Scanning and Finding out Vulnerabilities

By

Yemula Varnith

Submitted in partial Fullfillment of the Requirements
for the Course in cyber security from 1STOP in association with
TALAKUNCHI

April 2022

DECLARATION

I hereby declare that the contents presented in the project entitled "Scanning and Finding out Vulnerabilities", submitted in partial fulfillment for the course in cyber security from 1STOP in association with TALAKUNCHI, is a record of original work investigated by me.

ACKNOWLEDGEMENT

We sincerely thank all the teaching and non-teaching support staff of the 1stop and Talakunchi. We also like to thank all our college faculty members for their timely suggestions and motivation during the course of our project work. We thank our parents, who were the backbone behind our deeds.

Finally, we express our immense gratitude with pleasure to all individuals who have either directly or indirectly contributed to our needs at the right time for the development and success of our project work.

ABSTRACT

We are more vulnerable to the dangers of cyberspace because we live in an era where most tasks can be digitized. Our presence in the digital domain has exploded, from simple operations like placing a phone call to complicated processes like financial transactions via online banking. We put our information out there before, after, and during these activities, and if it isn't secure, we are vulnerable to cyberattacks. Cyber security is a branch of study that deals with these issues. The protection of our computer network and systems from hostile activity that can harm people, software, and hardware is known as cyber security.

TASK 1

Take some websites (vulnerable websites) Scan using OWASP ZAP Tool (quick/automated)

Set of vulnerabilities - make a report with mitigation

List of Tools used

Kali Linux

Kali Linux is a security Linux distribution based on Debian that was created with computer forensics and advanced penetration testing in mind. It was created by Mati Aharoni and Devon Kearns of Offensive Security, who rewrote BackTrack. Kali Linux comes with a large number of tools that are well-suited to a variety of information security tasks, including penetration testing, security research, computer forensics, and reverse engineering.

Owasp Zap

Owasp ZAP (short for Zed Attack Proxy) is a web application security scanner that is free and open-source. It's meant for beginners as well as experienced penetration testers.

It has been designated as a Flagship project by the Open Web Application Security Project (OWASP).

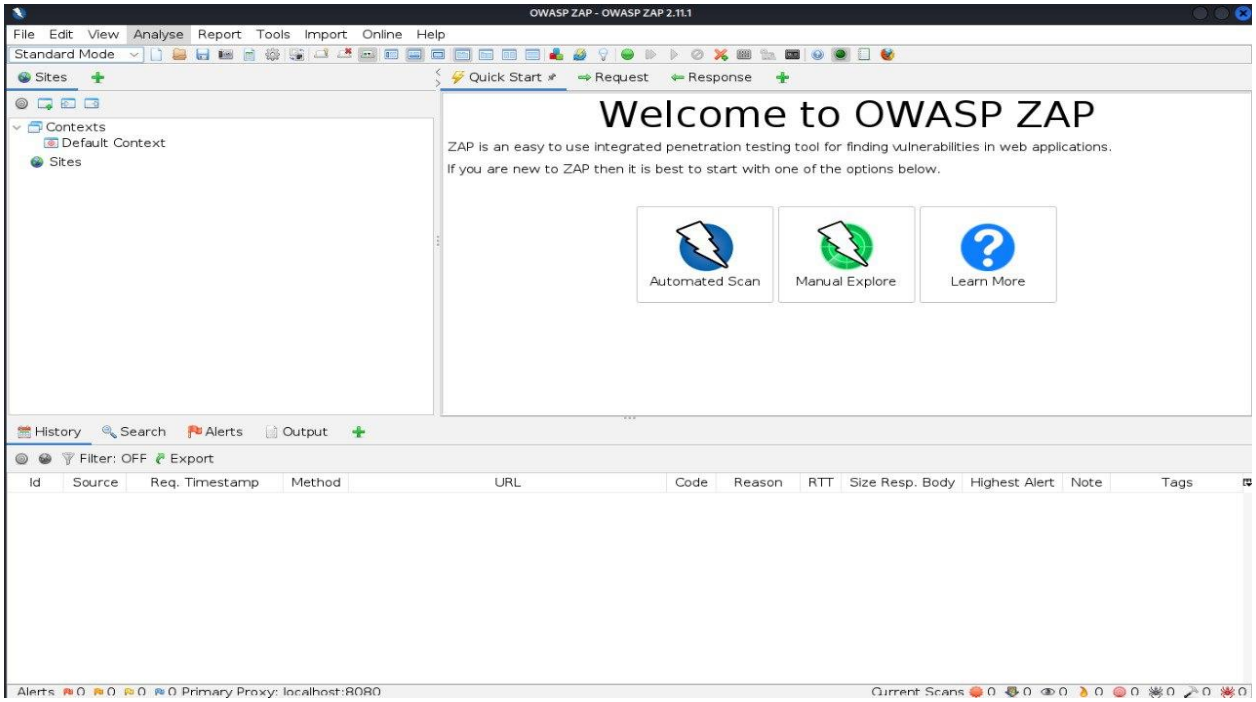
It can be used as a proxy server to modify all traffic that flows through it, including https traffic.

