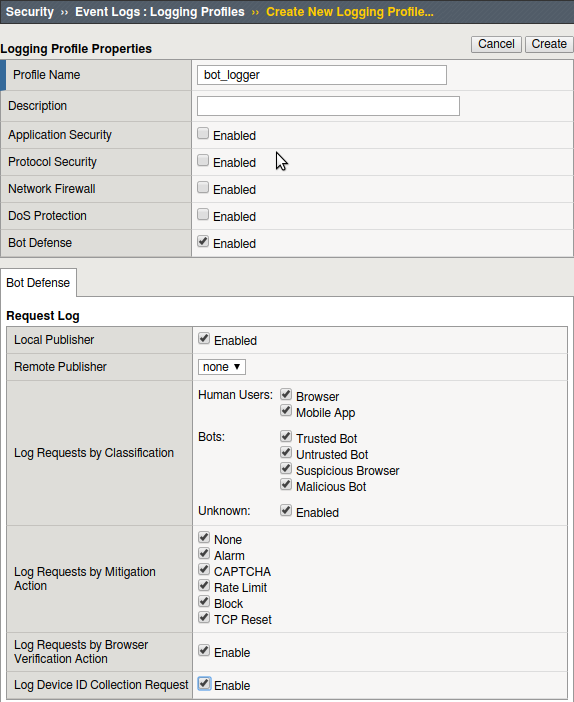
* + 1. Everything else can be left at the defaults, but feel free to review the different option
  1. **Mitigation Settings** can also be left at the current defaults. If you had left the **Enforcement** **Mode** at **Transparent** the mitigation enforcement cases at the bottom would have defaulted to **Disabled**. Click the question marks for more information on the cases.
  2. **Microservice Protection** is left at the defaults.
  3. **Browser Verification** can be left at the defaults. Settings here will determine, if and when we will challenge client browser to determine if it actually a BoT.
  4. **Mobile Applications** can be left at the defaults, but this section works in conjunction with the Anti-Bot mobile SDK to determine if mobile devices have been compromised.
  5. **Signature Enforcement** will be left at default. This allows you more granular and quicker enforcement of BoT signatures.
  6. **Whitelist** will be left at the default.

1. Select **Save** in the upper right corner.

Create a BoT Logging profile

Like with Advance WAF aka ASM you will create a logging profile to capture DoS events.

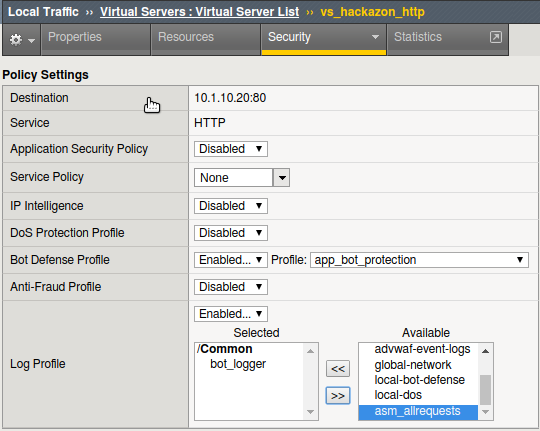
1. Go to **Security ›› Event Logs : Logging Profiles** and select **Create**. Name your profile **bot\_logger**.
   1. You could also modify the **asm\_allrequests** logging profile to include BoT logging, but a separate could be used on multiple virtual servers where only BoT protection is required.
   2. Check **Bot Defense** enable box.
      1. In the **DoS Protection** tab enable the **Local Publisher**.
      2. In the **Bot Defense** tab check **ALL** the boxes.
      3. Hit **Create**.



Add the BoT profile to a virtual server

The DOS profile has only Bot Signatures enabled.

