CPSC 455 - Web Security

Project 1 - Fall 2024

This project must be completed in a team of five or six. The instructor will assign teams for this project in Canvas.

See the following sections of the Canvas documentation for instructions on group submission:

- How do I submit an assignment on behalf of a group?

What are we trying to accomplish?

The textbook describes Cross-Site Request Forgery (CSRF) attacks, along with several different mitigation techniques. In this project, you will start with a vulnerable Node.js web application, cpsc455-project1. You will exploit the vulnerability, then implement the mitigations described in the textbook and verify that they function as intended.

Clone the repository and run the code

On Kali Linux, use the following commands:

```
$ git clone https://github.com/ProfAvery/cpsc455-project1.git
$ cd cpsc455-project1
$ npm install
$ npm start
```

Open your browser to `http://bank.127.0.0.1.nip.io:8080`, and explore the application. Try logging in as user `ProfAvery` (password `cpsc455`) and then as user `student` (password `blackhat`).

Note: One of the CSRF attacks that you will conduct involves setting up a second website on pretending to be the original. Since both sites will run on the same machine, they will need to be assigned different TCP ports. Rather than trying to remember which port corresponds to each site, you can use the nip.io service to assign multiple, meaningful hostnames that all resolve to `localhost`.

Transfer money

While the website does not implement a user interface for transferring money between accounts, there is an API. (Perhaps it is used behind the scenes by a mobile app.) Locate the code for executing transfers and make sure that you understand how it works.

Attack 1 - Misleading link

Log in as the user `student` and use the API directly from your browser's address bar to transfer a dollar to one of `ProfAvery`'s accounts.

Now switch users to `ProfAvery`, and verify that a dollar has been added.

Compose a phishing email that `ProfAvery` could send to `student` in order to steal another dollar. Use a URL-shortening service to obscure the link.

Attack 2 - Malicious page

Of course, no self-respecting student is going to allow a professor to pick their pocket: it's time to strike back. If one user can exploit a CSRF vulnerability, so can another.

Set up a new website, use the built-in HTTP server. On Kali Linux, use the following commands:

```
$ mkdir ~/evil
$ cd ~/evil
$ python -m http.server
```

Access this new website at `http://evil.127.0.0.1.nip.io:8000`.

Create an HTML page on the new website that will transfer $50 to `student` if `ProfAvery` is still logged into the bank when he visits the page. Verify that the money is transferred.

Attack 3 - Malicious deposit form

Of course, ProfAvery isn't going to randomly click on links or access the REST API directly. You will need to convince him that he is still on the Bank's website.

Examine `http://bank.127.0.0.1.nip.io:8080/deposit` to determine how money is added to an account.

Note: Presumably a real bank wouldn't just let you create money out of thin air. If it makes you feel better, imagine uploading a photo of a check.

Create an additional page on your malicious website that looks exactly like the bank's deposit page, except that any money deposited is always placed into `student`'s account, regardless of who is logged in.

Still logged into the bank as `ProfAvery`, visit this new page and deposit $1,000. Verify that `student` is now independently wealthy.

Note: for additional verisimilitude, access your bogus form though a similar but not-quite-right URL like `http://bank.127.0.0.1.nip.io.evil-7f000001.nip.io:8000/deposit.html`.

Mitigate the vulnerability

Chapter 6 of the textbook describes three separate mitigations for CSRF attacks. Modify the `cpsc455-project1` code to implement each mitigation separately and demonstrate that each mitigation prevents each of the three attacks that you executed.

Finally implement all three mitigations, and verify that the continues to function correctly.

Document your results

Create a report in PDF format documenting the results of each attack and mitigation.

At the beginning of your report, include:

- The names of each member of the team who participated in the project
- The semester, course number, and section number.
- The project number

For each attack and mitigation:

- Identify what is to be done.
- Document performing the step itself, including any commands that were run and any configuration or code that was modified.
- Describe your results, including definitions, diagrams, screenshots, or code as appropriate.
- Comment on the results, including references you consulted, variations you considered, or issues you encountered.

At the end of the report, include an appendix evaluating the contribution of each member of the team. If your team is able to reach consensus, you may simply assign a percentage contribution to each member. If your team cannot reach an agreement on each member's level of contribution, do not attempt to assign percentages; instead, describe in detail the tasks that each member performed.

Submit your work as a `.pdf` file through Canvas by the due date. Only one submission is required for a team.

Grading

The project will be evaluated on the following five-point scale, inspired by the general rubric used by Professor Michael Ekstrand at Boise State University:

Exemplary (5 points)

The project is a success. All requirements met. The quality of the work is high.

Basically Correct (4 points)

The project is an overall success, but some requirements are not met completely, or the quality of the work is inconsistent.

Solid Start (3 points)

The project is mostly finished, but some requirements are missing, or the quality of the work does not yet meet professional standards.

Serious Issues (2 points)

The project has fundamental issues in its implementation or quality.

Did Something (1 point)

The project was started but has not been completed enough to assess its quality fairly or is on the wrong track.

Did Nothing (0 points)

The project was not submitted, contained work belonging to someone else, or was of such low quality that there is nothing to assess.