Lab 3:

The vuln.c file was given to us and we did not have to make any changes to it. The file had declared main so when I named my file main.c instead of exploit.c. We start by declaring the headers. Then we define the variables (offset, dfs and Notar). As the offset value changes, our address changes. We then added an array that has the names of the char. We then create a function that finds the stack and return it. Then in the main function we write a function with the argc and argv arguments given to us. In the main we declare the variable, store the buffer, and declare the address. We then find the stack address and copy the notar to the pointer. Then we use a for loop from i = 0 to find the length of the code and then we exit. We also have to find the overflow, so we move the string to copy and put it into the buffer using the environmental variables. Lastly, we close, sleep, and overflow the system. Lastly, we return it to 0 and run. Initially, I was having errors because my file was not being read but then it finally performed the smash attack, and I got the result. I also had OS constrains but was able to figure them out. I also realized that I was able to run my main.c file in an online c compiler. I ran the vuln.c file by itself and wrote whoami on the resulting terminal and got runner as my result. When I ran the main.c file, I got the output that gave me the address and the runner which makes me feel that what I did was how the output is supposed to be.

Text

Description automatically generated

And then I realized my code doesn’t print “The buffer says..” and I was having issues with the buffer and overflow but it finally worked.

Text

Description automatically generated

And lastly, since I used an online complier, it prints it as a runner instead of my name.