We Test Pens Incorporated

COMP90074 - Web Security Assignment

Jiahao Chen

1118749

**PENETRATION TEST REPORT FOR**

**Bank of UniMelb Pty. Ltd. - WEB APPLICATION**

**Report delivered: 01/06/2023**

# Executive Summary

Introduction

This penetration test report aims to present five discovered vulnerabilities in the self-service banking system launched by Bank of UniMelb. Penetration tests are performed using Burp Suite, Python and DirBuster. The request frequency is limited to 30 payloads per minute and the number of threads used in DirBuster is limited to 5.

Vulnerabilities

1. Bypassing developer login - Extreme

Authentication of developer login is placed on the client-side, which is easy for attackers to bypass and get developer access of the system. It is essential to put all authentication mechanisms on the server side.

2. Takeover branch manager’s account - Extreme

Attackers can reset passwords of other users in the *Settings* page, enabling them to take control of administrative accounts, such as a branch manager. It is necessary to ensure that users can only reset their own credentials.

3. Privilege escalation - Extreme

Users can promote themselves in the *Admin* page, which could to lead to unauthorized actions. Proper access controls could help to resolve this vulnerability.

4. IDOR in User Profile - High

Users can visit other users’ profiles by passing a hidden parameter *id*, which may lead to privacy disclosure. All users should only have access to their own resources.

5. Information leakage in testing folders - Medium

Some files are left behind during testing and development, which may cause information leakage. It is better to remove these files in a production environment.

Security Posture Assessment:

The identified vulnerabilities mentioned above indicate that the system is currently at risk of potential security hazards. In the following sections, each vulnerability will be described in detail, including its corresponding proof of concept, consequence, likelihood, risk rating, and recommended mitigations. While acknowledging the limitations of the company’s time and budget, it is still recommended to apply some useful security measures before the system's launch, including access controls, rate limiting, MFA and log monitoring.

Table of Contents

[Executive Summary 1](#_Toc136480194)

[Summary of Findings 3](#_Toc136480195)

[Detailed Findings 4](#_Toc136480196)

[Finding 1 – Bypassing developer login 4](#_Toc136480197)

[**Description** 4](#_Toc136480198)

[**Proof of Concept** 4](#_Toc136480199)

[**Consequence** 4](#_Toc136480200)

[**Likelihood** 4](#_Toc136480201)

[**Risk Rating** 4](#_Toc136480202)

[**Recommendation** 4](#_Toc136480203)

[**References** 4](#_Toc136480204)

[Finding 2 – IDOR in User Profile 5](#_Toc136480205)

[**Description** 5](#_Toc136480206)

[**Proof of Concept** 5](#_Toc136480207)

[**Consequence** 5](#_Toc136480208)

[**Likelihood** 5](#_Toc136480209)

[**Risk Rating** 5](#_Toc136480210)

[**Recommendation** 5](#_Toc136480211)

[**References** 5](#_Toc136480212)

[Finding 3 – Takeover branch manager’s account 6](#_Toc136480213)

[**Description** 6](#_Toc136480214)

[**Proof of Concept** 6](#_Toc136480215)

[**Consequence** 6](#_Toc136480216)

[**Likelihood** 6](#_Toc136480217)

[**Risk Rating** 6](#_Toc136480218)

[**Recommendation** 6](#_Toc136480219)

[**References** 6](#_Toc136480220)

[Finding 4 – Privilege escalation 7](#_Toc136480221)

[**Description** 7](#_Toc136480222)

[**Proof of Concept** 7](#_Toc136480223)

[**Consequence** 7](#_Toc136480224)

[**Likelihood** 7](#_Toc136480225)

[**Risk Rating** 7](#_Toc136480226)

[**Recommendation** 7](#_Toc136480227)

[**References** 7](#_Toc136480228)

[Finding 5 – Information leakage in testing folders 8](#_Toc136480229)

[**Description** 8](#_Toc136480230)

[**Proof of Concept** 8](#_Toc136480231)

[**Consequence** 8](#_Toc136480232)

[**Likelihood** 8](#_Toc136480233)

[**Risk Rating** 8](#_Toc136480234)

[**Recommendation** 8](#_Toc136480235)

[**References** 8](#_Toc136480236)

[Appendix 1 - Risk Matrix 9](#_Toc136480237)

[Appendix 2 - Additional Information 9](#_Toc136480238)

# Summary of Findings

A brief summary of all findings appears in the table below, sorted by risk rating.

|  |  |  |
| --- | --- | --- |
| **Risk** | **Reference** | **Vulnerability** |
| Extreme | [Finding 1](#_Finding_1_–) | Authentication of developer login is placed on the client-side, which is easy for attackers to bypass and get developer access of the system. |
| Extreme | [Finding 3](#_Finding_3_–) | Attackers can reset passwords of other users in the *Settings* page, enabling them to take control of a branch manager’s account. |
| Extreme | [Finding 4](#_Finding_4_–) | Users can promote themselves in the *Admin* page, which could to lead to unauthorized actions. |
| High | [Finding 2](#_Finding_2_–_1) | Users can visit other users’ profiles by passing a hidden parameter *id*, which may lead to privacy disclosure. |
| Medium | [Finding 5](#_Finding_5_–_1) | Some files are left behind during testing and development, which may cause information leakage. |

# Detailed Findings

This section provides detailed descriptions of all the vulnerabilities identified.

