BLG460E - Secure Programming

Take-Home Exam 4 : SQL Injection & XSS Attacks

Due Date: May 13, 2019, 23:59 PM

Part1: SQL Injection Attacks

This part aims to provide hands-on experience on SQL injection attacks. You need to study on SQL injection attacks and analyze the given code to bypass simple security flaws.

A new Ubuntu based virtual machine provided for this assignment. You can download it from `https://drive.google.com/file/d/1WueGOEejZY-8E5TLKHruBmbeP6Qc5BYM/view?usp=sharing`. The login password is **12345**. Note that this virtual machine is different from the previous ones. Thus, you need to download this one for this assignment.

The code you will examine is ready in the Eclipse IDE's workspace. When you run the Eclipse you will see the code. You can run the program and start studying without making any changes. When you run the program in Eclipse, you can work on the console window displayed in the window below.

In the virtual machine, LAMP (Linux Apache MySQL PHP) stack is up and running. You can type `localhost/phpmyadmin` in the URL bar of the browser to use phpMyAdmin program. Login information for pypMyAdmin is **root** for username and **12345** for password. When you logged in, you can see the MySQL databases created. As an example, we created Customer database. Please examine the tables in the database.

Please answer the questions below in the report. While answering the questions you can assume that you know the names of the database and the tables except the bonus question.

- Can you login to the system without knowing any username and any password?
- Can you obtain any information about users (e.g., names, orders, details)?
- By means of SQL injections, is it possible to add new entries to the tables or remove any entries?
- What are the possible countermeasures against injection attacks. Give at least two types of countermeasures. Explain how they work. Provide implementations in the report.
- **(BONUS)** Can you obtain any information without knowing the names of the database and the tables? Explain the methods you would use.

Part2: XSS Attacks

In this part, you are given some python files that works as servers and some html files to make an interface of each servers. The files are uploaded to ninova and can also be found in the `Desktop` directory of the aforementioned virtual machine image. Each part must be answered individually, so delete every change you made in the previous parts when starting a new part.

You should use python version 2 to run the files. Then you can test the servers in the browser.

2.1. Set and Display Cookies

Run **main.py, attacker.py,** and **sdattacker.py** in different terminals. While they are running you can reach all three servers in the browser as follows:

- For **main.py** type `localhost:8080/app`,
- For **attacker.py** type `127.0.0.1:8070/attacker`,
- For **sdattacker.py** type `localhost:8100/sdattacker` in URL bar.

Alternatively, you can type `localhost:8080/app?user=recep&pass=1234` in URL bar to set cookies. Here, `user=recep` and `pass=1234` values are arbitrary examples. You can write anything you want.

Note: In this assignment you will be dealing with exactly two cookies, therefore, please give only two cookies.

When you set the cookies, click the first button in `localhost:8080` to see the cookies are displayed properly as shown in the Figure 1.

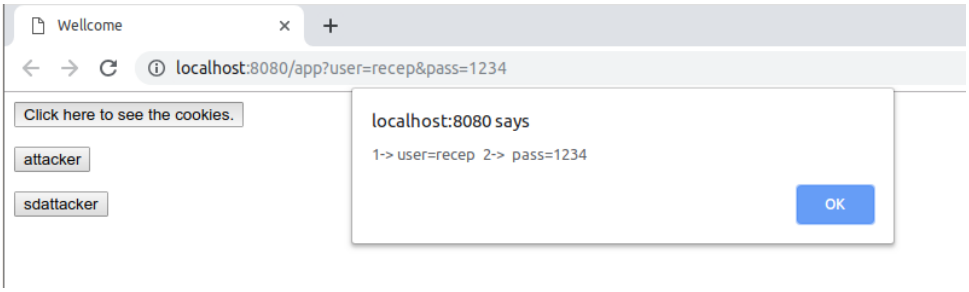

Figure 1: First button is clicked.

Then, click the two buttons below and observe the outputs in the terminals of **attacker.py** and **sdattacker.py**. After observing the outputs, answer the questions below in the report.

- What information can be gathered with an HTTP GET request?
- What do the buttons do?
- What are the differences between the outputs in the terminals of both files?

2.2. JavaScript Prevention

In the Figure 1 and the **PATH**– parts in the outputs of the terminals, cookies can be seen due to a JavaScript function. Observe the files and answer the questions below in the report.

