National Taiwan Normal University
CSIE Information Security: A Hands-on Approach

Instructor: Po-Wen Chi
Due Date: 12 06, 2021, AM 11:59

Assignment 4

Policies:

- Zero tolerance for late submission.
- Please pack all your submissions in one zip file. **RAR is not allowed!!**
- I only accept **PDF**. MS Word is not allowed.
- Hand-writing is not allowed.

4.1 SEED Lab (40 pts)

Return-to-libc Attack Lab
https://seedsecuritylabs.org/Labs_20.04/Software/Return_to_Libc/

Please record all your steps with screen captures and answer all questions. Undoubtedly, you need to complete all tasks even they are marked with **optional**.

4.2 randomize_va_space (15 pts)

In this class, I have taught you how ASLR works. I have also shown you that **randomize_va_space** have three options: 0, 1, 2. Undoubtedly, 0 is to disable ASLR. What is the difference between 1 and 2? Please give your description and use a lab to verify your description. What will happen if you set other numbers like 3 and 4?

4.3 Enter (15 pts)

In this class, I have introduced how **enter** works. The first number is used to allocate memory for local variables. However, I never explain the second argument of **enter**. Would you please explain what the second argument means and why it must be set to 0 for C?

enter 0, 0

4.4 Stack Layout (15 pts)

Given the following C code.

```
#include <stdio.h>
#include <stdint.h>
void g(int8_t n, int8_t x, int8_t y);
void f(int8_t x, int8_t y);

int main()
{
    f(4,5);
}

void f(int8_t x, int8_t y)
{
    g(2,x,y);
    return;
}

void g(int8_t n, int32_t x, int8_t y)
{
    int8_t z;
    if (n == 0)
    {
        printf("Values: [%d,%d]\n",x,y);
        return;
    }
    else
    {
        z = x+y;
        g(n- 1 , z, z+1);
    }
    return;
}
```

According to C calling convention, what is the content of the stack at the point when the code prints "Values: ...". Assume that there is nothing on the stack up to the call to function f. When showing values on the stack show **integer values** whenever possible as opposed to variable name.

4.5 Defeat Dash's Countermeasure with ROP (15 pts)

You all know **dash**'s countermeasure, right? It is time for you to use the ROP technique to get a root shell from a set-UID process with a overflow vulnerability. You should use **stack_rop.c** from the website and design a tutorial. That is, you need to list all your steps.