



Web Application Security Assessment

An assessment of the security of the Hacklab
Pizza's Web Application

Adam Board

CMP319: Ethical Hacking 2

BSc Ethical Hacking Year 3

2022/23

Note that Information contained in this document is for educational purposes.

Abstract

Web applications in the modern day are used in every aspect of life, including but not limited to, online banking, social media, and e-commerce. These websites hold billions of users' sensitive data, dramatically increasing the risk factor for these websites. With this increased risk, the organisations or individuals that host the web applications will need increased security to match the risk. Should these web applications become compromised by a breach, the consequences would be tremendous, Sensitive information leaked, web application being defaced, losing up to £20 million or 4% of annual global turnover - whichever ends up being a higher value, or the possibility of the business having to shut down; All of these are a pragmatic example of what could happen a business should their website be breached. Since this is the case, Mr Rick Astley has requested a professional in the cyber security industry to conduct a full security assessment of their web application and produce a report based on the assessment.

This report incorporates a complete web application penetration test, using the OWASP Web Application Testing Methodology. The aforementioned methodology is in use to ensure that a thorough test of Mr Astley's food ordering web application is completed to mitigate any vulnerabilities that are currently present in the previously mentioned web application.

During the security assessment, Mr Astley's food ordering web application was found to contain a myriad of vulnerabilities was found, ranging from SQL Injection, Stored Cross Site Scripting, lack of cryptography, and weak user passwords. The list of vulnerabilities here is not exhaustive. Using these vulnerabilities, the tester was able to breach user accounts and an admin account, which gave full access to admin-only webpages. The report given to Mr Astley will contain a summary of recommendations, which should be followed by a developer to secure the website from malicious users attempting to gain unauthorised access to administrative sections of the website.

Contents

1	Introduction	1
1.1	Background	1
1.2	Aims	1
2	Procedure and Results	3
2.1	Overview of Procedure	3
2.2	Information Gathering	4
2.2.1	Fingerprint Web Server	4
2.2.2	Review Webserver Metafiles for Information Leakage	5
2.2.3	Enumerate Applications on Webserver	5
2.2.4	Review Webpage Comments and Metadata for Information leakage	6
2.2.5	Identify Application Entry Points	6
2.2.6	Map Execution Paths Through Application	8
2.2.7	Fingerprint Web Application Framework	9
2.3	Configuration and Deployment Management Testing	11
2.3.1	Test Application Platform Configuration	11
2.3.2	Enumerate Infrastructure and Application Admin Interfaces	17
2.3.3	Test HTTP Strict Transport Security	18
2.4	Identity Management Testing	18
2.4.1	Test Role Definitions	18
2.4.2	Test User Registration Process	19
2.4.3	Testing for Weak or Unenforced	21
2.5	Authentication testing	22
2.5.1	Testing for Credentials Transported over Unencrypted Channels	22
2.5.2	Testing for Default Credentials	23
2.5.3	Testing for Weak Lock Out Mechanism	24
2.5.4	Test Remember Password Functionality	25
2.5.5	Testing for Weak Password Policy	25
2.5.6	Testing for Weak Password Change or Reset Functionalities	25
2.6	Authorisation Testing	27
2.6.1	Testing Directory Traversal File Include	27
2.7	Session Management Testing	28

2.7.1	Testing for Session Management Schema.....	28
2.7.2	Testing for Cookies Attributes	28
2.7.3	Testing for Session Fixation	29
2.7.4	Testing for Logout Functionality	30
2.8	Input Validation testing	31
2.8.1	Testing for Reflected Cross Site Scripting	31
2.8.2	Testing for Stored Cross Site Scripting.....	31
2.8.3	Testing for SQL Injection.....	32
2.9	Error Handling.....	32
2.9.1	Testing for Improper Error Handling	32
2.10	Cryptography	33
2.10.1	Testing for Weak Transport Layer Security	33
2.11	Business Logic Testing.....	34
2.11.1	Test Ability to Forge Requests	34
2.11.2	Test Number of Times a Function Can Be Used Limits.....	34
2.11.3	Test Upload of Unexpected File Types	34
3	Discussion	35
3.1	Discovered Vulnerabilities and Countermeasures	35
3.1.1	Robots.txt Vulnerability	35
3.1.2	Local File Inclusion Vulnerability	35
3.1.3	Reversible Cookie Vulnerability	35
3.1.4	Cookie Attributes Vulnerability	35
3.1.5	Directory Browsing Vulnerability	36
3.1.6	Unlimited Login Attempts Vulnerability	36
3.1.7	No HTTPS Vulnerability	36
3.1.8	PHP Information Disclosure Vulnerability	36
3.1.9	Cross Site Request Forgery (CSRF) Vulnerability	36
3.1.10	SQL Injection Vulnerability	37
3.1.11	User Enumeration Vulnerability	37
3.1.12	Brute-Forceable Admin Password	37
3.2	Generic Issues	37
3.2.1	X-Powered-By Header.....	37
3.2.2	Clickjacking.....	38

3.2.3	X-XSS-Protection Header	38
3.2.4	X-Content-Type-Options Header	38
3.2.5	Shellshock Vulnerability	38
3.2.6	HTTP TRACE	38
3.2.7	Mod_negotiation	38
3.2.8	PHPMyAdmin	38
3.3	General Discussion	39
3.4	Future Work	39
4	References	40
Appendices part 1		41
Appendix A – Omitted Methodology		41
Appendix B – Data and files within the website		41
4.1.1	URLS and Files Section	41
Appendix C – Console Output of tools		49
4.1.2	Section 1 – Nikto Output	49
4.1.3	Section 2 – Dirb Output	51
Appendix D – GUI Tool Output		55
4.1.4	OWASP Zap Scan Report Output	55

1 INTRODUCTION

1.1 BACKGROUND

Over the past decade, the number of internet users has more than doubled from 2.18 billion users at the beginning of 2012 to 4.96 billion at the beginning of 2022 (datareportal, 2022). With so many users on the internet, businesses require a significant presence on the internet to maximise sales to be successful. However, with the increase in users on the internet, there will also be an increase in cybercrime and data breaches.

The statistics for cyber-attacks in 2022 show that 30,000 websites are hacked daily and every 39 seconds a new cyber-attack will happen which is about 2,244 different attacks a day. In 2021 alone, there were 22 billion breached records. Finally, 64% of companies worldwide had experienced some form of cyber-attack. (techjury.net, 2022)(zippia, October 2022)

These statistics show how important it is for any organisation to have proper measures in place to protect against cyber-attacks. For organisations that don't take cyber-security seriously, it will eventually lead to a data breach that both the organisation and the customers will suffer from. This can include leaked users' data such as passwords, emails, and payment details. Because of the data leaks, the organisation will have to pay a huge fine in accordance with the General Data Protection Act (GDPR) which could potentially affect the day-to-day running of the organisation or in the worst-case scenario, the organisation will have to shut down.

A vital step to avoid these data breaches and fines is preparation. Therefore, Mr Astley has requested a penetration tester to conduct a penetration test of the organisation's website and create a report based on the test's findings. The report will also discuss methods for fixing security issues that are found. The description given to the penetration tester describes the website and "buggy but mostly functional" and was developed externally then given to Mr Astley. The client, Mr Rick Astley has given a virtual machine which replicates his website and login credentials to test, so the day-to-day running of the website is not disrupted for customers because of the penetration test.

1.2 AIMS

The penetration test aims to discover and evaluate all vulnerabilities found within the web application and format the found vulnerabilities into a clear and concise report for the client. A penetration test is usually conducted in the mindset of a hacker, or malicious actor, which attempts to make the scenario as realistic as possible to simulate the same process that would occur with a genuine cyber-attack. This means the tester will seek to gain privilege escalation on the web application, aim to obtain as much information regarding users as possible, and finally, gain complete control in any way possible, deface the website, or stop typical usage of the web application. To produce the report, the penetration tester uses

an industry-standard methodology, which uses a multitude of tools to complete specific objectives in a determined order to ensure the testing of the web application is as thorough as possible. More information regarding this can be found in the Overview and Procedure section. This methodology was used to maximise the number of vulnerabilities found, and mitigation recommendations can be given to the client regarding the vulnerabilities.

2 PROCEDURE AND RESULTS

2.1 OVERVIEW OF PROCEDURE

For this assessment, the tester made use of the OWASP Web Application Security Testing methodology version 4.2 (OWASP, no date). This methodology was the best choice as it is thorough and in a set of sequential steps to follow, furthermore this was created by a well-regarded and certified organisation. The currently released stable version was used so to ensure no errors from changes to the methodology can occur.

This methodology is comprised of these sequential steps:

1. Information Gathering
2. Configuration and Deployment Management Testing
3. Identity Management Testing
4. Authentication Testing
5. Authorisation Testing
6. Session Management Testing
7. Input Validation Testing
8. Error Handling
9. Cryptography
10. Business Logic Testing
11. Client-side Testing

Due to the scope of the testing, sections of the OWASP methodology had been excluded and since there is many subsections within the methodology, the omitted sections have been listed within Appendix A of this report under the heading, "Omitted Methodology". One full section that was completely omitted was Client-side testing as any pertinent testing included in this section was examined somewhere else.

The tester made use of their own tools which includes a Kali Linux machine virtual machine with many of the preinstalled tools required to achieve a successful test, Cyber chef, and OWASP tools including Mantra and Zap.

2.2 INFORMATION GATHERING

2.2.1 Fingerprint Web Server

“Fingerprinting” is described as determining the software and version of the workstation that the target is operating within. This is completed by running various tools that scans the server automatically, these tools include Nmap, and Nikto.

For this security assessment the penetration tester chose to use the industry standard tool, nmap, which is primarily used for mapping networks. The tester used the -sV flag to determine which services were operating on the target workstation. This can be seen in figure 1.

```
(kali㉿kali)-[~]
$ nmap -sV 192.168.1.10
Starting Nmap 7.93 ( https://nmap.org ) at 2022-12-06 09:55 EST
Nmap scan report for 192.168.1.10
Host is up (0.0013s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      ProFTPD 1.3.4a
80/tcp    open  http     Apache httpd 2.4.3 ((Unix) PHP/5.4.7)
3306/tcp  open  mysql    MySQL (unauthorized)
Service Info: OS: Unix

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 15.73 seconds
```

Figure 1 Nmap output for 192.168.1.10

Figure 1 shows that the web server uses a Unix-based operating system with three different services running.

- File Transfer Protocol (FTP) on the port 21 using ProFTPD version 1.3.4a.
- A website is open on port 80 with Hypertext Transfer Protocol, using Apache version 2.4.3 with PHP.
- On port 3306, MySQL is running.

The information gathered from this scan reveals various options for exploitation in the later stages, especially the MySQL service as this could be vulnerable to SQL injection if the service is poorly configured. Testing for this vulnerability is covered in section 2.8. Another service that was of interest is HTTP, as this means the site does not have encryption on the traffic being sent between the server and the client. This can lead to traffic being intercepted. This is covered in section 2.3. Finally, the version of each service could have specific vulnerabilities based on that exact version of the service.

2.2.2 Review Webserver Metafiles for Information Leakage

This section is regarding meta files. Metafiles are files that give information related to the site itself, A common metafile is “robots.txt” file, which lists the files and directories that can be accessed by search engine crawlers through indexing. This is accessible on the client’s website by inputting “/robots.txt” at the end of the URL. This is shown in figure 2.

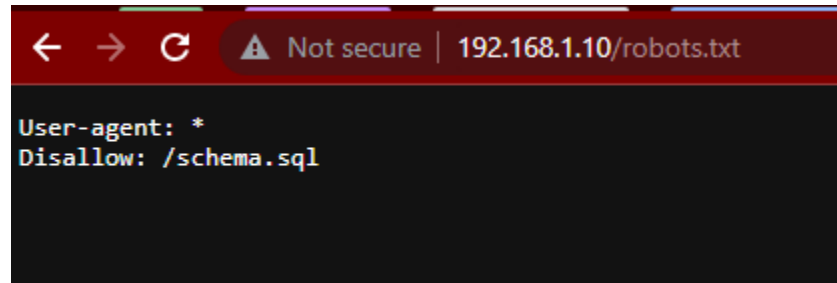


Figure 2 Browsing to /robots.txt

Figure 2 shows a single webpage that is disallowed for indexing, which is specified as “schema.sql”. appending this to the end of the URL reveals a file that is filled with plain text relating to the database schema of the website. The schema is the group of objects that are related with a database. The information within the file can be found in Appendix B.

2.2.3 Enumerate Applications on Webserver

Enumerating Applications is carried out in a similar method to the fingerprinting section above. Using the tool Nmap, the target web server is scanned to detect any applications on ports that are open outside the standard port range. The scan results can be seen in figure 3.

```
(kali@kali)-[~]
$ nmap -sV -Pn -sT -p- 192.168.1.10
Starting Nmap 7.93 ( https://nmap.org ) at 2022-12-06 11:35 EST
Stats: 0:00:18 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 11.00% done; ETC: 11:38 (0:02:34 remaining)
Stats: 0:00:52 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 34.81% done; ETC: 11:38 (0:01:39 remaining)
Nmap scan report for 192.168.1.10
Host is up (0.00081s latency).
Not shown: 65532 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      ProFTPD 1.3.4a
80/tcp    open  http     Apache httpd 2.4.3 ((Unix) PHP/5.4.7)
3306/tcp  open  mysql    MySQL (unauthorized)
Service Info: OS: Unix

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 131.20 seconds
```

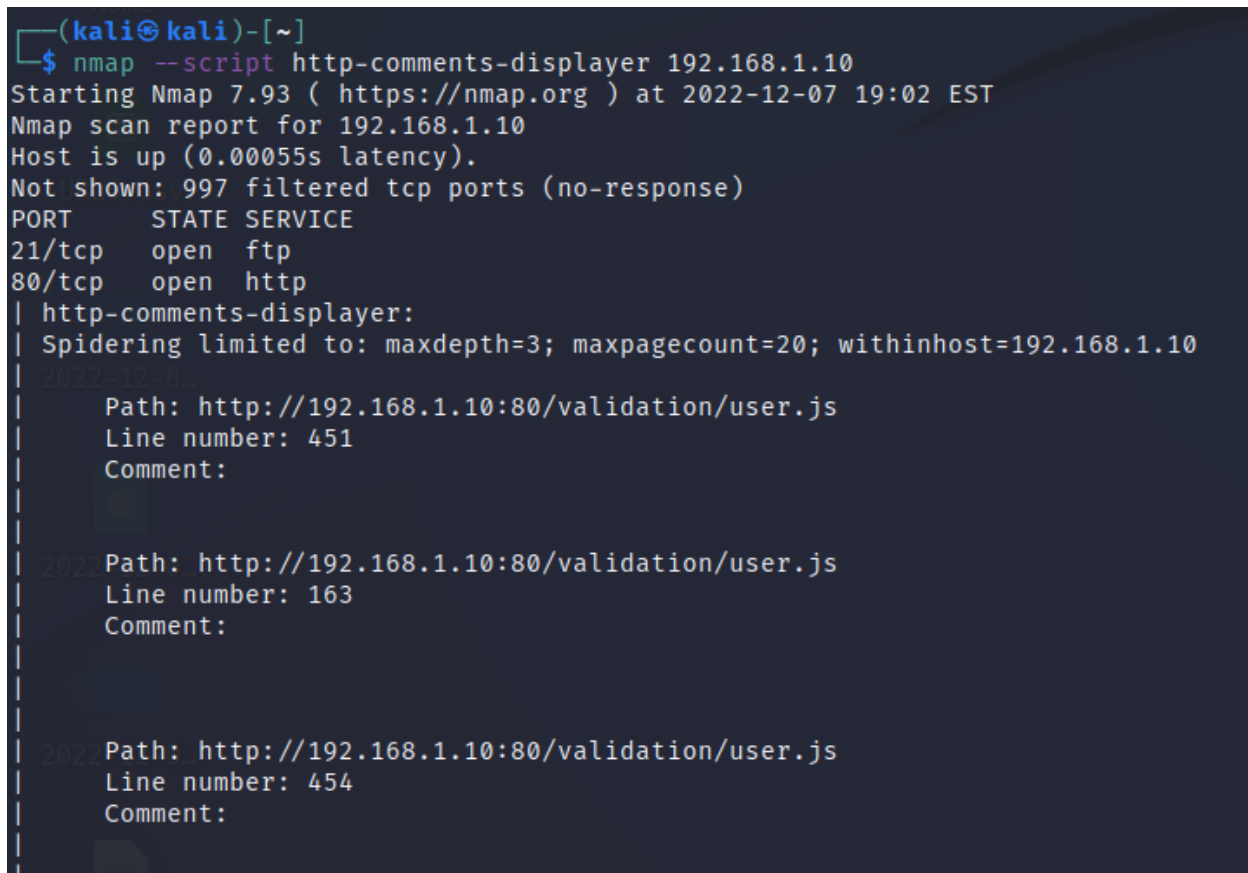
Figure 3 Output of nmap scan for all ports

Figure 3 shows that no new services were discovered after expanding the scope of the scan. Therefore, the penetration tester determined these services are the only ones that are running on an external port on the target web server.

2.2.4 Review Webpage Comments and Metadata for Information leakage

When producing a website, developers typically add comments and metadata on their project to provide guidance for other developers assisting in the creation of the web application. However, the web developers occasionally forget to remove these comments and metadata when the project is finished and this can provide explicit information regarding the processes of the web application, which could lead to security issues. Finding the comments and metadata is the point of this section.

