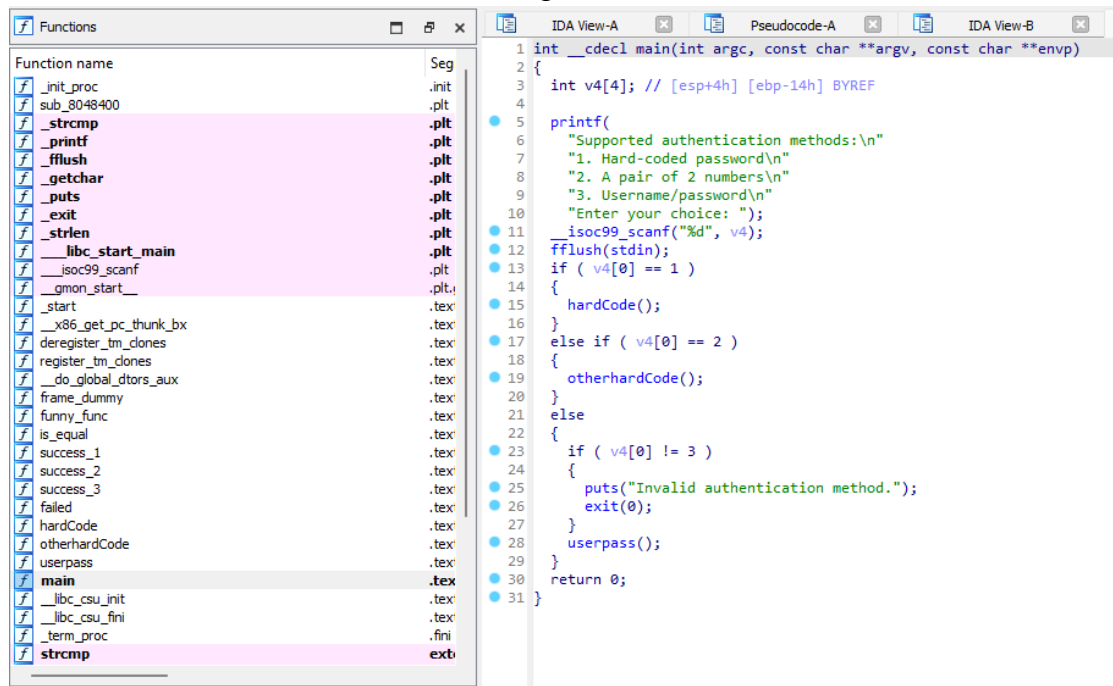


Lab 4 - Kỹ thuật dịch ngược

Hồ Trung Kiên - 22520704

Yêu cầu 1: xác định passphrase

Phân tích mã giả của hàm main



The screenshot shows the IDA Pro interface. On the left, the 'Functions' window lists various functions, with 'main' highlighted. On the right, the 'Pseudocode-A' window displays the disassembly of the 'main' function. The code is as follows:

```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     int v4[4]; // [esp+4h] [ebp-14h] BYREF
4
5     printf(
6         "Supported authentication methods:\n"
7         "1. Hard-coded password\n"
8         "2. A pair of 2 numbers\n"
9         "3. Username/password\n"
10        "Enter your choice: ");
11    __isoc99_scanf("%d", v4);
12    fflush(stdin);
13    if ( v4[0] == 1 )
14    {
15        hardCode();
16    }
17    else if ( v4[0] == 2 )
18    {
19        otherhardCode();
20    }
21    else
22    {
23        if ( v4[0] != 3 )
24        {
25            puts("Invalid authentication method.");
26            exit(0);
27        }
28        userpass();
29    }
30    return 0;
31 }
```

Hàm main tạo một mảng int gồm 4 phần tử mang tên v4, cho người dùng nhập vào và kiểm tra nội dung của ký tự đầu tiên.

