OWASP Broken Authentication and Session Management 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 broken authentication mechanism. This lab covers how to reset password using a GET request, how to bypass multifactor authentication, and how decode session tokens.

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-brokenauth
```

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.**

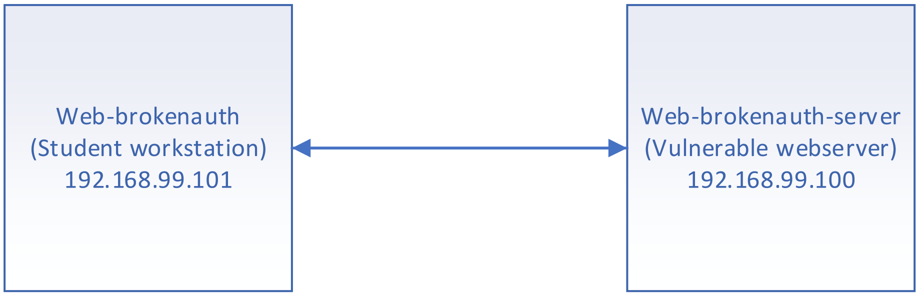
The resulting virtual terminal is connected to the student workstation; you will have OWASP ZAP and Firefox located on this workstation.

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-brokenauth`) is configured to have IP address `192.168.99.101` while the vulnerable webserver (`web-brokenauth` -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 192.168.99.100
```

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`
- 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 traversal is not an option, save the entire 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 what is path traversal and how can it be exploited?

4.6 Review and decode Token. Session vs token content

The objective of this task is to review user sessions and tokens. according to OWASP WSTG-SESS-04, Session Tokens (Cookie, SessionID, Hidden Field), if exposed, will usually enable an attacker to impersonate a victim and access the application illegitimately. It is important that they are protected from eavesdropping at all times, particularly whilst in transit between the client browser and the application servers.

While testing for GET & POST vulnerabilities, its important to note that in general, GET requests should not be used, as the Session ID may be exposed in Proxy or Firewall logs. They are also far more easily manipulated than other types of transport, although it should be noted that almost any mechanism can be manipulated by the client with the right tools.

To start testing us the following steps

- In Firefox, log in to Jims account
- Find his user token by using this command in the search box, if the token is not displayed than complete the chart below.

```
        <script>alert(document.cookie);</script>
```

- Decode his token by opening a new tab and type in the following URL: https://jwt.io/
- In the Encoded section paste the token, you should see the username/email and a hashed password.
- Break the hashed password using google and/or a MD5 hashing tool.
- Complete the chart below

Token	Email	Hashed Password	Cracked password
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJtZXNzYWdlIjoiSldUIFJ1bGVzISIsImlhdCI6MTQ1OTQ0ODExOSwiZXhwIjoxNDU5NDU0NTE5fQ.-yIVBD5b73C75osbmwwshQNRC7frWUYrqaTjTpza2y4			
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c			
eyJasdXAiOiJKV1QiLCJhbdsfghfJIUzI1NiJ9.eyJtZXNzYWdlIjoiSldUIFJdfgh2341I6MTQ1OTQ0ODExOSwiZXhwIjoxNDU5NDU0NTE5fQ.-yIVBD5b73C75osbmwwshQNRC7frWUYrqaTjTpza2y4			

Record in Item #4 of your report fill out the chart above and put your finding in the report.

Record in Item #5 of your report what type of hashing was used?

4.7 Login Password Cracking

The most prevalent and most easily administered authentication mechanism is a static password. The password represents the keys to the kingdom but is often subverted by users in the name of usability. In each of the recent high-profile hacks that have revealed user credentials. According to OWASP WSTG-ATHN-07, the ability to determine the resistance of the application against brute force password guessing using available password dictionaries by evaluating the length, complexity, reuse and aging requirements of passwords. Being able to take a hash and run it through a rainbow table or even a simple google search.

Example 1 – User login

- In Firefox, review all user credentials by performing an SQL injection. From a web browser using the Juice shop search section type in the following command:

```
http://192.168.99.100:3000/rest/products/search?q=%27))%20union%20select%20null,id,email,password,totpsecret,null,null,null,null%20from%20users--
```

- The above SQL statement should list all users. The responses to the call should look like as follows:

```
id              null
name            1
description     "admin@juice.org"
price           "0192023a7bbd73250516f069df18b500"
deluxePrice     ""
image           null
createdAt       null
updatedAt       null
deletedAt       null
```

- Notice that the price is a hashed value. This is actually the password! We can find the real password associated with the hash value.
- Fill in the chart below:

ID	User Emails	Hash password	Cracked password if possible
1			
2			
3			
4			
5			
6			

Record in Item #6 of your report fill out the chart above and record your results in the report.

Record in Item #7 of your report what were the total number of users found?