It can also be set up to run as a daemon, which can be controlled using a REST API.

Intercepting proxy server, Traditional and AJAX Web crawlers, Automated scanner, Passive scanner, Forced browsing, Fuzzer, WebSocket support, Scripting languages, and Plug-n-Hack support are just a few of the built-in capabilities. It offers a plugin-based design and an online "marketplace" where users can contribute new or updated functionality. The graphical user interface (GUI) control panel is simple to use.

Step 1: Opening OWASP ZAP

When OWASP ZAP eventually launches, it should look something like this.



Welcome to OWASP ZAP

ZAP is an easy to use integrated penetration testing tool for finding vulnerabilities in web applications.

If you are new to ZAP then it is best to start with one of the options below.

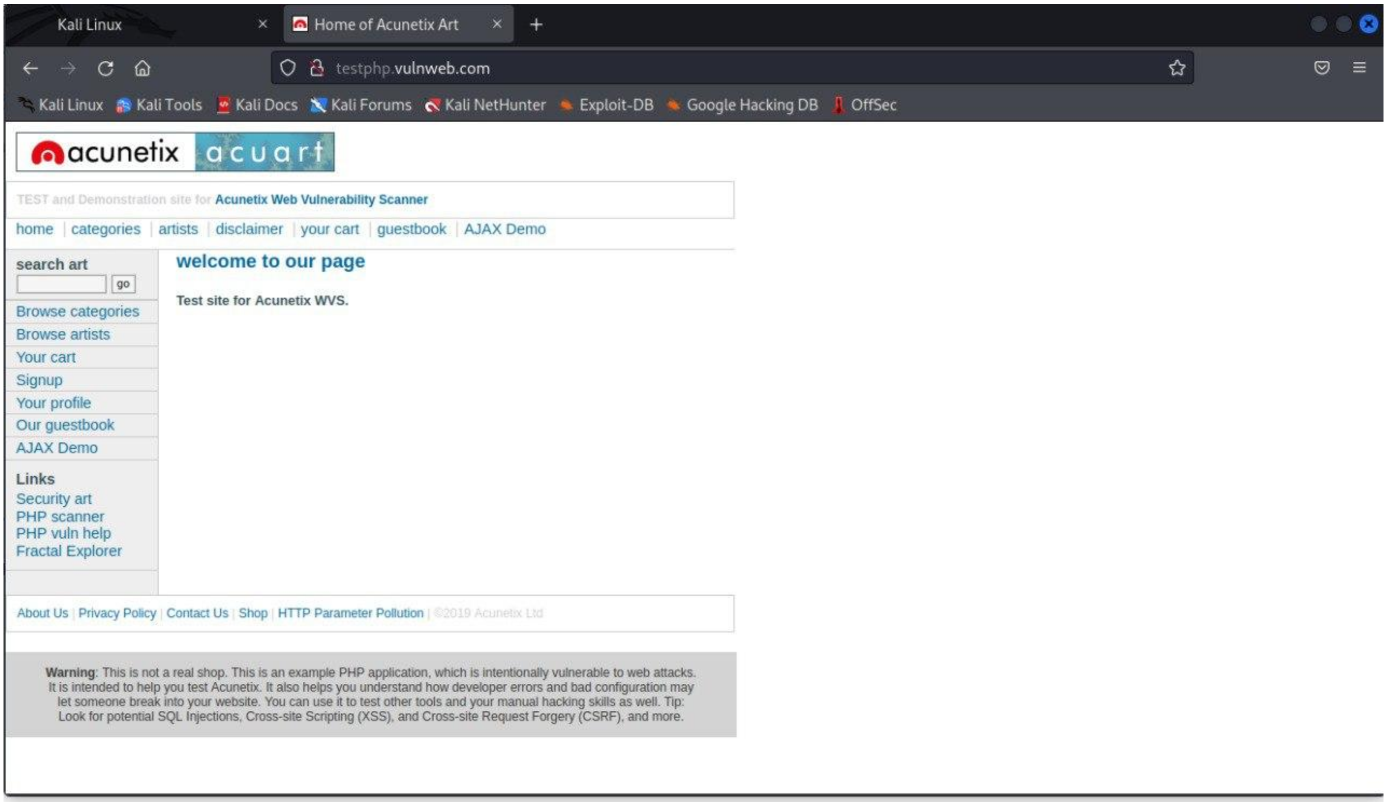
Primary Proxy: localhost:8080

This tool has a lot of useful functions, but we'll focus on the "Automated Scan" function in the wide right-hand window for now. In this mode, OWASP ZAP goes to the website we specify and starts scanning for vulnerabilities.

After clicking on the automated scan, it asks for the URL to attack or scan.

Step 2: Attacking a Website

Let's take a look at www.testphp.vulnweb.com/login.php, which was previously left insecure and safe to test.



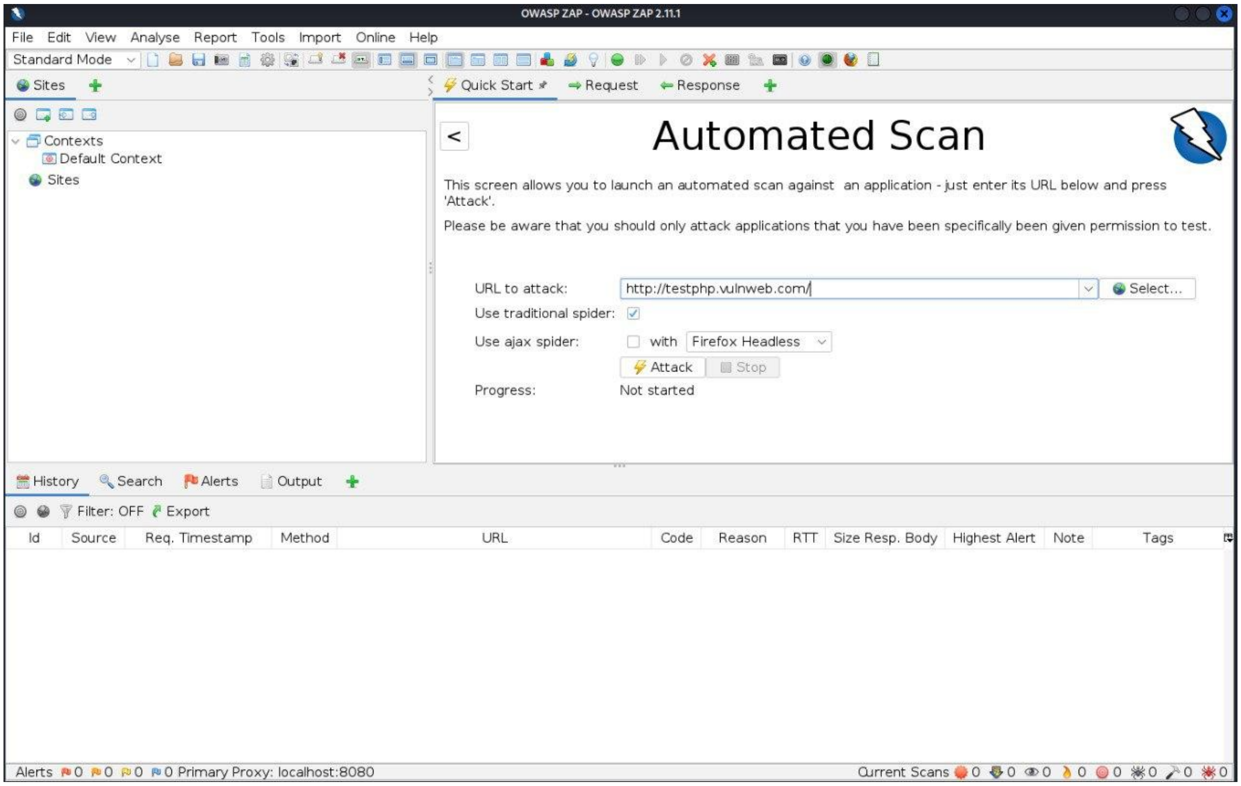
TEST and Demonstration site for **Acunetix Web Vulnerability Scanner**

welcome to our page

Test site for Acunetix WVS.