Nếu $v4[0] == 1$: tiếp tục vào hàm `hardCode()`

Nếu $v4[0] == 2$: tiếp tục vào hàm `otherhardCode()`

Nếu $v4[0] != 3$: In ra "Invalid authentication method" và thoát chương trình (`exit`)

Các trường hợp còn lại ($v4[0] == 3$): tiếp tục vào hàm `userpass()`

Vậy trong yêu cầu 1 ta sẽ xem xét hàm `hardCode()`

```

1 int hardCode()
2 {
3     char s1[1008]; // [esp+8h] [ebp-3F0h] BYREF
4
5     getchar();
6     puts("Enter the hard-coded password (option 1):");
7     __isoc99_scanf("%s", s1);
8     printf("Your input hard-coded password: %s\n", s1);
9     if ( !strcmp(s1, "Speak one way and act another") )
10         return success_1();
11     else
12         return failed();
13 }

```

Biến s1 sẽ là một mảng char gồm 1008 phần tử, nhiệm vụ của người dùng là nhập vào s1, chương trình sẽ so sánh s1 và chuỗi "Speak one way and act another".

Nếu !strcmp(s1,"Speak one way and act another"): trả về kết quả hàm success_1()

Các trường hợp còn lại: trả về kết quả hàm failed()

Vậy ta sẽ kiểm tra nội dung trả về của 2 hàm success_1() và failed():

```

int success_1()
{
    return puts("Congrats! You found the hard-coded secret, good job :).");
}

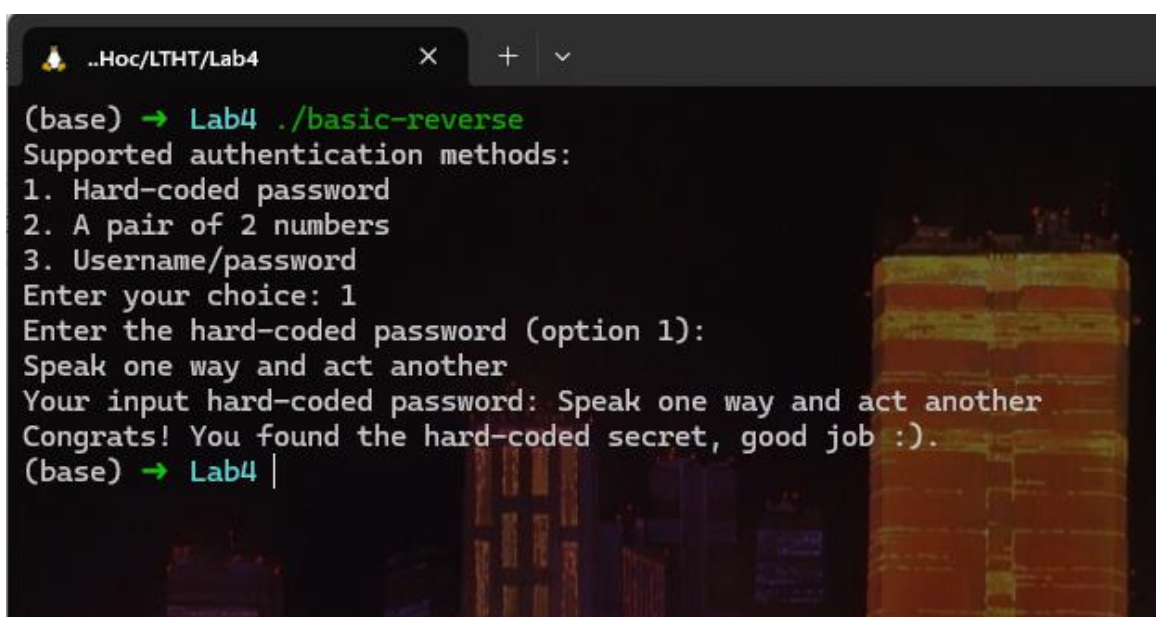
```

```

1 int failed()
2 {
3     return puts("Nice try but that is not the answer.");
4 }

```

Vậy mục tiêu của chúng ta sẽ là trả về kết quả hàm success_1() và làm cho !strcmp(s1,"Speak one way and act another") có kết quả là **true**. Để làm điều này ta sẽ input vào s1 chuỗi sau: Speak one way and act another



