Secure System Design: Threats and Countermeasures
CS392

`Date: 23rd Feb 2021` `MidSem Assignment`
`Submission Filename: rollno.pdf` `Due Date: 23rd Feb 2021`
`Full Marks 25`

1 Assignment Overview

In this assignment, a program with a format-string vulnerability is provided; the primary task is to develop a scheme to exploit the vulnerability.

It should be noted that the outcome of this assignment is operating system dependent. The description and discussion are based on Ubuntu Linux. It should also work in the most recent version of Ubuntu. However, if you use different operating systems, different problems and issues might come up.

2 Assignment Task: Exploit the vulnerabilit

For this task, turn off the address randomization. You can use the following command to turn off the address randomization:

```
sudo sysctl -w kernel.randomize_va_space=0
```

In the following program, you will be asked to provide an input, which will be saved in a buffer called `user_input`. The program then prints out the buffer using `printf`. The program is a `root` owned set-uid program, i.e., it runs with the root privilege. Unfortunately, there is a format-string vulnerability in the way how the `printf` is called on the user inputs. You need to exploit this vulnerability and see how much damage you can achieve.

The program has two secret values stored in its memory, and you are interested in these secret values. However, the secret values are unknown to you, nor can you find them from reading the binary code (for the sake of simplicity, we hardcode the secrets using constants 0x44 and 0x55). Although you do not know the secret values, in practice, it is not so difficult to find out the memory address (the range or the exact value) of them (they are in consecutive addresses), because for many operating systems, the addresses are exactly the same anytime you run the program. In this assignment, we just assume that you have already known the exact addresses. To achieve this, the program "intentionally" prints out the addresses for you. With such knowledge, your goal is to achieve the followings (one at a time):

(i) Crash the program.

(ii) Print out the secret[1] value.

(iii) Modify the secret[1] value.

(iv) Modify the secret[1] value to a pre-determined value (choose any number between 80-100).

Note that the binary code of the program is only readable/executable by you, and there is no way you can modify the code. Namely, you need to achieve the above objectives without modifying the vulnerable code. However, you do have a copy of the source code, which can help you design your attacks.

```
/* vul_prog.c */
#include<stdio.h>
#include<stdlib.h>

#define SECRET1 0x44
#define SECRET2 0x55

int main(int argc, char *argv[])
{
    char user_input[100];
    int *secret;
    long unsigned int_input;
    int a, b, c, d; /* other variables, not used here.*/
    /* The secret value is stored on the heap */
    secret = (int *) malloc(2*sizeof(int));

    /* getting the secret */
    secret[0] = SECRET1; secret[1] = SECRET2;

    printf("The variable secret's address is 0x%8x\n",
                    (unsigned int)&secret);
    printf("The variable secret's value is 0x%8x\n",
                    (unsigned int)secret);
    printf("secret[0]'s address is 0x%8x\n",
                    (unsigned int)&secret[0]);
    printf("secret[1]'s address is 0x%8x \n",
                    (unsigned int)&secret[1]);

    printf("Please enter a decimal integer\n");
    scanf("%lu", &int_input);  /* getting an input from user */
    printf("Please enter a string\n");
    scanf("%s", user_input); /* getting a string from user */

    /* Vulnerable place */
    printf(user_input);
    printf("\n");

    /* Verify whether your attack is successful */
    printf("The original secrets: 0x%x -- 0x%x\n", SECRET1, SECRET2);
    printf("The new secrets:      0x%x -- 0x%x\n", secret[0], secret[1]);
    return 0;
}
```

3 Submission

You need to submit a detailed report to describe what you have done and what you have observed; you also need to provide explanation to the observations that are *interesting* or *surprising*. Add necessary snapshots of your experiment wherever applicable in support of your observation. Upload your file (**rollno.pdf** or **rollno.doc**) using following google link only.

https://docs.google.com/forms/d/1q3gt11BIXWIoBtdSEW6RrCh91As6uqPj_rc7bV-A2SM