

BÁO CÁO CHI TIẾT LAB05 – NHÓM 6

Thành viên của nhóm 6:

- Nguyễn Hải Phong – 22521088
- Nguyễn Chí Thành – 22521350
- Trần Văn Thuận – 22521448

Dùng IDA cho phiên bản x86 (đây là file ELF 32-bit). Ta sẽ xem được flowchart của chương trình:

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
    int line; // eax
    int v4; // eax
    int v5; // eax
    int v6; // eax
    char *s; // [esp+0h] [ebp-Ch]

    puts(
        "Welcome to UIT's bomb lab.\n"
        "You have to deactivate our bomb by solving 5 phases with the correct inputs consecutively, and otherwise the bomb will be blown up!");
    if ( argc == 1 )
    {
        infile = (FILE *)stdin;
    }
    else
    {
        if ( argc != 2 )
        {
            printf("Usage: %s [<input_file>]\n", *argv);
            exit(8);
        }
        infile = fopen(argv[1], "r");
        if ( !infile )
        {
            printf("%s: Error: Couldn't open %s\n", *argv, argv[1]);
            exit(8);
        }
    }

    puts("\n[*] Phase 1\n- Hint: Numbers are always magical!");
    line = read_line();
    phase1(line);
    defuse_bomb();
    puts("\n[*] Phase 2\n- Hint: You must answer your secret question!");
    v4 = read_line();
    phase2(v4);
    defuse_bomb();
    puts("\n[*] Phase 3\n- Hint: Many cases make everything so confusing.");
    v5 = read_line();
    phase3(v5);
    defuse_bomb();
    puts("\n[*] Phase 4\n- Hint: Let's dig in to recursive function :)");
    v6 = read_line();
    phase4(v6);
    defuse_bomb();
    puts("\n[*] Phase 5\n- Hint: No hint is also a hint :)");
    s = (char *)read_line();
    phase5(s);
    defuse_bomb();
    return 0;
}
```

Ta thấy được chương trình trên sử dụng đến 5 hàm (tương ứng với 5 challenges mà chúng ta cần phải giải quyết). Nếu ta cung cấp đúng 5 input thỏa mãn yêu cầu của 5 challenges trên thì ta đã phá bom thành công. Trong đó 5 challenges tương ứng với 5 phase từ 1 đến 5 của đoạn chương trình này.

Câu 1: Về challenge phase 1:

Đầu tiên, để có thể giải nhanh chóng bài này, ta có thể xem qua mã giả của hàm phase1 này:

```
1 int __cdecl phase1(int a1)
2 {
3     int result; // eax
4     int v2[6]; // [esp+Ch] [ebp-2Ch] BYREF
5     int v3; // [esp+24h] [ebp-14h]
6     int v4; // [esp+28h] [ebp-10h]
7     int i; // [esp+2Ch] [ebp-Ch]
8
9     v4 = __isoc99_sscanf(a1, "%d %d %d %d %d %d", v2, &v2[1], &v2[2], &v2[3], &v2[4], &v2[5]);
10    if ( v4 != 6 )
11        explode_bomb();
12    v3 = 10;
13    result = v2[0];
14    if ( v2[0] < 10 )
15        explode_bomb();
16    for ( i = 1; i <= 5; ++i )
17    {
18        result = v2[i];
19        if ( result != 2 * v2[i - 1] )
20            explode_bomb();
21    }
22    return result;
23 }
```

Ta có thể thấy rằng đầu tiên hàm này sẽ yêu cầu ta nhập vào tổng cộng là 6 input với kiểu dữ liệu là số (dựa trên format %d có trong hàm scanf).

```
v4 = __isoc99_sscanf(a1, "%d %d %d %d %d %d", v2, &v2[1], &v2[2], &v2[3], &v2[4], &v2[5]);
if ( v4 != 6 )
    explode_bomb();
```

Đồng thời hàm sẽ kiểm tra liệu chúng ta đã nhập vào đủ 6 số chưa, nếu không thỏa điều kiện thì chương trình sẽ gọi đến hàm explode_bomb ám chỉ chúng ta đã không thỏa điều kiện đề bài.

```
void __noreturn explode_bomb()
{
    puts("BOMB!!!!\nThe bomb has blown up. Try again.");
    exit(0);
}
```

Tiếp theo thì chương trình sẽ tiến hành lưu 6 số mà ta nhập vào trong một mảng array v2 gồm 6 phần tử. Lúc đó hàm cũng sẽ kiểm tra liệu phần tử đầu tiên liệu có nhỏ hơn 10, nếu có thì sẽ gọi đến hàm explode_bomb ám chỉ chúng ta đã không thỏa điều kiện đề bài.

```
v3 = 10;  
result = v2[0];  
if ( v2[0] < 10 )  
    explode_bomb();
```

Sau đó hàm sẽ chạy qua được một vòng lặp duyệt qua các phần tử trong mảng v2, trong đó phải thỏa điều kiện sao cho phần tử sau phải gấp 2 lần phần tử trước. Nếu không điều kiện trên thì hàm sẽ gọi đến hàm explode_bomb

```
for ( i = 1; i <= 5; ++i )  
{  
    result = v2[i];  
    if ( result != 2 * v2[i - 1] )  
        explode_bomb();  
}  
return result;  
}
```

Sau khi nắm được các điều kiện đặt ra của challenge phase1 này thì mình đã viết code python để mô phỏng lại cách tìm ra input của challenge này.

