CSC 583 Fall 2024 Final Project

Dr. Si Chen

December 2, 2024

Final Project Overview

This project consists of two main components:

1. **Quiz (Final Project Hint), 10%:** Available on D2L. This quiz will provide crucial hints for the final project.
2. **Capture the Flag on a Remote Server, 40%:** This challenge will be a key part of your final assessment.

Server Details for Final Project

- **Target IP:** 143.244.149.32
- **Target Port:** 8888
- **Vulnerable program:** final
- **Target File (flag):** flag.txt
- **Security Measures:** ASLR is Off, XN is on
- **Libc Info:** Libc library has been loaded at 0x40835000 on the remote server
- **Gadget Info:** check class website, "Gadget Information" for remote libc gadgets

Exploit Objectives for Final Project

Students will be required to apply their skills in a realistic environment, targeting the server details provided.

Steps for Final Project

Students should follow the process similar to previous labs, adapting their methods to the unique challenges of this project.

Deliverables: A detailed report in PDF format, documenting the steps taken to complete the final project, including any code, screenshots, and the contents of the 'flag.txt' file.

1 Hints

Buffer Overflow Vulnerability: Observe the buffer overflow vulnerability in the 'vulnerable' function. The function reads more data into the buffer than its allocated size. This is your opportunity to manipulate the memory beyond the buffer's boundary.

Understanding 'not_called': Notice the 'not_called' function. It isn't invoked during the normal execution of the program. Your challenge is to manipulate the program's control flow to execute this function.

Global Variable 'x': The global variable 'x' plays a crucial role in triggering the 'not_called' function. Understand how global variables are stored and how you might alter 'x' through buffer overflow.

Exploit Objective: Your goal in this exploit is to overflow the buffer, set 'x' to '0xdeadbeef', and redirect execution to 'not_called'. This will require careful planning and understanding of the stack's layout.

Memory Layout and Control Flow: Consider the program's memory layout, especially how the stack is structured. Your exploit needs to carefully manipulate the stack to change the program's control flow.

Practical Steps: Begin by determining the necessary offsets to reach and overwrite the return address. Then, decide on the right values to inject to achieve your objectives.

Pleas also refer to the quiz on D2L for helpful hints.

Submission for Final Project

- Consult the course website for submission deadlines.
- Upload your project report to D2L.

- Ensure that your report is comprehensive and adheres to academic integrity guidelines.