4.8 Modifying and/or Updating passwords

The objective of this task is to review password modification and change/updating requests. According to OWASP WSTG-ATHN-09, the password change and reset function of an application is a self-service password change or reset mechanism for users. This self-service mechanism allows users to quickly change or reset their password without an administrator intervening. When passwords are changed they are typically changed within the application. When passwords are reset they are either rendered within the application or emailed to the user. This may indicate that the passwords are stored in plain text or in a decryptable format. The main object while testing is to demine the resistance of the application to subversion of the account change process allowing someone to change the password of an account. Testing procedure according to OWASP is to review information such as: what information is needed and how do resets occur. A common mistake is watching a present change their password while giving a presentation. To get a better understanding of how to test for and update weak password management review the two examples below.

Example 1. – Updating a password using HTTP GET

From the student workstation

- In Firefox try logging into benders account with the username bender@juice.org' and any password.
- Attempt to change Bender's password to abcde with the "existing" password of abcde.
- In OWASP ZAP inspect the history and look at the GET request for a password change. There should be a GET (?!).
- Try modifying a HTTP GET request to update his password. Type the following
  ```
  http://192.168.99..100:3000/rest/user/change-password?current=abcde&new=PASSWORD&repeat=PASSWORD
  ```
- A return of a 401 unauthorized, with the message 'Current password is not correct', but that was expected.
- Remove the section of code without quote "`current=abcde`" and see if the password can get set that way.
- Try to repeat the steps using the Admin user and updating the passwords to P@ssword#1

Example 2 – Watching a user type in information during presentation

- Watch BeNeLux Day 2018: Juice Shop: OWASP's Most Broken Flagship - Björn Kimminich
- This conference talk recording immediately dives into a demo of the Juice Shop application in which Bjoern starts registering a new account 3:59 into the video (`https://youtu.be/Lu0-kDdtVf4?t=239`)
- Bjoern picks *Name of your favorite pet?* as his security question and - live on camera - answers it truthfully with "Zaya", the name of his family's adorable three-legged cat.
- Visit `https://192.168.99..100:3000/#/forgot-password` and provide bjoern@owasp.org as your *Email*.
- In the subsequently appearing form, provide Zaya as *Name of your favorite pet?*
- Then type any *New Password* and matching *Repeat New Password*
- Click *Change* to solve this challenge.

Record in Item #8 of your report why is using GET to push passwords a security risk?

4.9 GDPR User Violation

The objective of this task is to see if the web site follows GDPR when it comes to user rights to be erased. According to OWASP WSTG-ATHN-04, authentication is the process of attempting to verify the digital identity of the sender of a communication. Though when necessary a user has the right to remove themselves from a system when it's a compliance requirement. Thus, being able to verify that a user who has been removed has truly been removed.

From the student workstation

- In Firefox Reviewing all user credentials by performing an SQL injection. From a web browser using the Juice shop search section type in the following command:

- The below SQL statement should list all users. The user chris.pike@juice.org should stand out as this is a deleted user. http://192.168.99.100:3000/rest/products/search?q=%27))%20union%20select%20null,id,email,password,totpsecret,null,null,null,null%20from%20users--
- Log in as Chris using the following username without the quote “chris.pike@juice.org’--” and select any password. Does the log in work? If so log out.
- If the first method for login worked, then the second login will be using the username without quote “\' or deletedAt IS NOT NULL--” and select any password. Does the log in work? Chris is the only user who was deleted.
- The above command will use “deletedAt” to determin delted user, and the Not Null only shows users who were soft deleted.

Record in Item #9 of your report why is verification that users are deleted an issue when it comes to GDPR and data retention?

4.10 Bypassing Multifactor Authentication

The objective of this lab is to understand how two-factors authentication works, what it looks like when a user has this feature enabled, and how to work on bypassing it when is possible. According to OWASP WSTG-ATHN-10, even if the primary authentication mechanisms do not include any vulnerabilities, it may be that vulnerabilities exist in alternative legitimate authentication user channels for the same user accounts. Tests should be undertaken to identify alternative channels and, subject to test scoping, identify vulnerabilities. Verifying even multifactor authentication is function and without flaws is also necessary. the following steps will guide you through how to verify if there are vulnerabilities.

From the student workstation

- In Firefox we need to ensure that there is an Authenticator present. In Firefox, go to Tools->Add-ons. Search for Authenticator click the result for the Authenticator by mymindstorm, and click “Add to Firefox”. It will appear as a QR code icon the top right corner of your browser.
- Reviewing all user credentials by performing an SQL injection. From a web browser using the Juice shop search section type in the following command:

  http://192.168.99.100:3000/rest/products/search?q=%27))%20union%20select%20null,id,email,password,totpsecret,null,null,null,null%20from%20users--

- The above SQL statement should list all users. Looking at response the user wurstbrot has a unique entry that is shown below:

  ```
  IFTXE3SPOEYVURT2MRYGI52TKJ4HC3KH
  ```

- Open the Authenticator, press edit, press the + symbol -> Manual Entry -> Anything for Issuer, and the unique entry above for the secret. Hit okay.
- Login using SQL injection for wurstbrot by typing the command for the user name without the quote “wurstbrot@juice.org’--" and type anything for the password.

- Go back to the Authenticator button and copy the code being shown into Juice Shop.

Record in Item #10 of your report why is dual factor authentication important in todays world if it can be bypassed?

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 #11 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 12.

5 Stop the Labtainer

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

```
stoplab web-brokenauth
```

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