## Finding 1 – Bypassing developer login

|  |  |
| --- | --- |
| **Description** | Attackers can easily get developer access to the bank system since the authentication of developer login is placed on the client side. The password is hidden in the *authenticate()* function. |
| **Proof of Concept** | [Appendix 2 – Bypassing developer login](#_Finding_1_–_1) |
| **Consequence** | **Attackers may be able to manipulate the database if they have developer access. They can also inject malicious code into the system, which could lead to disclosure of sensitive information and huge financial losses for the bank and users. Moreover, they may modify the log files to hide their activities.** |
| **Likelihood** | **It is almost certain that this vulnerability could be exploited as bypassing client-side authentication is straightforward.** |
| **Risk Rating** | **Extreme**  The consequence of this vulnerability could range from major to catastrophic, depending on the privilege of developers. Meanwhile, it is almost certain to be exploited, thereby the risk rating extreme. |
| **Recommendation** | **Most importantly, it is necessary to ensure that all authentication mechanisms must be placed in the server side instead of the client side.**  **In addition, if developers have high privileges, it is better to implement MFA for developer login to confirm the identify of developers.** |
| **References** | JJDecode: <https://www.53lu.com/tool/jjencode/>  Lecture slides |

[Back to Summary of Findings](#_Summary_of_Findings)

## Finding 2 – IDOR in User Profile

|  |  |
| --- | --- |
| **Description** | In the *User Profile* page, users can visit other users’ profiles by passing a hidden parameter *id*, which may lead to privacy disclosure such as usernames. |
| **Proof of Concept** | [Appendix 2 – IDOR in User Profile](#_Finding_2_–) |
| **Consequence** | **Attackers could obtain other users’ usernames, which may help to performing other attacks, such as takeover other users’ accounts, resulting in potentially catastrophic consequences. Besides, if sensitive information is included in a user profile, this vulnerability could lead to privacy disclosure.** |
| **Likelihood** | **It is possible that this vulnerability could be exploited. The hidden parameter *id* is commonly used.** |
| **Risk Rating** | **High**  The consequence could be major for a bank system, and it is possible that the hidden parameter is discovered. Therefore, the risk rating is high. |
| **Recommendation** | **Implement access controls to ensure the users can only have access to their own resources.**  **Use indirect object references such as unique identifiers, which are more difficult to guess.**  **Logging and monitoring could also be helpful to identify abnormal activities, for example, visiting other users’ profiles.**  **Rate limiting could restrict automated scraping of user profiles to avoid widespread privacy leakage.** |
| **References** | Regex: <https://regex101.com/>  Lecture slides |

[Back to Summary of Findings](#_Summary_of_Findings)

## Finding 3 – Takeover branch manager’s account

|  |  |
| --- | --- |
| **Description** | The *Settings* page provides a password reset functionality. However, attackers can reset passwords of other users by removing the parameter for the old password and specifying a username. If attackers know an administrative username, for example, a branch manager, they can successfully take over the account, granting them unauthorized access to perform malicious actions. |
| **Proof of Concept** | [Appendix 2 – Takeover branch manager’s account](#_Finding_3_–_1) |
| **Consequence** | **Attackers may take control over other users’ accounts, resulting in denial of service or even financial losses for the affected users.**  **If a branch manager’s account is compromised, attackers can have access to administrative data and perform malicious actions. Moreover, it is difficult to identify the attackers if no log is kept, potentially leading to the branch manager being held legally liable.**  **In the worst-case scenario where a significant number of usernames are leaked, the consequence could be catastrophic as numerous accounts could be under unauthorized control.** |
| **Likelihood** | **It is likely that this vulnerability could be exploited. If attackers know other usernames, it is straightforward to take over user accounts.** |
| **Risk Rating** | **Extreme**  The vulnerability is likely to be exploited and the consequence could range from major to catastrophic for a bank. Therefore, the risk rating for this vulnerability is extreme. |
| **Recommendation** | **Firstly, the sever side need to ensure that the old password is correctly entered, and users can only reset their own passwords.**  **A more robust way is to use MFA when user attempts to reset the password. Even if user forgets the old password, MFA can still help to confirm the user’s identity.** |
| **References** | Lecture slides |

[Back to Summary of Findings](#_Summary_of_Findings)

## Finding 4 – Privilege escalation

|  |  |
| --- | --- |
| **Description** | Users can promote their privileges in the *Admin* page by setting the parameter *admin* to true, entering their usernames and finding an available role group. Attackers can therefore grant themselves unauthorized access to administrative data or actions. Besides, the promotion can be revoked so that attackers can possibly hide their activities. |
| **Proof of Concept** | [Appendix 2 – Privilege escalation](#_Finding_4_–_1) |
| **Consequence** | **Without takeover of administrative accounts, attackers can have access to administrative data and perform malicious actions, which may lead to sensitive data disclosure and management disorder. Moreover, if no log monitoring is implemented, it is difficult to identify the attacker as they can hide themselves easily by revoking the promotion.** |
| **Likelihood** | **It is likely that this vulnerability could be exploited since attackers are able to directly promote themselves by intercepting the transmission, without knowing other credentials.** |
| **Risk Rating** | **Extreme**  The vulnerability is likely to be exploited and the consequence is major for a bank system, which makes the risk rating extreme. |
| **Recommendation** | **Firstly, it is necessary to improve the implementation of access controls. For example, the verification of admin should be placed in the server side instead of the client side.**  **In addition, for administrative actions such as promoting users, it is better to have a further layer of authentication to confirm the identity of the administrator. MFA could be taken into consideration.**  **Finally, implementation of a log monitoring system is useful to identifying abnormal activities. For example, each administrative action should be reviewed to avoid vulnerabilities such as privilege escalation.** |
| **References** | Lecture slides |

[Back to Summary of Findings](#_Summary_of_Findings)

