OWASP Vulnerable Components Lab Exercise

The development of this document is/was funded by three grants from the DOD Grant Number H98230-19-1-0301

1 Overview

This Labtainer exercise explores using components with known vulnerabilities. You might have totally secured your own code, but what about the dependencies you are using? Have you checked them or just imported them into your code? There is a high chance that one or more of them are vulnerable.

2 Lab Environment

This lab runs in the Labtainer framework, available at http://my.nps.edu/web/c3o/labtainers. That site includes links to a pre-built virtual machine that has Labtainers installed, however Labtainers can be run on any Linux host that supports Docker containers.

From your labtainer-student directory start the lab using:

```
labtainer web-vulcom
```

On most Linux systems, these are links that you can right click on and select "Open Link". **If you chose to edit the lab report on a different system, you are responsible for copying the completed report back to the displayed path on your Linux system before using "stoplab" to stop the lab for the last time.**

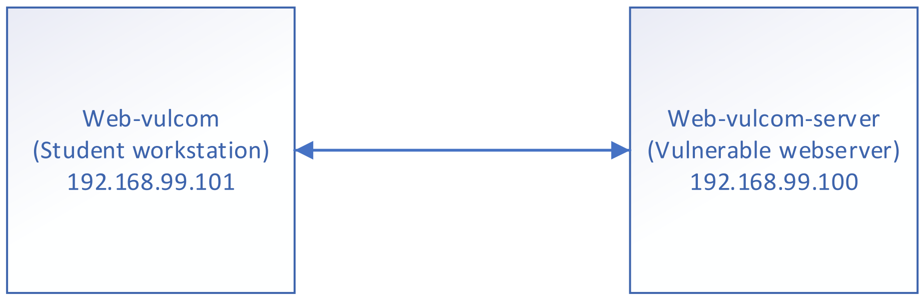
The resulting virtual terminal is connected to the student workstation; you will have OWASP ZAP and Firefox located on this workstation.

There are several accounts that are configured

Username	Password	Systems
admin@juice.org	admin123	Admin for web login
jim@juice.org	ncc-1701	web login
mc.safesearch@juice.org	Mr. N00dles	web login
Ubuntu	ubuntu	Student workstation
Ubuntu	ubuntu	vulnerable server
Ubuntu	ubuntu	Attacker workstation

3 Network Configuration

The student workstation (`web-vulcom`) is configured to have IP address `192.168.99.101` while the vulnerable webserver (`web-vulcom` -server) is `192.168.99.100;`



4 Lab Tasks

It is assumed that the student has received instruction or independent study on the basic operation of web operations.

4.1 Verify connectivity between student workstation and web server

A simple ping from the student workstation system will be sufficient.

```
 ping 192.16899.100
```

Note: to stop the ping use CTRL + C

4.2 Open Firefox and browse to the web server

At a terminal on the student workstation type:

```
firefox &
```

this will load Firefox, and type in the IP of the web server:

```
http://192.168.99.100
```

Record in Item #1 of your report why firefox might have been chosen to be the web browser used.

4.3 Open & Set up OWASP ZAP

At the terminal of the student workstation, type:

```
owasp-zap &
```

Note: if Firefox is running at the terminal and the "&" was not included then Firefox is not running in the background. Close Firefox and reopen using "Firefox &" at the terminal

OWASP ZAP Application should be open and it should be prompting the user for input.

- OWASP ZAP user input: select "yes, I want to persist this session with the name based on the current timestamp" then click start. This will open ZAP application.
- If you are prompted to "Manage Add-on" click close

4.4 Configure Firefox to use OWASP-ZAP as a Proxy

The objective of this task is to set up OWASP ZAP to be function and to allow the capture of traffic from the student workstation. Within the preference section of Firefox configure the following steps:

- In the student workstations Firefox, open "Preferences"
- In the find window type "proxy"
- In Network Proxy Setting, select "Settings"

- Select “Manual proxy configuration”
- In the HTTP Proxy section: use “127.0.0.1” and Port “8080”
- Also select “Use this proxy server for all protocols”
- Click “ok” to accept the settings

The above setting ensures that Firefox will use OWASP Zap as the proxy. Perform the following steps to ensure the Firefox is connecting and using ZAP as a proxy

- Refresh the webpage “192.168.99.100”
- A security warning stating “Your connection is not secure” will be displayed.
- This warning message must be accepted. To do that click on “Advanced”
- It will display the SSL certificate and should show a “SEC_ERROR_UNKOWN ISSUE” it is ok to use this cert, click “Add Exception”
- A confirmation window will pop up, confirm the exception by clicking the “Confirm Security Exception”

Record in Item #2 of your report why set up a proxy.