```
first_number = 10  
v2_arr = []  
v2_arr.append(first_number)  
for i in range(1,6):  
    v2_arr.append(2*v2_arr[i-1])  
  
for i in range(len(v2_arr)):  
    print(v2_arr[i], end=" ")
```

```
first_number = 12  
v2_arr = []  
v2_arr.append(first_number)  
for i in range(1,6):  
    v2_arr.append(2*v2_arr[i-1])  
  
for i in range(len(v2_arr)):  
    print(v2_arr[i], end=" ")
```

Vì ở trên, ta nhớ rằng đề bài chỉ yêu cầu số đầu tiên phải lớn hơn 10, nên ta có thể tạo ra nhiều chuỗi 6 số để thỏa yêu cầu của đề bài.

Minh chứng:

Với số đầu tiên là 10, thì mình đã tạo ra một chuỗi số có thể thỏa yêu cầu challenge đầu tiên này.

```

[?] /mnt/e/Stuff or sth like that/System_Programming/ThucHanh/Lab05 09:47:45
python3 exploiti.py
10 20 40 80 160 320 %

[?] /mnt/e/Stuff or sth like that/System_Programming/ThucHanh/Lab05 09:50:01
./nt209-uit-bomb
Welcome to UIT's bomb lab.
You have to deactivate our bomb by solving 5 phases with the correct inputs consecutively, and otherwise the bomb will be blown up!

[*] Phase 1
- Hint: Numbers are always magical!
10 20 40 80 160 320
Good job! You've cleared the first phase!

```

Với một số khác (miễn thỏa lớn hơn 10) thì mình cũng có thể tạo ra chuỗi 6 số thỏa mãn yêu cầu đề bài (giả sử mình chọn con số 12):

```

[?] /mnt/e/Stuff or sth like that/System_Programming/ThucHanh/Lab05 09:51:08
python3 exploiti.py
12 24 48 96 192 384 %

[?] /mnt/e/Stuff or sth like that/System_Programming/ThucHanh/Lab05 09:52:39
./nt209-uit-bomb
Welcome to UIT's bomb lab.
You have to deactivate our bomb by solving 5 phases with the correct inputs consecutively, and otherwise the bomb will be blown up!

[*] Phase 1
- Hint: Numbers are always magical!
12 24 48 96 192 384
Good job! You've cleared the first phase!

```

Câu 2: Về challenge phase2

Đầu tiên, để giải quyết được challenge này, ta cần coi qua mã giả của hàm phase2 này:

```

1 int __cdecl phase2(int a1)
2 {
3     int result; // eax
4     char *s1; // [esp+Ch] [ebp-1Ch]
5     char *s2; // [esp+10h] [ebp-18h]
6
7     s2 = ANSWERS[QA_MAP[9]];
8     s1 = (char *)transfer(a1);
9     if ( !*s2 || (result = is_equal(s1, s2)) == 0 )
10         explode_bomb();
11     return result;
12 }

```

Đầu tiên hàm sẽ yêu cầu nhập vào một chuỗi ký tự sao cho thỏa mãn yêu cầu của challenge này. Vậy ta cần phải hiểu được yêu cầu đề bài là gì.

Đầu tiên hàm sẽ truy xuất giá trị 2 mảng giá trị, đó là mảng QA_MAP và mảng ANSWERS. Đầu tiên ta sẽ nhìn qua giá trị của mảng QA_MAP:

.data:0804B0EB	db	0
.data:0804B0EC	db	3
.data:0804B0ED	db	0
.data:0804B0EE	db	0
.data:0804B0EF	db	0
.data:0804B0F0	db	4
.data:0804B0F1	db	0
.data:0804B0F2	db	0
.data:0804B0F3	db	0
.data:0804B0F4	db	5
.data:0804B0F5	db	0
.data:0804B0F6	db	0
.data:0804B0F7	db	0
.data:0804B0F8	db	6
.data:0804B0F9	db	0
.data:0804B0FA	db	0
.data:0804B0FB	db	0
.data:0804B0FC	db	7
.data:0804B0FD	db	0
.data:0804B0FE	db	0
.data:0804B0FF	db	0
.data:0804B100	db	8
.data:0804B101	db	0
.data:0804B102	db	0
.data:0804B103	db	0
.data:0804B104	db	9
.data:0804B105	db	0
.data:0804B106	db	0
.data:0804B107	db	0
.data:0804B108	db	0Ah
.data:0804B109	db	0
.data:0804B10A	db	0

Thì sau khi nhìn qua các phần tử của mảng này, thì mình đã tìm ra tại index thứ 9 của mảng sẽ có giá trị bằng 9. Tiếp tục thì mảng ANSWERS sẽ lấy giá trị 9 của mảng QA_MAP để tiến hành truy xuất giá trị tại mảng ANSWERS này. Bây giờ ta cần phải xem qua mảng này tại index bằng 9 chứa dữ liệu gì:

```

.data:0804B160 ANSWERS dd offset aKrwqMdxwp ; DATA XREF: phase2+2A↑r
.data:0804B160 ; "Krwq Mdxwp"
.data:0804B164 | dd offset aKjwptxt ; "Kjwptxt"
.data:0804B168 dd offset aRwoxavjcrxwBnl ; "Rwoxavjcrxw Bnldarch"
.data:0804B16C dd offset aByarwp ; "Byarwp"
.data:0804B170 dd offset aFrwmxf ; "Frwmxf"
.data:0804B174 dd offset aBrwpjyxan ; "Brwpjyxan"
.data:0804B178 dd offset aErncwjvnb ; "Erncwjvnb"
.data:0804B17C dd offset a941913 ; "94/1913"
.data:0804B180 dd offset aKrwqMdxwp ; "Krwq Mdxwp"
.data:0804B184 dd offset aBnenw ; "Bnenw"
.data:0804B188 dd offset aRwlxaanlcuh ; "Rwlxaanlcuh"
.data:0804B18C dd offset aErncwjvnb ; "Erncwjvnb"
.data:0804B190 dd offset a91726141991 ; "91726141991"
.data:0804B194 dd offset aWc198DrcKxvk ; "wc198-drc-kxvk"
.data:0804B198 dd offset aLqrljpx ; "Lqrljpx"
.data:0804B19C dd offset aJacrorlrjuRwcn ; "Jacrorlrju Rwcnuurpnwln"
.data:0804B1A0 dd offset aRbjknuuj ; "Rbjknuuj"
.data:0804B1A4 dd offset aDwrenabrchXoRw ; "Dwrenabrch Xo Rwoxavjcrxw Cnlqwxuxph"
.data:0804B1A8 dd offset aSxuurnnn ; "Sxuurnnn"
.data:0804B1AC dd offset aWlDrcNmdEw ; "wl.drc.nmd.ew"
.data:0804B1B0 dd offset aAdbbrj ; "Adbbrj"
.data:0804B1B4 dd offset a97951995 ; "97/95/1995"
.data:0804B1B8 dd offset aJwcjalcrly ; "Jwcjalcrly"
.data:0804B1BC dd offset aCqnPanjcFjuuXo ; "Cqn Panjc Fjuu Xo Lqrwj"
.data:0804B1C0 dd offset aAnprbcnab ; "Anprbcnab"
.data:0804B1C4 dd offset aVnaldah ; "Vnaldah"
.data:0804B1C8 dd offset aPvDrcNmdEw ; "pv.drc.nmd.ew"
.data:0804B1CC dd offset aEjwVrndZdxlCdP ; "Ejw Vrnd Zdxl Cd Prjv"
.data:0804B1D0 dd offset aRwcnaunc ; "Rwcnaunc"

```

Thì ta nhận thấy rằng trong mảng ANSWERS này thì tại index 9 thì sẽ có giá trị là một chuỗi ký tự **“Bnenw”**. Vậy giá trị của s1 sẽ là **“Bnenw”**

Còn giá trị tại s2 sẽ là chuỗi mà người dùng nhập vào và được biến đổi thông qua hàm transfer. Bây giờ ta phải xem qua hàm transfer này làm gì với input của người dùng:

```

1 int __cdecl transfer(int a1)
2 {
3     char v2; // [esp+Ah] [ebp-6h]
4     char v3; // [esp+Bh] [ebp-5h]
5     int i; // [esp+Ch] [ebp-4h]
6
7     for ( i = 0; *(_BYTE *)(i + a1); ++i )
8     {
9         v3 = *(_BYTE *)(i + a1);
10        if ( (v3 <= 96 || v3 > 122) && (v3 <= 64 || v3 > 90) )
11        {
12            if ( v3 > 47 && v3 <= 57 )
13                v3 = (v3 - 48 + 9) % 10 + 48;
14        }
15        else
16        {
17            if ( v3 <= 96 )
18                v2 = 65;
19            else
20                v2 = 97;
21            v3 = (v3 - v2 + 9) % 26 + v2;
22        }
23        *(_BYTE *)(a1 + i) = v3;
24    }
25    return a1;
26 }

```

Để tóm gọn lại thì hàm transfer này sẽ duyệt qua từng phần tử có trong chuỗi mà người dùng nhập vào và tiến hành biến đổi từng ký tự có trong chuỗi thành những ký tự mới theo quy tắc nhất định, đó là rotation 9 (tức là thay thế từng chữ cái bằng chữ thứ 9 sau nó trong bảng chữ cái). Sau khi biến đổi sau khi sẽ trả về một chuỗi ký tự hoàn toàn mới

Sau khi tiến hành biến đổi thành chuỗi ký tự mới từ chuỗi từ người dùng nhập vào. Thì chương trình sẽ tiến hành kiểm tra xem liệu chuỗi ký tự sau khi được biến đổi từ chuỗi người dùng có bằng với chuỗi của s1 (tức chuỗi “**Bnenw**”). Nếu không đúng thì chương trình sẽ gọi đến hàm explode_bomb để ám chỉ rằng chúng ta không đáp ứng yêu cầu của challenge này.

```

if ( !*s2 || (result = is_equal(s1, s2)) == 0 )
    explode_bomb();
return result;

```

Sau khi hiểu được quy luật của chương trình, để giải được bài này thì mình cần tìm chuỗi input sao cho qua hàm transfer thì sẽ thành chuỗi **Bnenw**. Vì vậy mình đã

tiến hành dịch ngược từ chuỗi **Bnenw** thông qua một đoạn code python để lấy được input thỏa yêu cầu này.