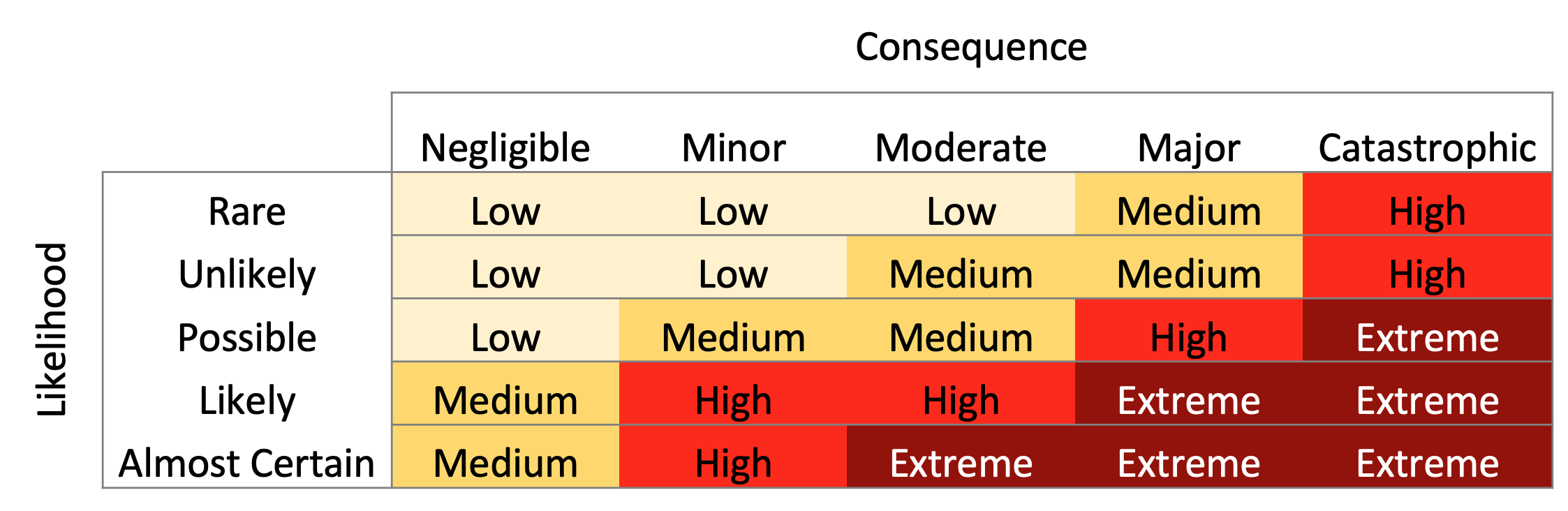
## Finding 5 – Information leakage in testing folders

|  |  |
| --- | --- |
| **Description** | A *test* folder is discovered, which may be left behind during testing and development. A *.git* folder is included in the test folder, which could lead to potential information leakage. |
| **Proof of Concept** | [Appendix 2 – Information leakage in testing folders](#_Finding_5_–) |
| **Consequence** | **Sensitive information stored in the test folder could be leaked, including credentials, source code or log files. Attackers may also find other vulnerabilities based on such files.** |
| **Likelihood** | **It is possible for the attackers to find *test* and *.git* since such directories are commonly seen during development.** |
| **Risk Rating** | **Medium**  Currently, there is no obviously sensitive information included in the *test* folder so that the consequence may be Minor. Since the likelihood is possible, the risk rating is considered as medium. |
| **Recommendation** | **It is better to ensure that testing and development folders are not included in the production environment. Despite developers following good security practices, it is hard to guarantee that attackers cannot find anything valuable from such files.**  **Besides, sensitive files could be prohibited from exposing to users by implementing access controls.**  **Moreover, rate limiting could be implemented to restrict automated tools such as DirBuster to better protect the hidden files.** |
| **References** | DirBuster: <https://gitlab.com/kalilinux/packages/dirbuster>  Wordlist: <https://chat.openai.com>  Regex: <https://regex101.com/>  Lecture slides |

[Back to Summary of Findings](#_Summary_of_Findings)

# Appendix 1 - Risk Matrix

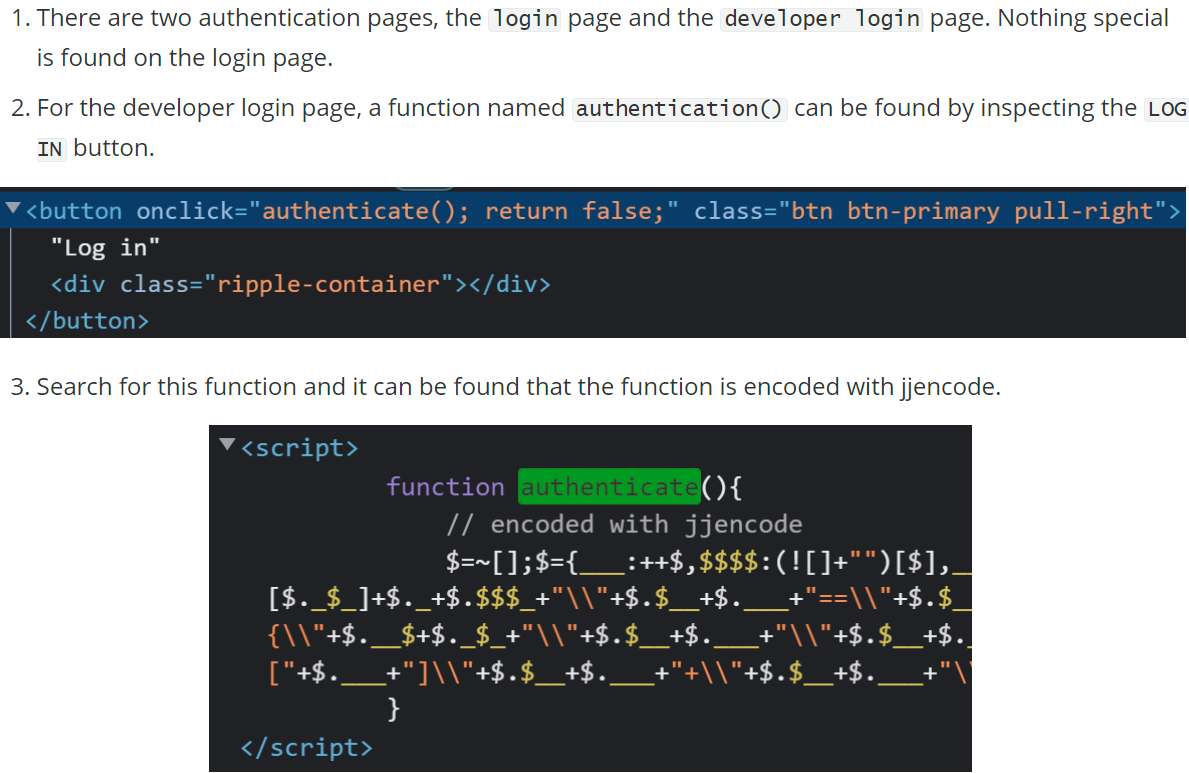
All risks assessed in this report are in line with the ISO31000 Risk Matrix detailed below:

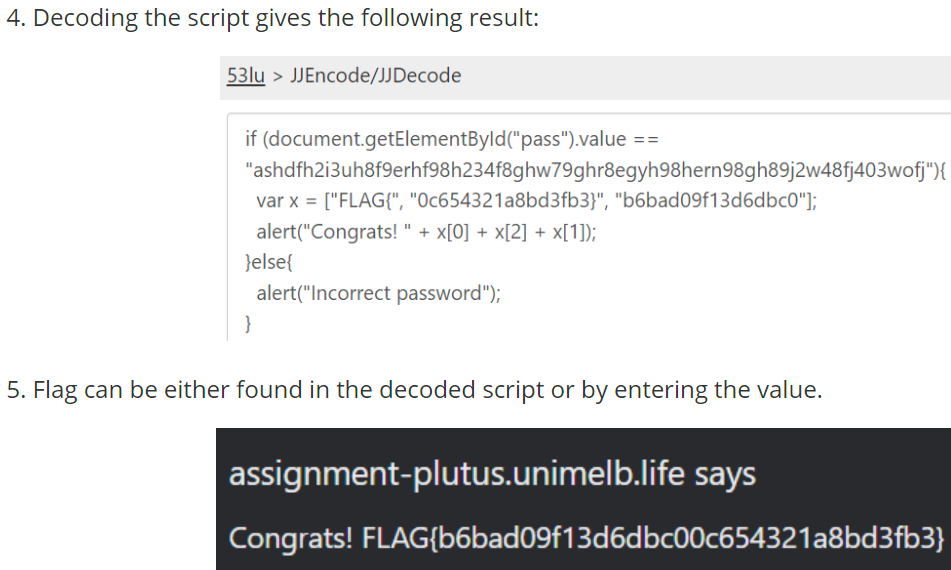


# Appendix 2 - Additional Information

Proof of concept for each vulnerability are presented here in detail.

