Christopher Gonzalez

Professor Jennifer Holst

CSCI 401

21 March 2025

**pwn.college Lab 3**

A screenshot of a computer

AI-generated content may be incorrect.

**Challenge 1: What’s the password?**

A screenshot of a computer

AI-generated content may be incorrect.

To obtain this flag, I started the challenge and opened the VSCode Workspace. As the name of this challenge implies, my task was to provide the password to the /challenge.runme program. The instructions for this challenge stated to read the program, which I did by typing the command vim /challenge/runme in the terminal. This revealed the contents of the program. One of the lines contained the password information, which was mlsgeypk. With this information, I returned to the terminal, started the program by writing /challenge/runme, and entered the password when prompted. The program checked and confirmed the password, which revealed the flag.

**Challenge 2: … and again**

A screenshot of a computer

AI-generated content may be incorrect.

To obtain this flag, I started the challenge and opened the VSCode Workspace. In short, this challenge was identical to the previous one. The only difference was the password. I discovered the password from typing vim /challenge/runme, which was ssxokgox. I then started the program, waited for the prompt to provide the password, provided the password, and the flag was given to me after confirmation.

**Challenge 3: Newline Troubles**

A screenshot of a computer

AI-generated content may be incorrect.

To obtain this flag, I started the challenge and opened the VSCode Workspace. This challenge introduced a potential challenge when it comes to the newline. According to the instructions, the program no longer ignores the Enter that the user presses when providing the password. As a result, the entered user will contain a newline, which the correct password doesn’t have. I had to find a way to provide the password to get around this issue. The solution I choose was to use the echo -n command, but first I read the contents of the /challenge/runme program for the password, which was tpjwnzzm. I then entered echo -n “tpjwnzzm” | /challenge/runme to the terminal. This approach allows the terminal to enter the password for the /challenge/runme program within the quotation marks. Since the information inside the quotation marks did not require hitting the Enter key, there will be no new line, which granted me the flag.

**Challenge 4: Reasoning about files**

A screenshot of a computer

AI-generated content may be incorrect.

To obtain this flag, I started the challenge and opened the VSCode Workspace. The instructions for this challenge were quite vague as the only hint provided was that the program doesn’t read the password from the terminal. First, I had to understand the /challenge/runme program by vimming it. Upon investigation, I learned the program expects to read the password from a file titled ryoy (and the password was mgisghaz). I returned to the terminal and used the echo command once again. However, this time, the echo command will be used to store the password in a ryoy file. To accomplish this, I entered echo -n “mgisghaz” > ryoy. Another reason why the echo command was used was because like the previous challenge, the program doesn’t ignore the Enter key. Once this was done, I executed the /challenge/runme program, resulting in the flag.

**Challenge 5: Specifying Filenames**

A screenshot of a computer

AI-generated content may be incorrect.

To obtain this flag, I started the challenge and opened the VSCode Workspace. This challenge was almost identical to the last one, but a slight twist was added. In addition to having a new password (oepkpuyh), there was no filename specified. This means I was allowed to store the password in any filename of my choice. For the purposes of this assignment, I named the file for this challenge as lab3. I also used the printf command to store the password rather than echo since it is more powerful. After vimming the program, I entered printf “oepkpuyh” > lab3 and started the program like so: /challenge/runme lab3. After the program checked my work, I was given the flag.

**Challenge 6: Binary and Hex Coding**

A screenshot of a computer

AI-generated content may be incorrect.

To obtain this flag, I started the challenge and opened the VSCode Workspace. This challenge introduced me to binary/hex coding, or more accurately how to read binary/hex. In all previous challenges, the password would have a b right before the password. This was because it reveals the format used to store the password: bytes. However, this time, the password was stored as a hex value. After vimming the program, I determined the password to be 0xf0, but 0x is not relevant to the password so it’s just f0. With this information, I returned to the terminal, executed the program, entered f0 as the password, and got the flag.

**Challenge 7: More Hex**

A screenshot of a computer

AI-generated content may be incorrect.

To obtain this flag, I started the challenge and opened the VSCode Workspace. This challenge is an add-on to what I had to do in the previous challenge. Once again, the password was stored as a hexadecimal, but instead of two hex values, I had to work with 16 hex values. In the program’s code, this hex value was stored like so: \xbe\xde\xa1\xf6\xbe\xa7\xd6\xf1. I had to enter this password, but without the \x (bedea1f6bea7d6f1). I returned to the terminal, started the program, entered the password without the \x, and retrieved the flag.

**Challenge 8: Decoding Hex**

A screenshot of a computer

AI-generated content may be incorrect.

To obtain this flag, I started the challenge and opened the VSCode Workspace. Unlike the previous two challenges, the password was stored without using \x, but the program expects this information when providing the password. I first retrieved the password by vimming to the program. The password was 88b5ceecae959ff8. The instructions for this challenge recommended be to enter the password using echo -e -n, which is what I did. I returned to the terminal, typed echo -e -n “\x88\xb5\xce\xec\xae\x95\x9f\xf8” | /challenge/runme, and got the flag.