Xuan(James) Zhai

CS3339 Lab

April 16th, 2021

CS 3339 - Lab 7 - Lab Report

1: Get stack pointer

```
┌──(student㉿kali)-[~/Desktop/Lab7]
└─$ gcc -o getsp get_sp.c

┌──(student㉿kali)-[~/Desktop/Lab7]
└─$ ls
a.out  EC.c  getsp  get_sp.c

┌──(student㉿kali)-[~/Desktop/Lab7]
└─$ ./getsp
Stack pointer (ESP): 0xbffff158

┌──(student㉿kali)-[~/Desktop/Lab7]
└─$
```

2: Check EC & get_sp

```
┌──(student㉿kali)-[~/Desktop/Lab7]
└─$ ./getsp
Stack pointer (ESP): 0xbffff158

┌──(student㉿kali)-[~/Desktop/Lab7]
└─$ ./getsp
Stack pointer (ESP): 0xbffff158

┌──(student㉿kali)-[~/Desktop/Lab7]
└─$ gcc -fno-stack-protector -z execstack -mpreferred-stack-boundary=2 -o EC -ggdb EC.c
EC.c: In function 'main':
EC.c:7:2: warning: implicit declaration of function 'strcpy' [-Wimplicit-function-declaration]
    7 |  strcpy(theString, argv[1]);
      |  ^~~~~~
EC.c:7:2: warning: incompatible implicit declaration of built-in function 'strcpy'
EC.c:2:1: note: include '<string.h>' or provide a declaration of 'strcpy'
    1 | #include <stdio.h>
  +++ |+#include <string.h>
    2 |

┌──(student㉿kali)-[~/Desktop/Lab7]
└─$ ls
EC  EC.c  getsp  get_sp.c

┌──(student㉿kali)-[~/Desktop/Lab7]
└─$ chmod u+s EC

┌──(student㉿kali)-[~/Desktop/Lab7]
└─$ ./EC helloWorld
You entered: helloWorld

┌──(student㉿kali)-[~/Desktop/Lab7]
└─$
```

3: Find the number of overwrite.

```
┌──(student㉿kali)-[~/Desktop/Lab7]
└─$ ./EC `perl -e "print 'A'x405";`
You entered: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAA

┌──(student㉿kali)-[~/Desktop/Lab7]
└─$ ./EC `perl -e "print 'A'x407";`
You entered: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

┌──(student㉿kali)-[~/Desktop/Lab7]
└─$ ./EC `perl -e "print 'A'x408";`
You entered: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
zsh: segmentation fault  ./EC `perl -e "print 'A'x408";`

┌──(student㉿kali)-[~/Desktop/Lab7]
└─$ 
```

```
(gdb) run `perl -e "print 'A'x411";`
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/student/Desktop/Lab7/EC `perl -e "print 'A'x411";`
You entered: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

Program received signal SIGSEGV, Segmentation fault.
0x00414141 in ?? ()
(gdb) info reg ebp eip
ebp            0x41414141          0x41414141
eip            0x414141            0x414141
(gdb) run `perl -e "print 'A'x412";`
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/student/Desktop/Lab7/EC `perl -e "print 'A'x412";`
You entered: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

Program received signal SIGSEGV, Segmentation fault.
0x41414141 in ?? ()
(gdb) info reg ebp eip
ebp            0x41414141          0x41414141
eip            0x41414141          0x41414141
(gdb) 
```

4: Open the terminal

5: What are some malicious ways this attack could be used?

Since the Buffer Overflow may overwrite other parts of the program or even things at the outside of the program, an attacker could overwrite parts of the program they shouldn't have access to and even get program to execute their own code.

6: How could you protect against this type of attack?

The attack should be prevented at the beginning. For example, the development should use a memory safe language with kernel memory protection. Also, if the program requires a user to enter an input, the program needs to have some error-check mechanism to prevent input get out of bound.