Warning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.

Simply type the URL into the box next to "URL to attack," then click the "Attack" button underneath it.



Automated Scan

This screen allows you to launch an automated scan against an application - just enter its URL below and press 'Attack'.

Please be aware that you should only attack applications that you have been specifically been given permission to test.

URL to attack: http://testphp.vulnweb.com/

Use traditional spider: ☑

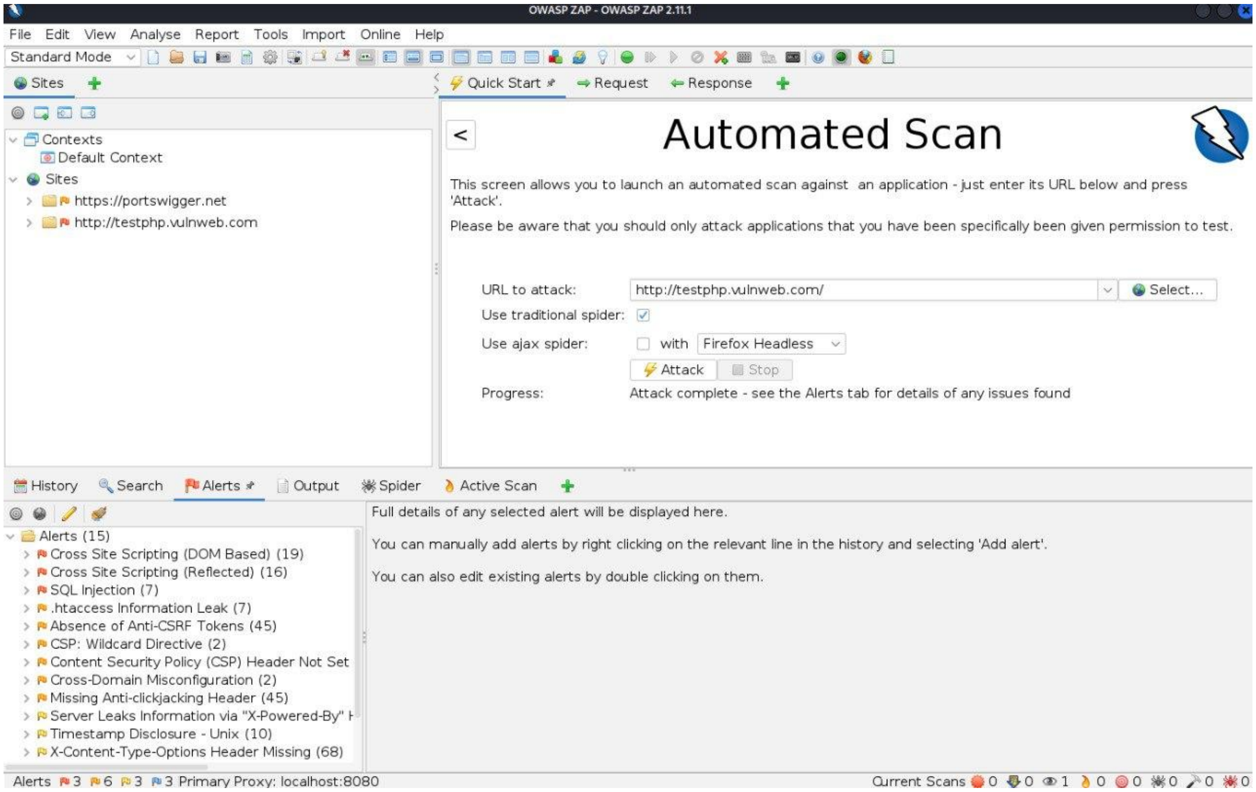
Use ajax spider: □ with Firefox Headless

Progress: Not started

OWASP ZAP will now begin spidering and testing the web application for a variety of flaws.

STEP3 :Attack Result and Alerts

When it has completed its work (this can be considerable time for large websites), you should see a screen like that below.



Contexts
Default Context
Sites
https://portswigger.net
http://testphp.vulnweb.com

Automated Scan

This screen allows you to launch an automated scan against an application - just enter its URL below and press 'Attack'.

Please be aware that you should only attack applications that you have been specifically been given permission to test.