1. Go 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 **BoT Defense Profile** and select the **app\_bot\_protection** profile.
3. Add **bot\_logger** to the **Log Profile**.
4. For purposes of this lab, **Disable** the **Application Security Policy** and remove **asm\_allrequests** from the **Log Profile.**



1. Finally, select **Update**.

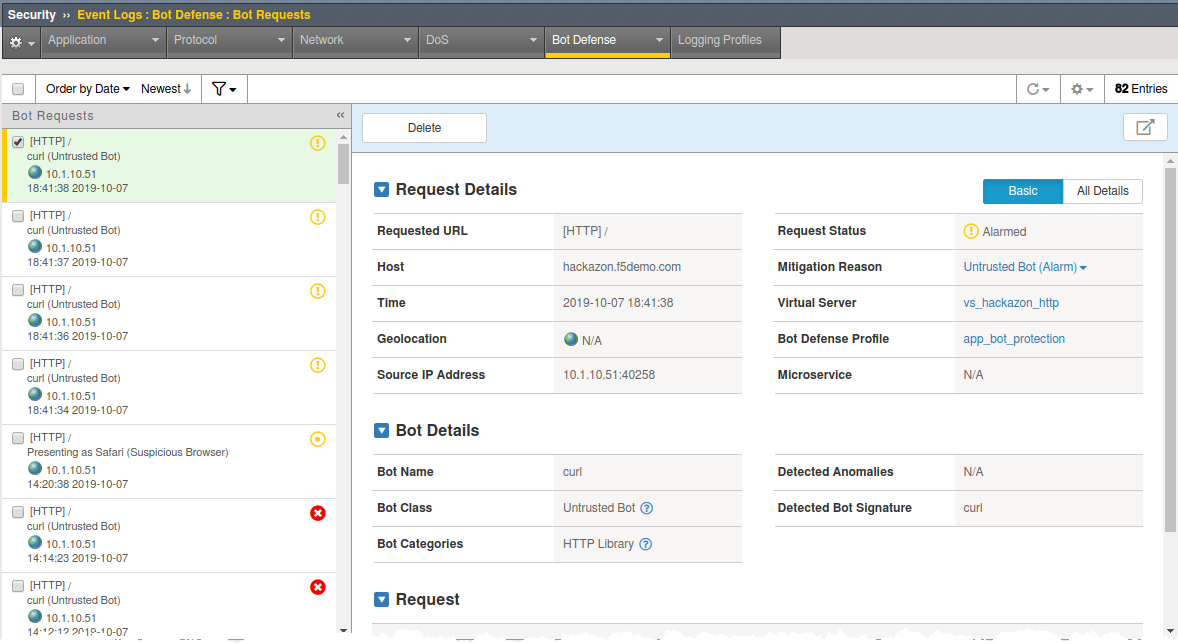
Attack with BoTs and observe the results

First let’s use an **HTTP Library (Benign)** tool, **curl**. Remember just because tools are in the **Benign** category doesn’t mean they can’t be used for nefarious purposes. Reporting will at allow you to know that these tools are being used against your site.