Searching entirely manually would be inefficient and could lead to possible comments being missed, so the tester has used the Nmap script “http-comments-displayer” to search for comments and the tester searched for metadata manually throughout the web application. Output of Nmap script is shown in figure 4.



```
(kali㉿kali)-[~]
$ nmap --script http-comments-displayer 192.168.1.10
Starting Nmap 7.93 ( https://nmap.org ) at 2022-12-07 19:02 EST
Nmap scan report for 192.168.1.10
Host is up (0.00055s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT      STATE SERVICE
21/tcp    open  ftp
80/tcp    open  http
| http-comments-displayer:
| Spidering limited to: maxdepth=3; maxpagecount=20; withinhost=192.168.1.10
| 2022-12-07
|   Path: http://192.168.1.10:80/validation/user.js
|   Line number: 451
|   Comment:
|
| 2022-12-07 Path: http://192.168.1.10:80/validation/user.js
|   Line number: 163
|   Comment:
|
| 2022-12-07 Path: http://192.168.1.10:80/validation/user.js
|   Line number: 454
|   Comment:
```

Figure 4 Output of http-comments-displayer

After thoroughly searching reviewing the output of the Nmap scan, the tester found comments within the JavaScript validation file called “user.js”. the comments explain what each function’s purpose is and where the variables come from.

2.2.5 Identify Application Entry Points

This section is related to finding the various entry points into the application, for example, login prompts, registration pages, etc. Essentially any place where user data can be entered. The tester has conducted a manual search for the different entry points within the website, which revealed these different entry points:

- Pre-Login
 - Login prompt
 - Email
 - Password
 - Register Prompt
 - FirstName
 - LastName
 - Email
 - Password
 - Confirm Password
 - Security Question
 - Security Answer
 - Admin login page
- Post-Login
 - Commenting on Ratings
 - Comment Box
 - Change your Password
 - Old Password
 - New Password
 - Confirm New Password
 - Delivery/Bill Address
 - First Name
 - Street Address
 - P.O. Box
 - City
 - Mobile No.
 - Landline No.
 - Change Profile Pic
 - Upload Profile Picture

Whilst enumerating the various entry points, the tester used the network feature in google chrome's inspect feature to determine the use of GET and POST requests being sent to the server. Both GET and POST are vital to the transmission of data throughout the web, as these requests are used to transmit data from client host machines to server host machines and vice versa. However, when sending a request to change password through POST it has the user account ID in the URL, this can be seen in figure 5 and 6 with a new account that had been created to confirm this.

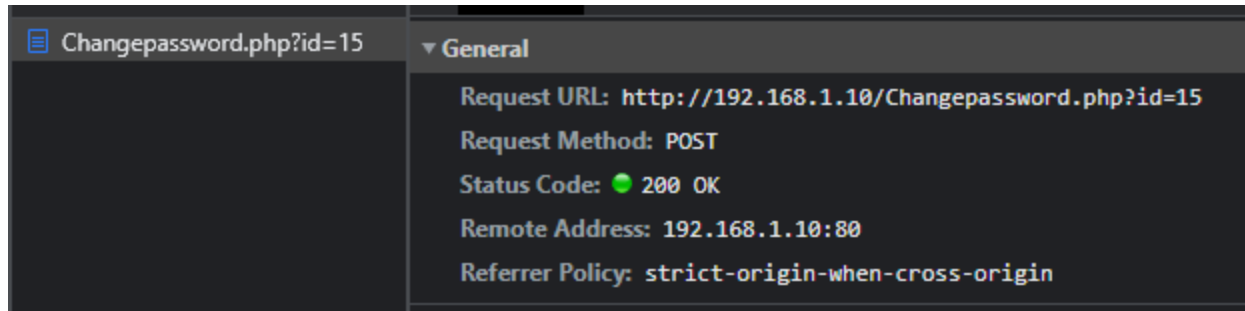


Figure 5 Changing password exposes user ID for hacklab@hacklab.com

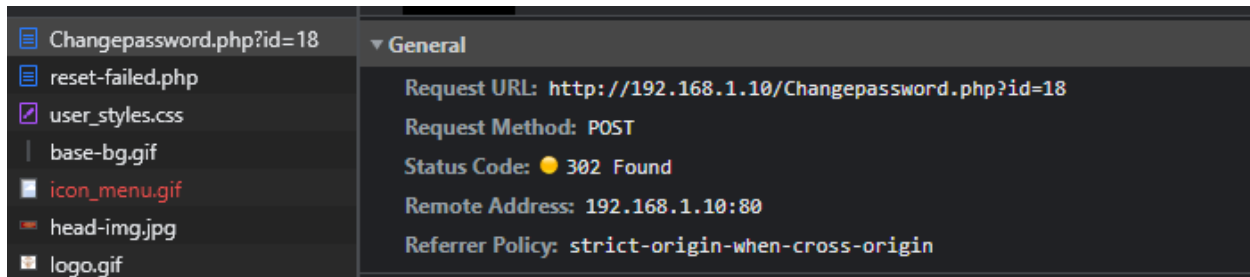


Figure 6 Changing password exposes user ID for test@test.com

2.2.6 Map Execution Paths Through Application

The section is focused on discovering the structure of the website and how this being mapped can be used for later exploitation. This process was done efficiently using the OWASP Zap tool. Using the built-in spidering tool of OWASP Zap, all URLs that are related with the domain 192.168.1.10 in this instance, will be enumerated and listed in a clear format. The spidering tool found 236 different URLs, all of which can be found in Appendix B, and the full report from OWASP ZAP can be found in Appendix D. A snippet of the scan is shown in Figure 7.

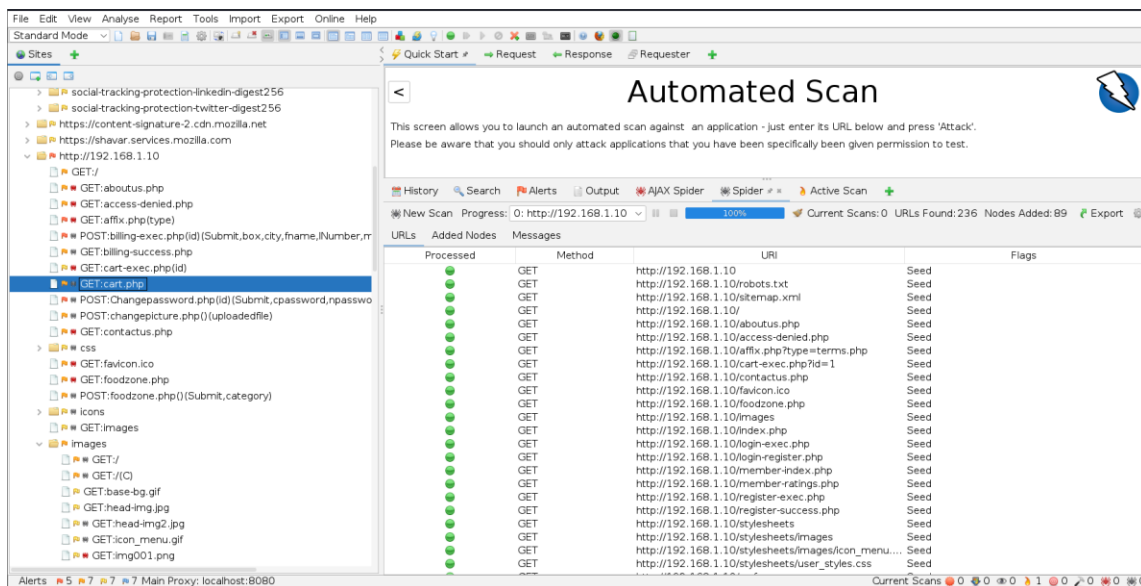


Figure 7 OWASP Zap output from scanning 192.168.1.10

The next step the tester took was to use a brute-force approach for enumeration to discover concealed files, URLs, directories, and anything else that may be of interest within scope. The two tools that were used are, Nikto and “dirbuster”, or more specifically Dirb, which is the non-GUI command line version of “dirbuster”. Dirb can enumerate and locate files and folders within a website, whilst Nikto can determine which vulnerabilities exist on the server and where they exist, which can be used for later exploitation phases. The results of both tools scans can be found in Appendix C Section 1 and Section 2. A snippet from Dirb can be seen in figure 8.

```
(kali@kali)-[~]
$ dirb http://192.168.1.10 -o ./Desktop/DIRBoutput.txt

DIRB v2.22
By The Dark Raver

OUTPUT_FILE: ./Desktop/DIRBoutput.txt
START_TIME: Wed Dec 7 18:34:09 2022
URL_BASE: http://192.168.1.10/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

GENERATED WORDS: 4612

-- Scanning URL: http://192.168.1.10/ --
=> DIRECTORY: http://192.168.1.10/~admin/
=> DIRECTORY: http://192.168.1.10/admin/
+ http://192.168.1.10/admin.cgi (CODE:403|SIZE:975)
+ http://192.168.1.10/admin.pl (CODE:403|SIZE:975)
+ http://192.168.1.10/AT-admin.cgi (CODE:403|SIZE:975)
+ http://192.168.1.10/cachemgr.cgi (CODE:403|SIZE:975)
+ http://192.168.1.10/cgi-bin/ (CODE:403|SIZE:989)
> DIRECTORY: http://192.168.1.10/css/
```

Figure 8 Snippet of Output of dirb scan

The tester found a vital file that reveals information regarding the webserver and website from the Nikto scan, which is the “phpinfo.php” file. This file should not be publicly viewable and gives the tester information to assist in further exploitation. For example, the physical paths and the full PHP configuration settings are contained within the file.

2.2.7 Fingerprint Web Application Framework

The final section of the Information Gathering section is related to fingerprinting the target and discovering data that is associated with the target system. Using the Wappalyzer extension, the tester could determine the target’s framework, the website used PHP 5.4.7 which is outdated compared the latest stable release of PHP 8.1.13 that can be found on the PHP website (PHP, November 2022). This is shown in figure 9.

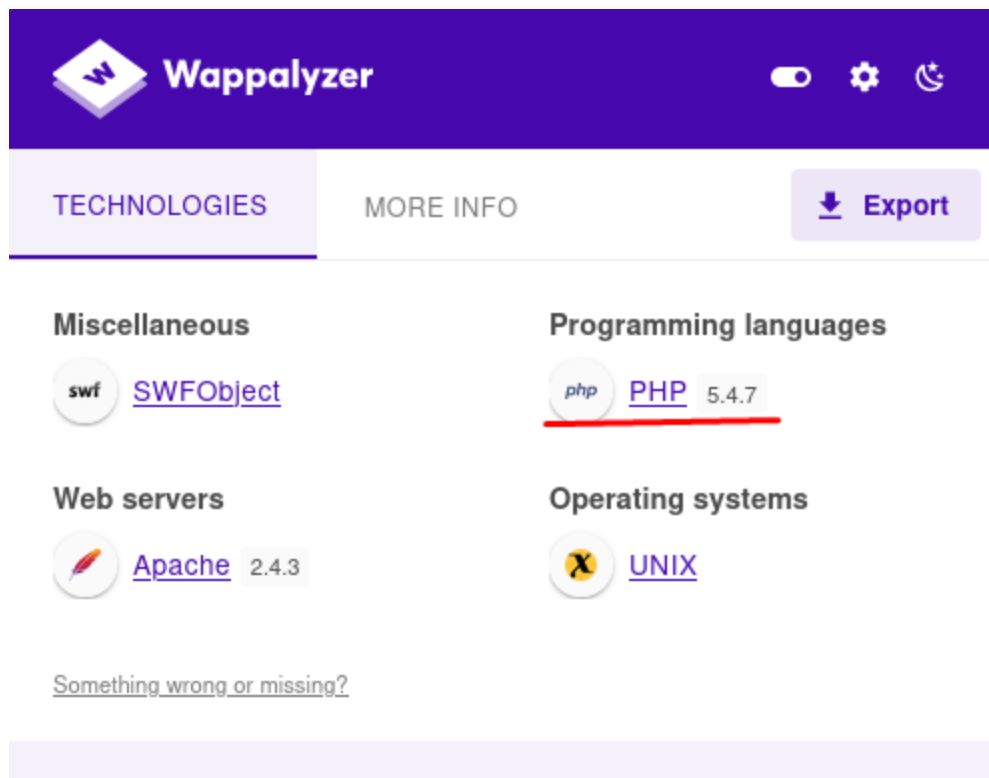


Figure 9 Wappalyzer used to determine systems in use

Afterwards the tester used Curl to confirm this, which did indeed give the tester confirmation that PHP 5.4.7 was used for the framework of the target. This can be seen in figure 10 (cyberciti, 2008).

```
(kali㉿kali)-[~]
└─$ curl -I http://192.168.1.10:80
HTTP/1.1 200 OK
Date: Wed, 07 Dec 2022 15:05:36 GMT
Server: Apache/2.4.3 (Unix) PHP/5.4.7
X-Powered-By: PHP/5.4.7
Content-Type: text/html
```

Figure 10 Using Curl to confirm PHP version

Next the tester used the output from the previous Nikto scan to determine other security concerns with the framework of the target. This resulted in issues being discovered with the anti-clickjacking x-frame-options header not being present, X-XSS-Protection was not defined, and the X-Content-Type-Options header was also not set. This can be seen in Figure 11. Other security concerns than the ones described here that were found by Nikto will be discussed in later sections.

```
1- Nikto v2.1.6/2.1.5
2+ Target Host: 192.168.1.10
3+ Target Port: 80
4+ GET Retrieved x-powered-by header: PHP/5.4.7
5+ GET The anti-clickjacking X-Frame-Options header is not present.
6+ GET The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
7+ GET 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
```

Figure 11 Nikto scan results reveal lack of vital headers

2.3 CONFIGURATION AND DEPLOYMENT MANAGEMENT TESTING

2.3.1 Test Application Platform Configuration

Within a web server, there will be three different things that could appear, which is, default files and known files, a file covering the logging systems with the web application, and finally, files and extensions which were used for debugging, etc, during the development stages. This section is associated with enumerating these three things in the application. The tester made use of the information within the previous scans for Nikto and OWASP ZAP.

From the Nikto scan, both a multitude of vulnerabilities and a list of directories and files which are known had been found. The results from the scan can be seen in figure 12.

```

9 + GET Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. See http://www.wisec.it/sectou.php?id=4698ebdc59d15. The following
alternatives for 'index' were found: HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var,
HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var,
10 + HEAD PHP/5.4.7 appears to be outdated (current is at least 7.2.12). PHP 5.6.33, 7.0.27, 7.1.13, 7.2.1 may also current release for each branch.
11 + HEAD Apache/2.4.3 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch.
12 + OSVDB-112084: GET /cgi-bin/printenv: Site appears vulnerable to the 'shellshock' vulnerability (CVE-2014-6271).
13 + OSVDB-112084: GET /cgi-bin/printenv: Site appears vulnerable to the 'shellshock' vulnerability (CVE-2014-6278).
14 + KUPKNRPG Web Server returns a valid response with junk HTTP methods, this may cause false positives.
15 + OSVDB-877: TRACE HTTP TRACE method is active, suggesting the host is vulnerable to XST
16 + GET /phpinfo.php: Output from the phpinfo() function was found.
17 + GET Cookie PHPSESSID created without the httponly flag
18 + OSVDB-12184: GET /%PHPE9568F36-D428-11D2-A769-00AA003ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
19 + OSVDB-12184: GET /%PHPE9568F36-D428-11D2-A769-00AA003ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
20 + OSVDB-12184: GET /%PHPE9568F36-D428-11D2-A769-00AA003ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
21 + OSVDB-12184: GET /%PHPE9568F36-D428-11D2-A769-00AA003ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
22 + OSVDB-3268: GET /css/: Directory indexing found.
23 + OSVDB-3092: GET /css/: This might be interesting...
24 + OSVDB-3268: GET /install/: Directory indexing found.
25 + OSVDB-3092: GET /install/: This might be interesting...
26 + OSVDB-3268: GET /stylesheets/: Directory indexing found.
27 + OSVDB-3092: GET /stylesheets/: This might be interesting...
28 + OSVDB-3233: GET /cgi-bin/printenv: Apache 2.0 default script is executable and gives server environment variables. All default scripts should be removed. It may also allow XSS types of
attacks. BID-4431.
29 + OSVDB-3233: GET /cgi-bin/test-cgi: Apache 2.0 default script is executable and reveals system information. All default scripts should be removed.
30 + OSVDB-3233: GET /phpinfo.php: PHP is installed, and a test script which runs phpinfo() was found. This gives a lot of system information.
31 + OSVDB-3233: GET /info.php: PHP is installed, and a test script which runs phpinfo() was found. This gives a lot of system information.
32 + OSVDB-3268: GET /icons/: Directory indexing found.
33 + OSVDB-3268: GET /images/: Directory indexing found.
34 + OSVDB-3268: GET /docs/: Directory indexing found.
35 + OSVDB-3233: GET /icons/README: Apache default file found.
36 + OSVDB-5292: GET /info.php?file=http://cirt.net/rfiinc.txt: RFI from RSNAKE's list (http://ha.ckers.org/weird/rfi-locations.dat) or from http://osvdb.org/
37

```

Figure 12 Nikto Scan Output

From figure 12, it can be determined that there are multiple vulnerabilities, one of which allows for filenames to be brute forced with ease, and the more critical vulnerability being “Shellshock”, which can allow for execution of arbitrary commands on the system. Meaning a reverse shell could be opened on the host device. Additionally, figure 12 revealed directories that can be viewed publicly which have been included in the screenshots below. With this information, the tester had determined the possibility of Remote Code Execution and a directory traversal attack.