URL to attack: http://testphp.vulnweb.com/ Select...

Use traditional spider: ☑

Use ajax spider: □ with Firefox Headless

Attack Stop

Progress: Attack complete - see the Alerts tab for details of any issues found

Alerts (15)
- Cross Site Scripting (DOM Based) (19)
- Cross Site Scripting (Reflected) (16)
- SQL Injection (7)
- .htaccess Information Leak (7)
- Absence of Anti-CSRF Tokens (45)
- CSP: Wildcard Directive (2)
- Content Security Policy (CSP) Header Not Set
- Cross-Domain Misconfiguration (2)
- Missing Anti-clickjacking Header (45)
- Server Leaks Information via "X-Powered-By" H
- Timestamp Disclosure - Unix (10)
- X-Content-Type-Options Header Missing (68)

Full details of any selected alert will be displayed here.

You can manually add alerts by right clicking on the relevant line in the history and selecting 'Add alert'.

You can also edit existing alerts by double clicking on them.

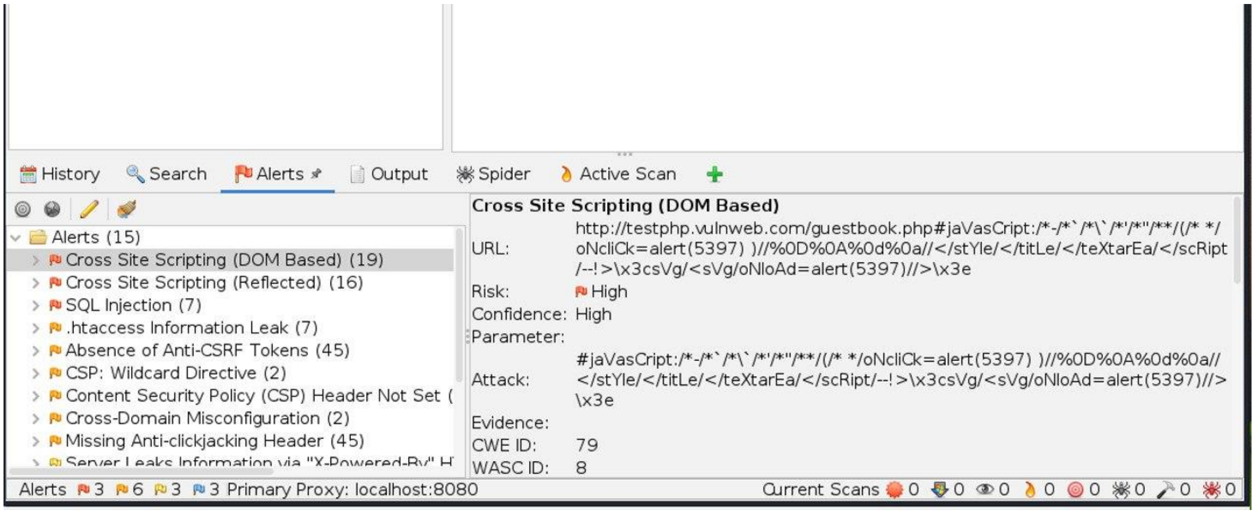
Alerts 3 6 3 3 Primary Proxy: localhost:8080 Current Scans 0 0 1 0 0 0 0 0

OWASP ZAP has provided us with 15 alerts, as you can see in the lower left window. The type of vulnerability is used to categorize these notifications. These are the ones in this case:

- Cross Site Scripting (DOM Based)
- Cross Site Scripting (Reflected)
- SQL Injection
- htaccess Information Leak
- Absence of Anti-CSRF Tokens
- CSP: Wildcard Directive
- Content Security Policy (CSP) Header Not Set
- Cross-Domain Misconfiguration
- Missing Anti-clickjacking Header
- Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)
- Timestamp Disclosure - Unix

- X-Content-Type-Options Header Missing
- Charset Mismatch (Header Versus Meta Content-Type Charset)
- Information Disclosure - Suspicious Comments
- Re-examine Cache-control Directives