1. From a terminal window on the jumpbox run the following several times:

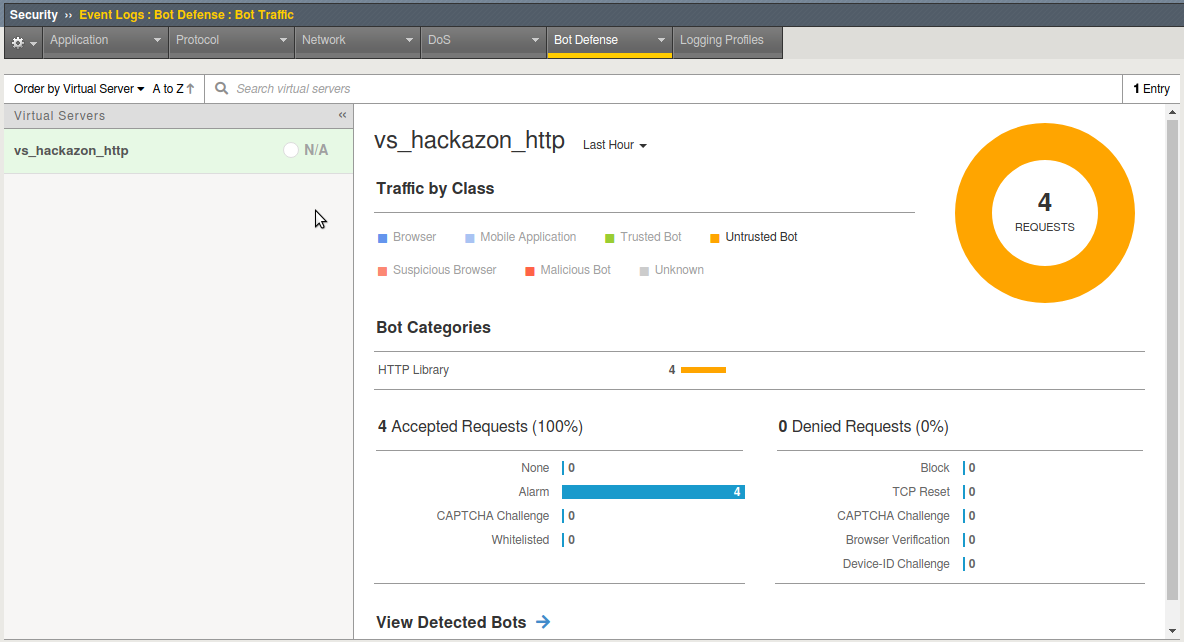
curl <http://hackazon.f5demo.com>

In the **Security ›› Event Logs : Bot Defense : Requests** you should see entries similar to this:



The BoT signature and category are logged and the BoT is allowed because the **Untrusted Bot** category is set to **Alarm** only.

In the **Security ›› Event Logs : Bot Defense : Bot Traffic** screen it may take a few minutes for the data to show up, but you should see something similar to the following:

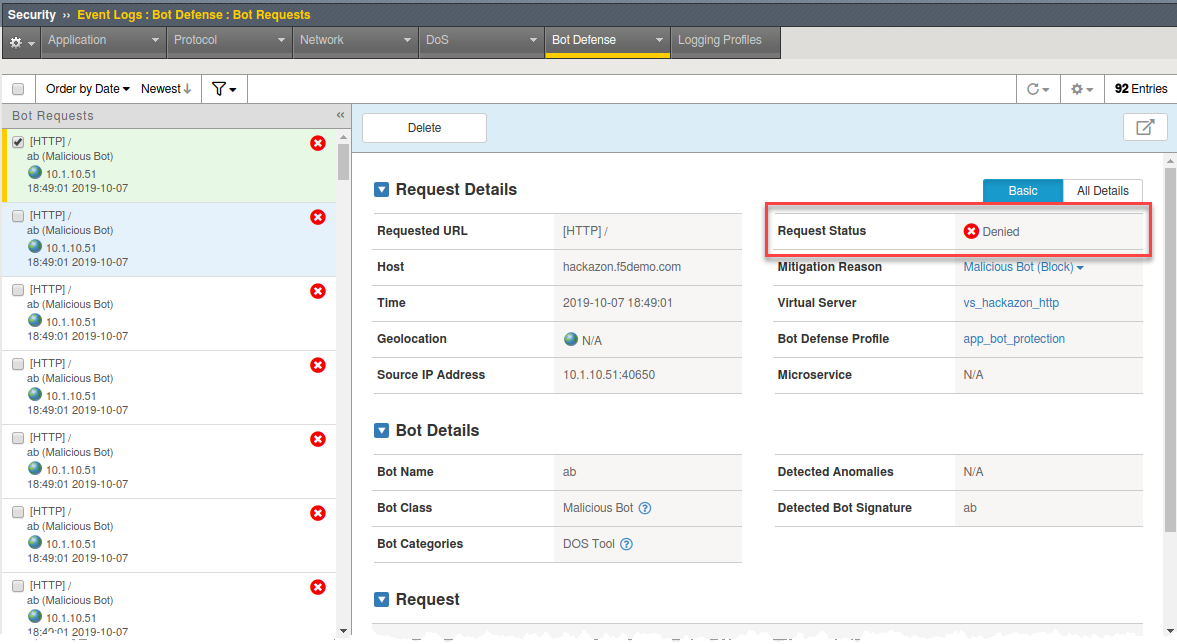


This time we will use the apache bench (ab) BoT from the **DOS Tools (Malicious)** category. The BoT was originally design for benchmark testing but is mostly used for those nefarious purposes I spoke of earlier.