```
# Dictionary to lookup the index of alphabets
dict1 = {'A' : 1, 'B' : 2, 'C' : 3, 'D' : 4, 'E' : 5,
        'F' : 6, 'G' : 7, 'H' : 8, 'I' : 9, 'J' : 10,
        'K' : 11, 'L' : 12, 'M' : 13, 'N' : 14, 'O' : 15,
        'P' : 16, 'Q' : 17, 'R' : 18, 'S' : 19, 'T' : 20,
        'U' : 21, 'V' : 22, 'W' : 23, 'X' : 24, 'Y' : 25, 'Z' : 26}

dict1_lower = {'a' : 1, 'b' : 2, 'c' : 3, 'd' : 4, 'e' : 5,
              'f' : 6, 'g' : 7, 'h' : 8, 'i' : 9, 'j' : 10,
              'k' : 11, 'l' : 12, 'm' : 13, 'n' : 14, 'o' : 15,
              'p' : 16, 'q' : 17, 'r' : 18, 's' : 19, 't' : 20,
              'u' : 21, 'v' : 22, 'w' : 23, 'x' : 24, 'y' : 25, 'z' : 26}

# Dictionary to lookup alphabets
# corresponding to the index after shift
dict2 = {0 : 'Z', 1 : 'A', 2 : 'B', 3 : 'C', 4 : 'D', 5 : 'E',
        6 : 'F', 7 : 'G', 8 : 'H', 9 : 'I', 10 : 'J',
        11 : 'K', 12 : 'L', 13 : 'M', 14 : 'N', 15 : 'O',
        16 : 'P', 17 : 'Q', 18 : 'R', 19 : 'S', 20 : 'T',
        21 : 'U', 22 : 'V', 23 : 'W', 24 : 'X', 25 : 'Y'}

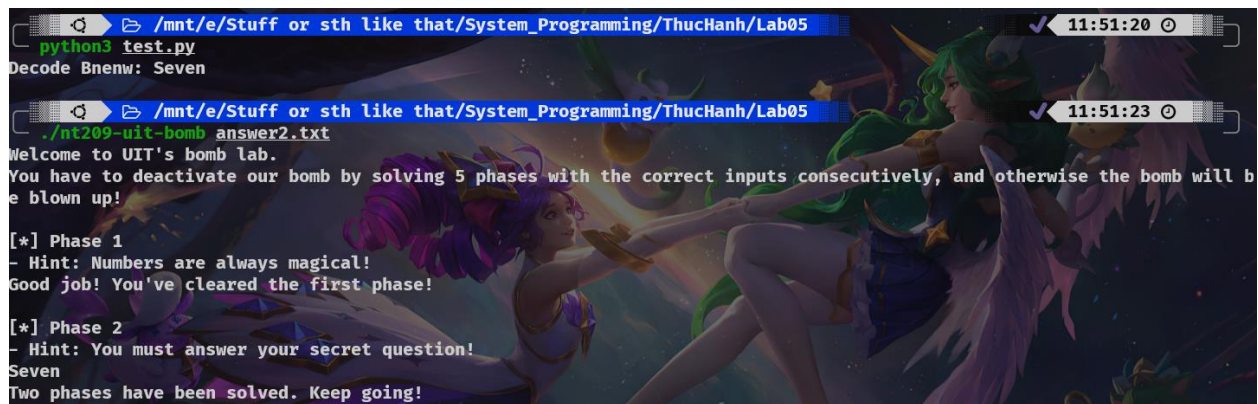
dict2_lower = {0 : 'z', 1 : 'a', 2 : 'b', 3 : 'c', 4 : 'd', 5 : 'e',
              6 : 'f', 7 : 'g', 8 : 'h', 9 : 'i', 10 : 'j',
              11 : 'k', 12 : 'l', 13 : 'm', 14 : 'n', 15 : 'o',
              16 : 'p', 17 : 'q', 18 : 'r', 19 : 's', 20 : 't',
              21 : 'u', 22 : 'v', 23 : 'w', 24 : 'x', 25 : 'y'}
```



```
def transfer(message, shift):
    decipher = ''
    for letter in message:
        # checks for space
        if(letter != ' '):
            # looks up the dictionary and
            # subtracts the shift to the index
            if (letter.isupper()):
                num = ( dict1[letter] - shift + 26) % 26
                decipher += dict2[num]
            else:
                num = ( dict1_lower[letter] - shift + 26) % 26
                decipher += dict2_lower[num]
            # looks up the second dictionary for the
            # shifted alphabets and adds them
        else:
            # adds space
            decipher += ' '
    return decipher
print(transfer("Bnenw", 9))
```

Như mình đã nói ở trên thì vì chuỗi **Bnenw** là kết quả của quá trình rotate 9 cho nên để dịch ngược được chuỗi này thì mình chỉ cần cho chuỗi trên rotate 9 nữa là sẽ ra được chuỗi ban đầu

Minh chứng:



```
/mnt/e/Stuff or sth like that/System_Programming/ThucHanh/Lab05 11:51:20
python3 test.py
Decode Bnenw: Seven

./nt209-uit-bomb answer2.txt 11:51:23
Welcome to UIT's bomb lab.
You have to deactivate our bomb by solving 5 phases with the correct inputs consecutively, and otherwise the bomb will be blown up!

[*] Phase 1
- Hint: Numbers are always magical!
Good job! You've cleared the first phase!

[*] Phase 2
- Hint: You must answer your secret question!
Seven
Two phases have been solved. Keep going!
```

Câu 3: Về challenge ở phase3

Đầu tiên, để có thể giải quyết được challenge nằm ở hàm phase3, ta cần xem qua mã giả của challenge này.

