Blue Team Data Analysis

In the second part of this project we are analyzing the resulting data from attacks on a dvwa web browser. We are using splunk as our SIEM and the first thing we do is set up our splunk to receive data from the dvwa.

Graphical user interface, text

Description automatically generated

Text

Description automatically generated

Graphical user interface, text

Description automatically generated

(Continues next page)

A screenshot of a computer

Description automatically generated with medium confidence

Once Splunk is receiving logs we can begin our attacks so that we can have some usable real-world data to see.

First, we ran a dirb on our DVWA to see what files and directories the site had.

Text

Description automatically generated

It generated many 400 error code status logs. It did this because dirbuster launches thousands of GET requests at a website and attempts to find all the files and directories of this site. Here is a picture of just some of the logs.

Graphical user interface, text, application

Description automatically generated

We then create an alert in Splunk that notifies us of these types of attacks we call it Bad 404’s.

A screenshot of a computer

Description automatically generated with medium confidence

Graphical user interface

Description automatically generated with low confidence

Dirbuster Mitigation: The most effective mitigation of this type of attack is to have an IDS in place so that it can monitor and recognize this type of traffic. Other mitigations are more similar to best practices and these include; securing all content not meant for public access, access control lists are up to date, cleaning out old logs, having automated tools to remove empty files and directories, and rebuilding a clean image of your site regularly.

The next attack was a SQL Injection. First we run the command to see a list of all the users.

Graphical user interface, website

Description automatically generated

Text

Description automatically generated with medium confidence

Next we see if we can find the password column of the table but the site said the password column doesn’t exist so we ran a command that would generate all the column names.

(Continues next page)

A picture containing background pattern

Description automatically generated

From this we saw that the password is under the ‘users’ column so we ran a command to join the user and the password together.

Text

Description automatically generated

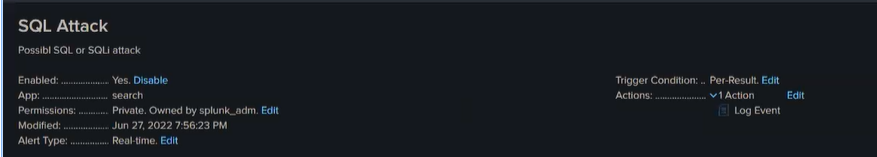
It gives us the password hashes and all we need to do to continue the attack is crack these hashes.

On the Blue Team side, it has been generating logs for us to analyze.

Graphical user interface, text

Description automatically generated

As you can see, under the id field in Splunk it shows the raw SQL input that we used on our DVWA site. The next step is to create an alert for this attack.



We ran the attack a few more times to see if alerts are being generated and we see that they are. It shows us these logs.

(continues next page)

Text

Description automatically generated

SQL Injection mitigation: The best mitigation for an SQL attack is with input validation. The ‘ is indicative of a SQL attack. It is also possible to remove Database error messages so that the attack doesn’t gain additional information with attempting a SQL attack.

The last attack that we get logs for a cross site scripting attack where we steal session cookies.

First we set up our http site.

Text

Description automatically generated

We then setup the attack on the https bar.

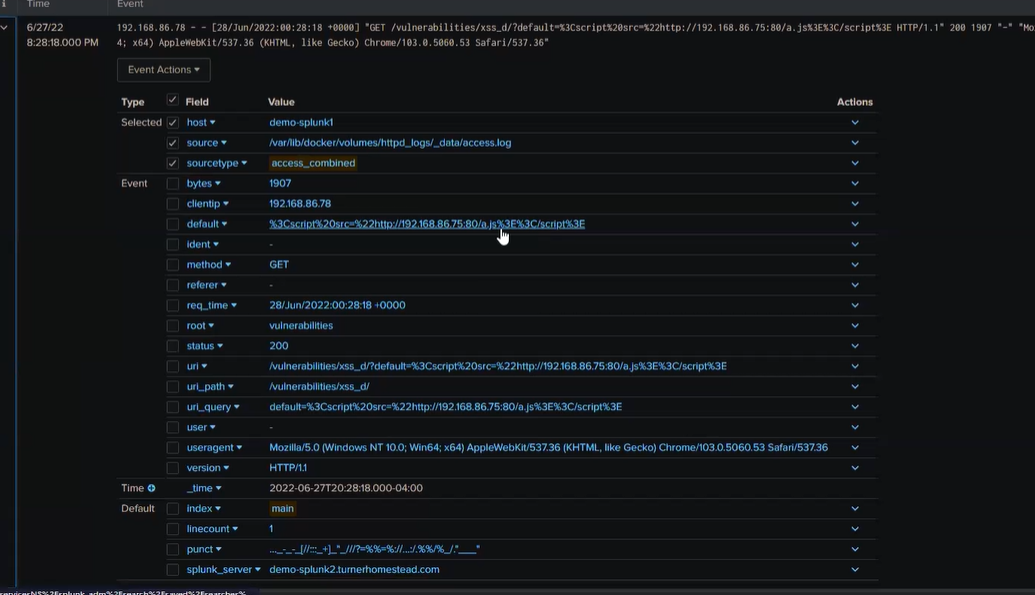


We then set up as simple script to grab the image of the session cookie and bring it back to our host. The script is the a.js to match the script we inputted into the https bar.

Text

Description automatically generated

When we pull our logs after launching this attack this is what we see



Cross site scripting attack mitigation: The .js or anything java script is indicative of this attack. Input validation is a good mitigation against cross site scripting. Also setting the appropriate response headers to help stop HTML or JavaScript where these are not intended.