

**Bachelor of Science with Honours in Cyber Security**

**COURSEWORK 2**

**Module Name : SECURE PROGRAMMING**  
**Module Code : E20/300IT/03**  
**Weighting : 60%**  
**Due Date : 25 October 2021**

**SUBMITTED BY**

**NAME : CHUN YOOJIN**  
**CU Index No. : 11059949**  
**Word Count : 2078**

## **Abstract**

Nowadays, with the development of computer technology, cybercrime is also increasing seriously. Hackers steal critical information such as account, username, and password from vulnerable websites. To prevent this, it is important to identify security vulnerabilities and create a web page with enhanced security. In this project, I will develop a vulnerable website to analyse vulnerabilities, and mitigate this web application.

## Table of Contents

<b>1</b>	<b>Introduction .....</b>	<b>4</b>
<b>2</b>	<b>System Environments.....</b>	<b>4</b>
<b>3</b>	<b>Project Strategies.....</b>	<b>5</b>
<b>4</b>	<b>UML .....</b>	<b>5</b>
<b>5</b>	<b>Insecure Web Application .....</b>	<b>7</b>
5.1	<i>SQL Injection.....</i>	<i>7</i>
5.1.1	Penetration Test.....	7
5.1.2	Mitigation .....	9
5.2	<i>Cross-Site Scripting (XSS).....</i>	<i>10</i>
5.2.1	Penetration Test.....	10
5.2.2	Mitigation .....	12
5.3	<i>Data Plaintext Transmission .....</i>	<i>13</i>
<b>6</b>	<b>Secure Web Application .....</b>	<b>14</b>
6.1	<i>Vulnerabilities Scanning.....</i>	<i>14</i>
6.2	<i>Penetration Test.....</i>	<i>14</i>
6.2.1	SQL Injection.....	14
6.2.2	Cross-Site Scripting (XSS).....	15
<b>7</b>	<b>Conclusion.....</b>	<b>15</b>
<b>8</b>	<b>References.....</b>	<b>16</b>

## 1 Introduction

This project performs vulnerability diagnosis to determine whether information on critical systems can be stolen, manipulated, or destroyed. In order to perform checking vulnerabilities based on the project strategies, a web application that may pose a threat is implemented and a penetration test is conducted from the attacker's point of view. Through this project, I can understand diagnostic items correctly and produce better results. Understanding what kind of error the web server causes according to the input value, it will be easy to understand the principles of web vulnerabilities. Find solutions and mitigate web applications according to the vulnerabilities founded in the web application.

## 2 System Environments

I have used two different servers to build insecure and secure web applications. The environment in which web services were developed and built is as follows below.

- Insecure Web Application

OS	VMware Fusion Professional Version 12.1.2(17964953)
	Ubuntu 64-bit 20.04.3
Server	Apache 2.4.51/2.4.37
	Tomcat 10.0.12
Development	Visual Studio Code 1.60.2 (Universal)
	PHP 7.3.11
	MySQL 10.5.12 Maria DB
	Firefox Browser 92.0.1 (64-bit)
Test	Kali Linux amd64
	Chromium
	SQLMap
	HTTrack
	Nessus 8.15.2
Frequency	Nmap 7.91

The environment of insecure web application was composed of APM (Apache, PHP, MySQL MariaDB), and because it aims to implement vulnerable websites, the server's security settings were set to a minimum and the server's web service daemon was given the highest authority.

- Secure Web Application

OS	VMware Fusion Professional Version 12.1.2(17964953)
	CentOS 8 64-bit
Server	Apache 2.4.51/2.4.37
	Tomcat 10.0.12
Development	Visual Studio Code 1.60.2 (Universal)
	Python 3.9.7
	MySQL 10.5.12 Maria DB
	Firefox Browser 92.0.1 (64-bit)
Test	Kali Linux amd64
	Chromium
	SQLMap
	HTTrack
	Nessus 8.15.2
Frequency	Nmap 7.91