```

v5 = 0;
v5 = __isoc99_sscanf(a1, "%d %c %d", &v4, &v2, &v3);
if ( v5 <= 2 )
    explode_bomb();
switch ( v4 )
{
    case 0:
        v6 = 'y';
        if ( v3 != 499 )
            explode_bomb();
        return result;
    case 1:
        v6 = 'w';
        if ( v3 != 586 )
            explode_bomb();
        return result;
    case 2:
        v6 = 'q';
        if ( v3 != 787 )
            explode_bomb();
        return result;
    case 3:
        v6 = 'l';
        if ( v3 != 774 )
            explode_bomb();
        return result;
    case 4:
        v6 = 'v';
        if ( v3 != 522 )
            explode_bomb();
        return result;
    case 5:
        v6 = 'z';
        if ( v3 != 529 )
            explode_bomb();
        return result;
    case 6:
        v6 = 'l';
        if ( v3 != 229 )
            explode_bomb();
        return result;
    case 7:
        v6 = 'q';
        if ( v3 != 513 )
            explode_bomb();
        return result;
    default:
        v6 = 'q';
        explode_bomb();
}
result = v2;
if ( v6 != v2 )
    explode_bomb();
return result;
}

```

Cụ thể hơn, thì đầu tiên chương trình này sẽ yêu cầu người dùng nhập vào gồm 2 số và 1 ký tự (dựa vào việc format của scanf lần lượt là %d %c %d)

Đồng thời nếu như không đáp ứng đủ số lượng input đầu vào thì chương trình sẽ gọi đến hàm `explode_bomb` ám chỉ rằng chúng ta đã thất bại

Sau đó hàm sẽ dùng switch case (tương tự như if else) để kiểm tra input của người dùng có thỏa một trong các case có sẵn của switch case ấy. Nếu không thỏa bất kỳ case nào thì chương trình sẽ gọi đến hàm `explode_bomb` ám chỉ rằng chúng ta đã thất bại

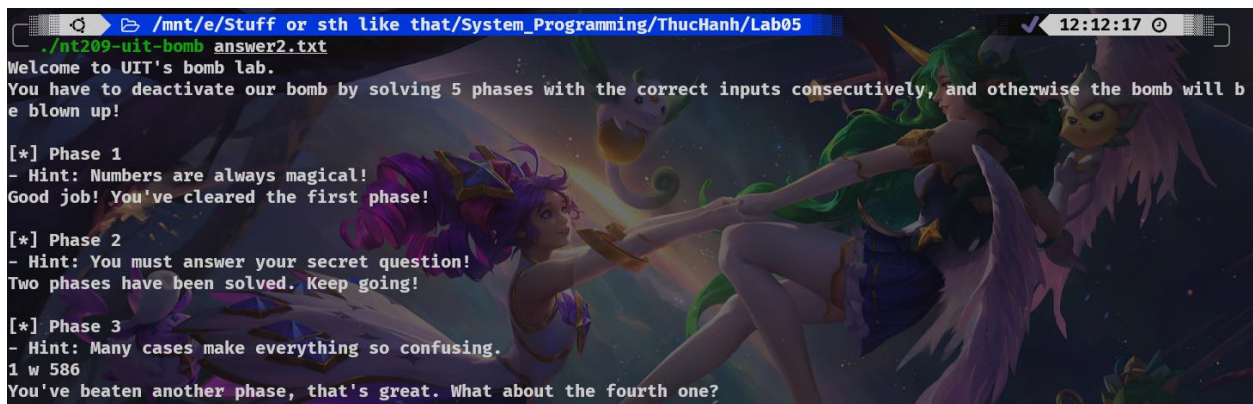
Vậy để có thể làm được challenge này thì ta cần cung cấp các input sao cho thỏa một trong số các case trong đoạn code switch case kia.

Tất cả các input đầu vào có thể thỏa điều kiện của challenge này là:

- 0 y 499
- 1 w 586
- 2 q 787
- 3 l 774
- 4 v 522
- 5 z 529
- 6 l 229
- 7 q 513

Mình chứng:

Giả sử mình chọn input đầu vào là 1 w 586:



```

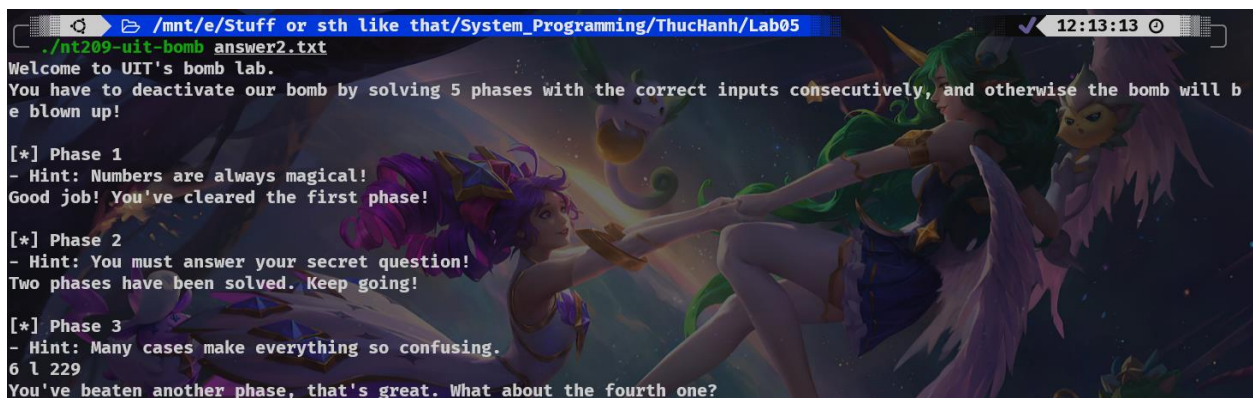
/mnt/e/Stuff or sth like that/System_Programming/ThucHanh/Lab05
./nt209-uit-bomb answer2.txt
Welcome to UIT's bomb lab.
You have to deactivate our bomb by solving 5 phases with the correct inputs consecutively, and otherwise the bomb will be blown up!