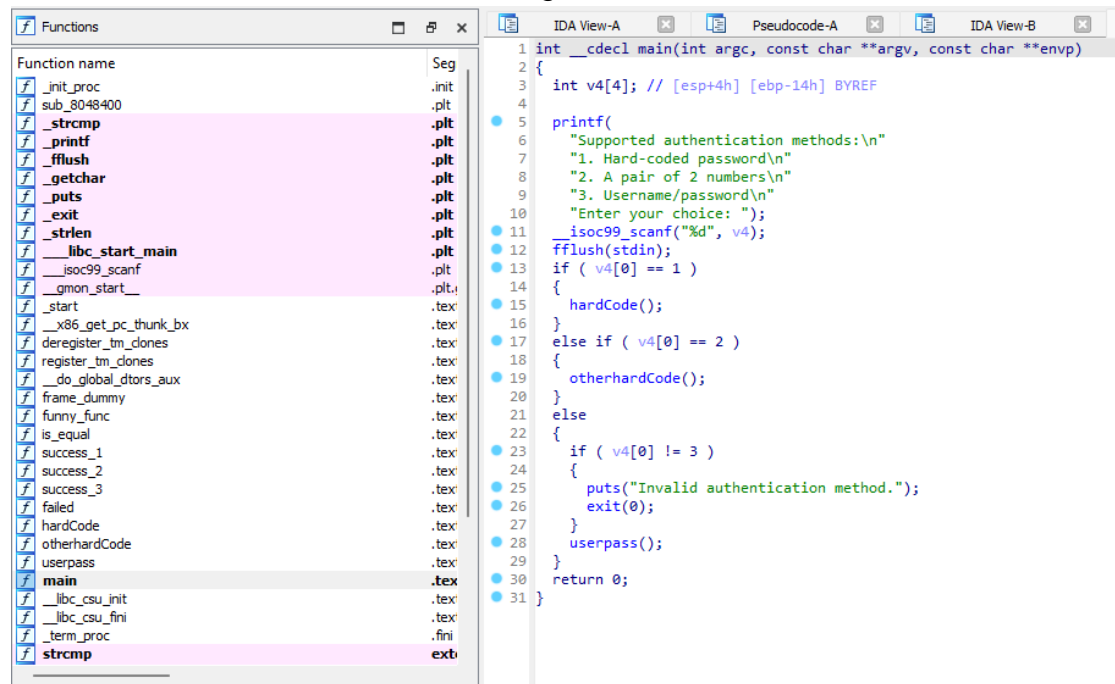
```

..Hoc/LTHT/Lab4
(base) → Lab4 ./basic-reverse
Supported authentication methods:
1. Hard-coded password
2. A pair of 2 numbers
3. Username/password
Enter your choice: 1
Enter the hard-coded password (option 1):
Speak one way and act another
Your input hard-coded password: Speak one way and act another
Congrats! You found the hard-coded secret, good job :).
(base) → Lab4 |

```

Yêu cầu 2: tìm cặp số nguyên

Phân tích mã giả của hàm main



```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     int v4[4]; // [esp+4h] [ebp-14h] BYREF
4
5     printf(
6         "Supported authentication methods:\n"
7         "1. Hard-coded password\n"
8         "2. A pair of 2 numbers\n"
9         "3. Username/password\n"
10        "Enter your choice: ");
11    __isoc99_scanf("%d", v4);
12    fflush(stdin);
13    if ( v4[0] == 1 )
14    {
15        hardCode();
16    }
17    else if ( v4[0] == 2 )
18    {
19        otherhardCode();
20    }
21    else
22    {
23        if ( v4[0] != 3 )
24        {
25            puts("Invalid authentication method.");
26            exit(0);
27        }
28        userpass();
29    }
30    return 0;
31 }
```

Nếu ta chọn option 2, hàm main() sẽ thực thi hàm otherhardCode().

```
1 int otherhardCode()
2 {
3     int v0; // eax
4     int v2; // [esp+4h] [ebp-14h] BYREF
5     int v3[4]; // [esp+8h] [ebp-10h] BYREF
6
7     getchar();
8     puts("Enter your 2 numbers (separated by space) (option 2):");
9     __isoc99_scanf("%d %d", v3, &v2);
10    printf("Your input: %d %d\n", v3[0], v2);
11    v3[1] = 3;
12    if ( v3[0] == 3 && (v0 = funny_func(funny_seq[3], 3), v0 == v2) )
13        return success_2();
14    else
15        return failed();
16 }
```

(dòng 2,3,4) Chương trình ban đầu khởi tạo 3 biến int lần lượt là v0,v2,v3[4].

