Mandatory assignment I

University of Bergen – INF226

Autumn 2021

In this exercise you will exploit buffer overflows in some exercise programs. The programs run on a remote server and the goal in each exercise is to take control over the program so that it outputs the file 'flag'.

Evaluation

This project will be graded on a scale from 0–10 points. This project will count 10% towards your final grade. To pass this assignment you must score at least 3 points.

Individual work

This is an individual exercise, each student hands in their own solution.

Files

For each exercise you have access to both the program binary running on the server and the source code. These files can be found in the MittUiB files section.

Report

Your submission to this assignment should consist of a report which explains how you got the flag in each exercise. Describe what the vulnerability is and how you exploited it. Any scripts you wrote or manual steps you performed should be included and described.

The score you get on each exercise is not just determined by getting the flag, but you showing that you understand the vulnerability and the exploit.

Tools

The main tools for this analysis are:

- Pwntools
- 'objdump -d'
- checksec
- GDB

Pwntools is a python library for CTF exploit development. You can use it to connect to the server and communicate with the program running there. It can also be used to pack payloads from hexadecimal strings.

Using `objdump -d` you can dissassemble the program to understand how it behaves at runtime. You can also use `checksec` to investigate what buffer overflow mitigations were enabled by the compiler.

You can also glean some info from running the program in **GDB**, but note that the execution of the program might be slightly different on your machine from what it is on the server.

How do I get started?

Last year our TA, Kenneth Fossen, produced this write up: `https://spydx.github.io/inf226.h20/mandatory1.html` on how to get started. **Notice:** The subdomain (`ctf21.softwaresecurity.no`) for this year's exercises is different from the one last year ('oblig1.softwaresecurity.no').

Exercises

The exercise programs run on a server located at `ctf21.softwaresecurity.no`. Each exercise has its own port number.

0x00 – 3 points

This exercise is found on port 7000.

Source

```
#include <assert.h>
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char **argv){
    struct{
        char buffer[16];
        int32_t check;
    }locals;
    locals.check=0xabcdc3cf;
    printf("Input an argument to pass\n");
    fflush(stdout);
    assert(fgets(locals.buffer, 512, stdin) != NULL);
    if(locals.check == 0x79beef8b){
        printf("Well done, you can get the flag\n");
        fflush(stdout);
        system("cat flag");
    }
    else{
        printf("Uh oh, value is not correct.
        please try again. Goodbye.\n");
    }
    return 0;
}
```

0x01 – 3 points

This exercise is found on port 7001.

Source

```
#include <stdio.h>
#include <stdlib.h>
#include <assert.h>

void getFlag(){
    printf("Congrats! you can get the flag\n");
    fflush(stdout);
    system("cat flag");
}

int main(int argc, char **argv){
    struct {
        char buffer[16];
        volatile int (*funcPointer)();
    } stores;

    stores.funcPointer = NULL;
    printf("Try to get flag by inputing argument\n");
    fflush(stdout);
    assert(fgets(stores.buffer, 512, stdin) != NULL);
    if(stores.funcPointer){
        printf("Function is goint to %p\n",
        stores.funcPointer);
        fflush(stdout);
        stores.funcPointer();
    }
    else{
        printf("oh no, please try again!\n");
    }
    return 0;
}
```

0x02 – 2 points

This exercise is found on port 7002.

Source

```
#include <stdio.h>
#include <stdlib.h>
#include <assert.h>

void getFlag(){
    printf("Congrats! you can get the flag\n");
    fflush(stdout);
    system("cat flag");
}

int main(int argc, char **argv){
    char buffer[16] = {0};
    int offset = 0;
    printf("What does the canary say?\n");
    fflush(stdout);
    scanf("%d", &offset);
    getchar();
    printf("%lu\n", *(unsigned long*)(buffer+8+offset));
    fflush(stdout);
    printf("Try to get flag by inputing value\n");
    fflush(stdout);
    assert(fgets(buffer, 512, stdin) != NULL);
    return 0;
}
```

0x03 – 2 points

This exercise is found on port 7003.

Source

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <assert.h>


void getFlag(){
    printf("Well done, you can get the flag\n");
    fflush(stdout);
    system("cat flag");
    return;
}

int main(int argc, char ** argv){

    unsigned long val = 5;
    struct {
        char buffer[32];
        unsigned long* pt0;
    }locals;

    locals.pt0 = &val;

    while(locals.buffer[0] != 'q'){
        printf("Do not, for one repulse, forego the
        purpose that you resolved to effect -William
        Shakespeare, The Tempest\n");
        fflush(stdout);

        assert(fgets(locals.buffer, 512, stdin) != NULL);

        printf("%lx\n", *locals.pt0);

        fflush(stdout);

    }

    return 0;
}
```