A number appears next to each kind of warning, indicating the number of occurrences of that type of vulnerability. When you click the arrow next to the alert, it expands to show you every instance of the vulnerability.



```
Alerts (15)
  Cross Site Scripting (DOM Based) (19)
  Cross Site Scripting (Reflected) (16)
  SQL Injection (7)
  .htaccess Information Leak (7)
  Absence of Anti-CSRF Tokens (45)
  CSP: Wildcard Directive (2)
  Content Security Policy (CSP) Header Not Set (
  Cross-Domain Misconfiguration (2)
  Missing Anti-clickjacking Header (45)
  Server Leaks Information via "X-Powered-By" H
```

```
Cross Site Scripting (DOM Based)
            http://testphp.vulnweb.com/guestbook.php#jaVasCript:/*-/*`/*\`/*'/*"/**/(/* */
URL:        oNcliCk=alert(5397) )//%0D%0A%0d%0a//</stYle/</titLe/</teXtarEa/</scRipt
            /--!>\x3csVg/<sVg/oNloAd=alert(5397)//>\x3e
Risk:       High
Confidence: High
Parameter:
            #jaVasCript:/*-/*`/*\`/*'/*"/**/(/* */oNcliCk=alert(5397) )//%0D%0A%0d%0a//
Attack:     </stYle/</titLe/</teXtarEa/</scRipt/--!>\x3csVg/<sVg/oNloAd=alert(5397)//>
            \x3e
Evidence:
CWE ID:     79
WASC ID:    8
```

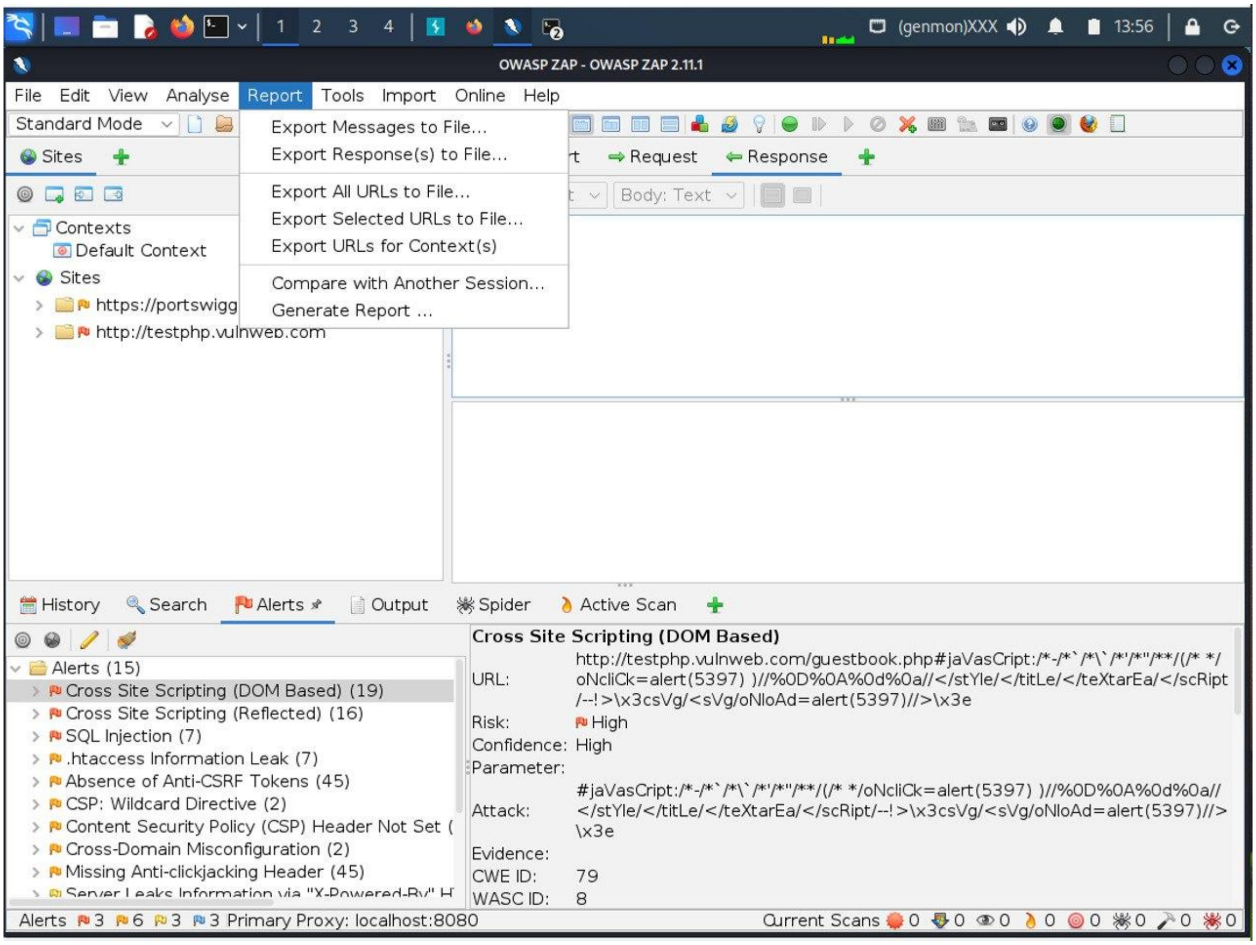
```
Alerts 3 6 3 3 Primary Proxy: localhost:8080          Current Scans 0 0 0 0 0 0 0 0
```