(dòng 9) Sau đó, chương trình cho ta nhập 2 số v3[0] và v2

(dòng 11) Gán cho v3[1] = 3

(dòng 12, 13) Nếu v3[0] == 3 && (v0 = funny_fuc(funny_seq[3], 3), v0 == v2): trả về kết quả hàm success_2()

(dòng 14, 15) Các trường hợp còn lại: trả về kết quả hàm failed()

Kiểm tra nội dung hàm `success_2()` và `failed()`:

```
int success_2()
{
    return puts("Congrats! You found a secret pair of numbers :).");
}
```

```
int failed()
{
    return puts("Nice try but that is not the answer.");
}
```

Vậy mục tiêu của chúng ta sẽ là làm cho `v3[0] == 3` và `v0 == v2` với `v2` là giá trị ta sẽ nhập vào, vậy ta sẽ kiểm tra giá trị của `v0 = funny_fuc(funny_seq[3], 3)`.

Hàm `funny_func()`:

```
int __cdecl funny_func(int a1, int a2)
{
    return (a1 + a2 - 1) * (a1 + a2);
}
```

Hàm này sẽ trả về kết quả của $(a1 + a2 - 1) * (a1 + a2)$ với `a1` và `a2` là 2 đối số truyền vào.

Các ô nhớ tại `funny_seq`:

```

.rodata:08048A60                                     public funny_seq
.rodata:08048A60 ; int funny_seq[10]
.rodata:08048A60 funny_seq                          dd 1
.rodata:08048A64                                     db    3
.rodata:08048A65                                     db    0
.rodata:08048A66                                     db    0
.rodata:08048A67                                     db    0
.rodata:08048A68                                     db    5
.rodata:08048A69                                     db    0
.rodata:08048A6A                                     db    0
.rodata:08048A6B                                     db    0
.rodata:08048A6C                                     db    7
.rodata:08048A6D                                     db    0
.rodata:08048A6E                                     db    0
.rodata:08048A6F                                     db    0
.rodata:08048A70                                     db    9
.rodata:08048A71                                     db    0
.rodata:08048A72                                     db    0
.rodata:08048A73                                     db    0
.rodata:08048A74                                     db    2
.rodata:08048A75                                     db    0
.rodata:08048A76                                     db    0
.rodata:08048A77                                     db    0
.rodata:08048A78                                     db    4
.rodata:08048A79                                     db    0
.rodata:08048A7A                                     db    0
.rodata:08048A7B                                     db    0
.rodata:08048A7C                                     db    6
.rodata:08048A7D                                     db    0
.rodata:08048A7E                                     db    0
.rodata:08048A7F                                     db    0
.rodata:08048A80                                     db    8
.rodata:08048A81                                     db    0
.rodata:08048A82                                     db    0
.rodata:08048A83                                     db    0
.rodata:08048A84                                     db    0
.rodata:08048A85                                     db    0
.rodata:08048A86                                     db    0
.rodata:08048A87                                     db    0

```

Giá trị của `funny_seq[3]` sẽ là 7 vì giá trị được lưu theo dạng little-endians và ta có:

- **DB:** Định nghĩa Byte. 8 bit.
- **DW:** Định nghĩa Word. Thường là 2 byte trên hệ thống x86 32 bit tiêu chuẩn.
- **DD:** Định nghĩa double word. Thường là 4 byte trên hệ thống x86 32 bit tiêu chuẩn.