[*] Phase 1
- Hint: Numbers are always magical!
Good job! You've cleared the first phase!

[*] Phase 2
- Hint: You must answer your secret question!
Two phases have been solved. Keep going!

[*] Phase 3
- Hint: Many cases make everything so confusing.
1 w 586
You've beaten another phase, that's great. What about the fourth one?
  
```

Giả sử mình chọn input là 6 l 229:



```

/mnt/e/Stuff or sth like that/System_Programming/ThucHanh/Lab05
./nt209-uit-bomb answer2.txt
Welcome to UIT's bomb lab.
You have to deactivate our bomb by solving 5 phases with the correct inputs consecutively, and otherwise the bomb will be blown up!

[*] Phase 1
- Hint: Numbers are always magical!
Good job! You've cleared the first phase!

[*] Phase 2
- Hint: You must answer your secret question!
Two phases have been solved. Keep going!

[*] Phase 3
- Hint: Many cases make everything so confusing.
6 l 229
You've beaten another phase, that's great. What about the fourth one?
  
```

Câu 4: Về challenge của phase4:

Đầu tiên, để giải quyết được challenge này thì ta sẽ nhìn qua xem mã giả để có cái nhìn về chương trình hoạt động như thế nào:

```

1 int __cdecl phase4(int a1)
2 {
3     int result; // eax
4     int input2; // [esp+Ch] [ebp-1Ch] BYREF
5     unsigned int input1; // [esp+10h] [ebp-18h] BYREF
6     int v4; // [esp+14h] [ebp-14h]
7     int v5; // [esp+18h] [ebp-10h]
8     int v6; // [esp+1Ch] [ebp-Ch]
9
10    v6 = __isoc99_sscanf(a1, "%d %d", &input1, &input2);
11    if ( v6 != 2 || input1 >= 15 )
12        explode_bomb();
13    v5 = 43;
14    v4 = func4(input1, 0, 14);
15    if ( v4 != v5 || (result = input2, input2 != v5) )
16        explode_bomb();
17    return result;
18 }

```

Đầu tiên thì chương trình sẽ tiến hành yêu cầu người dùng nhập vào 2 số (lý do là hàm scanf sử dụng format nhập vào là %d và %d)

Đồng thời chương trình cũng sẽ kiểm tra liệu chúng ta liệu đã nhập đủ 2 số theo yêu cầu của đề bài chưa, ngoài ra chương trình còn kiểm tra liệu xem số đầu tiên mà người dùng nhập vào đã thỏa phải nhỏ hơn 15. Nếu không thỏa 1 trong 2 điều kiện trên thì chương trình sẽ gọi đến hàm `explode_bomb` ám chỉ chúng ta đã không thỏa điều kiện đề bài.

```

if ( v6 != 2 || input1 >= 15 )
    explode_bomb();

```

Sau đó chương trình sẽ tiến hành sử dụng số đầu tiên mà ta nhập vào làm tham số đầu tiên trong một hàm tên là **func4** gồm có 3 tham số (trong khi đó 2 tham số còn lại được mặc định là 0 và 14)

Bây giờ ta sẽ tiến hành xem mã giả của hàm **func4** để có thể hiểu được nguyên lý hoạt động của hàm này:

```

int __cdecl func4(int a1, int a2, int a3)
{
    int v4; // [esp+Ch] [ebp-Ch]

    v4 = (a3 - a2) / 2 + a2;
    if ( v4 > a1 )
        return func4(a1, a2, v4 - 1) + v4;
    if ( v4 >= a1 )
        return (a3 - a2) / 2 + a2;
    return func4(a1, v4 + 1, a3) + v4;
}

```

Ở đây ta thấy chương trình dùng đến hàm đệ quy để có thể tính toán ra kết quả cuối cùng.

Trong đó biến `v4` sẽ là giá trị của biểu thức $(a3 - a2) / 2 + a2$ (với lần lượt `a1`, `a2`, `a3` là 3 tham số đầu vào của hàm **func4**). Đồng thời hàm sẽ so sánh với biến `v4` với biến `a1`:

- Nếu như $v4 > a1$ thì hàm sẽ đệ quy bằng cách gọi lại hàm **func4** với 3 tham số `a1`, `a2` và `v4 - 1` và dùng kết quả đó cộng với `v4` hiện tại
- Còn nếu $v4 < a1$ thì hàm sẽ đệ quy bằng cách gọi lại hàm **func4** với 3 tham số `a1`, `v4 + 1`, `a3` và dùng kết quả đó cộng với `v4` hiện tại
- Nếu không thỏa hai điều kiện trên thì hàm sẽ trả về biến `v4` tức là trả về kết quả cuối cùng

Cuối cùng, hàm sẽ tiến hành đối chiếu 2 điều kiện với 2 số mà ta vừa nhập vào, nếu như kết quả của hàm đệ quy mà trong đó có tham số đầu vào là số thứ nhất mà người dùng nhập vào không bằng biến `v5` hoặc là số thứ 2 không bằng biến `v5` (cụ thể biến `v5` ở đây được gán giá trị mặc định là 43) thì chương trình sẽ gọi đến hàm `explode_bomb` ám chỉ chúng ta đã không thỏa điều kiện đề bài.