Index of /css

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 DT_bootstrap.css	2013-02-20 00:02	3.6K	
 bootstrap-responsive..>	2013-06-20 14:41	22K	
 bootstrap.css	2013-07-21 20:11	122K	
 datepicker.css	2012-01-25 14:56	7.4K	
 demo.css	2013-04-24 20:22	3.5K	
 diapo.css	2013-07-11 17:29	3.0K	
 docs.css	2013-08-26 17:29	27K	
 font-awesome.css	2022-09-20 14:10	27K	
 normalize.css	2013-04-24 20:24	8.5K	
 style.css	2013-07-14 18:49	1.3K	

Figure 13 Index of /css

Index of /install



<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 coming_soon.txt	2016-06-13 10:12	103	

Figure 14 Index of /install

Index of /stylesheets



<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 user_styles.css	2017-01-15 11:56	3.2K	

Figure 15 Index of /stylesheets

Index of /icons


























<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 a.gif	1999-08-24 05:33	246	
 a.png	2001-05-30 07:54	293	
 alert.black.gif	1999-08-24 05:33	242	
 alert.black.png	2001-05-30 07:54	279	
 alert.red.gif	1999-08-24 05:33	247	
 alert.red.png	2001-05-30 07:54	298	
 apache_pb.gif	1999-08-24 05:33	2.3K	
 apache_pb.png	2001-05-30 07:54	1.4K	
 apache_pb2.gif	2001-05-03 04:30	2.4K	
 apache_pb2.png	2001-05-30 07:54	1.4K	
 apache_pb2_anl.gif	2001-05-03 04:30	2.1K	
 back.gif	1999-08-24 05:33	216	
 back.png	2001-05-30 07:54	284	
 ball.gray.gif	1999-08-24 05:33	233	
 ball.gray.png	2001-05-30 07:54	277	
 ball.red.gif	1999-08-24 05:33	205	
 ball.red.png	2001-05-30 07:54	265	
 binary.gif	1999-08-24 05:33	246	
 binary.png	2001-05-30 07:54	296	
 binhex.gif	1999-08-24 05:33	246	
 binhex.png	2001-05-30 07:54	304	
 blank.gif	1999-08-24 05:33	148	
 blank.png	2001-05-30 07:54	195	
 bomb.gif	1999-08-24 05:33	308	

Figure 16 Index of /icons (incomplete)

Index of /images



























<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 base-bg.gif	2016-06-13 10:12	587	
 head-img.jpg	2017-01-15 11:33	51K	
 head-img2.jpg	2016-06-13 10:12	78K	
 icon_menu.gif	2016-06-13 10:12	668	
 img001.png	2016-06-02 18:26	135K	
 img002.png	2016-07-21 06:30	135K	
 img003.png	2016-06-02 18:26	131K	
 img004.png	2016-06-02 18:26	122K	
 img005.png	2016-06-02 18:26	131K	
 img006.png	2016-06-02 18:26	121K	
 img007.png	2016-06-02 18:26	138K	
 img008.png	2016-06-02 18:26	152K	
 img009.png	2016-06-02 18:26	124K	
 img010.png	2016-06-02 18:26	136K	
 img011.png	2016-06-02 18:26	140K	
 img012.png	2016-06-02 18:26	144K	
 img013.png	2016-06-02 18:26	127K	
 img014.png	2016-06-02 18:26	128K	
 img015.png	2016-06-02 18:26	140K	
 img016.png	2016-06-02 18:26	143K	
 img017.png	2016-06-02 18:26	132K	
 img018.png	2016-06-02 18:26	123K	
 img019.png	2016-06-02 18:26	160K	
 img020.png	2016-06-02 18:26	129K	
 .			

Figure 17 Index of /images (incomplete)

Index of /docs







<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 changelog.txt	2016-06-13 10:12	1.1K	
 install.txt	2016-06-13 10:12	1.1K	
 readmefirst.txt	2016-06-13 10:12	1.8K	
 support.txt	2016-06-13 10:12	813	
 ~\$C 409 TERM PROJECT..>	2016-06-13 10:12	162	

Figure 18 Index of /docs

Next the tester used the results from the previous OWASP Zap's scan and more specifically, the "active scan" to confirm the vulnerabilities that were found from the Nikto scan and obtaining more data associated with the vulnerabilities that were found. This information can be found in the OWASP scan report in Appendix D, and a snippet of the vulnerabilities can be found in figure 19.

Alert type	Risk	Count
Cross Site Scripting (Persistent)	High	1 (5.3%)
Path Traversal	High	1 (5.3%)
SQL Injection	High	2 (10.5%)
SQL Injection - MySQL	High	2 (10.5%)
Absence of Anti-CSRF Tokens	Medium	17 (89.5%)
Application Error Disclosure	Medium	54 (284.2%)
Content Security Policy (CSP) Header Not Set	Medium	85 (447.4%)
Directory Browsing - Apache 2	Medium	54 (284.2%)
Missing Anti-clickjacking Header	Medium	81 (426.3%)
Vulnerable JS Library	Medium	2 (10.5%)

Figure 19 Snippet of vulnerabilities found by OWASP ZAP (Incomplete)

2.3.2 Enumerate Infrastructure and Application Admin Interfaces

The purpose of this section is to enumerate the web application for the existence of an admin interface, to do this the tester used information from the previous Dirb scan which can be found in Appendix C. From the scan, a directory called /admin was discovered, navigating to this directory brought the tester to an admin login panel. This can be seen in figure 20 and 21 respectively. However, without any admin credentials it was impossible to log on at this stage, this login panel will be used in later stages during exploitation.

```
DIRB v2.22
By The Dark Raver

OUTPUT_FILE: ./Desktop/DIRBoutput.txt
START_TIME: Wed Dec 7 18:34:09 2022
URL_BASE: http://192.168.1.10/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

GENERATED WORDS: 4612

— Scanning URL: http://192.168.1.10/ —
⇒ DIRECTORY: http://192.168.1.10/~admin/
⇒ DIRECTORY: http://192.168.1.10/admin/
+ http://192.168.1.10/admin.cgi (CODE:403|SIZE:975)
+ http://192.168.1.10/admin.pl (CODE:403|SIZE:975)
+ http://192.168.1.10/AT-admin.cgi (CODE:403|SIZE:975)
+ http://192.168.1.10/cachemgr.cgi (CODE:403|SIZE:975)
+ http://192.168.1.10/cgi-bin/ (CODE:403|SIZE:989)
```

Figure 20 Discovery of admin directory

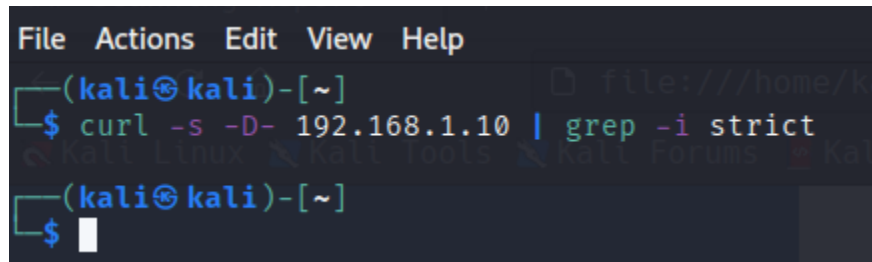


Figure 21 Admin login panel

2.3.3 Test HTTP Strict Transport Security

HTTP Strict Transport (HSTS) is a system used for preventing Man-In-The-Middle attacks. The mechanism does this by prohibiting connections to a website unless done through a secure connection like HTTPS, which means if the web server doesn't have this present, a malicious user can intercept the unencrypted traffic going between the server and a client to steal information such as bank details and user credentials.

To test for this, a straightforward command is run to check for the existence of a HSTS header. This command can be seen in figure 22. If no output is returned when the command is executed, then there is no HSTS header, and the web server is insecure.



```
File Actions Edit View Help
(kali@kali)-[~]
$ curl -s -D- 192.168.1.10 | grep -i strict
(kali@kali)-[~]
$
```

Figure 22 Insecure server revealed as there was no output from the executed command

2.4 IDENTITY MANAGEMENT TESTING

2.4.1 Test Role Definitions

Within the OWASP Methodology a multitude of user role types have been defined, these include:

- Administrator, this role manages application functionalities
- Auditor, this role reviews application transactions and provide a detailed report
- Support Engineer, this role assists customers debug and fix issues with their accounts
- Customer, this role is the standard user of the application

The tester determined the existence of an administrator account as there is an admin login panel which is meant for administrators. To confirm this, the tester attempted different credentials to try login to the admin panel, which revealed the existence of an account called “admin”.

The tester gains access to this admin role, which is described in a subsequent section. Using the admin panel, the tester could see a list of members with the option to remove the users. After testing the “remove users” function, it was determined that there was no multi-factor authentication enabled to stop a malicious user from removing accounts at will. This can be seen in figure 23.



Members Management					
Home Categories Foods Accounts Orders Reservations Specials Staff Messages Options Logout					
MEMBERS LIST					
Member ID	First Name	Last Name	Email		
15	Rick	Astley	hacklab@hacklab.com	Remove Member	
16	Jim	Smith	J.Smith@hacklab.com	Remove Member	
17	Joe	Bloggs	joeblogs@hereandnow.com	Remove Member	

© 2012-2013 Hacklab Pizza. All Rights Reserved

Figure 23 Test user account was removed from list of members

The tester could also alter all the categories and foods within the web application, which again required no multi-factor authentication to do. This is shown in figure 24 where the tester added the category “salad”.

The screenshot shows a web application titled "Foods Management". At the top, there is a navigation bar with links: Home, Categories, Foods, Accounts, Orders, Reservations, Specials, Staff, Messages, Options, and Logout. Below this is a form titled "ADD A NEW FOOD". The form has fields for Name, Description, Price, Category (a dropdown menu showing "- select one option -"), Photo (a "Browse..." button), and an "Add" button. Below the form is a table titled "AVAILABLE FOODS". The table has columns: Food Photo, Food Name, Food Description, Food Price, Food Category, and Action(s). There is one row of data: Food Photo (a small image), Food Name (yummy salad), Food Description (a bowl of lettuce), Food Price (0), Food Category (salad), and Action(s) (a "Remove Food" link). At the bottom of the page, there is a copyright notice: "© 2012-2013 Hacklab Pizza. All Rights Reserved".

Figure 24 The tester adding the category Salad

2.4.2 Test User Registration Process

When registering user accounts, it is vital to have validation for the usability of the web application. The validation used on this web application is entirely automated as manual validation for a growing website would become inefficient and slow down the entire process. However, with validation done in this manner, misconfigurations can appear throughout the code. This section is associated with enumerating validation of user identification and user input.

The first validation that was checked by the tester was the process of creating duplicate accounts for the same user. The following credentials was used by the tester:

- First Name: test
- Last Name: test
- Email: test@test.com
- Password: test
- Confirm Password: test
- Security question answer: test

This can be seen in figure 25.

The screenshot shows a user registration form. On the left, the labels for the fields are: First Name, Last Name, Email, Password, Confirm Password, Security Question, and Security Answer. On the right, the corresponding input fields are shown. The "First Name" field contains "test", "Last Name" contains "test", "Email" contains "test@test.com", "Password" contains "test" (masked with dots), "Confirm Password" contains "test" (masked with dots), "Security Question" is a dropdown menu showing "Ever gonna give you up? v", and "Security Answer" contains "test". Above the input fields, there is a red asterisk and the text "* Required fields". At the bottom of the form, there are two buttons: "Clear Fields" and "Register".

Figure 25 Input of Credentials into register form

From inputting these user credentials into the registration field, the user account was registered successfully. However, this raises some security issues to be addressed.

First off, the password input of “test” should not work as it is too simple of a password which could be brute forced with ease. Password policies should include, a minimum character length, the use of both upper-case and lower-case letters, the use of numbers, and at least one special character. Based on this best-case password policy, this web applications password policy raises security concerns considering “test” was allowed as a password.

Next, the email “test@test.com” was used, and while it is a legit email address, there is no validation to check if it is legit. The only validation on the web application is to determine whether this email address is already in the database.

A good practice that has been used for the registration interface is the lack of ability to select the role or permissions during registration.

Lastly the tester attempted to register again but with a different email address as the validation caught the same email when it was checked previously. The email used was “test1@test.com”. which successfully worked and allowed the tester to register the same account but with a different email address. The tester determined a lack of validation for an account other than email which the tester makes use of in the exploitation stage of the security assessment. The same email validation attempt and the new email can be seen in Figures 26 and 27 respectively.

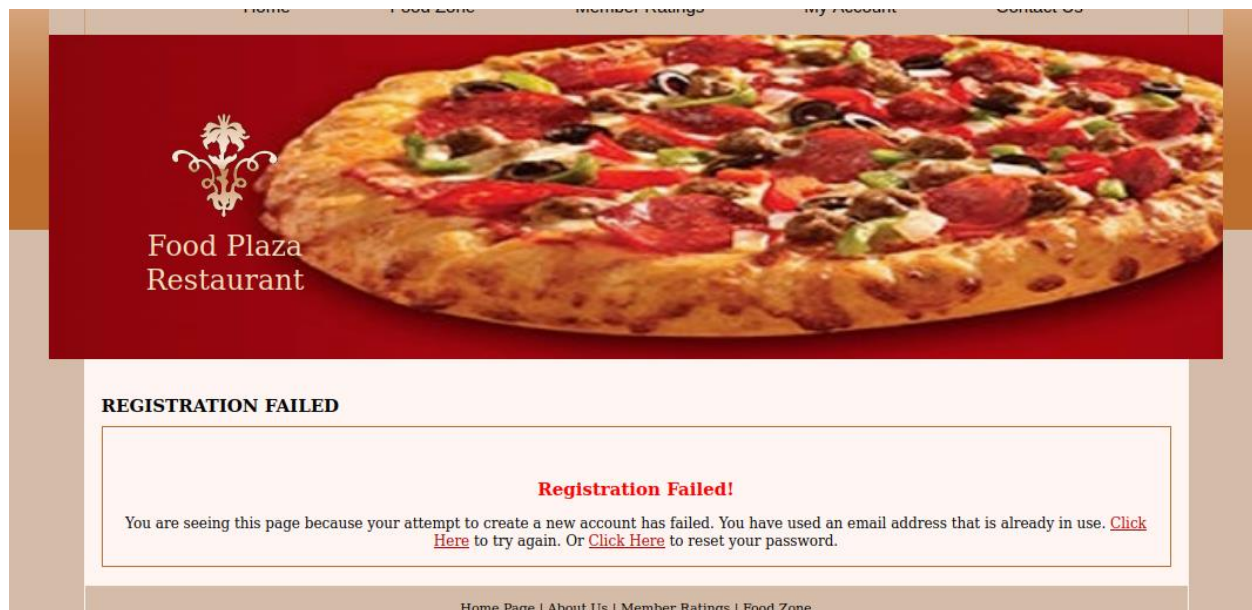


Figure 26 Showcasing the same email address validation working

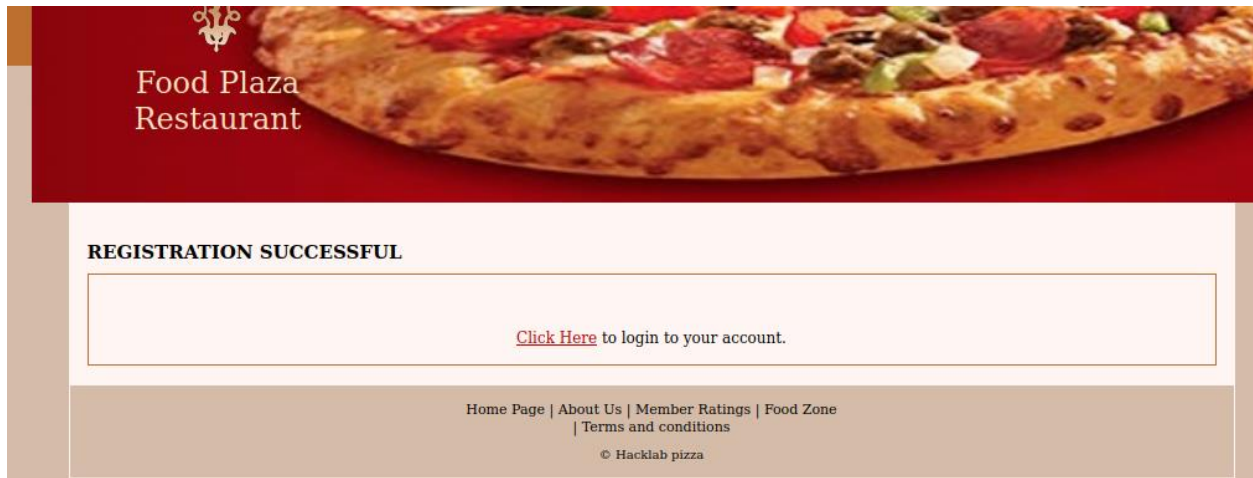


Figure 27 Showcasing slightly altered email address working with the same details for everything else

2.4.3 Testing for Weak or Unenforced

The last subsection is associated with testing restrictions on user input for registering an account. Two new accounts were created to test this, they are as follows

- Account 1
 - First name – “a”
 - Second name - “a”
 - Email - “a”
 - Password – “a”
 - Confirm Password – “a”
 - Security Answer – “a”
- Account 2
 - First name – “
 - Second name – “”
 - Email – “a”
 - Password – “”
 - Confirm Password – “”
 - Security Answer – “”

Figure 28 and figure 29 shows the accounts creation mechanism working successfully and an account with the name “a” and the name “” has been created.