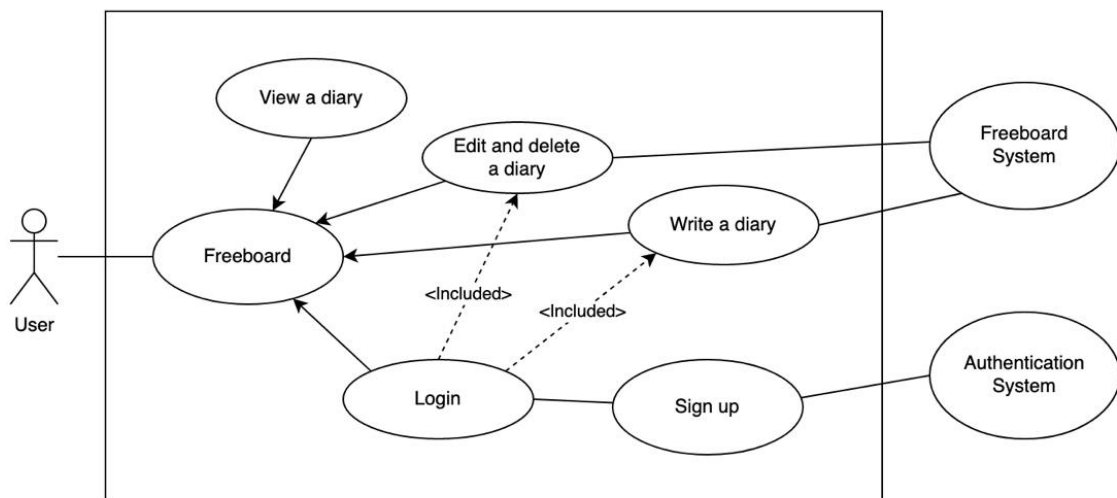
Use Django to create a web application that already have been secured. In order to prevent attacks conducted on vulnerable web application, the source code is modified, and vulnerability check verification is performed again. If there is a problematic script, block publication.

### 3 Project Strategies

- Modelling and developing a vulnerable web application.
- Exploring web applications via Nmap and Nessus scanning.
- Penetration Test (SQL Injection and Cross-Site-Scripting).
- Finding vulnerabilities and analyzing.
- Mitigating and developing a secure web application.
- Penetration Testing and checking vulnerabilities again.
- Repeating processes on and on.

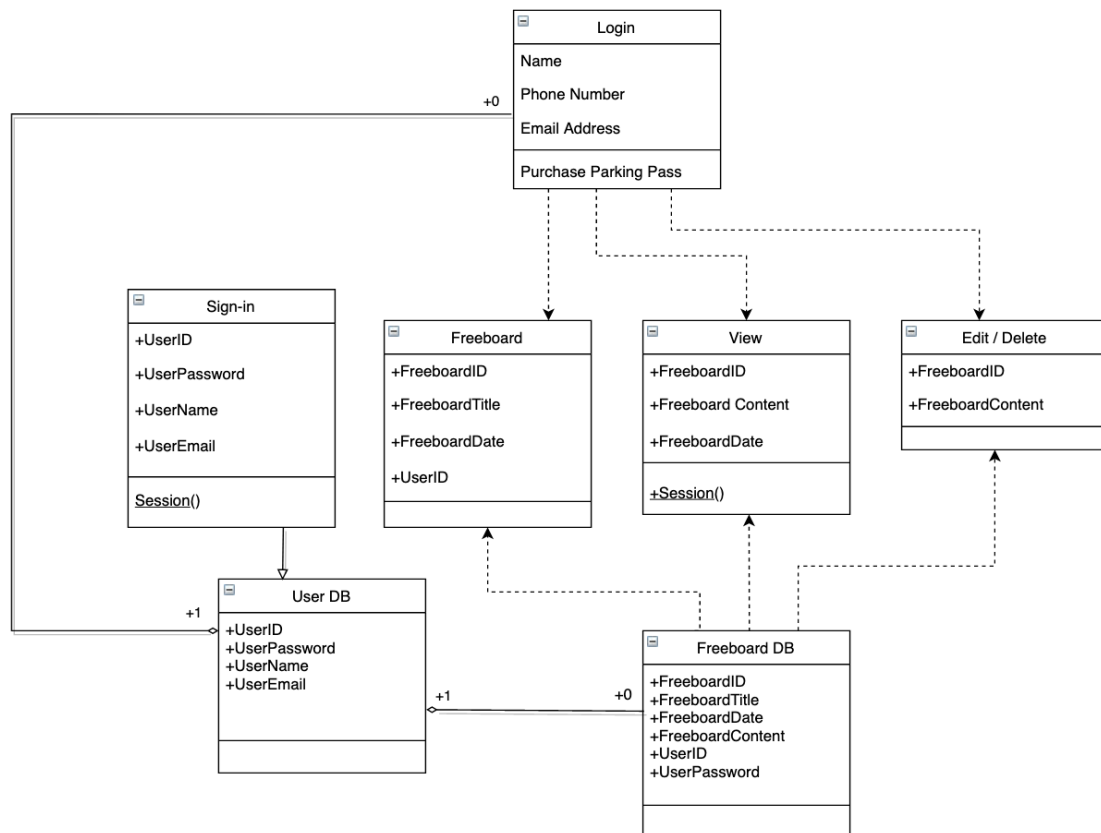
### 4 UML

It is a diagram of a free board with a login function. Login is required for writing, editing, and deleting on the free board. Sign up is required to Login to free board, and it is managed by Authentication system.



<Figure 4.1. Use case Diagram>

The diagram below is a Class diagram of the free board. User required to Login to write, edit, delete, and view posts of free board. User DB includes user ID, password, name, and e-mail address. And Free board DB includes freeboard ID, title, date, and content, and user ID and password.