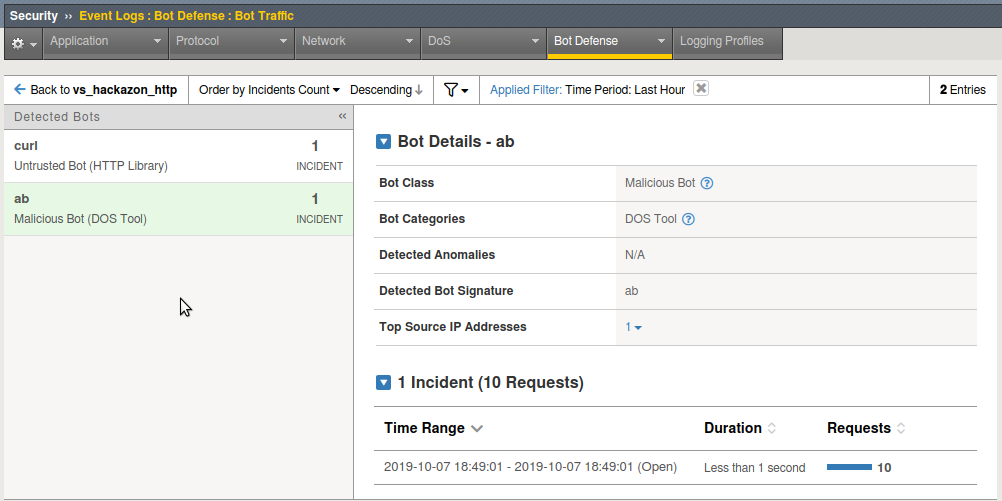
1. From a terminal window on the jumpbox run the following:

ab -c 10 -n 10 -r http://hackazon.f5demo.com/

In the DoS event log you can see this BoT was **Denied** (blocked) by sending a TCP reset immediately to the client.



If you go back to the **Bot Traffic** dashboards you will see the new attacks. If you select **View Detected Bots** at the bottom you will get a summary of the incidents.

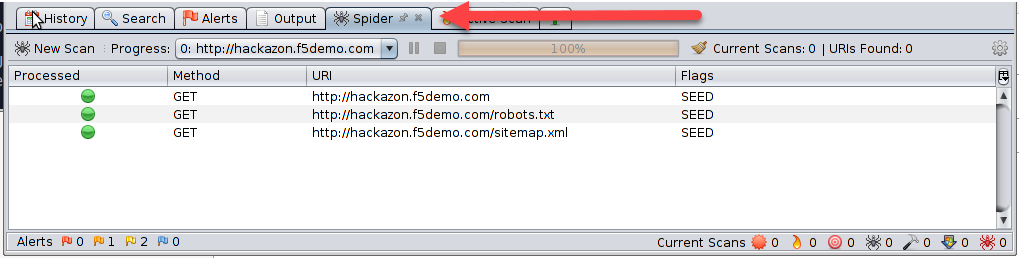


You are now going to use the OWASP ZAP tool to run a spider bot attack against the Hackazon website.

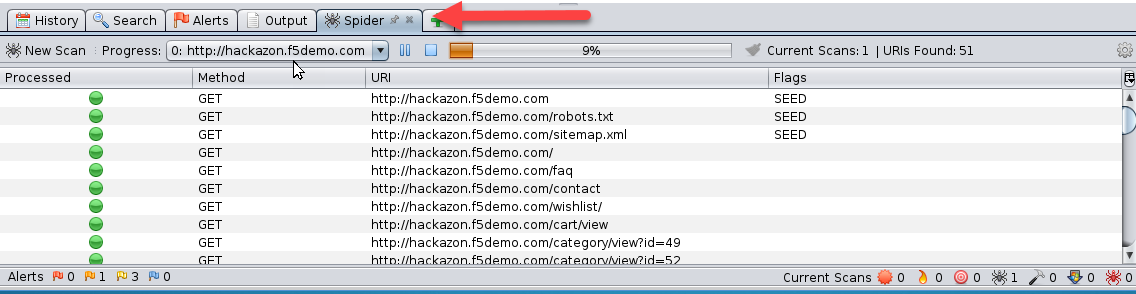
1. From the **Applications Menu** in the upper left-hand corner of the jumpbox go to **OWASP > Proxies > ZAP**.
2. Once ZAP is open, click on **Start** when asked **Do you want to persist the ZAP session**.
3. In the **Quick Start** tab, in the **URL Attack** box, enter <http://hackazon.f5demo.com> and hit the **Attack** button**.**



1. Once the attack has started a **Spider** tab should appear in the bottom ZAP window. You will see ZAP attempting to crawl the web site. **T**he attack will be short lived.