Figure 28 Logged in as user A

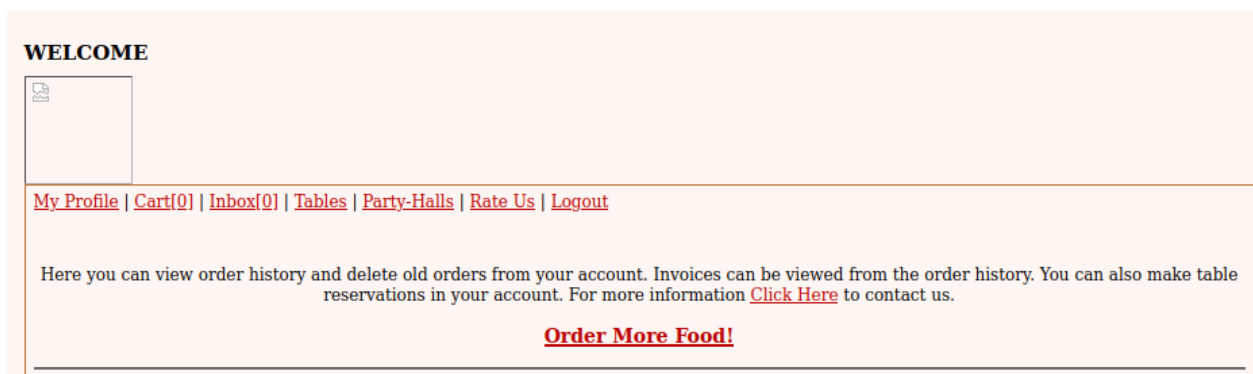


Figure 29 Logged in as user ""

2.5 AUTHENTICATION TESTING

2.5.1 Testing for Credentials Transported over Unencrypted Channels

This section is related to the sending of sensitive data, such as payment details, over unencrypted channels. Sending sensitive data over the internet using HTTP, is never advised. From the Nmap scans (figure 1 and figure 3) and checking for the padlock left of the URL address bar, the tester has determined this website uses HTTP for all communication which is insecure.

The tester chose to confirm this by using Curl to examine the login page of the web application while using the -kis flag to enable insecure connection. This can be seen in figure 30.

```
(kali㉿kali)-[~]
$ curl -kis http://192.168.1.10/member-index.php
HTTP/1.1 302 Found
Date: Mon, 24 Oct 2022 21:59:29 GMT
Server: Apache/2.4.3 (Unix) PHP/5.4.7
X-Powered-By: PHP/5.4.7
Set-Cookie: PHPSESSID=hdf9ushesqvc827tl1ncrdboe4; 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
location: access-denied.php
Content-Length: 0
Content-Type: text/html
```

Figure 30 Revealed phpsessid and headers on members page

In figure 28, the set-cookie field reveals an unencrypted PHPSESSID, which confirms that there is no SSL connection for transporting information and a user's session could be taken over by intercepting traffic and making use of stolen sensitive information.

Additionally, after the tester had logged into the test account and pressed F12 on google chrome to enter the developer panel and examined the cookies section of storage, which revealed a cookie called "SecretCookie". This can be seen in figure 31.

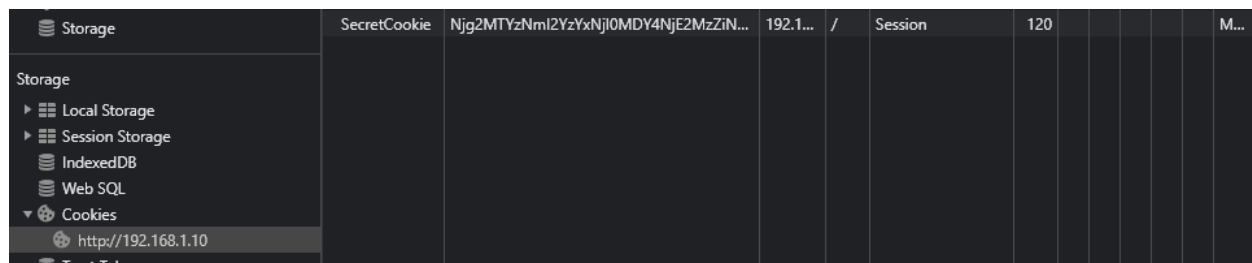


Figure 31 SecretCookie item

2.5.2 Testing for Default Credentials

During development of a web application, developers may create a set of default login credentials which are simple for ease of use, which will in turn be easily guessable. The tester chose to test different default creds, which are:

- test/test
- user/user
- admin/admin
- rick/rick

Out of these inputs, three of them gave a username not found on the admin panel and the "admin" test gave a different response, which was "login failed, please check username and password and try again". Meaning the username admin exists on the login panel and the rest do not exist on the backend database. This can be seen in figures 32 and 33.

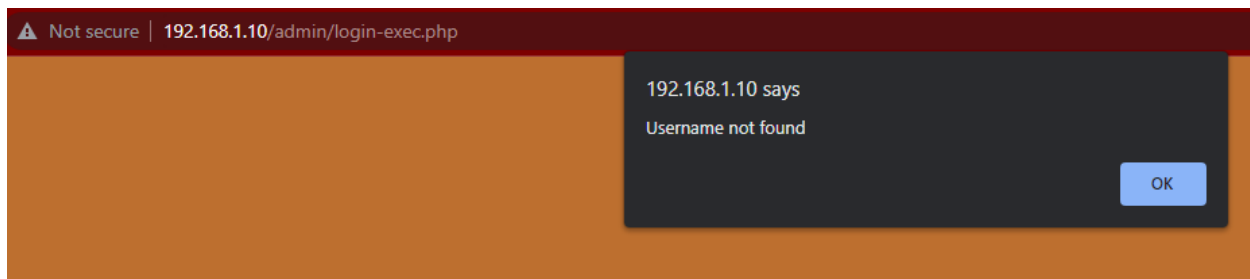


Figure 32 Username not found

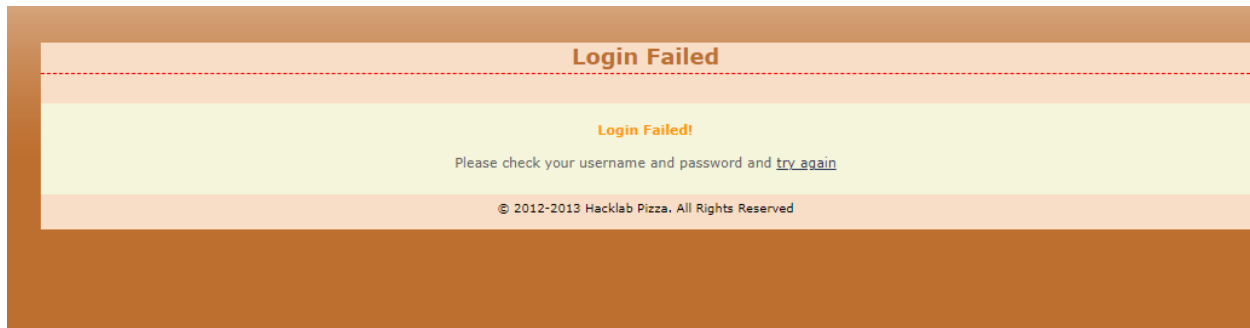


Figure 33 Login Failed

After the tester found the username “admin”, other default credentials were tested such as, “password”, “123456789”, “password123”, and “password!”. However, none of these gave any success but did reveal that there is no lockout rule with too many login attempts. This is a security concern, especially for an admin login panel. This is continued in the next subsection.

2.5.3 Testing for Weak Lock Out Mechanism

This section is associated with an account lockout process, which is when a user is locked out temporarily if they attempt to log in with too many incorrect attempts. If this mechanism is not in place, then an automated brute-force tool could be used to breach an account. Usually, the system will only allow between 3-5 wrong attempts before locking the user for a set period of time.

To test this, the account, “hacklab@hacklab.com” was used with the password hacklab to attempt to lock the login process temporarily. The tester used six wrong tries before attempting to log in with the correct details.

Once the six attempts were over, the tester was able to successfully log in to the hacklab account, which determined there is no lockout system on the login functionality.

Additionally, the tester was able to brute force the admin panel account to get the password “wormwood” using Burp Suite’s intruder function with the wordlist rockyou.txt. This can be seen in figure 34.

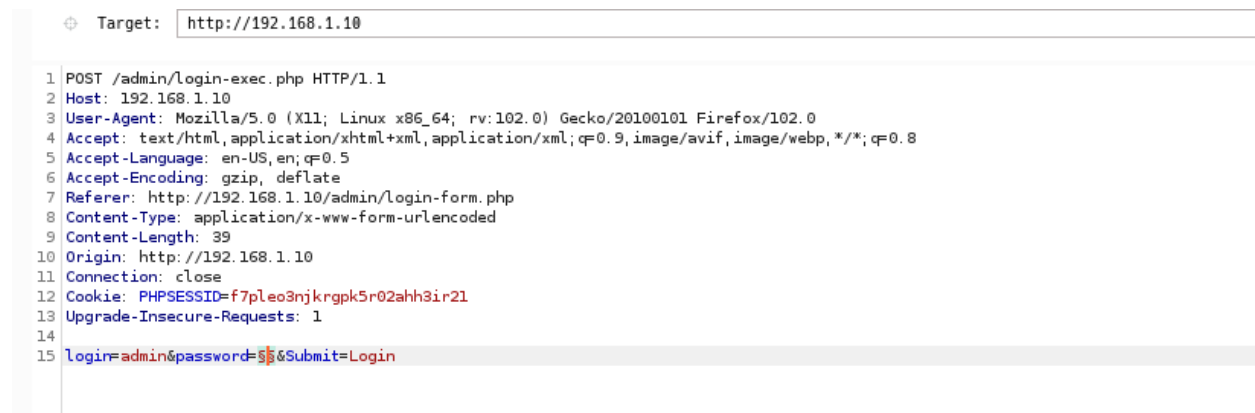


Figure 34 Burp Suite Intruder

2.5.4 Test Remember Password Functionality

Websites will use “Remember Password” to allow the user to login next time without having to type their password in, which is nice for the user but can raise some security concerns, especially if the token is set to never expire.

The tester used the account “a” to check the cookies and the “remember me” functionality to understand how the cookie’s expiry changes when “remember me” is turned on. The tester logged in with both the setting off and on. However, the cookie was still set to expire after session, which reveals the “remember me” is not functional as it should change the cookie expiry to a persistent value instead of a “session” value.

2.5.5 Testing for Weak Password Policy

The tester determined that this website does not use a password policy as an account with no username and only an email of “a” was created. This can be seen in figure 28 from a previous subsection. This means there is no minimum character length.

As mentioned previously in section 2.42, password policies should include:

- a minimum character length
- the use of both upper-case and lower-case letters
- the use of numbers
- at least one special character

However, the tester determined the website’s password policy does not meet any of these criteria, which makes brute-forcing a trivial task as users will typically choose a simple and insecure password. Additionally, passwords will most likely appear in dictionary-based attacks as the allowed passwords can be one-word passwords, which means hash cracking may not be needed for user accounts.

2.5.6 Testing for Weak Password Change or Reset Functionalities

Users will occasionally forget their password and will need to change it or will change their password as the user realised it is insecure and needs changed. This section covers security issues regarding these mechanisms such as if the password is sent in plaintext or not.

The tester found two different locations to reset a user's password on this web application, however, the reset password functionality for "forgot password" is not working due to programming errors. Therefore, the only mechanism to test is changing the password of a logged in account.

To do this the tester used the "test" account that was created previously and burp suite to intercept their traffic to be able to see and alter it. The account's password was changed from "test" to "tes", then the traffic was intercepted. This can be seen in figure 35.

```

1 POST /Changepassword.php?id=43 HTTP/1.1
2 Host: 192.168.1.10
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 56
9 Origin: http://192.168.1.10
10 Connection: close
11 Referer: http://192.168.1.10/member-profile.php
12 Cookie: PHPSESSID=vfpk81plad4ahqggk0pf00mn20; SecretCookie=NzQ2NTczNzQ0MDc0NjU3Mzc0MmU2MzZmNmQzYTc0NjU3Mzc0M2EzMTM2MzYzNjM2MzgzMjM2Mzg3NQ%3D%3D
13 Upgrade-Insecure-Requests: 1
14
15 opassword=test&npassword=tes&cpassword=tes&Submit=Change

```

Figure 35 Intercepted Traffic for password change

From figure 34, it shows that the password has been sent in without any confirmation required, such as an email to confirm the password change. A good practice is that the change does require the old password as well as the new password, which can slow down an attacker trying to change an account password.

The tester was particularly interested in the POST request at the top of the traffic. The POST request has the user id of the account inside of the request, meaning other account passwords could be brute forced to change their password. The tester confirmed this by using the account "a" by changing the password while in account "test". This can be seen in figure 36 and 37 respectively.

```

1 POST /Changepassword.php?id=45 HTTP/1.1
2 Host: 192.168.1.10
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 51
9 Origin: http://192.168.1.10
10 Connection: close
11 Referer: http://192.168.1.10/member-profile.php
12 Cookie: PHPSESSID=22vln8c1dkd2q4t69mc5sgvhd0; SecretCookie=NzQ2NTczNzQ0MDc0NjU3Mzc0MmU2MzZmNmQzYTc0NjU3Mzc0M2EzMTM2MzYzNjM2Mzg3NDM2MzIzNQ%3D%3D
13 Upgrade-Insecure-Requests: 1
14
15 opassword=a&npassword=aa&cpassword=aa&Submit=Change

```

Figure 36 Changed account id to account "a"

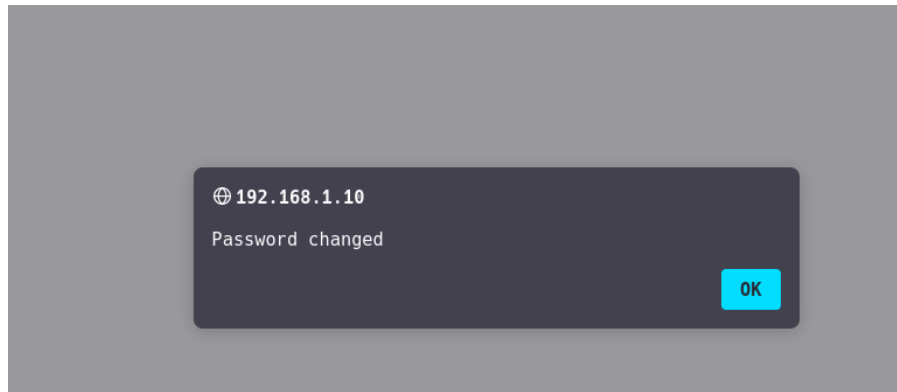


Figure 37 Password Changed Successfully

2.6 AUTHORISATION TESTING

2.6.1 Testing Directory Traversal File Include

This has been shown previously in section 2.3.1. Where the traversal of files was possible within this web application. By appending the known directory to the URL, for example, /images, the tester could view all the files and directories within /images. This can be seen in figure 17 in the section previously mentioned.

Additionally, the “affix.php” web page allows the passwd file to be read from the web page. This was found from the OWASP Zap scan found in Appendix D. The results of the directory traversal can be found in figure 38.

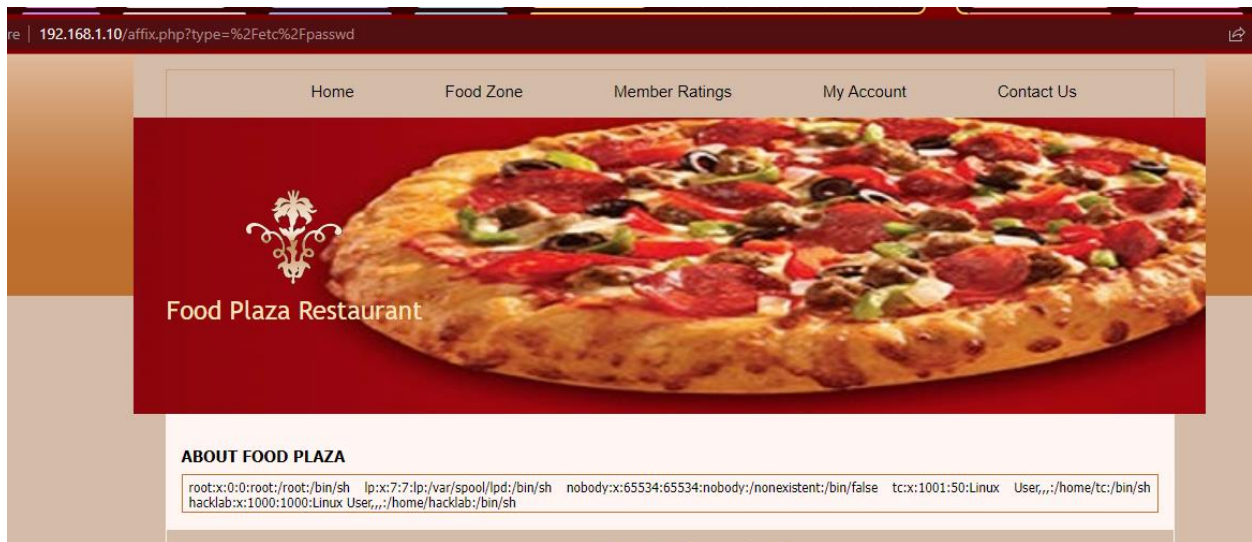


Figure 38 Reading Passwd through Local File Inclusion

2.7 SESSION MANAGEMENT TESTING

2.7.1 Testing for Session Management Schema

This section is associated with the process of a user's session management, for example, to ensure a user isn't logging back in every single webpage, cookies are implemented. As seen previously in section 2.5.1, a cookie called SecretCookie is created for all unique users and can be seen in developer tools as the website uses HTTP which is an unencrypted channel. The cookie from the account, "test" was decoded using CyberChef online web tool and then analysed by the tester (CyberChef, no date). The cookie was decoded from URL encoding, then Base64 encoding, and finally Changed from hex to plaintext to reveal the cookie. The cookie was created using, the username, the password, and a set of numbers which will be covered in the next section. This can be seen in figure 39.