<Figure 4.2. Class Diagram>

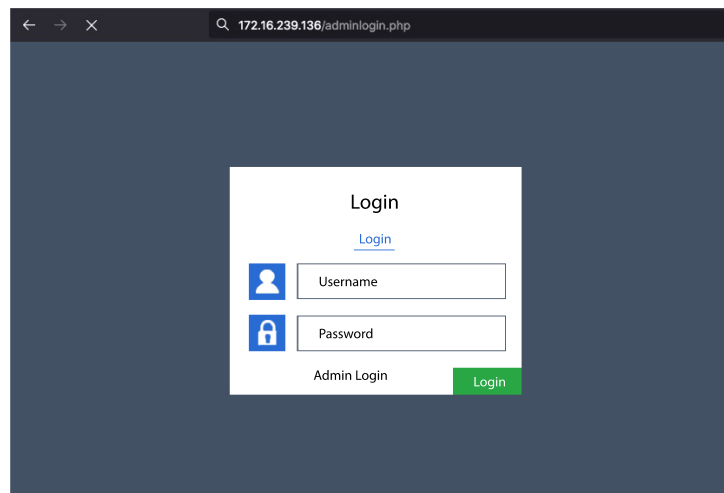
## 5 Insecure Web Application

This web application is deliberately made vulnerable for penetration testing and mitigation. Installing and setting Apache, PHP, and Maria DB is required to build vulnerable APM server.

### 5.1 SQL Injection

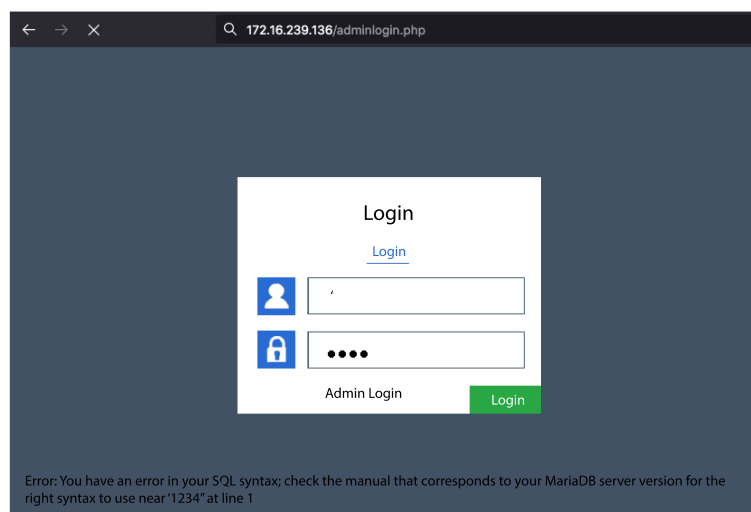
#### 5.1.1 Penetration Test

SQL Injection is an attack method that can attack the server's database by manipulating the client's input value with a technique of code injection. By injecting completed query statements using single quotation mark and annotation, attackers can take malicious actions such as login bypass and data extraction.



<Figure 5.1-1. Login Page>

The login page receives ID and Password and sends a request to the POST by clicking the login button. When logging in successfully, 'Login SUCCESS!' is displayed as a message, and in case of failure, error message 'Incorrect Login information' is displayed.

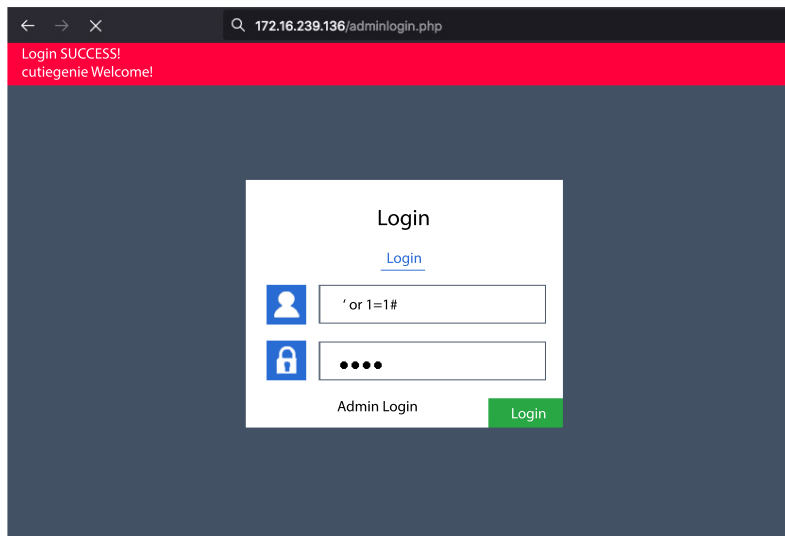


<Figure 5.1-2. Input Single Quotation Mart SQL Injection>

If the website does not properly process the special character input or causes the PHP error message to be displayed, there is a possibility of SQL Injection attack.