In the snapshot above, I initially clicked on the "Cross Site Scripting(DOM Based)" alert, which opened a window to the right with information representing the application's risk (High) and confidence level (High). The notice was then expanded to show each of the XSS flaws in this web app.

Of course, the next step is to test each of the reported vulnerabilities to see if they are valid.

STEP3 : Now generating report

Now to generate a detailed report of all the vulnerabilities that were found in the scanning. Click on the report tab in the top row, then click on "Generate report."

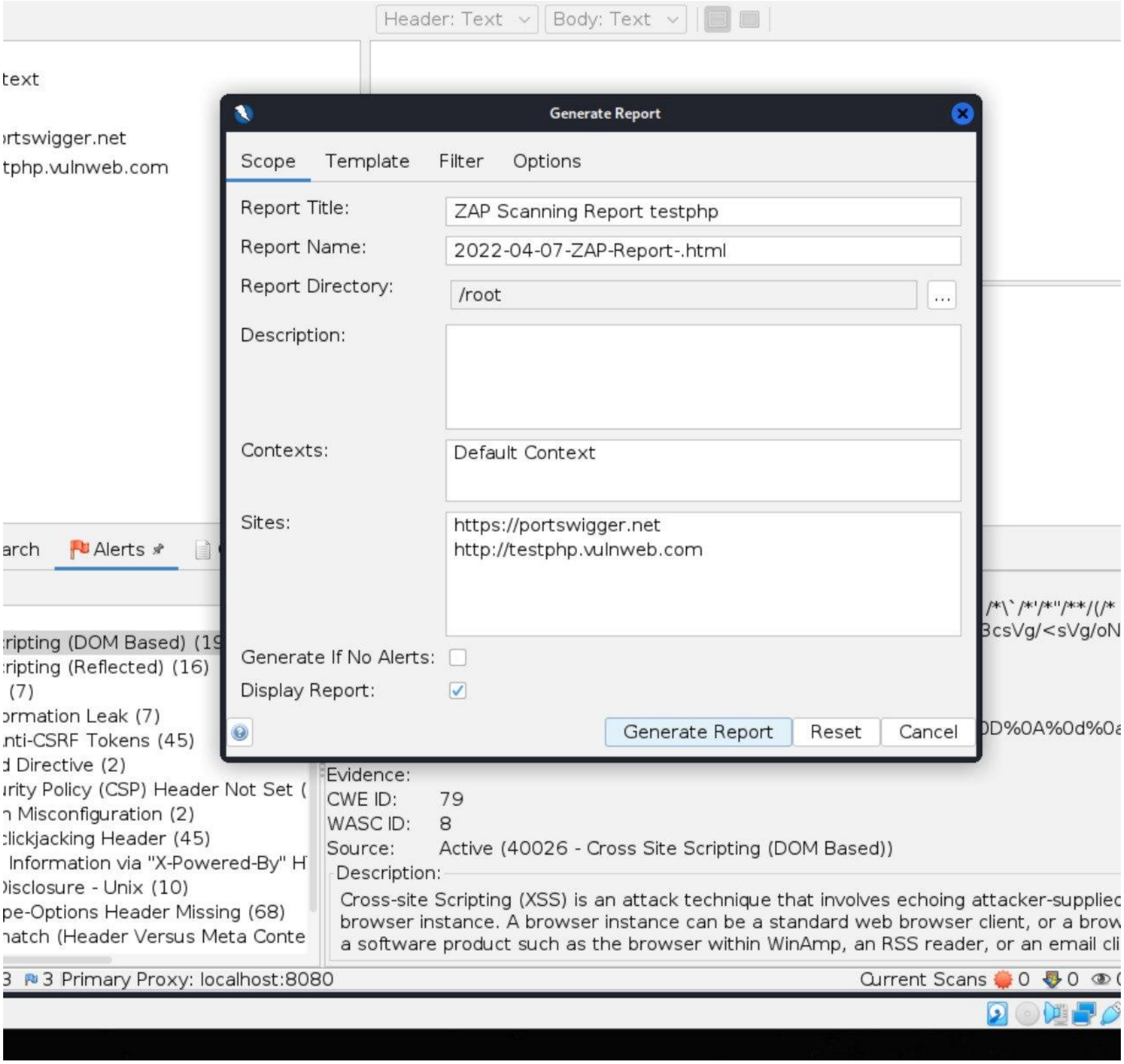

1 2 3 4
(genmon)XXX 13:56
OWASP ZAP - OWASP ZAP 2.11.1
File Edit View Analyse Report Tools Import Online Help
Standard Mode
Export Messages to File...
Export Response(s) to File...
Export All URLs to File...
Export Selected URLs to File...
Export URLs for Context(s)
Compare with Another Session...
Generate Report ...
Sites
Request
Response
Body: Text
Contexts
Default Context
Sites
https://portswigg
http://testphp.vulnweb.com
History
Search
Alerts
Output
Spider
Active Scan

```
Alerts (15)
  Cross Site Scripting (DOM Based) (19)
  Cross Site Scripting (Reflected) (16)
  SQL Injection (7)
  .htaccess Information Leak (7)
  Absence of Anti-CSRF Tokens (45)
  CSP: Wildcard Directive (2)
  Content Security Policy (CSP) Header Not Set (
  Cross-Domain Misconfiguration (2)
  Missing Anti-clickjacking Header (45)
  Server Leaks Information via "X-Powered-By" H
```

```
Cross Site Scripting (DOM Based)
URL:         http://testphp.vulnweb.com/guestbook.php#jaVasCript:/*-/*`/*\`/*'/*"/**/(/* */
             oNcliCk=alert(5397) )//%0D%0A%0d%0a//</stYle/</titLe/</teXtarEa/</scRipt
             /--!>\x3csVg/<sVg/oNloAd=alert(5397)//>\x3e
Risk:        High
Confidence:  High
Parameter:
Attack:      #jaVasCript:/*-/*`/*\`/*'/*"/**/(/* */oNcliCk=alert(5397) )//%0D%0A%0d%0a//