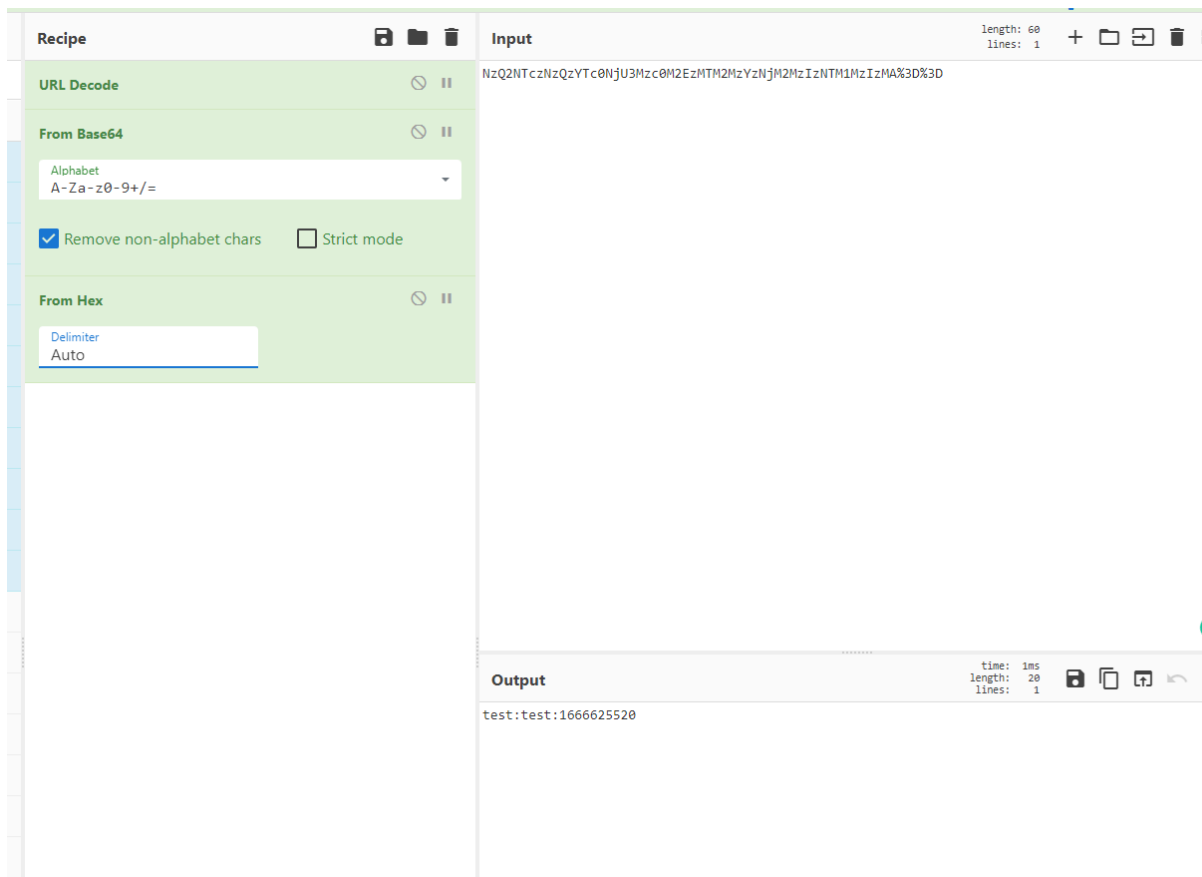


Figure 39 Decoding cookie of account, test

2.7.2 Testing for Cookies Attributes

The tester has confirmed in various sections, the website does not use HTTPS and only uses HTTP, which is unencrypted. Meaning cookies can be stolen and used by an unauthorised user as they are viewable during the session.

From the previous section, the cookie was decoded and had three separate values to create the cookie, the username, the password and a third value which the tester determined to be a timestamp that Unix uses to display the exact time the user logged into the account. However, the timestamp is roughly two

months behind the exact time and date. This can be seen in figure 40. This was confirmed after testing with different accounts.

Convert epoch to human-readable date and vice versa

Supports Unix timestamps in seconds, milliseconds, microseconds and nanoseconds.

Assuming that this timestamp is in **seconds**:

GMT : Monday, 24 October 2022 15:47:09

Your time zone : Monday, 24 October 2022 16:47:09 GMT+01:00 DST

Relative : 2 months ago

Figure 40 Conversion from Unix timestamp to human-readable

2.7.3 Testing for Session Fixation

Session Fixation was tested by authenticating into the given hacklab account, then taking the two cookies that belong to that account, afterwards, logging into the test account using Firefox's Developer tools to alter the cookie values after the authentication process to the hacklab account's values. This is known as session hijacking, where the tester will switch accounts by doing this without proper authentication. The tester successfully switched accounts to hacklab, which can be seen in figures 41 and 42.

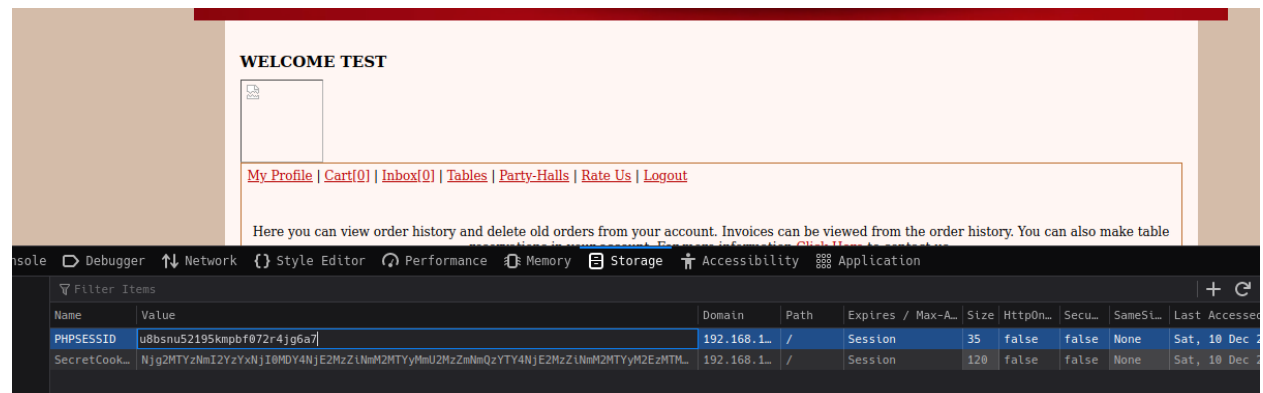


Figure 41 Changing cookie values to hacklab's cookies

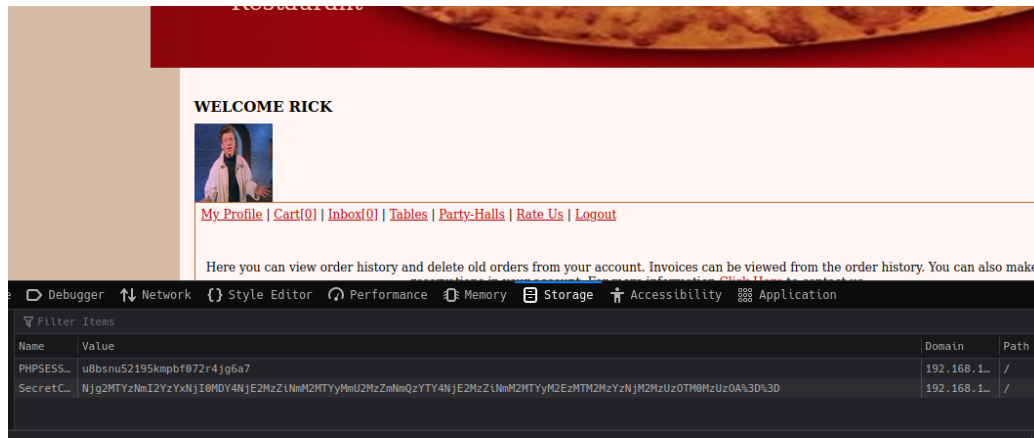


Figure 42 Successfully changed to hacklab without authentication

2.7.4 Testing for Logout Functionality

This section is associated with testing whether logout functionality properly terminates the session of the user. The tester used the account “test” for testing the logout functionality. After logging in and then logging out, the tester determined the logout functionality works server-side

as intended as the web application denies any entry into member only webpages. This can be seen in figure 43.

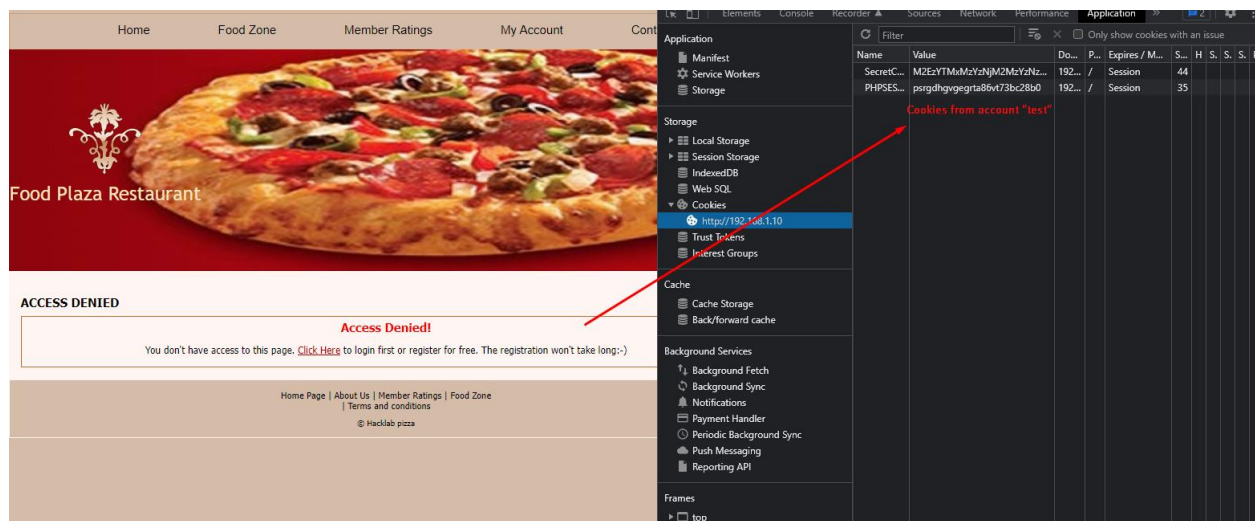


Figure 43 Displaying access denied webpage after logging out of "test" account

2.8 INPUT VALIDATION TESTING

2.8.1 Testing for Reflected Cross Site Scripting

The OWASP Zap scan results for vulnerabilities, as seen in section 2.3.1 evaluates the web application to be vulnerable to Cross Site Scripting (XSS) attacks. XSS attacks are performed by injecting executable code such as “<script>alert(1)</script>” inside of a user input field. A Reflected XSS attack, is both delivered and executed at the same time and is non-persistent (PortSwigger, no date).

After the tester attempted an XSS attack against all known user input fields, no Reflected XSS attack vulnerability was found. However, the tester found multiple cases of Stored XSS attacks, which is described in the next section.

2.8.2 Testing for Stored Cross Site Scripting

A Stored Cross Site Scripting attack occurs when an input field which is vulnerable to XSS is responsible for storing data into the database. The tester created a new account to test this vulnerability in two different user input fields. The first tested input field was the account registration fields which was all filled with <script>alert(1)</script> to prove the web application is vulnerable to stored XSS. The second user input field which also was used as a proof of the web application being vulnerable was member comments input. Both attempts at stored XSS can be seen in figure 44 and 45 respectively.

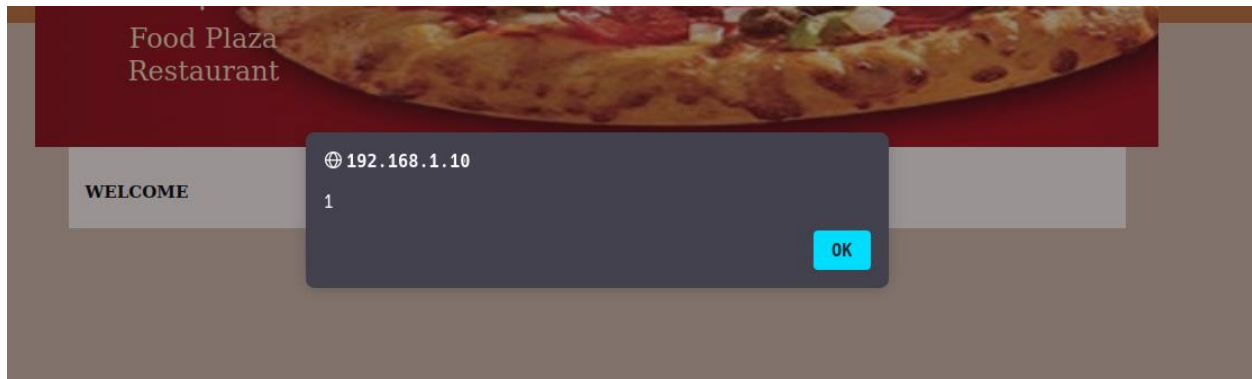


Figure 44 User Registration vulnerable to Stored XSS

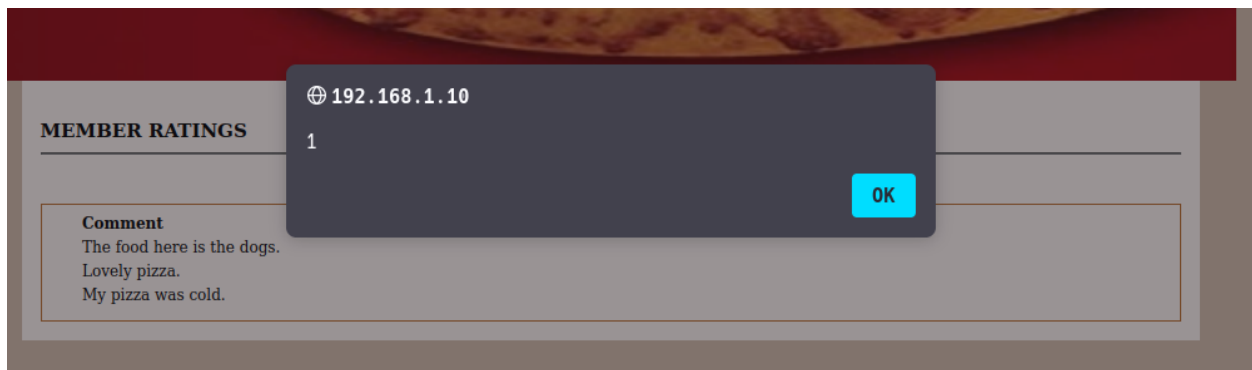


Figure 45 Member Comments vulnerable to Stored XSS

2.8.3 Testing for SQL Injection

To test for SQL Injection, the tester used the standard SQL injection code, "test' AND '1'='1" to evaluate if the login system is vulnerable to SQLI. The results can be seen in figure 46.

* Required fields

Email * test' AND '1'='1

Password * 1111

☐ Remember me

[Forgot password?](#)

Clear Fields Login

Figure 46 Testing SQLI

The tester successfully gained access to the user account which proves the user input field is vulnerable to SQLI as it threw no errors and accepted the input shown in figure 43. The tester also found the same results with the register user input fields.

The tester then tested the admin panel password field using "x' OR '1'='1" and the username "admin" which granted access to the admin control panel. This can be seen in figure 47.

Administrator Control Panel

[Profile](#) | [Categories](#) | [Foods](#) | [Accounts](#) | [Orders](#) | [Reservations](#) | [Specials](#) | [Staff](#) | [Messages](#) | [Options](#) | [Logout](#)

CURRENT STATUS									
Members Registered	Orders Placed	Orders Processed	Orders Pending	Table(s) Reserved	Table(s) Allocated	Table(s) Pending	PartyHall(s) Reserved	PartyHall(s) Allocated	PartyHall(s) Pending
6	1	0	1	0	0	0	1	0	1

CUSTOMERS' RATINGS (100%)

Food:

Excellent Good Average Bad Worse

© 2012-2013 Hacklab Pizza. All Rights Reserved

Figure 47 SQLI to Admin panel

2.9 ERROR HANDLING

2.9.1 Testing for Improper Error Handling

The tester deliberately navigated to webpages and content that does not exist to get a 404-error page. The error webpage displayed information associated with the system the web application is running on. This can be seen in figure 48.

Object not found!

The requested URL was not found on this server. If you entered the URL manually please check your spelling and try again.

If you think this is a server error, please contact the [webmaster](#).

Error 404

192.168.1.10
Apache/2.4.3 (Unix) PHP/5.4.7

Figure 487 404 Error page displays server information

2.10 CRYPTOGRAPHY

2.10.1 Testing for Weak Transport Layer Security

The tester has previously determined the web application to have no SSL or TLS certificate, which can be confirmed as the web application is using the HTTP protocol instead of HTTPS. This can be seen in figure 3. The tester decided to confirm this again using the tool sslscan. The results of this scan can be seen in figure 49.

```
(kali@kali)~$  
$ sslscan 192.168.1.10:80  
Version: 2.0.15-static  
OpenSSL 1.1.1q-dev xx XXX xxxx  
  
Connected to 192.168.1.10  
  
Testing SSL server 192.168.1.10 on port 80 using SNI name 192.168.1.10  
  
SSL/TLS Protocols:  
SSLv2 disabled  
SSLv3 disabled  
TLSv1.0 disabled  
TLSv1.1 disabled  
TLSv1.2 disabled  
TLSv1.3 disabled  
  
TLS Fallback SCSV:  
Connection failed - unable to determine TLS Fallback SCSV support  
  
TLS renegotiation:  
Session renegotiation not supported  
  
TLS Compression:  
Compression disabled  
  
Heartbleed:  
  
Supported Server Cipher(s):  
Unable to parse certificate  
Unable to parse certificate  
Unable to parse certificate  
Unable to parse certificate  
Certificate information cannot be retrieved.
```

Figure 498 SSLScan displays no valid certificates on web application

2.11 BUSINESS LOGIC TESTING

2.11.1 Test Ability to Forge Requests

As proved in a multitude of sections, the tester has confirmed this web application contains various vulnerabilities such as Local File Inclusion, Cross Site Scripting, SQL Injection, and session hijacking. Using these vulnerabilities, the tester gained access to unauthorised areas of the websites such as the entirety of user data and access to the admin panel using an admin account.