- Which JavaScript function allows the cookie display?
- How can you prevent JavaScript to reach the cookies? Explain the method you aim to use and give python implementation in the report for only the second cookie. You should add only the added/altered part in the report.

  Note: You are not expected to delete or change the cookie content. In fact, you should add some feature/features to the second cookie.

  Hint1: For python implementation you can refer to the `Cookie` module `https://docs.python.org/2/library/cookie.html`.

  Hint2: The only file you should change is **main.py**.

2.3. Origin Problem

When you observe the **HEADER**– parts in the outputs of the attacker terminals, you can see that one of them gets the cookies in the header while the other one does not. Try to find out what the reason is for this situation and answer the questions below in the report.

- Explain the reason of the difference mentioned above.
- Provide a solution to prevent sending the second cookie in the header to the **sdattacker**. Give python implementation as in the section 2.2.

  Hint: Again you need to add some feature/features to the second cookie.

2.4. Expired Cookies

If you inspect the file **main.py**, you can see that the second cookie expires in 2099. In this part you are expected to make changes on the second cookie so that it expires 30 seconds after it is set. Ensure that the cookie is not available after 30 seconds, and then put the changes you made to the report.

2.5. SSL

First, ensure that the pem file is in the same directory with the files and its name is `yourpemfile.pem`.

For this part stop the **main.py** and **attacker.py**. Run only **mainssl.py, attackerssl.py,** and **sdattacker.py**. Visit the addresses below and give permission or unsafely continue as shown in Figure 2.

- `https://localhost:4443/app`
- `https://localhost:4447/attacker`

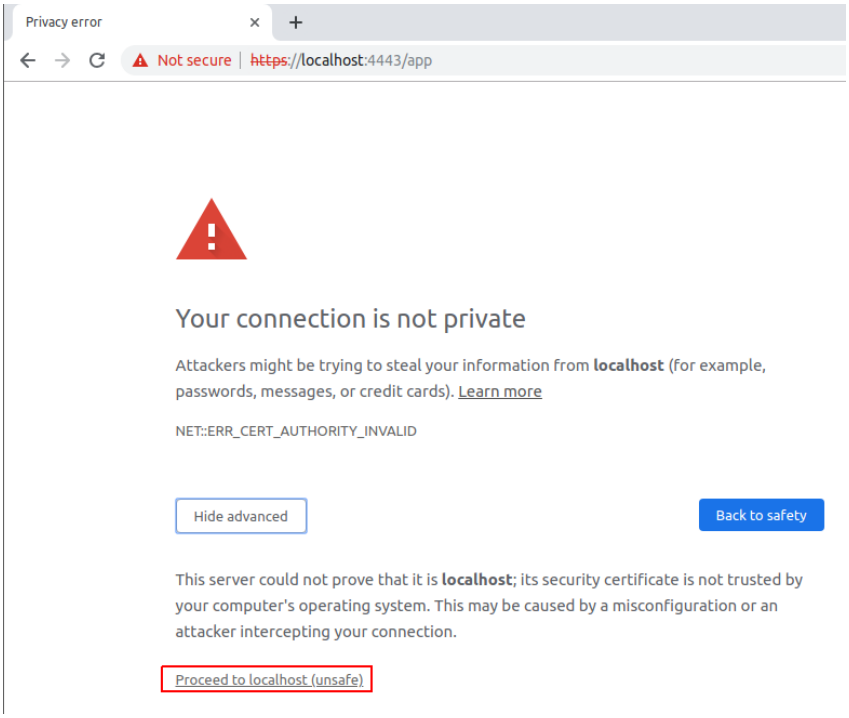

Figure 2: Proceed to localhost.

As you can check in the files, **mainssl.py** and **attackerssl.py** use SSL to encrypt the connection.

In this part, you are asked to prevent the second cookie to be sent over the unencrypted connection as it contains sensitive information. Find a solution and answer the questions below.

- Explain how you can prevent **mainssl.py** to send cookies over an unencrypted connection.
- Give python implementation for only the second cookie. In other words, two cookies must be set but only the first one must be sent to the **attacker.py** when its button is clicked. Since **attackerssl.py** also uses SSL, both cookies must be sent when its button is clicked.