Secure Programming Laboratory 2: Introduction

SP Lab Team

6th February 2018
This is the second Laboratory Session for Secure Programming

It is convened by Arthur, Connie, David and Margus.

Please take a copy of the handout. It is also available online via the course web page.
Solutions to previous lab

- See solution sheet on the Lab 1 web page.
What is this lab about?

There are 5 exercises showing exploits which result from **data corruption** as we have studied in Lectures 3-6.

- **Exercise 0.** Data corruption puzzles.
- **Exercise 1.** Stack overflows, corrupting memory.
- **Exercise 2.** Corrupting a network protocol (Java).
- **Exercise 3.** A more advanced buffer overflow.
- **Exercise 4.** A real-life data handling flaw in a version of OpenSSL.

The exercises are based on executable and source files provided in a Virtual Machine running Linux.

You’ll be given the exploits; ask us for the passwords when you’re ready.
What do we hope you will learn?

▶ More basics of low-level attacks, including manipulating executable code and memory.
▶ Thinking about program flaws in C and Java, and how they may be exploited.
▶ Practice repairing or preventing some attacks, and checking to see that they have been stopped.
Resources

► Use anything! You are encouraged to search on the web for help, tutorials, manuals, etc.

► You can get plenty of help this way. But it is probably more rewarding to try to solve the exercises for yourself first. Make sure to spend time experimenting, not only reading.

► **Warning**: experiment with care! If you download sample exploits, generation tools, etc, install and run these in the Virtual Machine, **not on the host DICE environment**. The VM already has several interesting tools provided.

► **Ask us!** We are here to help, as much as we can.

► **Ask each other!** There may be expert x86 programmers, C hackers, exploit developers(?) among you...
Timing

You probably won’t have time to complete everything.

▶ Don’t worry!
▶ Exercise 0 is optional; move on if you don’t see the answers.
▶ Exercise 3 is optional, try only if you found Exercise 1 easy.
▶ Exercise 4 involves a large software package, it will be difficult to understand the context.
▶ Of course: spend your own time later if you want but completing the lab is not essential. At least, try to look at each exercise a little bit, and review the solutions when they are released. The important thing is to understand the concepts well.
Discussion

We may enable a course forum for online discussion after the lab.

During the lab we will provide individual help and guidance, and also make announcements during the lab with hints and tips.
Feedback on your work

Besides discussing with us, we will give additional brief feedback *at the next lab* based on **submitted work**.

Submission is optional.

Each exercise has a series of **Checkpoints** which are questions that you can provide brief answers to. A plain text file `checkpoints.md` is provided, please fill this in and submit it electronically. The main point of the checkpoints is for you to check your own understanding, so please **don’t spent time writing long or very polished answers!**

Personalised feedback will only be available on submissions made by the deadline of **Monday 12th Feb, 4pm**. This is to discourage you from spending too much time.
Good Luck!

We hope you enjoy the lab.