2.11.2 Test Number of Times a Function Can Be Used Limits

As shown in section 2.5.3, this web application has a lack of lock out mechanism, meaning the web application is vulnerable to brute-force attack. Brute-forcing is when an unauthorised user rapidly attempts to breach a user account by password guessing if they have a username, or an attacker will brute-force both the username and the password systematically.

2.11.3 Test Upload of Unexpected File Types

The member's profile page allowed the user to upload a profile picture to allow the user to customise their account. The tester uploaded a PHP reverse shell to it which gave an error only allowing .jpg files. To get around this, the web proxy Burp Suite was used to intercept the request and alter the file to .jpg. This allowed the tester to upload the shell as seen in figure 50 Which means validation stopped the user from being able to upload a working PHP reverse shell as it does not work in .jpg format and allowed the tester to determine the validation method which was file extension based.





<u>NAME</u>	<u>DATE MODIFIED</u>	<u>SIZE</u>	<u>DESCRIPTION</u>
 Parent Directory		-	
 fluffy.jpg	2017-08-05 14:39	67K	
 php_reverse_shell.jpg	2022-10-25 10:34	5.4K	
 rick.jpg	2017-08-05 14:39	21K	

Figure 509 Uploaded PHP Reverse Shell

3 DISCUSSION

3.1 DISCOVERED VULNERABILITIES AND COUNTERMEASURES

3.1.1 Robots.txt Vulnerability

The “Robots.txt” file is usually for storing a list of directories and files which a web crawler is not allowed to access with spidering tools, which means it will not appear when the web application is searched for through a search browser. The web application’s Robots.txt file while the file itself is not a vulnerability, it is misconfigured as only one file is in the disallow list, “schema.sql”. This file contains information on the objects related to the database of the website. This information should not be publicly viewable.

To mitigate this vulnerability, “schema.sql” should be removed from robots.txt and not be publicly viewable as it is a private file with information associated to the backend of the website. Meaning the file should be removed from the directory where the web application is located.

3.1.2 Local File Inclusion Vulnerability

Using the “affix.php” file, a local file inclusion vulnerability is exploited. This vulnerability allows unauthorised access to files which are not supposed to be publicly available, in this case, /etc/passwd. There is a filter which is trivially bypassed by using a tool such as OWASP ZAP.

To mitigate this vulnerability, the files accessible by “affix.php” should be whitelisted with everything else being disallowed. However, considering “affix.php” is only being used to display the terms and conditions web page, an appropriate choice would be to display the terms and conditions in its own separate webpage.

3.1.3 Reversible Cookie Vulnerability

The web application has a login functionality which registers each user with a “SecretCookie” when they log in. This cookie was straightforward to reverse into its three components consisting of, the username, the password, and the timestamp of when the user logged in. When a user is already logged in, using “SecretCookie” and “PHPSESSID” of that user’s session will allow a malicious user to switch to the victim’s account when they attempt to validate their login. This process can be seen in section 2.7.3. However, even if an attacker does not session hijack, they will be able to obtain the username and password of a victim by reversing “SecretCookie”.

The mitigation for this vulnerability would be to encrypt the information with a hashing algorithm such as SHA256. Another solution would be to remove the cookie entirely as the “PHPSESSID” cookie deals with sessions which makes the “SecretCookie” redundant.

3.1.4 Cookie Attributes Vulnerability

While the cookie is reversible, it also has no attributes set, such as the HttpOnly attribute. Meaning the cookie is vulnerable to Cross Site Scripting attacks that use JavaScript. This attribute is not the be all and end all of protection but can reduce the chance of the cookie being stolen.

The mitigation is to set the HttpOnly attribute with newly created cookies.

3.1.5 Directory Browsing Vulnerability

The web application allows users to browse directories within the website, which allows an attacker to trivially enumerate the site. This is because an attacker can look through directories with sensitive information such as `/~admin`, which has the `sqlcm.bak` file that deals with filtering. This could potentially allow an attacker to forge a way to get around validation found within `sqlcm.bak`.

The mitigation for this vulnerability is to reconfigure the apache server which is hosting the web application. Changing the “Options+Indexes” to “Options-Indexes” in the “.htaccess” file will ensure users cannot browse through directories which makes enumeration difficult for attackers. (Simplified, no date)

3.1.6 Unlimited Login Attempts Vulnerability

When a user attempts to login to the web application, there is not a limited number of incorrect attempts. This means an attacker can use a brute-force attack without any kind of mechanism to slow down the attack.

The mitigation for this vulnerability is to rate limit the users attempts at logging in and introduce account locking which requires either a timeout period and/or send an email which asks the user if they had been trying to log in. These mechanisms put together would increase the security of the web application.

3.1.7 No HTTPS Vulnerability

The web application uses HTTP over HTTPS for all communication between hosts and the server, which is unencrypted. Meaning an attacker can monitor and intercept traffic using tools like “wireshark”. The intercepted traffic can include sensitive information like bank details and account credentials.

The mitigation for this is to switch the service to HTTPS and update to the latest version of SSL/TLS and finally enforce this for all traffic for security purposes. In doing this, all data will be safe from monitoring tools and software.

3.1.8 PHP Information Disclosure Vulnerability

The files “`phpinfo.php`” and “`info.php`” can be accessed by appending them onto the URL of the website, which is a security concern as these files reveal information associated with the website such as version numbers, the configurations, and the root directory of the web application. This makes enumeration easy for the attacker. (rapid7, no date)

The mitigation for this security issues is to remove the files from the web application before going live as they contain a large amount of sensitive information which is publicly viewable currently.

3.1.9 Cross Site Request Forgery (CSRF) Vulnerability

On the password update page, an attacker can send a link to a user and trick them to change their password. The id of each user is also displayed in the request which can be used to work out which account belongs to which. The major concern of the CSRF is that if it was used against an admin account then the entire web application could be breached.

The mitigation for this vulnerability is to implement a CSRF token in the appropriate requests. This token should only be validated before form execution, the token should not be easily reversible, and should be associated with the user’s validated session. This is to ensure that two CSRF tokens do not match.

3.1.10 SQL Injection Vulnerability

The login forms of both user and admin panel are vulnerable to SQL injection. It appears no filtering is done to stop SQL injection as the basic script, "x' OR '1'=1" was able to bypass the admin account password validation.

The mitigation for this security issue is use prepared statements. The prepared statements will especially important when a "untrusted" input is supplied. Additionally, the prepared statements should not include variable content, which will stop SQL injection attacks as they are ignored.

3.1.11 User Enumeration Vulnerability

When logging into the web application incorrectly, the user returns with the error message "user not found". However, when a correct username but an incorrect password is input, it returns a different error message. Which means a Malicious user can repeatedly guess or brute-force accounts until they find a username, then can attempt to access the newly found account through guessing or brute-forcing the password.

The mitigation for this security issues is to alter the error message to a generic "error with username/password, try again" message. This results in attackers not knowing if the password or username field was incorrect.

3.1.12 Brute-Forceable Admin Password

The admin account password, "wormwood" was obtained with the use of brute-force and the password can be found in the "rockyou.txt" wordlist that was used for password-cracking. The password does not follow the best practice for passwords such as a minimum password length of 8 characters. Additionally, this password can be found in any dictionary, and with the basic username, this admin account was trivially breached.

The mitigation for this vulnerability is to implement a password policy which includes, a minimum character length of 8 characters, check passwords against a list of commonly used passwords and deny if the password matches any on the list, include at least 1 number, include at least 1 special character, and use a mix of both upper-case and lower-case characters.

While it can seem like a good idea to have a password expiry policy, this is not the case. The National Institute of Standards and Technology password security standards listed in NIST SP800-63B states to not include a password expiration period. Additionally, it states that Multi-Factor Authentication should be enforced. (NIST, 2017 June).

3.2 GENERIC ISSUES

3.2.1 X-Powered-By Header

Most of the site headers include "X-Powered-By" which allows an attacker to determine the PHP version to be 5.4.7. Using this information, the attacker can use Metasploit or exploitDB to find vulnerabilities associated with that version of PHP.

The web application should suppress this header by changing the configuration of the "php.ini" file. The line that requires modification is "expose_php" and is set to "off".

3.2.2 Clickjacking

The nikto scan revealed that within the HTTP response, the “X-Frame-Options” header is not present, which allows the attacker to attempt Clickjacking attacks. This type of attack is done by layering an invisible button on top of a legitimate button on the web application. The invisible button could potentially share a malicious link to the victim.

To mitigate this, the “X-Frame-Header” should be enabled on all pages on the website, and considering the website does not use any frames, the header should be set to “DENY”.

3.2.3 X-XSS-Protection Header

The X-XSS-Protection Header is not present on the web application, which would normally be mitigated by setting a filter for Cross Site Scripting when it is detected. However, this mechanism is now deprecated, and “content-Security-Policy” header should be in place. The mentioned header enables restrictions on how assets on the page load.

3.2.4 X-Content-Type-Options Header

The web application lacks the “X-Content-Type-Options” header, this header is used to stop the browser from translating files as a different MIME type. This also applies to error pages.

This header should be set to “nosniff” throughout the entire web application to ensure MIME-sniffing is not performed.

3.2.5 Shellshock Vulnerability

This web application is vulnerable to Shellshock, which is a commonly known arbitrary code execution vulnerability. This vulnerability makes use of applications or services which allow malicious users to manipulate bash environment variables(infosecarticles, 2020).

To mitigate against this vulnerability is to update the system’s version of bash regularly as later versions of this software patches the shellshock vulnerability.

3.2.6 HTTP TRACE

HTTP TRACE is enabled on the web application; however, this is typically used during debugging. This can lead to a leak of sensitive information as HTTP TRACE enabled will make the web server echo TRACE method requests.

After development of the web application is completed, the TRACE method should be turned off to stop misuse when the website goes live.

3.2.7 Mod_negotiation

“mod_negotiation” is an Apache module which is used to choose documents that matches client capacity from several different documents. An attacker can use an invalid Accept header to reveal information when the server responds to it with an error.

3.2.8 PHPMyAdmin

PHPMyAdmin is publicly viewable and accessible, meaning an attacker could conduct a brute-force attack against this to obtain access to the MySQL database, to fix this, the web application should be configured so the default directory alias that is used to access phpMyAdmin is changed to an obscure alias that is hard to find(linode, 2022).

3.3 GENERAL DISCUSSION

The investigation of Mr Rick Astley's Hacklab Pizza web application revealed various misconfigurations, security concerns and vulnerabilities. The potential damage from the security issues can be financial and reputational as an unauthorised user could trivially obtain access to admin-only areas of the web application and vandalise the website or alter the prices of items to free.

The biggest security concern which needs addressed is the usage of HTTP over HTTPS. The web application has login and transaction functionalities, meaning sensitive data is being transferred over the channel of HTTP, which does not use encryption. Meaning the information could be intercepted by an attacker and used maliciously, which will affect Hacklab Pizza under the General Data Protection Act as information is not being transferred and stored securely.

Secondly, the web application makes use of an insecure cookie called "SecretCookie", which can be reversed into its three main components easily. Should the attacker intercept and reverse the information within the cookie, they will gain access to the user session which that cookie belongs to.

Finally, the numerous misconfigurations within the web application is causing information leakage, which allows an attacker to enumerate the different service's version for vulnerabilities related to that version. The information leaks make enumeration simple for an attacker and should be reconfigured to stop these leakages.

Overall, the website is not secure and should not be made live until the listed issues are mitigated using the countermeasures listed within this report. Each countermeasure would significantly increase the security of the web application.

3.4 FUTURE WORK

After countermeasures have been implemented, a second security assessment should take place to ensure the changes have been implemented properly with no new vulnerabilities appearing from the changes.

Additionally, the future test's scope could be extended to the organisation. For example, running a Phishing campaign to ensure all staff of the organisation are fully trained and aware of the process that should be taken if they are sent a phishing email that imitates a supplier or a trusted organisation.

4 REFERENCES

- (no date) *CyberChef*. Available at: [https://gchq.github.io/CyberChef/#recipe=From_Base64\('A-Za-z0-9%2B/%3D',true,false\)From_Hex\('Auto'\)&input=TnpRMk5UY3pOelEwTURjME5qVTN NemMwTW1VMk16Wm1ObVF6WVRjME5qVTNNemMwTTJFek1UTTJNell6TmpNM 016SXpNek0xTXpjek5BJTNEJTNE](https://gchq.github.io/CyberChef/#recipe=From_Base64('A-Za-z0-9%2B/%3D',true,false)From_Hex('Auto')&input=TnpRMk5UY3pOelEwTURjME5qVTN NemMwTW1VMk16Wm1ObVF6WVRjME5qVTNNemMwTTJFek1UTTJNell6TmpNM 016SXpNek0xTXpjek5BJTNEJTNE) (Accessed: December 4, 2022).
- (no date) *How to disable directory listing in Apache*. Available at: <https://www.simplified.guide/apache/disable-directory-listing> (Accessed: December 7, 2022).
- Boskamp, E. (2022) *30 worrisome cybersecurity statistics [2022]: Data, trends and more*, Zippia *30 Worrisome Cybersecurity Statistics 2022 Data Trends And More Comments*. Zippia. Available at: <https://www.zippia.com/advice/cybersecurity-statistics/#:~:text=Cybercrime%20costs%20the%20United%20States,and%20files%20are%20properly%20protected.> (Accessed: December 1, 2022).
- Downloads* (no date) *php*. Available at: <https://www.php.net/downloads.php> (Accessed: December 3, 2022).
- Fingerprint / identify remote web server - nixcraft* (no date). Available at: <https://www.cyberciti.biz/faq/find-out-remote-webserver-name/> (Accessed: December 4, 2022).
- Hackersploit (2022) *How to secure phpmyadmin, Linode Guides & Tutorials*. Linode. Available at: <https://www.linode.com/docs/guides/how-to-secure-phpmyadmin/> (Accessed: December 12, 2022).
- How many cyber attacks happen per day in 2022?* (no date) *Techjury*. Available at: <https://techjury.net/blog/how-many-cyber-attacks-per-day/#:~:text=Here%20is%20a%20sneak%20peek,form%20of%20a%20cyber%20attack.> (Accessed: December 1, 2022).
- Kemp, S. (2022) *Digital 2022: Global Overview Report - DataReportal – Global Digital Insights, DataReportal*. DataReportal – Global Digital Insights. Available at: <https://datareportal.com/reports/digital-2022-global-overview-report/#:~:text=Global%20internet%20users%3A%20Global%20internet,of%20the%20world's%20total%20population.> (Accessed: December 1, 2022).
- Mehndiratta, M. (2021) *Exploiting a shellshock vulnerability, INFOSEC ARTICLES*. INFOSEC ARTICLES. Available at: <https://www.infosecarticles.com/exploiting-shellshock-vulnerability/> (Accessed: December 10, 2022).

Owasp Web Security Testing Guide (no date) *OWASP Web Security Testing Guide* / *OWASP Foundation*. Available at: <https://owasp.org/www-project-web-security-testing-guide/> (Accessed: November 30, 2022).

Phpinfo() information leakage (no date) *Rapid7*. Available at: <https://www.rapid7.com/db/vulnerabilities/http-php-phpinfo-leak/#:~:text=The%20information%20leaked%20by%20the,the%20full%20PHP%20configuration%20settings>. (Accessed: December 8, 2022).

The principles (no date) *ICO*. Available at: <https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/principles/?q=fines> (Accessed: December 2, 2022).

What is cross-site scripting (XSS) and how to prevent it?: Web security academy (no date) *What is cross-site scripting (XSS) and how to prevent it? | Web Security Academy*. Available at: <https://portswigger.net/web-security/cross-site-scripting> (Accessed: December 4, 2022).