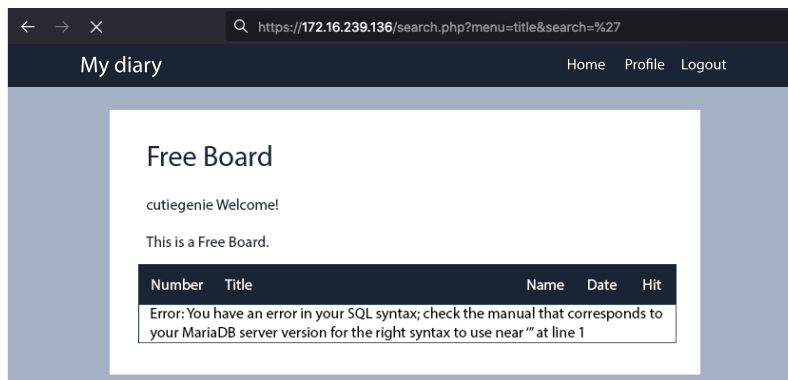
Attack by inserting SQL Injection into the form of entering the username.

- Input syntax: ' or 1=1#



<Figure 5.1-3. SQL Injection>

Through this SQL Injection, the result is always being true, and the use of '#' should not be interpreted by annotating the following construction. At this time, the value that brings information is the last value that comes when the query statement is executed, and login with other values can be successful using the sorting function as well.



<Figure 5.1-4. Single Quotation Mark Error Message>

After accessing the vulnerable bulletin board, enter a single quotation mark to check whether SQL Injection attack is possible by causing a grammar error. When the number of columns does not match, an error occurs, and when the number of columns matches, the search result and the position of the column may be checked.



- A screenshot of a web browser window. The address bar shows the URL: https://172.16.239.136/freeboard/search.php?menu=title&search=%27+UNION+SELECT+. The page title is 'My diary'. The main content area has a dark blue header with 'Home', 'Profile', and 'Logout' links. Below this, the text 'Free Board' is displayed in a large font. A message says 'cutiegenie Welcome!'. Below that, it says 'This is a Free Board.' and then a table with columns 'Number', 'Title', 'Name', 'Date', and 'Hit'. The table contains one row with the text 'Error: The used SELECT statements have a different number of columns'.

- Input syntax: 'UNION SELECT ALL 1,2,3,5,6,7,8#'

Search results for '3' by user '2' on 2021-10-25. The search input field shows the query: '1 UNION SELECT ALL 1,2,3,4,5,6,7,8#'.


Login works when a true query statement is created in a form where SQL Injection is possible, and login does not work when a false query statement is created in a form where it is not possible.


← → × 🔍 172.16.239.136/adminlogin.php

Login SUCCESS!  
cutleggie Welcome!

### Login

[Login](#)






Admin Login
Login


← → × 🔍 172.16.239.136/adminlogin.php

Incorrect Login information.

## Login

[Login](#)





Admin Login
Login

### 5.1.2 Mitigation

- When the user's input value dynamically affects DB query, it is verified whether the input value is intended by the developer. The following unintended input values should be verified and blocked.

- /\*, -, ', ", ?, #, (, ), :, @, =, \*, +, union, select, drop, update, from, where, join, substr, user\_tables, user table columns, information schema, sysobject, table schema, declare, dual, ...

- ```
UserInput = preg_replace("/[r\n\t'\\";"/>
```

```

if(preg_match('/(union|select|from|where)/i', $UserInput)) {

    $this->Error_popup('No SQL-Injection');

}

```

- Use saved procedure

The stored procedure refers to the pre-formatting query you want to use. The query is not executed unless it is the specified format data.

```

if ctype_digit($_POST['uId']) && is_init($_POST['uId'])) {

    $validateduId = $_POST['uId'];

    $pdo = new PDO('mysql:store.db');

    $stmt = $pdo->prepare('SELECT *FROM tb_user WHERE user_id = :uId');

    $stmt->bindParam(':uId', $validateduId, PDO::PARAM_INT);

    $stmt->execute();

} else {

    //reject id value

}

```

- Server security

Operate DB with minimal authority, remove unused saved procedures and built-in functions, or control authority. The authority of the query is modified according to the purpose and the common system object is also managed through access control. Access is allowed only to trusted networks and servers, and error messages are set not to be exposed to web pages.