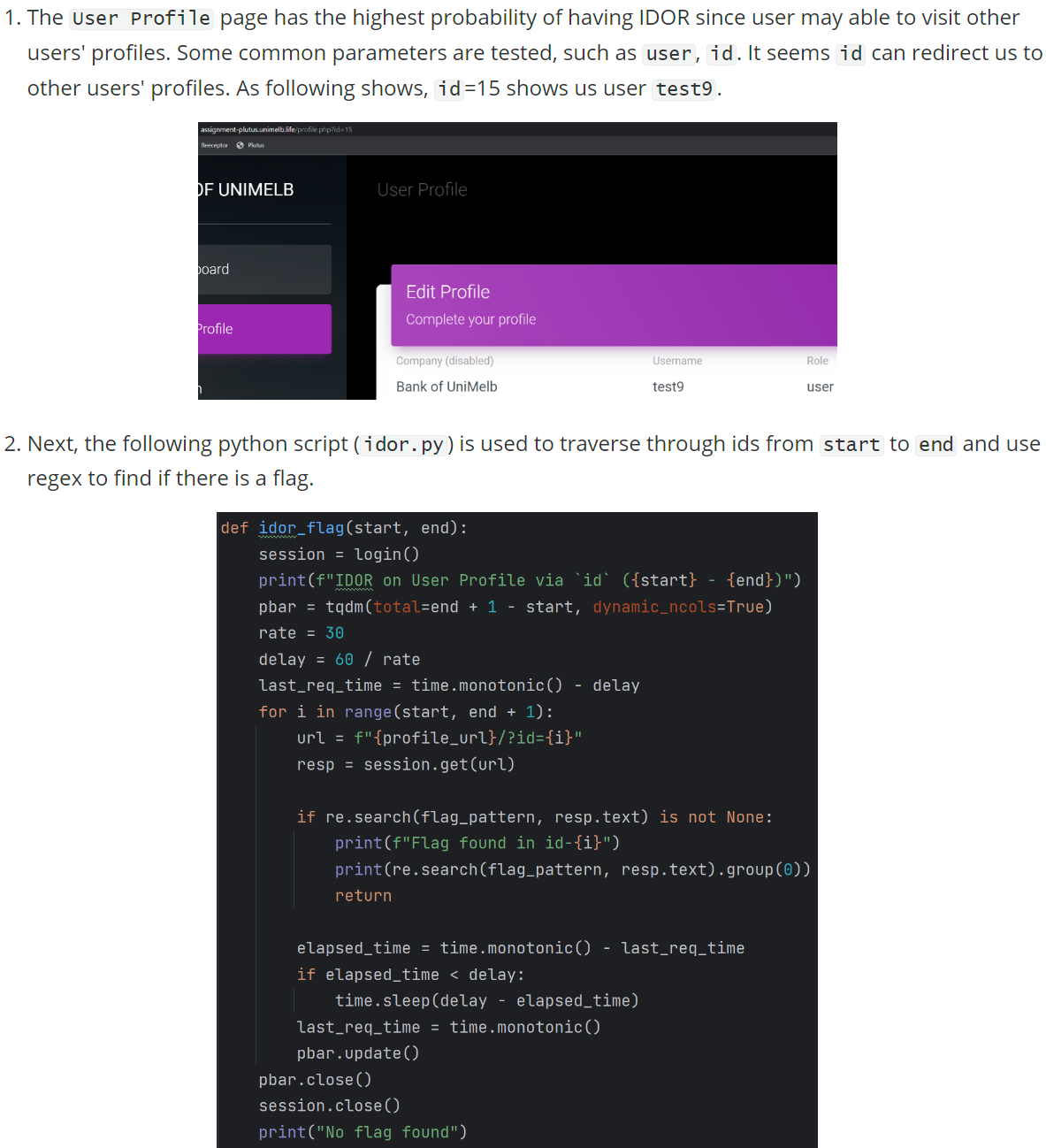
# Finding 1 – Bypassing developer login

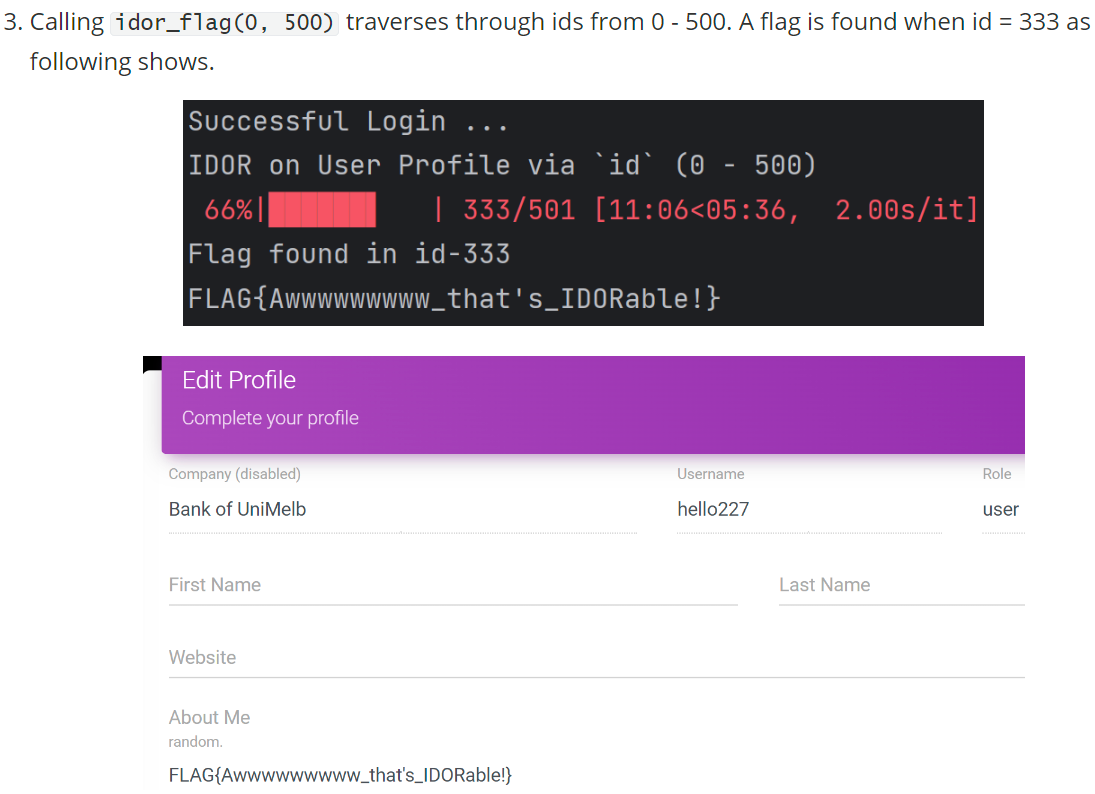




[Back to Finding 1](#_Finding_1_–)