Vậy ta có thể biết được các giá trị trong mảng `funny_seq` lần lượt là:

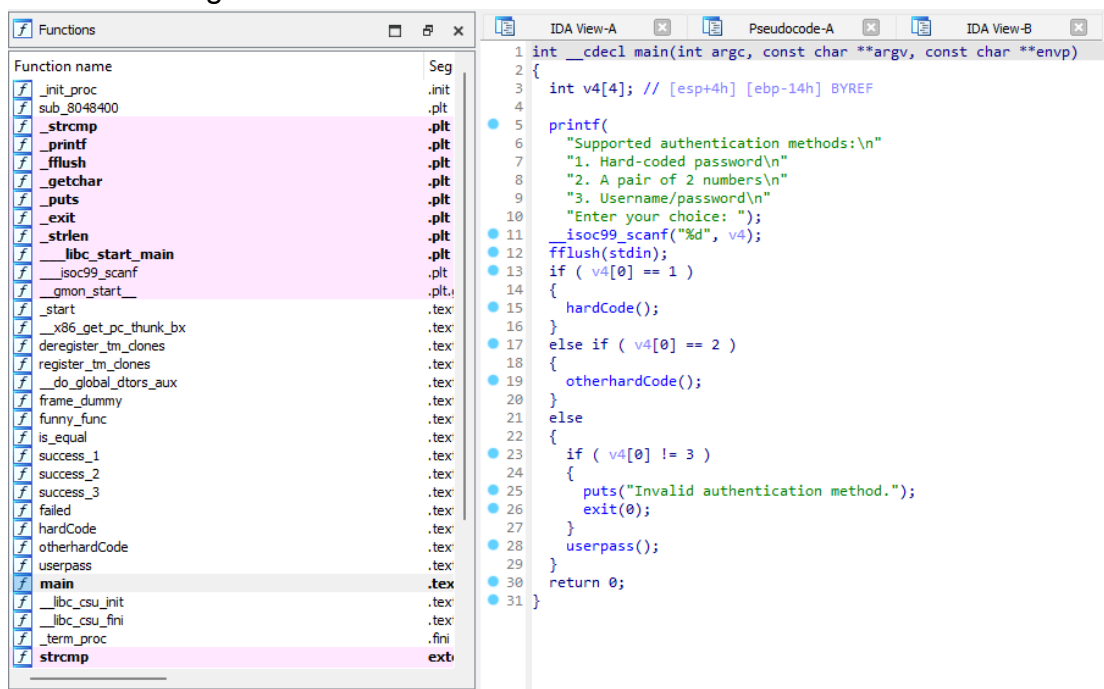
1,3,5,7,9,2,4,6,8,0.

Vậy ta biết `funny_seq[3] = 7` và `funny_func(7,3) = 90` vậy `v0 = 90`. Vậy ta nhập `v3[0] = 3` và `v2 = 90`.

```
..Hoc/LTHT/Lab4
(base) → Lab4 ./basic-reverse
Supported authentication methods:
1. Hard-coded password
2. A pair of 2 numbers
3. Username/password
Enter your choice: 2
Enter your 2 numbers (separated by space) (option 2):
3 90
Your input: 3 90
Congrats! You found a secret pair of numbers :).
(base) → Lab4 |
```

Yêu cầu 3: tìm username/password phù hợp

Phân tích mã giả của hàm main



The screenshot shows the IDA Pro interface with the main function selected. The left pane displays the function list, and the right pane shows the pseudocode for the main function.

Function name	Seg
__init_proc	.init
sub_8048400	.plt
__strcmp	.plt
__printf	.plt
__fflush	.plt
__getchar	.plt
__puts	.plt
__exit	.plt
__strlen	.plt
__libc_start_main	.plt
__isoc99_scanf	.plt
__gmon_start__	.plt
__start	.text
__x86_get_pc_thunk_bx	.text
deregister_tm_clones	.text
register_tm_clones	.text
__do_global_ctors_aux	.text
frame_dummy	.text
funny_func	.text
is_equal	.text
success_1	.text
success_2	.text
success_3	.text
failed	.text
hardCode	.text
otherhardCode	.text
userpass	.text
main	.text
__libc_csu_init	.text
__libc_csu_fini	.text
__term_proc	.fini
__strcmp	.extb

```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     int v4[4]; // [esp+4h] [ebp-14h] BYREF
4
5     printf(
6         "Supported authentication methods:\n"
7         "1. Hard-coded password\n"
8         "2. A pair of 2 numbers\n"
9         "3. Username/password\n"
10        "Enter your choice: ");
11    __isoc99_scanf("%d", v4);
12    fflush(stdin);
13    if ( v4[0] == 1 )
14    {
15        hardCode();
16    }
17    else if ( v4[0] == 2 )
18    {
19        otherhardCode();
20    }
21    else
22    {
23        if ( v4[0] != 3 )
24        {
25            puts("Invalid authentication method.");
26            exit(0);
27        }
28        userpass();
29    }
30    return 0;
31 }
```