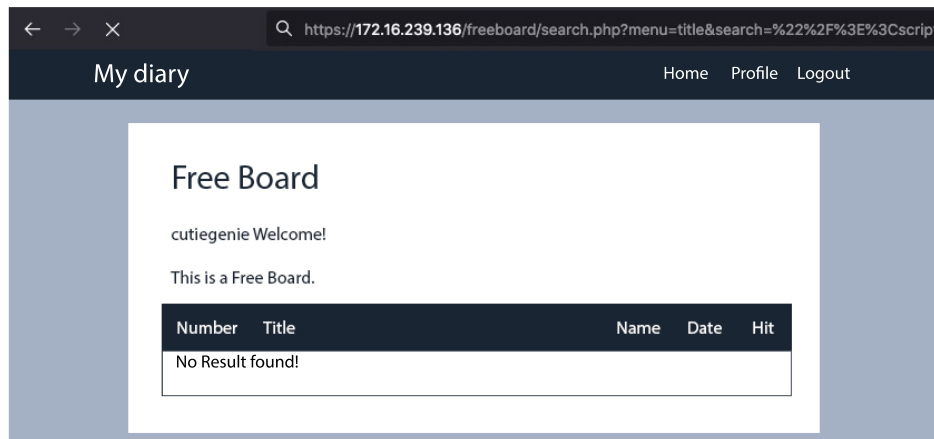
## 5.2 Cross-Site Scripting (XSS)

Cross-Site Scripting (XSS) is a vulnerability in which malicious scripts inserted or provided by an attacker run in the user's browser. Since it is a frequently used attack with various attack methods, accurate security measures must be taken depending on the service. A damage caused by XSS includes stealing user's information and cookie information, and it forces to redirect users to phishing sites, and let them download and execute malicious codes.

### 5.2.1 Penetration Test

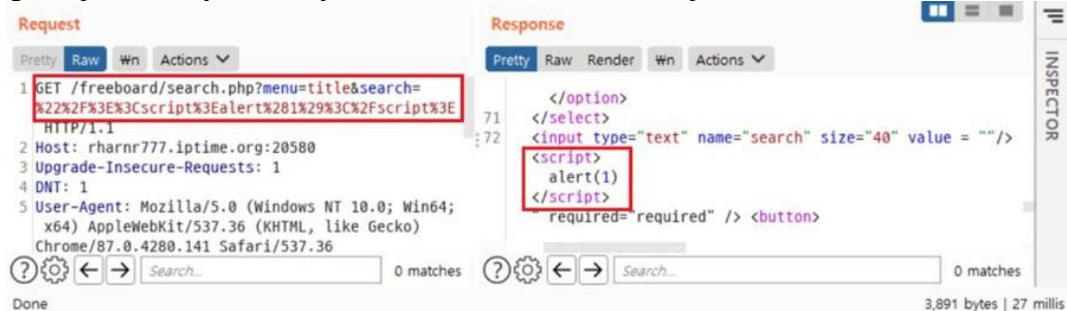
Check where the reflected XSS occurs and that the script is executed in the search bar of the post search.

- Input syntax: `/><script>alert(1)</script>`



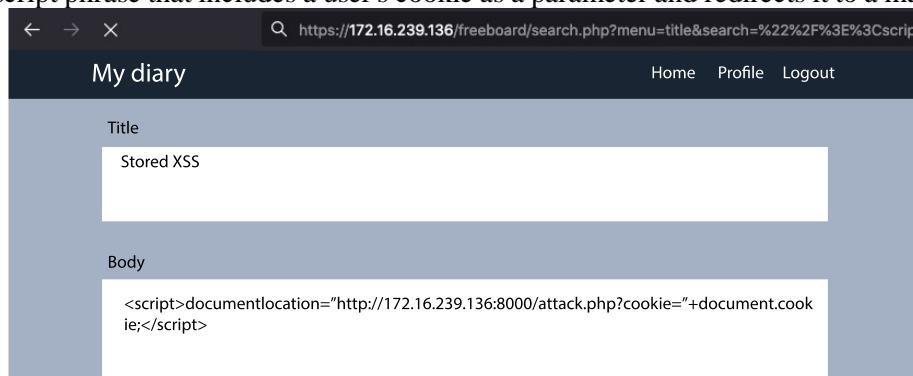
<Figure 5.2-1. Check the Location of Reflected XSS>

Through response analysis, it may be confirmed that the user's input value has been executed as a script.



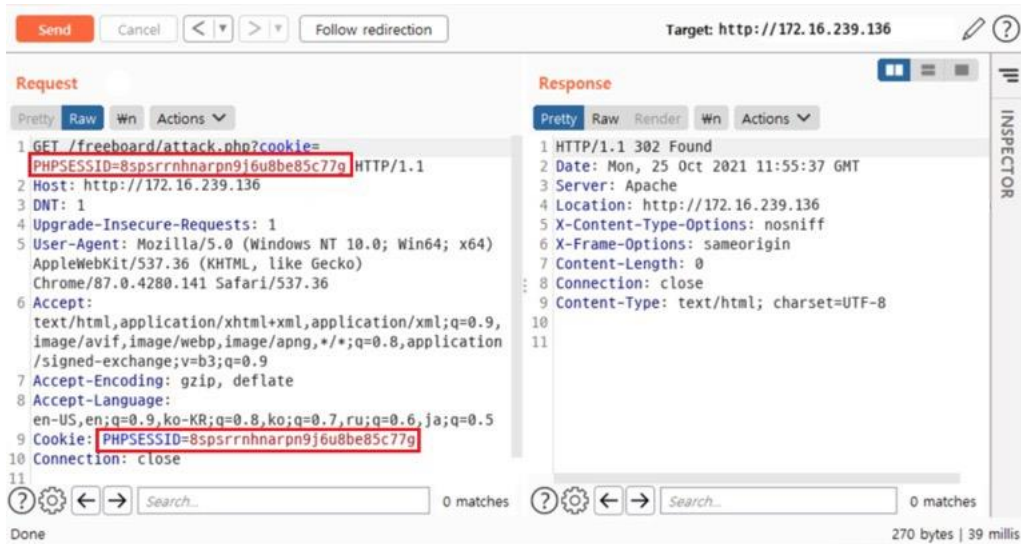
<Figure 5.2-2. Response Value of Reflected XSS>

