CSCI 400 Lab 6

**Name: Christopher Gonzalez**

**Class Section: CSCI 400 02 [35583]**

**Date: November 19, 2024**

**Instructions**:

* Login to your account at <https://pwn.college/>

**pwn.college username: Chris\_B\_Gonzalez**

**A screenshot of a computer

Description automatically generated**

* Go to the Intro to Cybersecurity dojo: <https://pwn.college/intro-to-cybersecurity/>
* Review short videos above challenges in Web Security: <https://pwn.college/intro-to-cybersecurity/web-security/>
* Complete challenges from Path Traversal 1 through Authentication Bypass 2 (10 challenges)

**Challenge 1: Path Traversal 1**

**A screenshot of a computer

Description automatically generated**

**To obtain the flag, I opened the terminal and started the challenge by typing “cat /challenge/server” on the tab on the left. I did this to understand how the server functioned and if there were any vulnerabilities I could uncover based on the code. After analysis, I noticed there was a weakness in the way it handles unusual paths made by a user. On a separate tab, I entered “curl -v challenge.localhost/..%2F..%2Fflag”, which effectively tricked the system and gave me the flag.**

**Challenge 2: Path Traversal 2**

**A screenshot of a computer

Description automatically generated**

**To obtain the flag, I opened the terminal and started the challenge. Like the previous challenge, I had to discover a way to trick the server to give me the flag. The key to this challenge is to move upward from the application root directory using the directories in the server. I discovered one of the directories of the server was “fortunes”. Using this information, I entered “challenge. localhost/fortunes/..%2F..%2F..%2Fflag which allowed me to move up the directory and eventually access the flag.**

**Challenge 3: CMDi 1**

**A screenshot of a computer

Description automatically generated**

**To obtain the flag, I opened the terminal and started the challenge. In the previous two challenges, all work required to get the flags was done through the terminal. However, for this challenge onward, I discovered that challenge.localhost is a website I can use to perform my tasks, which is arguably much easier than what I have done thus far. After starting the server, I opened the Internet and went to challenge.localhost, which asked me to enter a directory that will be searched using ls -l. In the challenge description, it was also hinted to use the semicolon to trick the system. The semicolon is used to end a command line, however entering other commands after the semicolon has the chance to confuse the server. I proceeded by entering “/; cat /flag”. The semicolon can be seen before the cat command. This resulted in the server listing all directories in the / directory and providing the flag at the very end.**

**Challenge 4: CMDi 2**

**A screenshot of a computer

Description automatically generated**

**To obtain the flag, I opened the terminal and started the challenge. Once the server was active, I went to the challenge.local host website and was given a similar prompt as the last challenge. However, the description for this assignment informed me that I cannot use the semicolon to trick the server. Instead, I was hinted to use pipes. A pipe, in the context of terminal commands, is used to combine two or more commands. Similar to what I did in the last challenge, I entered “/ | cat /flag”, where it can be seen that the command before the pipe is / and the command after is cat. Fortunately, the server accepted the piped command, which tricked it into giving me the flag.**

**Challenge 5: CMDi 3**

**A screenshot of a computer

Description automatically generated**

**To obtain the flag, I opened the terminal and started the challenge. Once the server was active, I went to the challenge.local host website. The instructions for this challenge mentioned that characters like the semicolon will not be interpreted as Linux syntax unless it is within a set of single quotes. To quote, “In the shell, all quotes must be matched with a partner, or the command is invalid. Make sure to craft your injection so that the resulting command is valid!”. This confirms the command I create must be within a set of single quotes to be valid. I entered /’; cat /flag’ (double quotes were not used to avoid confusion). Since the cat command is within the single quotes, the flag was printed.**

**Challenge 6: CMDi 4**

**A screenshot of a computer

Description automatically generated**

**To obtain the flag, I opened the terminal and started the challenge. Once the server was active, I went to the challenge.local host website. This time, the website acted as a service that prints the time of a given area, meaning that the user is expected to provide a location. The instructions for this challenge also made it evident that I would be tricking the server using a similar, but slightly different, command that I have been using prior. The cat command is essential for all previous challenges and this one, meaning the aspect that will change is the content before the semicolon. I decided it would be appropriate to enter “USA;cat /flag date” since the server is about telling time from a given area. After entering this, I managed to trick the system and obtain the flag.**

**Challenge 7: CMDi 5**

**A screenshot of a computer

Description automatically generated**

**To obtain the flag, I opened the terminal and started the challenge. Once the server was active, I went to the challenge.local host website. The information on the website informed me that the touch command will be performed on whatever file the user chooses. For context, the touch command is typically used to create empty files. Of course, my objective isn’t to create a new/empty file, but to retrieve the contents of an existing file. The instructions for this challenge also informed me that the website will not be printing the flag the way it has been for the past few challenges. The best solution I came up with was entering “; cat /flag > /tmp/a”. Of course, the semi colon ends the initial touch command, and the cat command is used to print the flag. However, I used the greater than symbol after the cat command to transfer the contents of the flag to /tmp/a. Then, I went to the terminal and listed the directories of /tmp to verify that file a was created. After confirming file a, I entered “cat /tmp/a” which printed the flag.**

**Challenge 8: CMDi 6**

**A screenshot of a computer

Description automatically generated**

**To obtain the flag, I opened the terminal and started the challenge. Once the server was active, I went to the challenge.local host website. The prompt given by the website asked me to choose a directory to list the files of. I decided to enter “/” just to see what would happen. The website response normally but was worth trying as the instructions for this assignment hinted that the solution should resemble something I’ve already done before. After much pondering, I considered that perhaps the hint was referring to the way I encrypted requests for the first two challenges. At first glance, one would think to perform this on the given search bar, however the encryption must be done on the url. I modified the url to “challenge.localhost /?directory=%0Acat %2Fflag”. After entering this new url, the server printed the flag.**

**Challenge 9: Authentication Bypass 1**

**A screenshot of a computer

Description automatically generated**

**To obtain the flag, I opened the terminal and started the challenge. Once the server was active, I went to the challenge.local host website. The website asked me to enter a username and password, and the objective is to login as the admin. Unfortunately, I did not know the admin credentials, or any other credentials in the beginning. I decided to then check the code for /challenge/server to see what information I can obtain. Upon doing this, I learned that the admin password is programmed to be a random set of 8 characters. Brute forcing this password is a real, but horrible option, but fortunately the code provided a second set of credentials for username “guest”. I entered “guest for username and “password” for the password, which successfully logged me in as a guest. I noticed that the url changed where the end of the url contained “session\_user=guest”. I proceeded by modifying the url to say “session\_user-admin”, which effectively logged me in as the admin and gave me the flag.**

**Challenge 10: Authentication Bypass 2**

**A screenshot of a computer

Description automatically generated**

**To obtain the flag, I opened the terminal and started the challenge. Once the server was active, I went to the challenge.local host website. This challenge is near identical to the previous one, but unfortunately, I would not be able to modify the url the same way I did previously. I began by logging in as a guest user and looked for a way to log in as the admin. After some research, I discovered I can perform function F12, which is a command used to open developer tools and modify web pages. Through trial and error, I stumbled upon the “Storage” tab, which contained a value of “guest”. I then modified this value to “admin” and refreshed the page. This proved to be successful as I was now logged in as the admin and was given the flag.**