ECE 404 Homework #10

Due: Tuesday 04/04/2023 at 5:59PM

Buffer Overflow Attacks

Included with this homework, you will find two socket programs written in C. One of them acts as a server and the other as a client. Your first task is to launch an attack that executes the "secret" function in the server side code by using the client program to send a carefully crafted string to the server. Following this, your second task must show how you would fix server.c to prevent such a buffer overflow attack. Here are the instructions for performing these two tasks:

- If you are using `gcc` on your Linux machine, you must compile the server and client programs with the `-fno-stack-protector` option. Refer to pages 36-39 of Lecture 21 for more details.
- You can test the programs with two different shell terminals: one terminal for a server and the other terminal for a client. You can also run the server on a Purdue ECN machine using a high numbered port like 7777 and the client on your own laptop.
- Use `gdb` to determine how you can develop a string to send (using the client program) to the server program and trigger the execution of `secretFunction()`. Refer to Lecture 21.6 for more details on how to do this. Here are some things to keep in mind when creating this string:
  1. While you send the data with the client program, you will have to run the server program with `gdb` to determine the buffer overflow string to use.
  2. When sending the string to the server program, note that you can send ASCII characters as well as hexadecimal-format bytes. You can send, for example, the hex byte `0xAD` using the format `\xAD`.
  3. As in the lecture notes, you will need to reverse the order of addresses sent to deal with big endian-little endian conversion problems.
- You may need to turn off ASLR on your operating system for the attack to work properly.
- After you have completed crafting your buffer overflow string and then triggering the execution of `secretFunction()`, modify `server.c` to remove the buffer overflow vulnerability. Your fix should allow the program to run without a buffer overflow (i.e. the program should not simply exit to deal with the buffer overflow attack).
- **Include comments** in the code explaining what the vulnerability was and how you fixed it.

Useful Notes

- Make sure that the server program is running before you start the client program.
- You can use whichever high-numbered port number you like, but you will need to update the `client.c` file and recompile each time you change it.

- Be mindful of any aliases you may have created for the `gcc` command in previous classes (you should be able to view such aliases, if they exist, in `~/.bashrc`). You may unknowingly be using additional compiler flags with `gcc` if it is in fact aliased. These additional compiler flags may result in errors when trying to perform the buffer overflow attack.

Submission Instructions

- For this homework you will be submitting a zip file titled `hw10_<last name>_<first name>.zip`, which consists of:
  - A pdf titled `hw10_<last name>_<first name>.pdf` containing:
    - the gdb commands and their respective outputs used to mount the buffer overflow attack along with comments for each command
    - the specially-crafted buffer overflow string
    - the modified server code along with an explanation of your fixes