Post articles including scripts that have malicious functions in the bulletin board writing function. This phrase is a script phrase that includes a user's cookie as a parameter and redirects it to a malicious page.



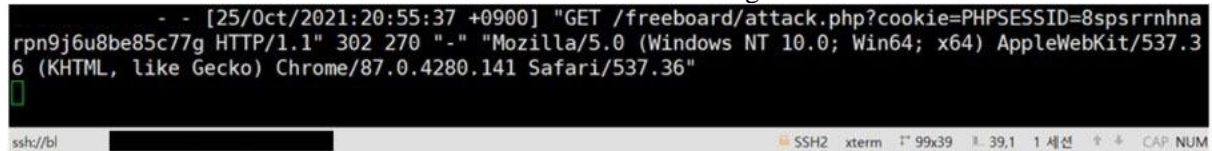
<Figure 5.2-3. Stored XSS>

When clicking on the post, the script operates, and the session value is transmitted to the attacker's domain.



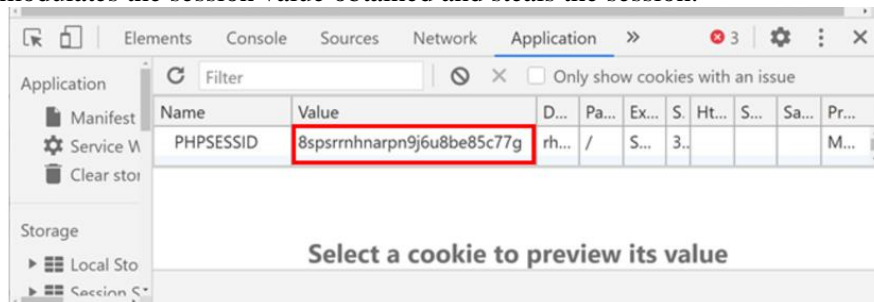
<Figure 5.2-4. Session Value Transmission>

The session value transmitted from the attacker's web server login can be checked.



<Figure 5.2-5. Session Value Confirm>

The attacker modulates the session value obtained and steals the session.



<Figure 5.2-6. Changing session values and stealing account rights>

## 5.2.2 Mitigation

- Must perform user input value verification at the server stage.
- Special characters that can be recognized as HTML codes in the user input string are replaced with general characters and processed.

| HTML special character | HTML Entity    |
|------------------------|----------------|
| <, >                   | &lt;. &gt;     |
| (, )                   | &#x28;, &#x29; |
| &                      | &amp;          |
| /                      | &#x2F          |
| “                      | &quot;         |
| ‘                      | &#x27;         |
| #                      | &#x23;         |

- If an HTML tag should be allowed on a bulletin board, etc., apply a method of allowing only the corresponding tag after selecting a whitelist of HTML tags.

### 5.3 Data Plaintext Transmission

The vulnerability is that critical information such as account information and credit information is exposed in plaintext when communicating between servers and clients. Since the data after decryption is completed is visible, diagnose as a wire shark, not a proxy.

|      |                                                 |                   |
|------|-------------------------------------------------|-------------------|
| 0330 | 31 2e 37 35 2e 31 31 2e 31 33 34 26 49 44 3d 74 | 1.75.11. 134&ID=t |
| 0340 | 65 73 74 74 65 73 74 31 32 33 26 78 3d 32 30 26 | esttest1 23&x=20& |
| 0350 | 79 3d 31 35 26 50 57 44 32 3d 25 43 36 25 44 30 | y=15&PWD 2=%C6%D0 |
| 0360 | 25 42 44 25 42 41 25 42 46 25 46 36 25 42 35 25 | %BD%BA%B F%F6%B5% |
| 0370 | 45 35 26 50 57 44 3d 31 32 33 31 32 33          | E5&PWD=1 23123    |