4.5 Accessing Restricted Areas

The objective of this section is to see what access is allowed and to determine even if you can access a restricted area if you will be allowed to do anything. According to OWASP WSTG-ATHZ-01, one of the first tasks when looking at a web site is to see what URLs and directories you have access to. Conducting a basic site survey using a web site crawler is the preferred method for data gathering on a web site.

Example 1 – Path Traversal
The following will allow OWASP Zap to crawl a website to determine if what paths are accessible.

- Open OWASP Zap and perform a site scan on the IP address:
  `192.168.99.100:3000`
- Under alerts you should see a section titled Path Traversal, see if you can see any paths that you would think are password protect or secured.
- Do you see any links to administrative pages?
- Please save your scan results from OWASP Zap, save it to the desktop and title it “traversal.html” if you don’t see traversal, that save the entire scan report.

Example 2 – feedback modification

- In Firefox navigate to the following URL
  `http://192.168.99.100:3000/#/administatration`
- Once you are at the administrative page are you able to remove feedback? Remove all 1-star feedback

Record in Item #3 of your report how should an administrative page be protected?

4.6 Vulnerable Components – Typosquatting

The objective of this task is to explore typosquatting. According to OWASP WSTG-ERRH-01, typically focuses on error handling, however, typosquatting infects dependencies and are often explored in error handling.

A review of dependencies is the first step in investigating typosquatting. While reviewing the directory ftp, certain files were located and will be reviewed.

- In Firefox, if you inspect the page you may view the dependencies, you may also view them in OWASP ZAP.
- Explore each of the files in the ftp directory and see what dependencies may be found.
- The package.json.bak contains not only runtime dependencies but also development dependencies under the dependencies section. Review and research the "epilogue-js" and compare it to "epilogue"
- Checking the dependencies using the following site: https://www.npmjs.com/package/
- Define what typosquatting is and how can we be sure that the above dependencies may have been comprised.
- Next we will view frontend based typosquatting by exploring the following URL:
- https://192.168.9.100:3000/3rdpartylicenses.txt
- Viewing the list of modules review each of them for known typosquatting vulnerabilities. Explore at minimum the dependency known as ng2-bar-rating.
- Checking the dependencies using the following site: https://www.npmjs.com/package/

Record in Item #4 of your report what is typosquating?

Record in Item #5 of your report what are decencies and how do they differ from packages used in supporting a web service?

4.7 Vulnerable Libraries

The objective of this task is to explore Libraries that may be vulnerable. According to OWASP WSTG-ERRH-01, typically focuses on error handling, however, supply chain attacks often explore error handling and other dependencies. Also, OWASP WSTG-CLNT-12, outlines what browsers provide client-side storage mechanisms for developers to store and retrieve data that maybe exploited.

A review of dependencies is the first step in investigating supply chain attacks. While reviewing the directory ftp, certain files were located and will be reviewed.

- In Firefox, if you inspect the page you may view the dependencies, you may also view them in OWASP ZAP.
- Explore each of the files in the ftp directory and see what dependencies may be found.
- The package.json.bak contains not only runtime dependencies but also development dependencies under the dependencies section. Research both "sanitize-html" and "express-jwt"
- Checking the dependencies in package.json.bak for known vulnerabilities online will give you the following information:
    - sanitize-html: Sanitization of HTML strings is not applied recursively to input, allowing an attacker to potentially inject script and other markup
    - express-jwt: Inherits an authentication bypass and other vulnerabilities from its dependencies

Record in Item #6 of your report what is a library when it comes to web services?

Record in Item #7 of your report how does a library become vulnerable?

4.8 Supply Chain Attacks

The objective of this task is to explore what are known as supply chain attacks. According to OWASP WSTG-ERRH-01, typically focuses on error handling, however, supply chain attacks often explore error handling and other dependencies.

A review of dependencies is the first step in investigating supply chain attacks. While reviewing the directory ftp, certain files were located and will be reviewed.

