Seunghyuk Baek

[Sbaek44@vt.edu](mailto:Sbaek44@vt.edu)

ECE 5480

Project 5

**Introduction**

This project is to demonstrate and test how cross-site scripting (XSS) exploits security vulnerabilities in a website based on scenario-based study. XSS is injection of malicious code (such as javascript code) into the victim’s web browser. This security attack often results in bleaching of confidential information of users.

**Task 1**: An attacker writes javascript code in his or her profile, attacking those who visiting attacker’s profile page.

A screenshot of a cell phone

Description automatically generated

Here, attacker is the user ‘Alice’. She wrote the simple javascript code to pop up the message box that says ‘XSS’.

A screenshot of a cell phone

Description automatically generated

If someone visits Alice’s profile page, XSS message pup-up will appear.

**Task 2**: The attacker now tries to exploit other users’ credential information

A screenshot of a social media post

Description automatically generated

Alice added another javascript code that reveals cookie information of visitors.

A screenshot of a social media post

Description automatically generated

The cookie of another user (this case, Boby) was revealed when he visited Alice’s profile page

**Task 3**: An attacker tries to reveal a victim’s cookie by HTTP request.A screenshot of a cell phone

Description automatically generated

Alice the hacker added javascript code that sends HTTP request to attacker’s computer at specified port (here port 5555 is used)

A screenshot of a cell phone

Description automatically generated

Boby visited Alice’s profile page.

A screenshot of a cell phone

Description automatically generated

Using netcat command, Alice created simple TCP server and listen to port 5555 for TCP transaction. The victim’s cookie is bleached (in red circle)

**Task 4**: Another hacker ‘Samy’ tries to add other user in his friend list. Any visitor who visits Samy’s profile will automatically add Samy as a friend.

A screenshot of a social media post

Description automatically generated

Frist, Samy need to know his own unique id in this web site. By sending add request to himself, he can easily identify his user id number (here it is 47).

A screenshot of a cell phone

Description automatically generated

Next, Samy needed to know how “add friend” requests are passed us URL. Here, it is /action/friends/add?firend=(user id)&other information.

A screenshot of a cell phone

Description automatically generated

Here, Samy added javascript code using the information above.

A screenshot of a cell phone

Description automatically generated

Boby had no friend in his friend list.

A screenshot of a cell phone

Description automatically generated

However, after visiting Samy’s profile page, Samy is added to boby’s friend list.

**Task 5**: Same method can be applied to alter a victim’s profile.

A screenshot of a cell phone

Description automatically generated

This is initial profile of Boby.

A screenshot of a social media post

Description automatically generated

Samy added malicious javascript code in his profile. This code will detect if the session is own by Samy and if it is not Samy’s account, the code will change profile “About Me” section into ‘Hacker Samy Here’ sentence.

A screenshot of a cell phone

Description automatically generated  
After visiting Samy’s profile page, Boby’s profile had changed.

**Conclusion**

I devoted about 5 to 6 hours to this project. This can be much simpler project if I knew javascript more. The most time I spent was to set up TCP session with another Virtual Machine because somehow, I could not receive anything by listening 5555 port. This project was very useful to understand how inject malicious script into web browser can create security exploit.