<Figure 5.3. Login Information Plaintext Transmission>

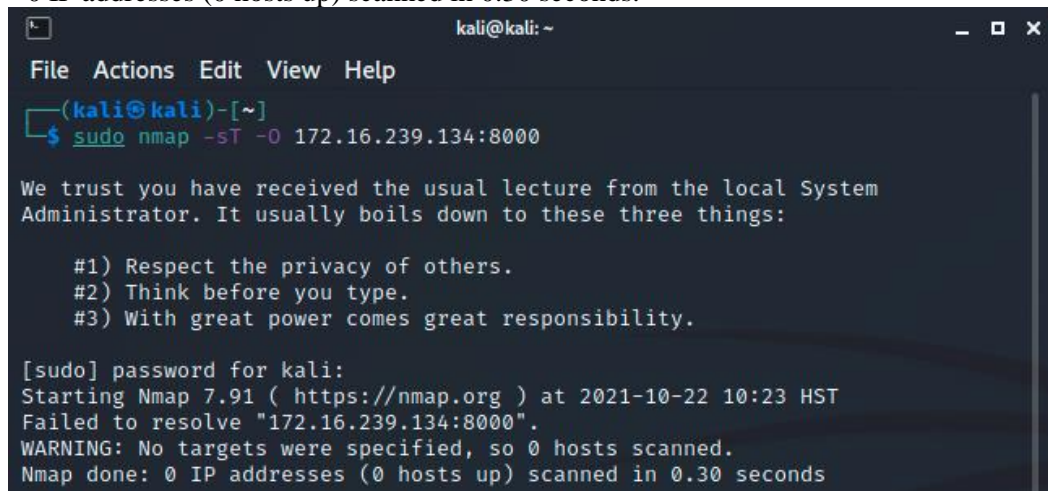
By checking the packet generated at the time of login with Wireshark, it can be confirmed that the input account information is transmitted in plain text. When data is transmitted in plaintext, there is a risk of account exposure due to sniffing attacks, so it must be encrypted and transmitted.

## 6 Secure Web Application

### 6.1 Vulnerabilities Scanning

I have scanned Nmap and MySQL in Kali Linux, but there was nothing scanned because developed web application is already secured well.


- Input syntax: `sudo nmap -sT -O 172.16.239.134:8000`  
No targets were specified, so 0 hosts scanned.  
0 IP addresses (0 hosts up) scanned in 0.30 seconds.



```
kali@kali: ~  
File Actions Edit View Help  
(kali@kali)-[~]  
$ sudo nmap -sT -O 172.16.239.134:8000  
  
We trust you have received the usual lecture from the local System  
Administrator. It usually boils down to these three things:  
  
#1) Respect the privacy of others.  
#2) Think before you type.  
#3) With great power comes great responsibility.  
  
[sudo] password for kali:  
Starting Nmap 7.91 ( https://nmap.org ) at 2021-10-22 10:23 HST  
Failed to resolve "172.16.239.134:8000".  
WARNING: No targets were specified, so 0 hosts scanned.  
Nmap done: 0 IP addresses (0 hosts up) scanned in 0.30 seconds
```

<Figure 6.1-1. Nmap Scanning>

- Input syntax: `-u root -p -h 172.16.239.134:8000`



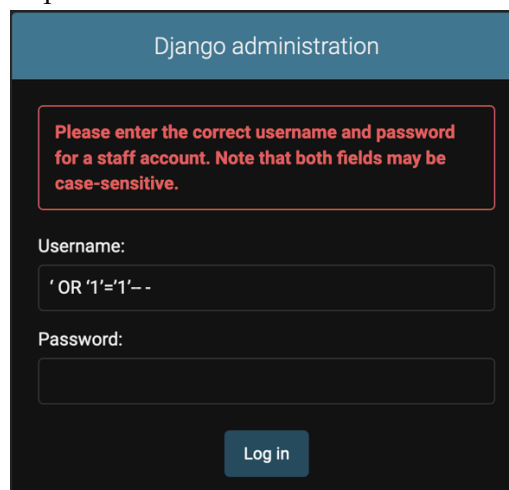
```
(kali@kali)-[~]  
$ mysql -u root -p -h 172.16.239.134:8000  
Enter password:  
ERROR 2005 (HY000): Unknown MySQL server host '172.16.239.134:8000' (-2)
```

<Figure 6.1-2. MySQL Scanning>

### 6.2 Penetration Test

#### 6.2.1 SQL Injection

- Input syntax: `' OR '1'='1' --`



Django administration

Please enter the correct username and password for a staff account. Note that both fields may be case-sensitive.

