Lab 4 – Buffer Overflow

Updated: 2023-11-16.

Introduction to the lab

This individual assignment may take 3 classes.

Learning outcomes

— Learn the basics of Buffer Overflow.

— Learn binary exploitation using the toolkit Pwntools.

Submission

You may submit as you go but be sure to complete and submit all activities up to 5 days after the last class (Wednesday at 11:50 pm – more than one month due to holidays).
We strongly recommend submitting the work at the end of each class as it is, and then, improving it.

Report structure

The report should have (at least) the following contents:

- Scope of this assessment;
- Summary of the activities of each module and evidence that you made each step (with screenshots);
- Print and attach the PDF after concluding the TryHackMe modules;
- Answer the questions that are raised in the document.

2.1 TryHackMe – Buffer Overflow Prep

This module will introduce you to stack buffer overflow.
Helpful links:

- https://github.com/joshua17sc/Buffer-Overflows
- https://github.com/V1n1v131r4/OSCP-Buffer-Overflow

2.2 TryHackMe – Intro To Pwntools

This module is an introductory room for the binary exploit toolkit Pwntools.

2.3 Player

The source code of this application is a single file written in C language designated *player.c*. This application is a simple MP3 player that prints some of the metadata present in the file. This application contains vulnerabilities that allow the execution of buffer overflow attacks. The goal is to find and exploit the bugs that may lead to an attack vector. For this analysis, the ASLR should be turned off and the player compiled with the flags “-no-pie” and “-fno-stackprotector”.

1. Use Flawfinder to identify possible entry points.
2. Create a playload that chashes the application.

3. Dump stack addresses using filename.
4. Dump stack addresses using MP3 tags.
5. Create a denial-of-service using format string attacks.
6. Perform a remote code execution attack.