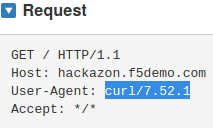
Without Bot protection it would have scan the site.



1. Check the Bot Requests you’ll find Non-browser presenting as Internet Explorer request. Go to **All Details** and check out the attack and how BIG-IP detected it.

Now let’s see how BIG-IP challenges Bots that don’t match up to the signatures. We already saw some of this with OWASP Zap, but here you will see the BIG-IP challenge the client to prove it’s not a Bot.

1. Go to Bot Request, find a curl request and look at the **Request** section. In it you will find the User-Agent set to curl/7.52.1

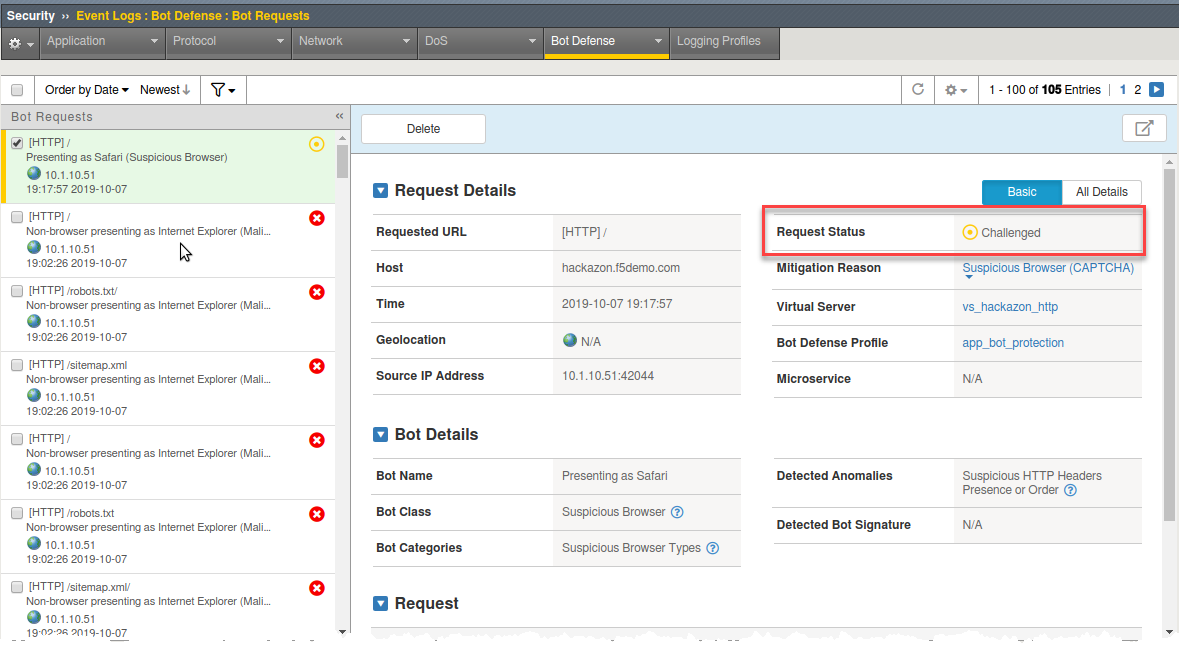


1. That makes it kind of easy to detect, but what if we changed the User-Agent to a legitimate browser. Could the BIG-IP still detect it?
   1. Run the following command from a terminal window:

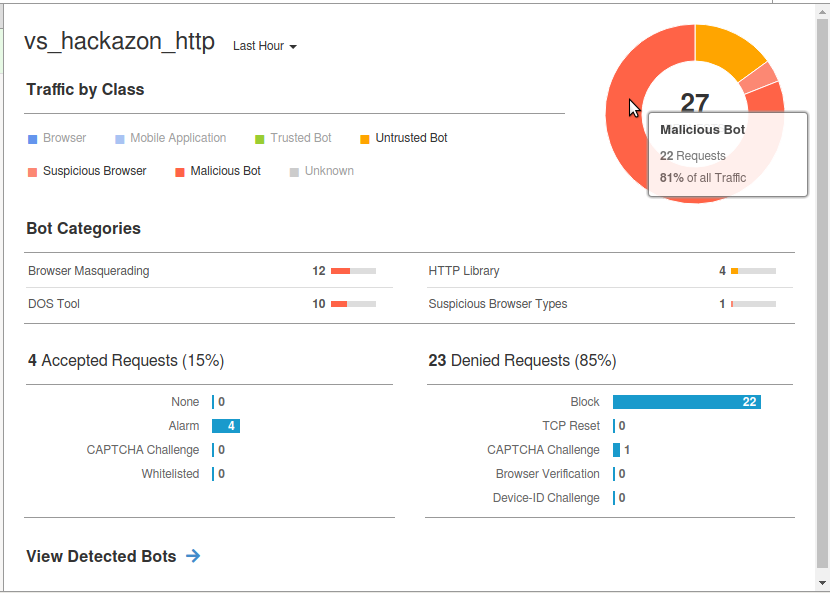
curl -A "Mozilla/5.0 (iPhone; U; CPU iPhone OS 4\_3\_3 like Mac OS X; en-us) AppleWebKit/533.17.9 (KHTML, like Gecko) Version/5.0.2 Mobile/8J2 Safari/6533.18.5" http://hackazon.f5demo.com

The resulting request failed, and you can see the obfuscated code and the BIG-IP block page and support ID at the bottom. Even though the signature is allowed, curl cannot meet the javascript challenge (returning ASM cookie, prefixed by TS, with the javascript results) presented it. Check the BoT Request log for the results showing the Bot was challenged and we sent a Captcha.





1. On you Bot Traffic dashboard you can see the Browser Masquerading category



Exercise 4 – Protecting Credentials with DataSafe

Objectives:

* The purpose of this lab is to show the new DataSafe perpetual license in 13.1.
* You will review the login page with and without DataSafe protections.
* You will enable and test encryption, obfuscation, and decoy fields.

Estimated completion time: 45 minutes

Exercise 1 – Review and Attack the Login Page

Task 1 – Review Form Fields with the Developer Tools

* Open Firefox and access [http://hackazon.f5demo.com/user/login.](http://hackazon.f5demo.com/user/login)
* Right-click inside the **Username or Email** field and select **Inspect Element**.

Question:

What is the **name** value for this field?

* Right-click inside the **Password** field and select **Inspect Element**.

Question:

What is the **name** value for this field?

**FOOD FOR THOUGHT:** How difficult would it be for malware to know which fields to grab to steal credentials from this page? How difficult would it be for an attacker to stuff credentials into these fields? They could simply put the stolen username into the “username” field and the stolen password in the “password” field. This is what you did with the Sentry MBA tool earlier in this lab.

Task 2 – Review Methods for Stealing Credentials

* In Firefox click the **FPS Demo Tools** bookmark.

This includes tools that behave like real malware.

* On the login page enter your first name as username and **P@ssw0rd!** as password but do not click **Sign In**.
* From the **Demo Tools** click **Steal Password**, and then click on the password field.

The “malware” is using JavaScript to grab the value of the password field out of the DOM (Document Object Model). This is one way that malware can steal credentials; even before the user has submitted them to the application.

* Click **OK**, then clear the password you entered.
* From the **Demo Tools** click **Start Keylogger**, and then enter the same password from earlier.
* Watch the top of the Demo Tools.

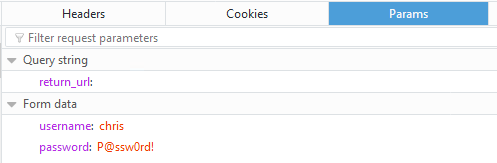


This is another way that malware can steal credentials. The “malware” is using a JavaScript keylogger to log the password as it is typed.

* In the developer tools select the **Network** tab, then click the trash can icon to delete the requests.
* On the login page (with your first name and **P@ssw0rd!** entered) click **Sign In**.