Username:  
' OR '1'='1' --

Password:

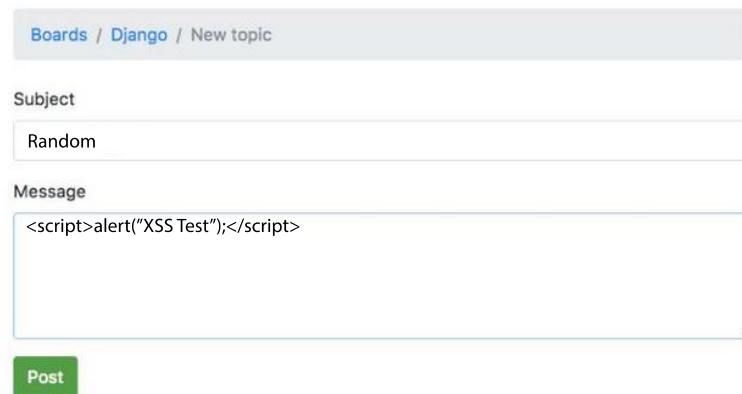
Log in

<Figure 6.2-1. SQL Injection>

I have tried SQL Injection attack to login as an Administrator, but it failed because it is already well secured, and an error message showed that please enter the correct username and password for a staff account.

### 6.2.2 Cross-Site Scripting (XSS)

If Cross-Site Scripting attack is successful, the script notification window appears as a pop-up, but nothing appears in a well-secured web application.



Boards / Django / New topic

Subject

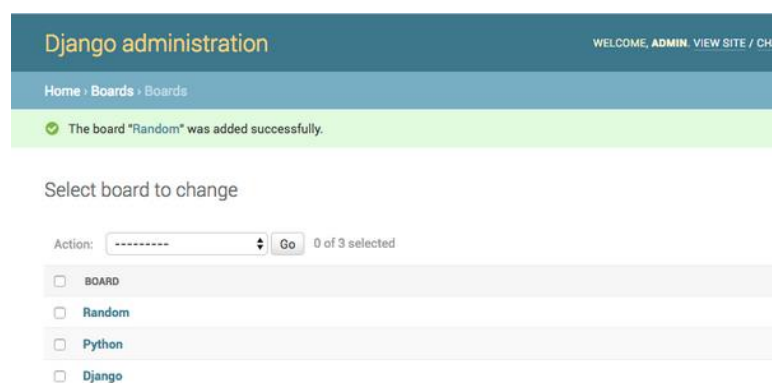
Random

Message

`<script>alert("XSS Test");</script>`

Post

<Figure 6.2-2. XSS>



Django administration

WELCOME, ADMIN. VIEW SITE / CHANGE PASSWORD

Home » Boards » Boards

✓ The board "Random" was added successfully.

Select board to change

Action: [dropdown] Go 0 of 3 selected

|                          |        |
|--------------------------|--------|
| <input type="checkbox"/> | BOARD  |
| <input type="checkbox"/> | Random |
| <input type="checkbox"/> | Python |
| <input type="checkbox"/> | Django |

<Figure 6.2-3. XSS Succeed>

## 7 Conclusion

Recently, security threats have been seriously occurring around the web application area. It is important to encrypt and store username and password because malicious attackers can acquire unencrypted data and commit crimes such as identity theft or credit card fraud. In addition, it is recommended to change the administrator ID and password periodically and use at least 10 digits of the password.

Physically, it is helpful to install a web firewall application, minimize network weaknesses, and access through SSL when accessing an external connection. Information protection management system certification through ISMS (Information Security Management System) and PIMS (Personal Information Management System) are also recommended, and periodic training for managers and executives is also required.

## 8 References

- [1] PortSwigger (2021) SQL injection. Available at <https://portswigger.net/web-security/sql-injection> (Accessed: 11 October 2021).
- [2] S. Kirsten (2021) Cross Site Scripting (XSS). Available at <https://owasp.org/www-community/attacks/xss/> (Accessed: 11 October 2021).
- [3] M. Shah, S. Ahmed, K. Saeed, M. Juaid, H. Khan (2019) Penetration Testing Active Reconnaissance Phase – Optimized Port Scanning With Nmap Tool. Available at <https://ieeexplore.ieee.org/abstract/document/8673520> (Accessed: 12 October 2021).
- [4] Q. Weizhong (2018) PrivGuard: Protecting Sensitive Kernel Data From Privilege Escalation Attacks. Available at <https://ieeexplore.ieee.org/abstract/document/8443329> (Accessed: 12 October 2021).
- [5] Z. Banach (2019) How you can disable directory listing on your web server – and why you should. Available at <https://www.netsparker.com/blog/web-security/disable-directory-listing-web-servers/> (Accessed: 18 October 2021).
- [6] D. Calle (2017) How to turn your website into a desktop app. Available at <https://ubuntu.com/blog/how-to-turn-your-website-into-a-desktop-app> (Accessed: 23 October 2021).
- [7] R. Peterson (2021) Your First PHP Web Application using MySQL an PHP with Examples. Available at <https://www.guru99.com/php-practical-example.html> (Accessed: 23 October 2021).
- [8] D. Karczewski (2020) 15 Amazing Django Website Development Examples You Should Look At. Available at: <https://www.ideamotive.co/blog/amazing-django-website-development-examples-you-should-see-right-now> (Accessed: 23 October 2021).