APPENDICES PART 1

APPENDIX A – OMITTED METHODOLOGY

The omitted methodology mentioned in the Overview of Procedure section can be found with this Appendix. Numerous sections of the OWASP Web Application Methodology were omitted due to the sections either being out of scope or having little to no abnormal activity which could be concluded as insecure. The methodology is very thorough, and the web application does not have all features and mechanisms which are mentioned within the OWASP Web Application Methodology. Below is a list of all the omitted sections due to the previously mentioned reasons:

- Information Gathering
 - Conduct Search Engine Discovery and Reconnaissance for Information Leakage
 - Map Application Architecture
- Configuration and Deploy Management Testing
 - Test Network/Infrastructure Configuration
 - Test File Extensions Handling for Sensitive Information
 - Backup and Unreferenced Files for Sensitive Information
 - Test HTTP Methods
 - Test RIA cross domain policy
- Identity Management Testing
 - Test Account Provisioning Process
 - Testing for Account Enumeration and Guessable User Account
 - Test Permissions of Guest/Training Accounts
 - Test Account Suspension/Resumption Process
- Authentication Testing
 - Testing for Credentials Transported over an Encrypted Channel
 - Testing for Bypassing Authentication Schema
 - Testing for Browser Cache Weaknesses
 - Testing for Weak Security Question/Answer
 - Testing for Weaker Authentication in Alternative Channel
- Authorization Testing
 - Testing for Bypassing Authorization Schema
 - Testing for Privilege Escalation
 - Testing for Insecure Direct Object References
- Session Management Testing
 - Testing for Exposed Session Variables
 - Testing for Cross Site Request Forgery
 - Testing Session Timeout
 - Testing for Session Puzzling
- Data Validation Testing
 - Testing for HTTP Verb Tampering
 - Testing for HTTP Parameter Pollution
 - Oracle Testing
 - MySQL Testing
 - SQL Server Testing

- Testing PostgreSQL
 - MS Access Testing
 - Testing for NoSQL Injection
 - Testing for LDAP Injection
 - Testing for ORM Injection
 - Testing for XML Injection
 - Testing for SSI Injection
 - Testing for XPath Injection
 - IMAP/SMTP Injection
 - Testing for Code Injection
 - Testing for Remote File Inclusion
 - Testing for Command Injection
 - Testing for Buffer overflow
 - Testing for Heap overflow
 - Testing for Stack overflow
 - Testing for Format string
 - Testing for incubated vulnerabilities
 - Testing for HTTP Splitting/Smuggling
- Error Handling
 - Analysis of Stack Traces
- Cryptography
 - Testing for Padding Oracle
 - Testing for Sensitive information sent via unencrypted channels
- Business Logic Testing
 - Test Business Logic Data Validation
 - Test Integrity Checks
 - Test for Process Timing
 - Testing for the Circumvention of Work Flows
 - Test Defenses Against Application Mis-use
 - Test Upload of Unexpected File Types
 - Test Upload of Malicious Files
- Client Side Testing (WHOLE SECTION)

APPENDIX B – DATA AND FILES WITHIN THE WEBSITE

4.1.1 URLs and Files Section

Processed,Method,URI,Flags

true,GET,http://192.168.1.10,Seed

true,GET,http://192.168.1.10/robots.txt,Seed

true,GET,http://192.168.1.10/sitemap.xml,Seed

true,GET,http://192.168.1.10/,Seed

true,GET,http://192.168.1.10/aboutus.php,Seed

true,GET,http://192.168.1.10/access-denied.php,Seed

true,GET,http://192.168.1.10/affix.php?type=terms.php,Seed
true,GET,http://192.168.1.10/cart-exec.php?id=1,Seed
true,GET,http://192.168.1.10/contactus.php,Seed
true,GET,http://192.168.1.10/favicon.ico,Seed
true,GET,http://192.168.1.10/foodzone.php,Seed
true,GET,http://192.168.1.10/images,Seed
true,GET,http://192.168.1.10/index.php,Seed
true,GET,http://192.168.1.10/login-exec.php,Seed
true,GET,http://192.168.1.10/login-register.php,Seed
true,GET,http://192.168.1.10/member-index.php,Seed
true,GET,http://192.168.1.10/member-ratings.php,Seed
true,GET,http://192.168.1.10/register-exec.php,Seed
true,GET,http://192.168.1.10/register-success.php,Seed
true,GET,http://192.168.1.10/stylesheets,Seed
true,GET,http://192.168.1.10/stylesheets/images,Seed
true,GET,http://192.168.1.10/stylesheets/images/icon_menu.gif,Seed
true,GET,http://192.168.1.10/stylesheets/user_styles.css,Seed
true,GET,http://192.168.1.10/swf,Seed
true,GET,http://192.168.1.10/swf/swfobject.js,Seed
true,GET,http://192.168.1.10/validation,Seed
true,GET,http://192.168.1.10/validation/user.js,Seed
true,GET,http://192.168.1.10/schema.sql,
false,GET,http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd,Out of Scope
false,GET,http://www.w3.org/TR/html4/loose.dtd,Out of Scope
false,GET,http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd,Out of Scope
true,GET,http://192.168.1.10/images/pizza-inn-map4-mombasa-road.png,
true,GET,http://192.168.1.10/images/img001.png,
true,GET,http://192.168.1.10/images/img002.png,
true,GET,http://192.168.1.10/cart-exec.php?id=2,

true,GET,http://192.168.1.10/images/img003.png,
true,GET,http://192.168.1.10/cart-exec.php?id=3,
true,GET,http://192.168.1.10/images/img004.png,
true,GET,http://192.168.1.10/cart-exec.php?id=4,
true,GET,http://192.168.1.10/images/img005.png,
true,GET,http://192.168.1.10/cart-exec.php?id=5,
true,GET,http://192.168.1.10/images/img006.png,
true,GET,http://192.168.1.10/cart-exec.php?id=6,
true,GET,http://192.168.1.10/images/img007.png,
true,GET,http://192.168.1.10/cart-exec.php?id=7,
true,GET,http://192.168.1.10/images/img008.png,
true,GET,http://192.168.1.10/cart-exec.php?id=8,
true,GET,http://192.168.1.10/images/,
true,GET,http://192.168.1.10/images/img009.png,
true,GET,http://192.168.1.10/cart-exec.php?id=9,
true,GET,http://192.168.1.10/images/img010.png,
true,GET,http://192.168.1.10/cart-exec.php?id=10,
true,GET,http://192.168.1.10/images/img011.png,
true,GET,http://192.168.1.10/cart-exec.php?id=11,
true,GET,http://192.168.1.10/images/img012.png,
true,GET,http://192.168.1.10/cart-exec.php?id=12,
true,GET,http://192.168.1.10/images/img013.png,
true,GET,http://192.168.1.10/cart-exec.php?id=13,
true,GET,http://192.168.1.10/images/img014.png,
true,GET,http://192.168.1.10/cart-exec.php?id=14,
true,GET,http://192.168.1.10/images/img015.png,
true,GET,http://192.168.1.10/cart-exec.php?id=15,
true,GET,http://192.168.1.10/images/img016.png,
true,GET,http://192.168.1.10/cart-exec.php?id=16,

true,GET,http://192.168.1.10/images/img017.png,
true,GET,http://192.168.1.10/cart-exec.php?id=17,
true,GET,http://192.168.1.10/images/img018.png,
true,GET,http://192.168.1.10/cart-exec.php?id=18,
true,GET,http://192.168.1.10/images/img019.png,
true,GET,http://192.168.1.10/cart-exec.php?id=19,
true,GET,http://192.168.1.10/images/img020.png,
true,GET,http://192.168.1.10/cart-exec.php?id=20,
true,GET,http://192.168.1.10/images/img021.png,
true,GET,http://192.168.1.10/cart-exec.php?id=21,
true,GET,http://192.168.1.10/images/img022.png,
true,GET,http://192.168.1.10/cart-exec.php?id=22,
true,GET,http://192.168.1.10/images/img023.png,
true,GET,http://192.168.1.10/cart-exec.php?id=23,
true,GET,http://192.168.1.10/images/img024.png,
true,GET,http://192.168.1.10/cart-exec.php?id=24,
true,GET,http://192.168.1.10/images/img025.png,
true,GET,http://192.168.1.10/cart-exec.php?id=25,
true,GET,http://192.168.1.10/member-profile.php,
true,GET,http://192.168.1.10/cart.php,
true,GET,http://192.168.1.10/inbox.php,
true,GET,http://192.168.1.10/tables.php,
true,GET,http://192.168.1.10/partyhalls.php,
true,GET,http://192.168.1.10/ratings.php,
true,GET,http://192.168.1.10/logout.php,
true,GET,http://192.168.1.10/pictures/,
true,GET,http://192.168.1.10/register-failed.php,
true,GET,http://192.168.1.10/stylesheets/,
true,GET,http://192.168.1.10/swf/,

false,GET,http://code.google.com/p/swfobject/,Out of Scope
false,GET,http://www.opensource.org/licenses/mit-license.php,Out of Scope
true,GET,http://192.168.1.10/validation/,
true,GET,http://192.168.1.10/images/?C=N;O=D,
true,GET,http://192.168.1.10/images/?C=M;O=A,
true,GET,http://192.168.1.10/images/?C=S;O=A,
true,GET,http://192.168.1.10/images/?C=D;O=A,
true,GET,http://192.168.1.10/images/base-bg.gif,
true,GET,http://192.168.1.10/images/head-img.jpg,
true,GET,http://192.168.1.10/images/head-img2.jpg,
true,GET,http://192.168.1.10/images/icon_menu.gif,
true,GET,http://192.168.1.10/images/logo.gif,
true,GET,http://192.168.1.10/images/logo2.gif,
true,GET,http://192.168.1.10/images/pizza/,
true,GET,http://192.168.1.10/images/special.jpg,
true,GET,http://192.168.1.10/icons/blank.gif,
true,GET,http://192.168.1.10/icons/back.gif,
true,GET,http://192.168.1.10/icons/image2.gif,
true,GET,http://192.168.1.10/icons/folder.gif,
true,GET,http://192.168.1.10/css/bootstrap.css,
true,GET,http://192.168.1.10/js/jquery.js,
true,GET,http://192.168.1.10/js/bootstrap.js,
true,GET,http://192.168.1.10/order-exec.php?id=3,
true,GET,http://192.168.1.10/order-exec.php?id=4,
true,GET,http://192.168.1.10/order-exec.php?id=5,
true,GET,http://192.168.1.10/order-exec.php?id=6,
true,GET,http://192.168.1.10/order-exec.php?id=7,
true,GET,http://192.168.1.10/order-exec.php?id=8,
true,GET,http://192.168.1.10/order-exec.php?id=9,

true,GET,http://192.168.1.10/order-exec.php?id=10,
true,GET,http://192.168.1.10/order-exec.php?id=11,
true,GET,http://192.168.1.10/order-exec.php?id=12,
true,GET,http://192.168.1.10/order-exec.php?id=13,
true,GET,http://192.168.1.10/order-exec.php?id=14,
true,GET,http://192.168.1.10/order-exec.php?id=15,
true,GET,http://192.168.1.10/order-exec.php?id=16,
true,GET,http://192.168.1.10/order-exec.php?id=17,
true,GET,http://192.168.1.10/order-exec.php?id=18,
true,GET,http://192.168.1.10/order-exec.php?id=19,
true,GET,http://192.168.1.10/order-exec.php?id=20,
true,GET,http://192.168.1.10/order-exec.php?id=21,
true,GET,http://192.168.1.10/order-exec.php?id=22,
true,GET,http://192.168.1.10/order-exec.php?id=23,
true,GET,http://192.168.1.10/order-exec.php?id=24,
true,GET,http://192.168.1.10/order-exec.php?id=25,
true,GET,http://192.168.1.10/order-exec.php?id=26,
true,GET,http://192.168.1.10/pictures/?C=N;O=D,
true,GET,http://192.168.1.10/pictures/?C=M;O=A,
true,GET,http://192.168.1.10/pictures/?C=S;O=A,
true,GET,http://192.168.1.10/pictures/?C=D;O=A,
true,GET,http://192.168.1.10/pictures/Professional_Logo_blur_bckng.png,
true,GET,http://192.168.1.10/pictures/fluffy.jpg,
true,GET,http://192.168.1.10/pictures/rick.jpg,
true,GET,http://192.168.1.10/pictures,
true,GET,http://192.168.1.10/swf/?C=N;O=D,
true,GET,http://192.168.1.10/swf/?C=M;O=A,
true,GET,http://192.168.1.10/swf/?C=S;O=A,
true,GET,http://192.168.1.10/swf/?C=D;O=A,

true,GET,http://192.168.1.10/swf/Carousel.swf,
true,GET,http://192.168.1.10/swf/default.xml,
true,GET,http://192.168.1.10/icons/unknown.gif,
true,GET,http://192.168.1.10/icons/text.gif,
true,GET,http://192.168.1.10/stylesheets/?C=N;O=D,
true,GET,http://192.168.1.10/stylesheets/?C=M;O=A,
true,GET,http://192.168.1.10/stylesheets/?C=S;O=A,
true,GET,http://192.168.1.10/stylesheets/?C=D;O=A,
true,GET,http://192.168.1.10/validation/?C=N;O=D,
true,GET,http://192.168.1.10/validation/?C=M;O=A,
true,GET,http://192.168.1.10/validation/?C=S;O=A,
true,GET,http://192.168.1.10/validation/?C=D;O=A,
true,GET,http://192.168.1.10/images/?C=N;O=A,
true,GET,http://192.168.1.10/images/?C=M;O=D,
true,GET,http://192.168.1.10/images/?C=S;O=D,
true,GET,http://192.168.1.10/images/?C=D;O=D,
true,GET,http://192.168.1.10/images/pizza/?C=N;O=D,
true,GET,http://192.168.1.10/images/pizza/?C=M;O=A,
true,GET,http://192.168.1.10/images/pizza/?C=S;O=A,
true,GET,http://192.168.1.10/images/pizza/?C=D;O=A,
true,GET,http://192.168.1.10/images/pizza/Romans.xcf,
true,GET,http://192.168.1.10/images/pizza/img001.png,
true,GET,http://192.168.1.10/images/pizza/img002.png,
true,GET,http://192.168.1.10/images/pizza/img003.png,
true,GET,http://192.168.1.10/images/pizza/img004.png,
true,GET,http://192.168.1.10/images/pizza/img005.png,
true,GET,http://192.168.1.10/images/pizza/img006.png,
true,GET,http://192.168.1.10/images/pizza/img007.png,
true,GET,http://192.168.1.10/images/pizza/img008.png,

true,GET,http://192.168.1.10/images/pizza/img009.png,
true,GET,http://192.168.1.10/images/pizza/img010.png,
true,GET,http://192.168.1.10/images/pizza/img011.png,
true,GET,http://192.168.1.10/images/pizza/img012.png,
true,GET,http://192.168.1.10/images/pizza/img013.png,
true,GET,http://192.168.1.10/images/pizza/img014.png,
true,GET,http://192.168.1.10/images/pizza/img015.png,
true,GET,http://192.168.1.10/images/pizza/img016.png,
true,GET,http://192.168.1.10/images/pizza/img017.png,
true,GET,http://192.168.1.10/images/pizza/img018.png,
true,GET,http://192.168.1.10/images/pizza/img019.png,
true,GET,http://192.168.1.10/images/pizza/img020.png,
true,GET,http://192.168.1.10/images/pizza/img021.png,
true,GET,http://192.168.1.10/images/pizza/img022.png,
true,GET,http://192.168.1.10/images/pizza/img023.png,
true,GET,http://192.168.1.10/images/pizza/img024.png,
true,GET,http://192.168.1.10/images/pizza/img025.png,
true,GET,http://192.168.1.10/images/pizza/unavailable.png,
true,GET,http://192.168.1.10/images/pizza,
false,GET,http://www.apache.org/licenses/LICENSE-2.0,Out of Scope
false,GET,http://twitter.github.com/bootstrap/javascript.html,Out of Scope
false,GET,http://www.modernizr.com/,Out of Scope
true,GET,http://192.168.1.10/billing-success.php,
true,GET,http://192.168.1.10/reserve-success.php,
true,GET,http://192.168.1.10/ratings-success.php,
true,GET,http://192.168.1.10/pictures/?C=N;O=A,
true,GET,http://192.168.1.10/pictures/?C=M;O=D,
true,GET,http://192.168.1.10/pictures/?C=S;O=D,
true,GET,http://192.168.1.10/pictures/?C=D;O=D,

true,GET,http://192.168.1.10/swf/?C=N;O=A,
true,GET,http://192.168.1.10/swf/?C=M;O=D,
true,GET,http://192.168.1.10/swf/?C=S;O=D,
true,GET,http://192.168.1.10/swf/?C=D;O=D,
false,GET,http://www.flshow.net/,Out of Scope
true,GET,http://192.168.1.10/stylesheets/?C=N;O=A,
true,GET,http://192.168.1.10/stylesheets/?C=M;O=D,
true,GET,http://192.168.1.10/stylesheets/?C=S;O=D,
true,GET,http://192.168.1.10/stylesheets/?C=D;O=D,
true,GET,http://192.168.1.10/validation/?C=N;O=A,
true,GET,http://192.168.1.10/validation/?C=M;O=D,
true,GET,http://192.168.1.10/validation/?C=S;O=D,
true,GET,http://192.168.1.10/validation/?C=D;O=D,
true,GET,http://192.168.1.10/images/pizza/?C=N;O=A,
true,GET,http://192.168.1.10/images/pizza/?C=M;O=D,
true,GET,http://192.168.1.10/images/pizza/?C=S;O=D,
true,GET,http://192.168.1.10/images/pizza/?C=D;O=D,
true,POST,http://192.168.1.10/login-exec.php,
true,POST,http://192.168.1.10/register-exec.php,
true,POST,http://192.168.1.10/foodzone.php,
true,POST,http://192.168.1.10/Changepassword.php?id=19,
true,POST,http://192.168.1.10/billing-exec.php?id=19,
true,POST,http://192.168.1.10/changepicture.php,
true,POST,http://192.168.1.10/update-quantity.php,
true,POST,http://192.168.1.10/reserve-exec.php?id=19,
true,POST,http://192.168.1.10/ratings-exec.php?id=19,
true,POST,http://192.168.1.10/reserve-exec.php?id=19,

APPENDIX C – CONSOLE OUTPUT OF TOOLS

4.1.2 Section 1 – Nikto Output

- Nikto v2.1.6/2.1.5

+ Target Host: 192.168.1.10

+ Target Port: 80

+ GET Retrieved x-powered-by header: PHP/5.4.7

+ GET The anti-clickjacking X-Frame-Options header is not present.

+ GET The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS

+ GET 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

+ GET Entry '/schema.sql' in robots.txt returned a non-forbidden or redirect HTTP code (200)

+ GET Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. See <http://www.wisec.it/sectou.php?id=4698ebdc59d15>. The following alternatives for 'index' were found: HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var

+ HEAD PHP/5.4.7 appears to be outdated (current is at least 7.2.12). PHP 5.6.33, 7.0.27, 7.1.13, 7.2.1 may also current release for each branch.