**NOTE:** Your login will fail, but your credentials were still sent to the web server.

* In the **Network** tab select the **/login?return\_url=** entry, and then examine the **Params** tab.



The user’s credentials are visible in clear text. This is another way that malware can steal credentials.

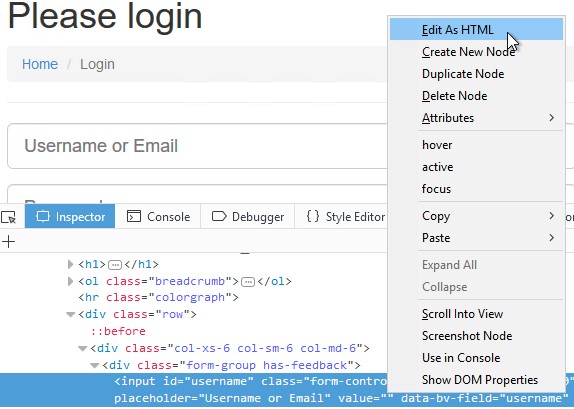
By “grabbing” the POST request and any data sent with it, including username and password.

Task 3 – Perform a Form Field “Web Inject”

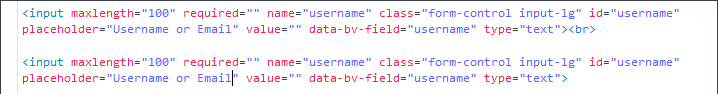
* Return to the <http://hackazon.f5demo.com/user/login>page.

**NOTE:** It should NOT have **?return\_url=** at the end of the URL in the address bar.

* Right-click inside the **Username or Email** field and select **Inspect Element** again.
* Right-click on the highlighted text and select **Edit as HTML**.



* Select all the text in the window and type **Ctrl+C** to copy the text.
* Click after the end of **type="text">** and type **<br>**, and then press the **Enter** key twice.
* Type **Ctrl+V** to paste the copied text.



* For the new pasted entry, change the **name**, **id**, and **data-by-field** values to **mobile**, and change the

**placeholder** value to **Mobile Phone Number**.



* Click outside of the edit box and examine the Hackazon login page.

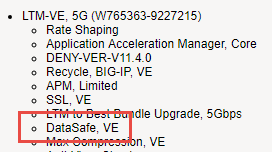
This is an example of the type of “web injects” that malware can perform to collect additional information. This same technique could be used to remove text or form fields. Note that this was done on the client side, in the browser, without any requests being sent to the server. The web application and any security infrastructure protecting it would have no idea this is happening in the browser.

* Close Firefox.

Exercise 2 – Review and Configure DataSafe Components

Task 1 – DataSafe Licensing and Provisioning

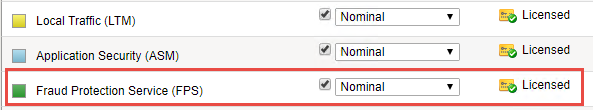
* In the Configuration Utility, navigate to **System** and right-click on **License**, and then select **Open Link in New Tab**, and then select the new tab.



**DataSafe** includes only the Application Layer Encryption (ALE) module of WebSafe. Unlike WebSafe, **DataSafe** is licensed perpetually per device, just like ASM, APM, or any other licensed module.

**DataSafe** is **NOT** included in the Best Bundle.

* Open the **System > Resource Provisioning** page.



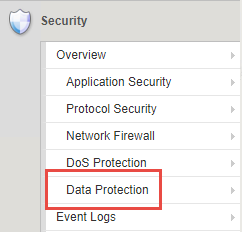
When **DataSafe** is licensed, **Fraud Protection Service (FPS)** will display as **Licensed**. This is different than **WebSafe**, where Fraud Protection Services will show up as N/A.



* Expand the **Security** menu.

There is a **Data Protection** option. This is different than WebSafe where this menu option is

###### Fraud Protection Service. DataSafe



###### WebSafe



#### Task 2 – DataSafe Configuration

* Open the **Security > Data Protection > DataSafe Profiles** page and click **Create**.
* For **Profile Name** enter **Hackazon-DS**.
* On the left menu click **URL List**, and then click **Add**.