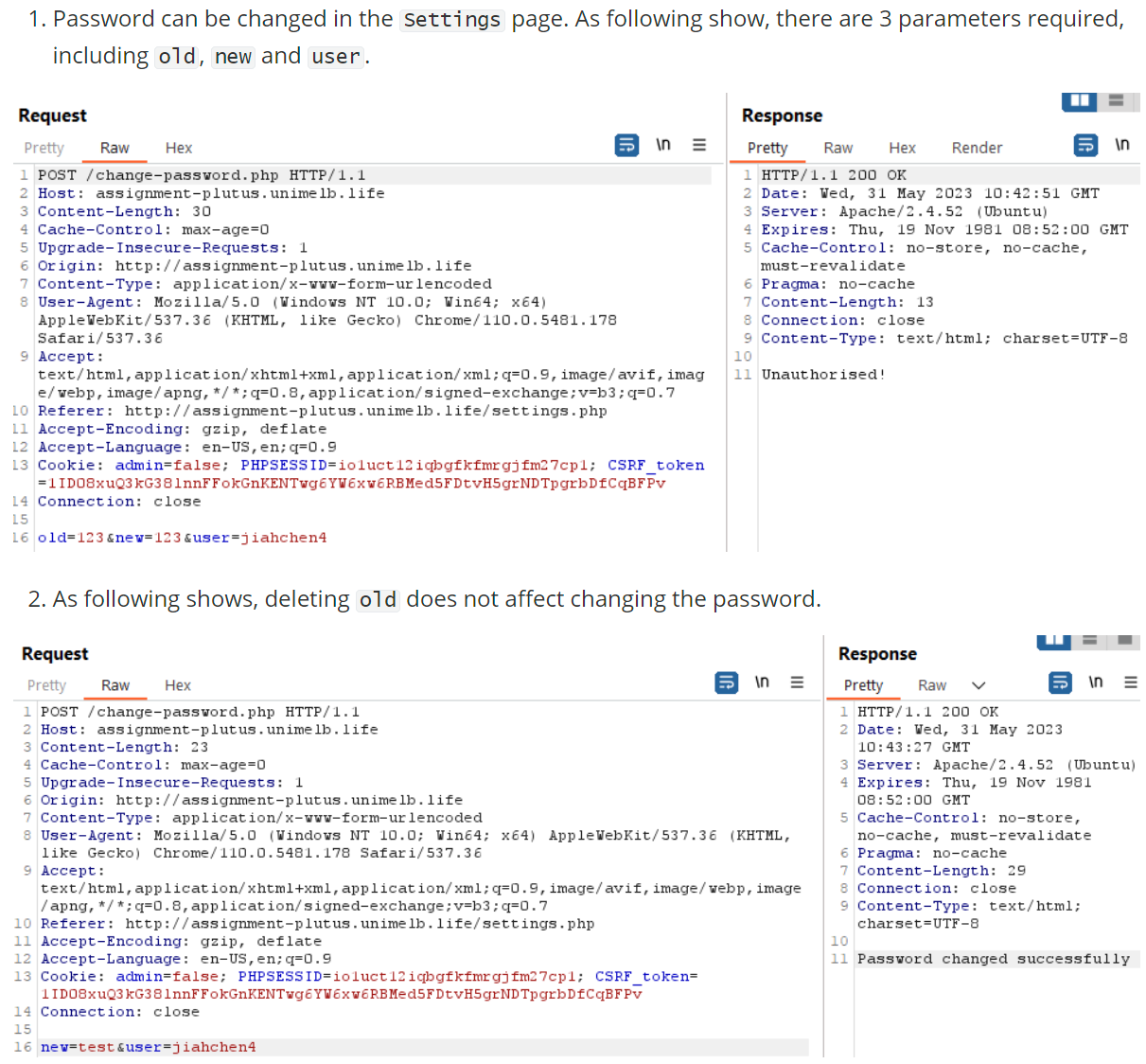
# Finding 2 – IDOR in User Profile

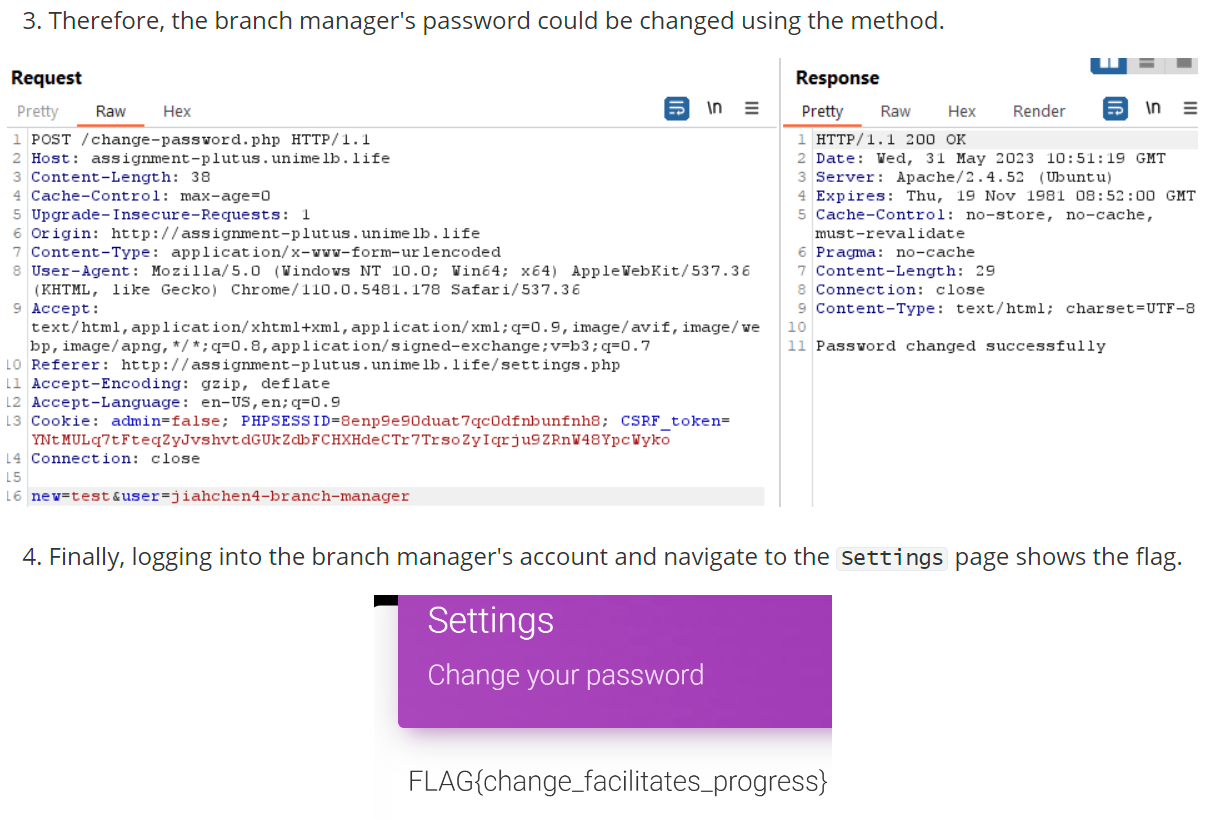




[Back to Finding 2](#_Finding_2_–_1)