             </stYle/</titLe/</teXtarEa/</scRipt/--!>\x3csVg/<sVg/oNloAd=alert(5397)//>
             \x3e
Evidence:
CWE ID:      79
WASC ID:     8
```

```
Alerts 3 6 3 3 Primary Proxy: localhost:8080          Current Scans 0 0 0 0 0 0 0 0
```

Then this dialogue box shown below will pop up and, if you want, you can edit the report title, name, or context and other things as shown below.



Report Title: ZAP Scanning Report testphp
Report Name: 2022-04-07-ZAP-Report-.html
Report Directory: /root
Description:
Contexts: Default Context
Sites: https://portswigger.net
http://testphp.vulnweb.com
Generate If No Alerts: □
Display Report: ☑

text
ortswigger.net
tphp.vulnweb.com

ripting (DOM Based) (19
ripting (Reflected) (16)
(7)
ormation Leak (7)
nti-CSRF Tokens (45)
d Directive (2)
urity Policy (CSP) Header Not Set (
n Misconfiguration (2)
Clickjacking Header (45)
Information via "X-Powered-By" H
Disclosure - Unix (10)
pe-Options Header Missing (68)
natch (Header Versus Meta Conte

/*\`/*'/*"/**/(/*
3csVg/<sVg/oN
0D%0A%0d%0a

Evidence:
CWE ID: 79
WASC ID: 8
Source: Active (40026 - Cross Site Scripting (DOM Based))
Description:
Cross-site Scripting (XSS) is an attack technique that involves echoing attacker-supplied
browser instance. A browser instance can be a standard web browser client, or a brow
a software product such as the browser within WinAmp, an RSS reader, or an email cli

Then it generates a full report of all the vulnerabilities found on the website, as well as an explanation of each vulnerability and the risk level connected with it. It also includes connections to resources and explanations for all of the flaws. The report also includes recommendations for how to address the vulnerabilities and make the website more secure. The report can be produced in a variety of formats. It is generated in pdf format, which is given below.