CSC 472 Fall 2022 Lab 3

Dr. Si Chen

October 28, 2022

Return-oriented programming (ROP)

The goals of this lab:

- Understanding the concepts of Return-oriented programming (ROP)
- Exploiting a Return-oriented programming (ROP) vulnerability

Objectives and Targets

Target 1: Understanding ROP

Please **draw a diagrams to illustrate the structure of your ROP payload for lab3. Explain why your paylod will work and how it forms the ROP chain.** In particular, please include the following in your diagram:

1. The order of dummy characters, parameters (if applicable), return address, and ROP gadgets
2. Length of the dummy characters to reach return address

Target 2: Launch Return-oriented programming (ROP) Attack

The provided C code (lab3.c) contains a stack buffer overflow vulnerability. Please create a ROP gadget chain. The high level idea is to overwrite the return address with the address of function *add_bin()* and then create ROP chain to execute *add_bash()* and *exec_string()* sequentially. You should get a new shell (e.g. executing a linux command '/bin/bash') if success.

You can use our Badger CTF system to complete this project.

Steps:

1) Copy the provided C code from the shared folder on BadgerCTF.

```
cp /workdir/ss2022/lab3/lab3.c /workdir/
```

2) Copy the provided binary file from the shared folder to your workdir on BadgerCTF (which you will be exploiting):

```
cp /workdir/ss2022/lab3/lab3 /workdir/
```

3) To run this program, type the following command:

```
./lab3
```

4) When you type a very long list of characters, you will notice lab3 crashes with memory segfault, this is because the return address has been overwritten by your data.

5) Now you can craft your payload using the provided skeleton Python script. Again, your goal is to overwrite the return address with the address of function **add_bin()** and then create ROP chain to execute **add_bash()** and **exec_string()**. **You also need to prepare proper arguments and store them in the stack before calling target function**.

GDB can be used to find these library addresses and test/debug your exploit. However, it should be noted that your final exploit (i.e., the final version of your Python script) should work outside of GDB. Just running

```
python3 rop_exp.py
```

You should get a new shell if success.

6) Provide a screenshot of you exploiting sort.

7) Have fun.

Deliverables: the Python script you crafted and a screenshot of the exploit. The string of your shellcode and the screenshot should be put into the PDF file.

More...

Please check lecture video (http://moozlab.com/ss2020/rop.html, example 1)

Submission

- The lab due date is available on our course website. Late submission will not be accepted;
- The assignment should be submitted to D2L directly.

- Your submission should include: A **detailed project report in PDF format** to describe what you have done, including screenshots and code snippets.
- **No copy or cheating is tolerated**. If your work is based on others', please give clear attribution. Otherwise, you **WILL FAIL** this course.