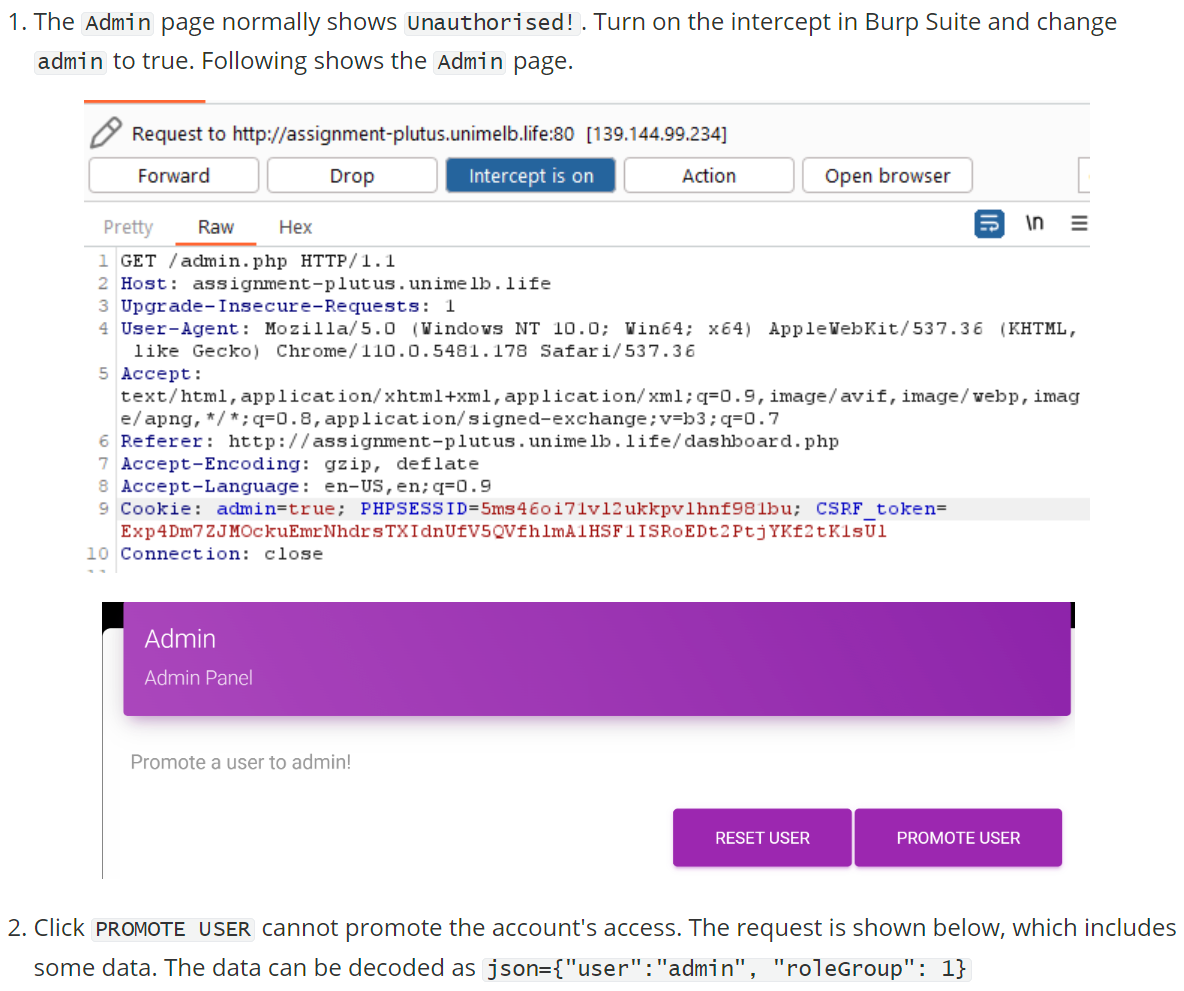
# Finding 3 – Takeover branch manager’s account



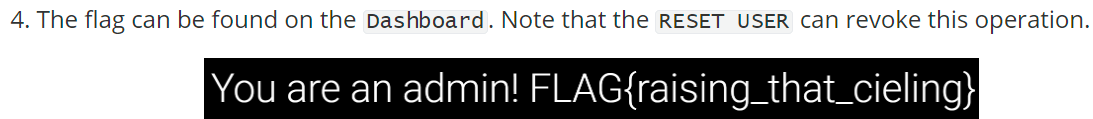


[Back to Finding 3](#_Finding_3_–)

