Christopher Gonzalez

Professor Jennifer Holst

CSCI 401

22 February 2025

**pwn.college Lab 1**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Challenge 1: Your First Register**

**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 served as a guide on how to write my first assembly in the workspace. I began by familiarizing myself with the workspace and located the option to open the terminal on the left. Afterwards, I found the option to create a file where I was expected to write my assembly code to. I named my file lab.s as the instructions informed me the file needed to be an .s extension. Once I created this file, I entered “mov rax, 60” and checked my work by entering /challenge/check lab.s on the terminal. After my work was verified, the flag was given to me (this is highlighted in white).

**Challenge 2: Your First Syscall**

**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 required me to use the same file I created in the last challenge (lab.s) and provide a syscall at the end of the file. For context, syscall is short for “system call”, and this commad makes requests from the operating system it is running on. The type of request it makes depends on the value provided in the rax. For this challenge, I was informed the syscall I needed to invoke was exit, which is represented by the number 60. Since I already had this, I simply added “syscall” to my existing file and ran /challenge/check lab.s on the terminal to check my work. After this, the flag was given to me.

**Challenge 3: Exit Codes**

**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 introduced me to a new register: rdi. This register is used to pass a given argument to functions in the x86-64 architecture. This challenge expected me to use rdi to set the program’s exit code (42), then set the system call number of the exit syscall (using move rax) and syscall-ing at the end. In short, all I had to do was add “mov rdi, 42” to the beginning of my existing file and verify my work using /challenge/check lab.s on the terminal. Once all of this was done, I received the flag.

**Challenge 4: Building Executables**

**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 revealed to me I would be building executable files. It was necessary for me to follow the steps provided, which included tasks like catt-ing my file (lab.s), providing “.intel\_syntax noprefix” in the beginning of my file, conforming the presence of my file using ls, using the as tool to read/assemble the code in my lab.s file and output it to an object file I titled

“lab.o”. Once this was done, I created an executable file by typing “ld -o exe lab.o”, and checking my work by invoking /challenge/check exe on the terminal, resulting in the flag.

**Challenge 5: Moving Between Registers**

**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 introduced me to a new register known as rsi. This register is like rdi in the sense the register can be used to “park some data”. I was tasked with using the secret value stored in rsi and have the program exit with said value. I wasn’t informed what the value was, meaning I needed to enter “mov rdi, rsi” to set the programs exit code to the value stored in rsi. I was able to keep “mov rax, 60” and “syscall” and proceeded to verify my work by entering /challenge/check lab.s on the terminal. Once this was done, I got the flag.