```

if ( v4 != v5 || (result = input2, input2 != v5) )
    explode_bomb();

```

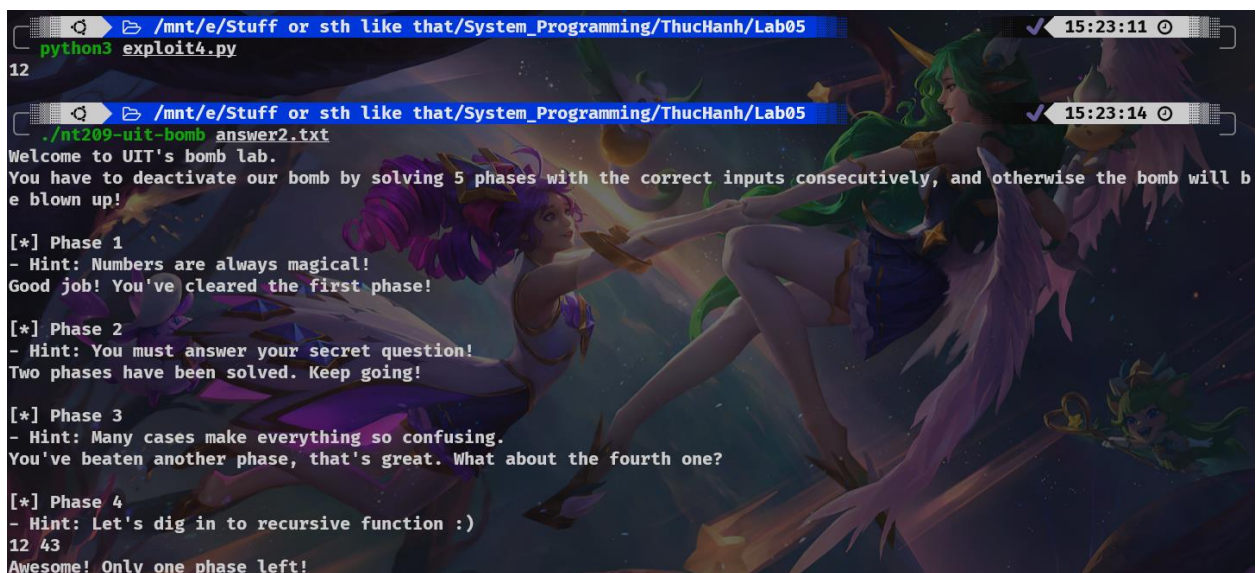
Sau khi hiểu được quy luật của challenge này thì đầu tiên mình đã có thể tìm ra đáp án cho số thứ hai mà người dùng cần phải nhập vào, đó là số 43. Còn về số thứ nhất thì để giải ra thì mình đã mô phỏng lại hàm đệ quy thông qua code python để tìm được số thỏa kết quả đệ quy bằng số 43:


```
def func4(a, b, c):
    result = 0
    result = ((c - b) / 2) + b
    if (result > a):
        return func4(a,b,result-1) + result
    elif (result < a):
        return func4(a,result+1,c) + result
    else:
        return result

for i in range(15):
    if (func4(i,0,14)==43):
        print(i)
        break
```

Ở đây thì mình chỉ chạy vòng lặp 15 số đầu vì như ở trên thì chương trình đã yêu cầu người dùng nhập số thứ nhất không vượt quá giá trị 15

Minh chứng:



```
/mnt/e/Stuff or sth like that/System_Programming/ThucHanh/Lab05 15:23:11
python3 exploit4.py
12
/mnt/e/Stuff or sth like that/System_Programming/ThucHanh/Lab05 15:23:14
./nt209-uit-bomb answer2.txt
Welcome to UIT's bomb lab.
You have to deactivate our bomb by solving 5 phases with the correct inputs consecutively, and otherwise the bomb will be blown up!

[*] Phase 1
- Hint: Numbers are always magical!
Good job! You've cleared the first phase!

[*] Phase 2
- Hint: You must answer your secret question!
Two phases have been solved. Keep going!

[*] Phase 3
- Hint: Many cases make everything so confusing.
You've beaten another phase, that's great. What about the fourth one?

[*] Phase 4
- Hint: Let's dig in to recursive function :)
12 43
Awesome! Only one phase left!
```

Câu 5: Về challenge cuối cùng trong phase5

Đầu tiên thì để có thể giải được challenge trong phase5 này thì ta cần đọc qua mã giả của đoạn chương trình này để nắm bắt được thông tin:


```

1 | B00L4 __cdecl phase5(char *s)
2 | {
3 |     B00L4 result; // eax
4 |     char result2[7]; // [esp+1h] [ebp-17h] BYR1
5 |     size_t v3; // [esp+8h] [ebp-10h]
5 |     int i; // [esp+Ch] [ebp-Ch]
7 |
3 |     v3 = strlen(s);
3 |     if ( v3 != 6 )
3 |         explode_bomb();
1 |     for ( i = 0; i <= 5; ++i )
2 |         result2[i] = array_3855[s[i] & 0xF];
3 |     result2[6] = 0;
4 |     result = is_equal(result2, "flames");
5 |     if ( !result )
5 |         explode_bomb();
7 |     return result;
3 | }

```

Ở đây thì đầu tiên chương trình sẽ yêu cầu người dùng nhập vào một chuỗi ký tự và phải thỏa sao cho chuỗi ký tự ấy có độ dài đúng bằng 6 thông qua hàm strlen. Nếu không thỏa điều kiện trên thì chương trình sẽ gọi đến hàm explode_bomb ám chỉ chúng ta đã không thỏa điều kiện đề bài.

Sau đó chương trình sẽ thông qua một vòng lặp tổng cộng là 6 lần để duyệt qua các phần tử trong biến result. Trong đó các phần tử của biến result sẽ là kết quả được truy xuất trong một mảng có tên là array_3855, mà index của mảng này lại phụ thuộc vào ký tự theo chuẩn mã ASCII của chuỗi ký tự người dùng nhập vào và thực hiện phép AND với 0xf (tức số 15). Bây giờ ta sẽ xem qua giá trị có trong mảng array_3855 này:

.data:0804B1E8	array_3855	db	6Dh
.data:0804B1E9		db	61h ; a
.data:0804B1EA		db	64h ; d
.data:0804B1EB		db	75h ; u
.data:0804B1EC		db	69h ; i
.data:0804B1ED		db	65h ; e
.data:0804B1EE		db	72h ; r
.data:0804B1EF		db	73h ; s
.data:0804B1F0		db	6Eh ; n
.data:0804B1F1		db	66h ; f
.data:0804B1F2		db	6Fh ; o
.data:0804B1F3		db	74h ; t
.data:0804B1F4		db	76h ; v
.data:0804B1F5		db	62h ; b
.data:0804B1F6		db	79h ; y
.data:0804B1F7		db	6Ch ; l

Đường như mảng này là một mảng gồm 16 ký tự và có giá trị là “maduiersnfotvbyl”

Sau đó trong vòng lặp ấy chương trình sẽ lấy từng ký tự có trong mảng array_3855 và gắn vào trong chuỗi ký tự mới tên là result (và đồng thời gắn tại vị trí index cuối cùng tại biến result là ‘\0’)

Sau đó hàm sẽ tiến hành đối chiếu kết quả của chuỗi result vừa tính ra được với chuỗi ký tự mặc định là “flames”. Nếu không thỏa thì chương trình sẽ gọi đến hàm explode_bomb ám chỉ chúng ta đã không thỏa điều kiện đề bài.

Hiểu được luồng thực thi của challenge cuối này thì mình đã tiến hành mô phỏng lại đoạn code của chương trình này để có thể dịch ngược được đầu vào sao cho thỏa yêu cầu của đề bài.

```
given = "maduiersnfotvbyl"

final_string = "flames"
search = ""

for i in range(len(final_string)):
    tempo = 33
    while tempo < 126:
        if (given[tempo & 0xf] == final_string[i]):
            print(tempo)
            search += chr(tempo)
            break
        else:
            tempo += 1
            continue

print(search)
```

Ở đây đoạn chương trình này có thể cho ra nhiều đầu vào mà vẫn đáp ứng được yêu cầu đề bài. Đây là 2 trong số nhiều chuỗi có thể đáp ứng điều đó:

```
/mnt/e/Stuff or sth like that/System_Programming/ThucHanh/Lab05
python3 exploit5.py
9?1057

/mnt/e/Stuff or sth like that/System_Programming/ThucHanh/Lab05
python3 exploit5.py
IOAPEG
```

Minh chứng:

Với chuỗi ký tự là 9?1057

```

[?] /mnt/e/Stuff or sth like that/System_Programming/ThucHanh/Lab05
./nt209-uit-bomb answer2.txt
Welcome to UIT's bomb lab.
You have to deactivate our bomb by solving 5 phases with the correct inputs consecutively, and otherwise the bomb will be blown up!

[*] Phase 1
- Hint: Numbers are always magical!
Good job! You've cleared the first phase!

[*] Phase 2
- Hint: You must answer your secret question!
Two phases have been solved. Keep going!

[*] Phase 3
- Hint: Many cases make everything so confusing.
You've beaten another phase, that's great. What about the fourth one?

[*] Phase 4
- Hint: Let's dig in to recursive function :)
Awesome! Only one phase left!

[*] Phase 5
- Hint: No hint is also a hint :)
9?1057
Amazing bomb solvers, the bomb has been deactivated. Enjoy your day :))
```

Với chuỗi ký tự là IOAPEG

```

[?] /mnt/e/Stuff or sth like that/System_Programming/ThucHanh/Lab05
./nt209-uit-bomb answer2.txt
Welcome to UIT's bomb lab.
You have to deactivate our bomb by solving 5 phases with the correct inputs consecutively, and otherwise the bomb will be blown up!

[*] Phase 1
- Hint: Numbers are always magical!
Good job! You've cleared the first phase!

[*] Phase 2
- Hint: You must answer your secret question!
Two phases have been solved. Keep going!

[*] Phase 3
- Hint: Many cases make everything so confusing.
You've beaten another phase, that's great. What about the fourth one?

[*] Phase 4
- Hint: Let's dig in to recursive function :)
Awesome! Only one phase left!

[*] Phase 5
- Hint: No hint is also a hint :)
IOAPEG
Amazing bomb solvers, the bomb has been deactivated. Enjoy your day :))
```