# Finding 4 – Privilege escalation

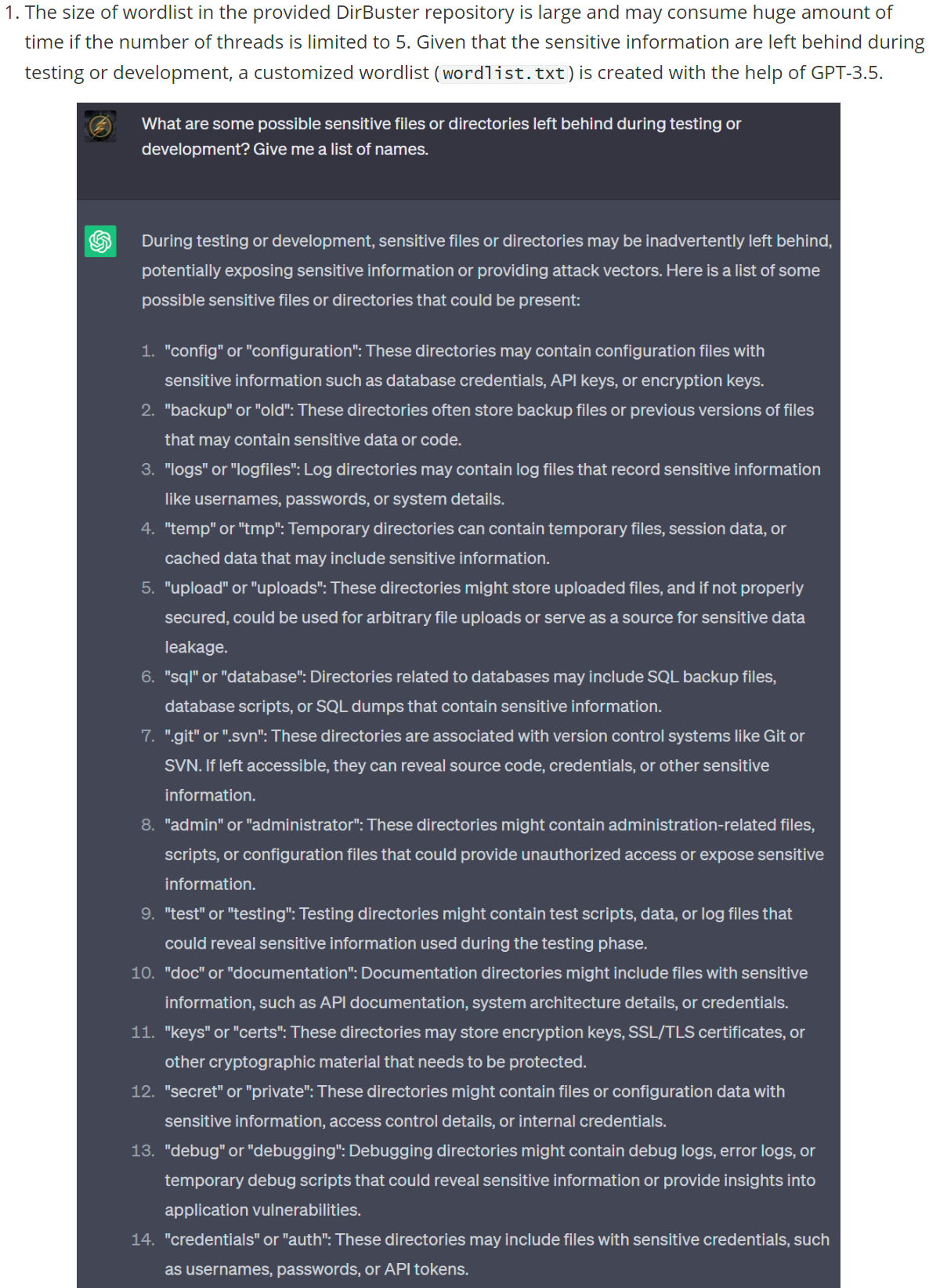


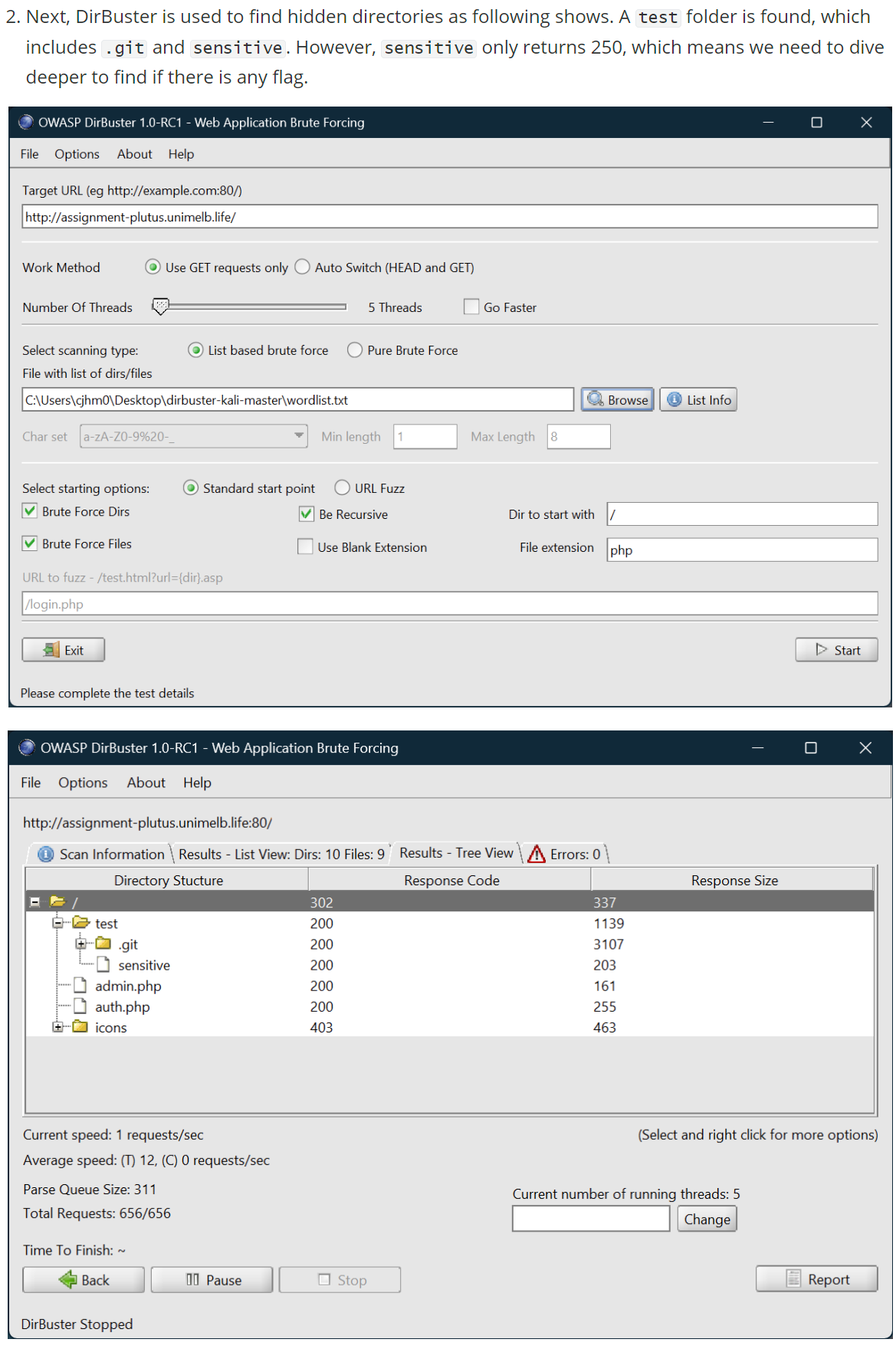




[Back to Finding 4](#_Finding_4_–)

# Finding 5 – Information leakage in testing folders







[Back to Finding 5](#_Finding_5_–_1)

[Back to Top](#_top)