Homework 2: Debugging Tools, Address Spaces, and Buffer

Xin Liu

Florida State University
xliu15j@fsu.edu

CIS 5370 Computer Security
https://xinliulab.github.io/cis5370.html
January 18, 2025

Question 1

Objective: Let us use debugging tools (including large language models) to understand how a program interacts with the operating system by invoking system calls.

Task:

1. Use a large language model, such as ChatGPT or Copilot, to help you write an x86-64 assembly program that prints `Hello World`.
2. Do not use `printf`. Instead, directly invoke a Linux system call to print `Hello World` and exit.
3. Use `gdb` commands such as `starti`, `layout asm`, and `si` to step through the program and explain each instruction.
4. Your answer should include:
   - Your assembly code.
   - Screenshots (2 to 3) of step-by-step debugging with `si`.
   - Explanation of each instruction.
5. The student submitting the smallest code will receive 1 point of extra credit.

Question 2

Objective: Use debugging tools to understand how the address space works by analyzing program outputs and instruction locations.

```c
#include <stdio.h>
int main()
{
    printf("%p\n", main);

    int x = *(int*)main;
    printf("%x\n", x);
}
```

Task:

1. Explain why the program produces two different outputs.
2. Specifically analyze how the first output relates to the second output.
3. Your answer should include:
   - Two outputs.
   - A detailed explanation with calculations of the relationship between the address, the instruction bytes, and the program's outputs.
   - Explanation supplemented by several screenshots of debugging using `objdump` or `gdb`.

Question 3

Objective: Understand how buffering works in the address space.

```c
#include <stdio.h>
#include <unistd.h>
int main() {
    for (int i = 0; i < 2; i++) {
        fork();
        printf("Hello\n");
    }
}
```

Execution: Run the above program using the following commands:

1. `gcc hello.c | ./a.out`
2. `gcc hello.c | ./a.out | cat`

Task:

1. Explain why the outputs of the two commands differ.
2. Your explanation must include:
   - The exact outputs of both commands.
   - A detailed analysis of the buffering mechanism and how it affects output.
   - Screenshots of debugging using tools such as `objdump` or `gdb`.