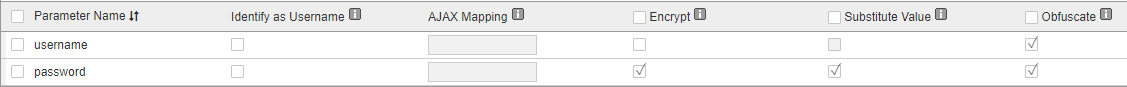


* For **URL Path** leave **Explicit** selected, and type **/user/login**.
* From the left panel open the **Parameters** page.

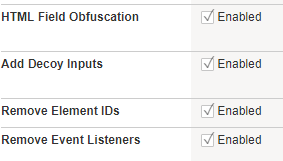
Remember from earlier you found that the username and password parameter names are **username**

and **password**.

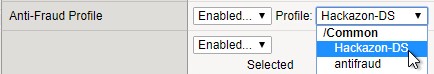
* Create a new parameter named **username**, and then click **Add**.
* Create a second parameter named **password**, and then click **Add**.
* Scroll to the right to view all the parameter options.
* For the **username** parameter select the **Obfuscation** checkbox.
* For the **password** parameter select the **Encrypt**, **Substitute Value**, and **Obfuscate** checkboxes.



* Scroll to the left, and from the left menu open the **Application Layer Encryption** page. Notice that most features are enabled by default.
* Review the explanations for the different features.
* Select the **Add Decoy Inputs** and **Remove Element IDs** checkboxes, and then click **Create**.



* Open the **Virtual Server List** page and click **vs\_hackazon\_http**, and then open the virtual server **Security > Policies** page.
* From the **Anti-Fraud Profile** list select **Enabled**.
* From the **Profile** list box, select **Hackazon-DS**, and then click **Update.**



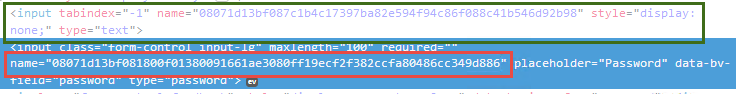
Exercise 3 – Testing DataSafe Protection

Task 1 – Review the Protected Hackazon Login Page

* Open a **private** Firefox window and access <http://hackazon.f5demo.com/user/login>.
* Right-click inside the **Password** field and select **Inspect Element**.

Question:

What is the **name** value for this field?

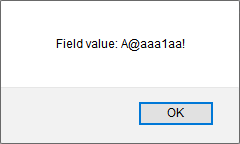


**Obfuscation** - Notice that the name of the password field (outlined in red) is now a long cryptic name and is changing every second. The same is true of the username field.

**Add Decoy Inputs** – Notice that there are other random inputs being added (outlined in green). The number and order of these inputs is changing frequently.

**FOOD FOR THOUGHT:** Considering this obfuscation and you earlier review of the Sentry MBA tool, do you think DataSafe could protect the login page from a credential stuffing tool like Sentry MBA?

* In Firefox click the **FPS Demo Tools** bookmark.
* On the login page enter **P@ssw0rd!** as password but do not click **Sign In**.
* From the **Demo Tools** click **Steal Password**, and then click on the password field.



**Substitute Value –** DataSafe is protecting the password field from malware JavaScript grabbing the value of the field from the DOM. Uppercase letters are replaced with “A”, lower case letters are replaced with “a”, and non-alphanumeric characters are replaced with “!”.

* Click **OK**, then clear the password you entered.
* From the **Demo Tools** click **Start Keylogger**, and then begin entering the same password from earlier while watching the demo tools title bar.



**Keylogger Protection** – DataSafe injected java script is injecting fake keystrokes to protect the page from software keyloggers common in some malware.

* Close the FPS Demo Tool.
* In the developer tools window select the **Network** tab, then click the trash can icon to delete any current requests.
* On the login page (with your first name and **P@ssw0rd!** entered) click **Sign In**.
* In the **Network** tab select the **/login?return\_url=** entry, and then examine the **Params** tab.

Questions:

What parameters were submitted?

Do you see a username or password field?

Do you see the username you submitted?

**Obfuscation** – DataSafe obfuscates the names of the parameters when they are submitted in a login request.

**Encryption** – DataSafe encrypted the value of the password field so that it is not a readable value in the login request.

These two features together protect sensitive parameters.