- In Firefox, if you inspect the page you may view the dependencies, you may also view them in OWASP ZAP.
- Explore each of the files in the ftp directory and see what dependencies may be found.
- The package.json.bak contains not only runtime dependencies but also development dependencies under the devDependencies section.
- Go through the list of devDependencies and perform research on vulnerabilities in them which would allow a Software Supply Chain Attack.
- What vulnerability is known to be vulnerable to a Supply Chain Attack?
- For the "eslint-scope" module you will learn about one such incident exactly in the pinned version 3.7.2

Record in Item #8 of your report what is a supply chain attack?

4.9 Forging Tokens

The objective of this task is to forge a token. According to OWASP WSTG-BUSL-02, forging requests is a method that attackers use to circumvent the front-end GUI application to directly submit information for back end processing. The goal of the attacker is to send HTTP POST/GET requests through an intercepting proxy with data values that is not supported, guarded against or expected by the applications business logic. Some examples of forged requests include exploiting guessable or predictable parameters or expose "hidden" features and functionality such as enabling debugging or presenting special screens or windows that are very useful during development but may leak information or bypass the business logic. Vulnerabilities related to the ability to forge requests is unique to each application and different from business logic data validation in that it focuses is on breaking the business logic workflow.

To forge an unsigned token, complete the following steps.

- Log in as the Jim user to receive a valid JWT in the Authorization header. The authorization header details can be found in OWASP ZAP.
- Copy the JWT (this means everything after Bearer in the Authorization header) and decode it. It uses a base64 encoding.
- Under the payload property, change the email attribute in the JSON to jwtn3d@juice.org.
- Change the value of the alg property in the header part from HS256 to none.
- Encode the header to base64url. Similarly, encode the payload to base64url. base64url makes it URL safe, a regular Base64 encode doesn't work!
- Join the two strings obtained above with a . (dot symbol) and add a . at the end of the obtained string. So, effectively it becomes base64url(header).base64url(payload).
- Change the Authorization header of a subsequent request to the retrieved JWT (prefixed with Bearer as before) and submit the request.

Record in Item #9 of your report how would one prevent a token from being forged?

Record in Item #10 of your report how does a session token differ from session cookies?

4.10 Overwriting Server Files

The objective of this task is to see if files can be manipulated via an upload and modifying files via a traversal attack. According to the OWASP WSTG-ATHZ-01, a path traversal attack (also known as directory traversal) aims to access files and directories that are stored outside the web root folder. By manipulating variables that reference files with “dot-dot-slash (../)” sequences and its variations or by using absolute file paths, it may be possible to access arbitrary files and directories stored on file system including application source code or configuration and critical system files. It should be noted that access to files is limited by system operational access control (such as in the case of locked or in-use files on the Microsoft Windows operating system). This attack is also known as “dot-dot-slash”, “directory traversal”, “directory climbing” and “backtracking”. The dot-dot-slash attacks allow for modifying of files when files are allowed to be uploaded.

You have learned that the site is vulnerable to ZIP base attacks via the complaint section.

- To take advantage of this research what are the common vulnerabilities. Play close attention to the zip vulnerabilities around “Zip Slip.”
- The file being replaced is located in the ftp directory. When a file is uploaded to the complaint section, there is no information in where the file is uploaded. Do some research on types of attacks that allow for directory traversal such as the approach known as iterative directory traversal
- On the student workstation prepare a zip file using “zip exploit.zip ../../ftp/legal.md”
- Log in as Jim and navigate to Compliant section.
- File a compliant using the exploit.zip and leave a comment.
- View the legal.md up the ftp directory and see if it was updated.

Record in Item #11 of your report what is a ZIP based attack?

4.11 Generate Report

In OWASP ZAP once the tester has found the different vulnerabilities, in the ZAP application under Report on the top and save a HTML report. Save the report to the home directory of the web-inject workstation, call the file “`report_zap`”

Record in Item #12 of your report any interesting finding of the report. Find three areas of interest and explain why they were important and how they may have an impact on the security of this website.

4.12 Review Journal output

This task allows the user to see output that the server would be generating when certain attacks or exploits have run against them. The learner will review journal output on the server.

- On the shell of the web server type the following command:
    - Sudo journalctl -u juice-shop
- Review the output and space bar until the end, record anything of note and if there are any possible exploits that had ran. Support your claim with output from the journal. Record this at the end of your report under question 13.

5 Stop the Labtainer

When the lab is completed, or you'd like to stop working for a while, run

```
stoplab web-vulcom
```

from the host Labtainer working directory. You can always restart the Labtainer to continue your work. When the Labtainer is stopped, a zip file is created and copied to a location displayed by the stoplab command. When the lab is completed, send that zip file to the instructor.

References