+ HEAD Apache/2.4.3 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch.

+ OSVDB-112004: GET /cgi-bin/printenv: Site appears vulnerable to the 'shellshock' vulnerability (CVE-2014-6271).

+ OSVDB-112004: GET /cgi-bin/printenv: Site appears vulnerable to the 'shellshock' vulnerability (CVE-2014-6278).

+ KUFKNPRG Web Server returns a valid response with junk HTTP methods, this may cause false positives.

+ OSVDB-877: TRACE HTTP TRACE method is active, suggesting the host is vulnerable to XST

+ GET /phpinfo.php: Output from the phpinfo() function was found.

+ GET Cookie PHPSESSID created without the httponly flag

- + OSVDB-12184: GET /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
- + OSVDB-12184: GET /?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
- + OSVDB-12184: GET /?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
- + OSVDB-12184: GET /?=PHPE9568F35-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
- + OSVDB-3268: GET /css/: Directory indexing found.
- + OSVDB-3092: GET /css/: This might be interesting...
- + OSVDB-3268: GET /install/: Directory indexing found.
- + OSVDB-3092: GET /install/: This might be interesting...
- + OSVDB-3268: GET /stylesheets/: Directory indexing found.
- + OSVDB-3092: GET /stylesheets/: This might be interesting...
- + OSVDB-3233: GET /cgi-bin/printenv: Apache 2.0 default script is executable and gives server environment variables. All default scripts should be removed. It may also allow XSS types of attacks. BID-4431.
- + OSVDB-3233: GET /cgi-bin/test-cgi: Apache 2.0 default script is executable and reveals system information. All default scripts should be removed.
- + OSVDB-3233: GET /phpinfo.php: PHP is installed, and a test script which runs phpinfo() was found. This gives a lot of system information.
- + OSVDB-3233: GET /info.php: PHP is installed, and a test script which runs phpinfo() was found. This gives a lot of system information.
- + OSVDB-3268: GET /icons/: Directory indexing found.
- + OSVDB-3268: GET /images/: Directory indexing found.
- + OSVDB-3268: GET /docs/: Directory indexing found.
- + OSVDB-3233: GET /icons/README: Apache default file found.
- + OSVDB-5292: GET /info.php?file=http://cirt.net/rfiinc.txt?: RFI from RSnake's list (<http://ha.ckers.org/weird/rfi-locations.dat>) or from <http://osvdb.org/>

4.1.3 Section 2 – Dirb Output

DIRB v2.22

Adam Board

By The Dark Raver

OUTPUT_FILE: ./Desktop/DIRBoutput.txt

START_TIME: Wed Dec 7 18:34:09 2022

URL_BASE: http://192.168.1.10/

WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

GENERATED WORDS: 4612

---- Scanning URL: http://192.168.1.10/ ----

+ http://192.168.1.10/admin.cgi (CODE:403|SIZE:975)

+ http://192.168.1.10/admin.pl (CODE:403|SIZE:975)

+ http://192.168.1.10/AT-admin.cgi (CODE:403|SIZE:975)

+ http://192.168.1.10/cachemgr.cgi (CODE:403|SIZE:975)

+ http://192.168.1.10/cgi-bin/ (CODE:403|SIZE:989)

+ http://192.168.1.10/index.php (CODE:200|SIZE:5978)

+ http://192.168.1.10/info.php (CODE:200|SIZE:266321)

+ http://192.168.1.10/phpinfo.php (CODE:200|SIZE:76831)

+ http://192.168.1.10/phpmyadmin (CODE:401|SIZE:1222)

```
==> DIRECTORY: http://192.168.1.10/pictures/
+ http://192.168.1.10/robots.txt (CODE:200|SIZE:36)
==> DIRECTORY: http://192.168.1.10/stylesheets/
==> DIRECTORY: http://192.168.1.10/swf/
==> DIRECTORY: http://192.168.1.10/validation/
==> DIRECTORY: http://192.168.1.10/videos/
```

```
---- Entering directory: http://192.168.1.10/~admin/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
      (Use mode '-w' if you want to scan it anyway)
```

```
---- Entering directory: http://192.168.1.10/admin/ ----
+ http://192.168.1.10/admin/admin.cgi (CODE:403|SIZE:975)
+ http://192.168.1.10/admin/admin.pl (CODE:403|SIZE:975)
+ http://192.168.1.10/admin/AT-admin.cgi (CODE:403|SIZE:975)
+ http://192.168.1.10/admin/cachemgr.cgi (CODE:403|SIZE:975)
+ http://192.168.1.10/admin/index.php (CODE:302|SIZE:0)
==> DIRECTORY: http://192.168.1.10/admin/stylesheets/
==> DIRECTORY: http://192.168.1.10/admin/validation/
```

```
---- Entering directory: http://192.168.1.10/css/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
      (Use mode '-w' if you want to scan it anyway)
```

```
---- Entering directory: http://192.168.1.10/docs/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
      (Use mode '-w' if you want to scan it anyway)
```

```
---- Entering directory: http://192.168.1.10/images/ ----
```

(!) WARNING: Directory IS LISTABLE. No need to scan it.

(Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.10/install/ ----

(!) WARNING: Directory IS LISTABLE. No need to scan it.

(Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.10/js/ ----

(!) WARNING: Directory IS LISTABLE. No need to scan it.

(Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.10/pictures/ ----

(!) WARNING: Directory IS LISTABLE. No need to scan it.

(Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.10/stylesheets/ ----

(!) WARNING: Directory IS LISTABLE. No need to scan it.

(Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.10/swf/ ----

(!) WARNING: Directory IS LISTABLE. No need to scan it.

(Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.10/validation/ ----

(!) WARNING: Directory IS LISTABLE. No need to scan it.

(Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.10/videos/ ----

(!) WARNING: Directory IS LISTABLE. No need to scan it.

Adam Board

(Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.10/admin/stylesheets/ ----

(!) WARNING: Directory IS LISTABLE. No need to scan it.

(Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.10/admin/validation/ ----

(!) WARNING: Directory IS LISTABLE. No need to scan it.

(Use mode '-w' if you want to scan it anyway)

END_TIME: Wed Dec 7 18:34:28 2022

DOWNLOADED: 9224 - FOUND: 15

APPENDIX D – GUI TOOL OUTPUT

4.1.4 OWASP Zap Scan Report Output

ZAP Scanning Report

Generated with The ZAP logoZAP on Wed 7 Dec 2022, at 09:37:11

Contents

About this report

Report parameters

Summaries

Alert counts by risk and confidence

Alert counts by site and risk

Alert counts by alert type

Alerts

Adam Board

Risk=High, Confidence=Medium (4)

Risk=Medium, Confidence=High (1)

Risk=Medium, Confidence=Medium (4)

Risk=Medium, Confidence=Low (1)

Risk=Low, Confidence=High (1)

Risk=Low, Confidence=Medium (4)

Risk=Informational, Confidence=Medium (2)

Risk=Informational, Confidence=Low (2)

Appendix

Alert types

About this report

Report parameters

Contexts

No contexts were selected, so all contexts were included by default.

Sites

The following sites were included:

<http://192.168.1.10>

(If no sites were selected, all sites were included by default.)

An included site must also be within one of the included contexts for its data to be included in the report.

Risk levels

Included: High, Medium, Low, Informational

Adam Board

Excluded: None

Confidence levels

Included: User Confirmed, High, Medium, Low

Excluded: User Confirmed, High, Medium, Low, False Positive

Summaries

Alert counts by risk and confidence

This table shows the number of alerts for each level of risk and confidence included in the report.

(The percentages in brackets represent the count as a percentage of the total number of alerts included in the report, rounded to one decimal place.)

		Confidence			
User Confirmed		High	Medium	Low	Total
Risk	High	0			
	(0.0%)	0			
	(0.0%)	4			
	(21.1%)	0			
	(0.0%)	4			
Medium	(21.1%)				
	Medium	0			
	(0.0%)	1			
	(5.3%)	4			
	(21.1%)	1			
Low	(5.3%)	6			
	(31.6%)				
	Low	0			

Adam Board

(0.0%)	1	
(5.3%)	4	
(21.1%)	0	
(0.0%)	5	
(26.3%)		
Informational	0	
(0.0%)	0	
(0.0%)	2	
(10.5%)	2	
(10.5%)	4	
(21.1%)		
Total	0	
(0.0%)	2	
(10.5%)	14	
(73.7%)	3	
(15.8%)	19	
(100%)		

Alert counts by site and risk

This table shows, for each site for which one or more alerts were raised, the number of alerts raised at each risk level.

Alerts with a confidence level of "False Positive" have been excluded from these counts.

(The numbers in brackets are the number of alerts raised for the site at or above that risk level.)

Risk	
High	
(= High)	Medium
(>= Medium)	Low

(>= Low) Informational

(>= Informational)

Site http://192.168.1.10 4

(4) 6

(10) 5

(15) 4

(19)

Alert counts by alert type

This table shows the number of alerts of each alert type, together with the alert type's risk level.

(The percentages in brackets represent each count as a percentage, rounded to one decimal place, of the total number of alerts included in this report.)

Alert type Risk Count

Cross Site Scripting (Persistent) High 1

(5.3%)

Path Traversal High 1

(5.3%)

SQL Injection High 2

(10.5%)

SQL Injection - MySQL High 2

(10.5%)

Absence of Anti-CSRF Tokens Medium 17

(89.5%)

Application Error Disclosure Medium 54

(284.2%)

Content Security Policy (CSP) Header Not Set Medium 85

(447.4%)

Directory Browsing - Apache 2 Medium 54

Adam Board

(284.2%)

Missing Anti-clickjacking Header	Medium	81
----------------------------------	--------	----

(426.3%)

Vulnerable JS Library	Medium	2
-----------------------	--------	---

(10.5%)

Cookie No HttpOnly Flag	Low	4
-------------------------	-----	---

(21.1%)

Cookie without SameSite Attribute	Low	4
-----------------------------------	-----	---

(21.1%)

Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)	Low	83
---	-----	----

(436.8%)

Server Leaks Version Information via "Server" HTTP Response Header Field	Low	226
--	-----	-----

(1,189.5%)

X-Content-Type-Options Header Missing	Low	160
---------------------------------------	-----	-----

(842.1%)

Content-Type Header Missing	Informational	2
-----------------------------	---------------	---

(10.5%)

Information Disclosure - Suspicious Comments	Informational	20
--	---------------	----

(105.3%)

Modern Web Application	Informational	6
------------------------	---------------	---

(31.6%)

User Controllable HTML Element Attribute (Potential XSS)	Informational	1
--	---------------	---

(5.3%)

Total	19
-------	----

Alerts

Risk=High, Confidence=Medium (4)

http://192.168.1.10 (4)

Cross Site Scripting (Persistent) (1)

GET http://192.168.1.10/member-ratings.php

Path Traversal (1)

GET http://192.168.1.10/affix.php?type=%2Fetc%2Fpasswd

SQL Injection (1)

POST http://192.168.1.10/login-exec.php

SQL Injection - MySQL (1)

POST http://192.168.1.10/billing-exec.php?id=19

Risk=Medium, Confidence=High (1)

http://192.168.1.10 (1)

Content Security Policy (CSP) Header Not Set (1)

GET http://192.168.1.10/

Risk=Medium, Confidence=Medium (4)

http://192.168.1.10 (4)

Application Error Disclosure (1)

GET http://192.168.1.10/images/

Directory Browsing - Apache 2 (1)

GET http://192.168.1.10/images/

Missing Anti-clickjacking Header (1)

GET http://192.168.1.10/

Vulnerable JS Library (1)

GET http://192.168.1.10/swf/swfobject.js

Risk=Medium, Confidence=Low (1)

http://192.168.1.10 (1)

Absence of Anti-CSRF Tokens (1)

GET http://192.168.1.10/

Risk=Low, Confidence=High (1)

http://192.168.1.10 (1)

Server Leaks Version Information via "Server" HTTP Response Header Field (1)

GET http://192.168.1.10/stylesheets/user_styles.css

Risk=Low, Confidence=Medium (4)

http://192.168.1.10 (4)

Cookie No HttpOnly Flag (1)

GET http://192.168.1.10/cart-exec.php?id=1

Cookie without SameSite Attribute (1)

GET http://192.168.1.10/cart-exec.php?id=1

Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s) (1)

GET http://192.168.1.10/

X-Content-Type-Options Header Missing (1)

GET http://192.168.1.10/stylesheets/user_styles.css

Risk=Informational, Confidence=Medium (2)

http://192.168.1.10 (2)

Content-Type Header Missing (1)

GET http://192.168.1.10/schema.sql

Modern Web Application (1)

GET http://192.168.1.10/swf/swfobject.js

Risk=Informational, Confidence=Low (2)

http://192.168.1.10 (2)

Information Disclosure - Suspicious Comments (1)

GET http://192.168.1.10/

User Controllable HTML Element Attribute (Potential XSS) (1)

POST http://192.168.1.10/foodzone.php

Appendix

Alert types

This section contains additional information on the types of alerts in the report.

Cross Site Scripting (Persistent)

Source raised by an active scanner (Cross Site Scripting (Persistent))

CWE ID 79

WASC ID 8

Reference

<http://projects.webappsec.org/Cross-Site-Scripting>

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

Path Traversal

Source raised by an active scanner (Path Traversal)

CWE ID 22

WASC ID 33

Reference

<http://projects.webappsec.org/Path-Traversal>

<http://cwe.mitre.org/data/definitions/22.html>

SQL Injection

Source raised by an active scanner (SQL Injection)

CWE ID 89

WASC ID 19

Reference

https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html

SQL Injection - MySQL

Source raised by an active scanner (SQL Injection - MySQL)

CWE ID 89

WASC ID 19

Reference

https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html

Absence of Anti-CSRF Tokens

Source raised by a passive scanner (Absence of Anti-CSRF Tokens)

CWE ID 352

WASC ID 9

Reference

<http://projects.webappsec.org/Cross-Site-Request-Forgery>

<http://cwe.mitre.org/data/definitions/352.html>

Application Error Disclosure

Source raised by a passive scanner (Application Error Disclosure)

CWE ID 200

WASC ID 13

Content Security Policy (CSP) Header Not Set

Source raised by a passive scanner (Content Security Policy (CSP) Header Not Set)

CWE ID 693

WASC ID 15

Reference

https://developer.mozilla.org/en-US/docs/Web/Security/CSP/Introducing_Content_Security_Policy

https://cheatsheetseries.owasp.org/cheatsheets/Content_Security_Policy_Cheat_Sheet.html

<http://www.w3.org/TR/CSP/>

<http://w3c.github.io/webappsec/specs/content-security-policy/csp-specification.dev.html>

<http://www.html5rocks.com/en/tutorials/security/content-security-policy/>

<http://caniuse.com/#feat=contentsecuritypolicy>

<http://content-security-policy.com/>

Directory Browsing - Apache 2

Source raised by a passive scanner (Directory Browsing)

CWE ID 548

WASC ID 16

Reference

<https://cwe.mitre.org/data/definitions/548.html>

Missing Anti-clickjacking Header

Source raised by a passive scanner (Anti-clickjacking Header)

CWE ID 1021

WASC ID 15

Reference

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options>

Vulnerable JS Library

Source raised by a passive scanner (Vulnerable JS Library (Powered by Retire.js))

CWE ID 829

Reference

<https://github.com/swfobject/swfobject/wiki/SWFObject-Release-Notes#swfobject-v21-beta7-june-6th-2008>

Cookie No HttpOnly Flag

Source raised by a passive scanner (Cookie No HttpOnly Flag)

CWE ID 1004

WASC ID 13

Reference

<https://owasp.org/www-community/HttpOnly>

Cookie without SameSite Attribute

Source raised by a passive scanner (Cookie without SameSite Attribute)

CWE ID 1275

WASC ID 13

Reference

<https://tools.ietf.org/html/draft-ietf-httpbis-cookie-same-site>

Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)

Source raised by a passive scanner (Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s))

CWE ID 200

WASC ID 13

Reference

<http://blogs.msdn.com/b/varunm/archive/2013/04/23/remove-unwanted-http-response-headers.aspx>

<http://www.troyhunt.com/2012/02/shhh-dont-let-your-response-headers.html>

Server Leaks Version Information via "Server" HTTP Response Header Field

Source raised by a passive scanner (HTTP Server Response Header)

CWE ID 200

WASC ID 13

Reference

<http://httpd.apache.org/docs/current/mod/core.html#servertokens>

http://msdn.microsoft.com/en-us/library/ff648552.aspx#ht_urlscan_007

<http://blogs.msdn.com/b/varunm/archive/2013/04/23/remove-unwanted-http-response-headers.aspx>

<http://www.troyhunt.com/2012/02/shhh-dont-let-your-response-headers.html>

X-Content-Type-Options Header Missing

Source raised by a passive scanner (X-Content-Type-Options Header Missing)

CWE ID 693

WASC ID 15

Reference

<http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx>

<https://owasp.org/www-community/Security-Headers>

Content-Type Header Missing

Source raised by a passive scanner (Content-Type Header Missing)

CWE ID 345

WASC ID 12

Reference

<http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx>

Information Disclosure - Suspicious Comments

Source raised by a passive scanner (Information Disclosure - Suspicious Comments)

CWE ID 200

WASC ID 13

Modern Web Application

Source raised by a passive scanner (Modern Web Application)

User Controllable HTML Element Attribute (Potential XSS)

Adam Board

Source raised by a passive scanner (User Controllable HTML Element Attribute (Potential XSS))

CWE ID 20

WASC ID 20

Reference

<http://websecuritytool.codeplex.com/wikipage?title=Checks#user-controlled-html-attribute>