Với lựa chọn 3 ta sẽ phân tích hàm userpass()

```

1 int userpass()
2 {
3     size_t v0; // ebx
4     size_t v2; // eax
5     size_t v3; // edx
6     char v4[9]; // [esp+Ah] [ebp-2Eh]
7     char v5[10]; // [esp+13h] [ebp-25h] BYREF
8     char s[10]; // [esp+1Dh] [ebp-18h] BYREF
9     char v7[5]; // [esp+27h] [ebp-11h] BYREF
10    int i; // [esp+2Ch] [ebp-Ch]
11
12    qmemcpy(v7, "30>(r", sizeof(v7));
13    getchar();
14    puts("Enter your username:");
15    __isoc99_scanf("%[^\n]", s);
16    getchar();
17    puts("Enter your password:");
18    __isoc99_scanf("%[^\n]", v5);
19    printf("Your input username: %s and password: %s\n", s, v5);
20    if ( strlen(s) != 9 )
21        return failed();
22    v0 = strlen(s);
23    if ( v0 != strlen(v5) )
24        return failed();
25    for ( i = 0; i <= 8; ++i )
26    {
27        if ( i > 1 )
28        {
29            if ( i > 3 )
30                v4[i] = v7[i - 4];
31            else
32                v4[i] = s[i + 5];
33        }
34        else
35        {
36            v4[i] = s[i + 2];
37        }
38    }
39    for ( i = 0; ; ++i )
40    {
41        v2 = strlen(s);
42        if ( v2 <= i || (s[i] + v4[i]) / 2 != v5[i] )
43            break;
44    }
45    v3 = strlen(s);
46    if ( v3 == i )
47        return success_3();
48    else
49        return failed();
50 }

```

(dòng 12) khởi tạo giá trị cho v7 bằng qmemcpy. v7[5] = "30>(r"

(dòng 14-18) Hàm này sẽ lưu username ta nhập vào (9 ký tự) vào trong s[10], lưu password ta nhập vào (9 ký tự) trong v5[10].

(dòng 20-24) Kiểm tra số ký tự trong username và password, username phải có đúng 9 ký tự, tương tự với password.

(dòng 24-38) Gán các giá trị từ v7 và s vào v4

(dòng 39-44) Kiểm tra điều kiện và dừng khi $(v2 \leq i \parallel (s[i] + v4[i]) / 2 \neq v5[i])$

(dòng 45-49) cho v3 là số ký tự của username (là 9) và so sánh:

Nếu $v3 == i$: trả về kết quả hàm success_3()

Các trường hợp còn lại: trả về kết quả hàm failed()

Vậy mục tiêu là phải không cho việc kiểm tra điều kiện dừng (dòng 39-44) trừ khi $v2 \leq i$ (vì $v2 = \text{strlen}(s) = 9 \Rightarrow i = 9 == v3$). Suy ra điều kiện $(s[i]+v4[i])/2 \neq v5[i]$ phải luôn **sai** hay $(s[i]+v4[i])/2 == v5[i]$ với mọi i từ 0 đến 8.

Để làm được điều trên ta phải biết được các giá trị của $v4$ và s từ đó biết được $v5$ (password) tương ứng.

Đoạn code python sau dựa trên mã giả của chương trình, dùng để tính toán $v5$ (password) với username "2252-0704".

```
s = b"2252-0704"
v7 = b'30>(r'

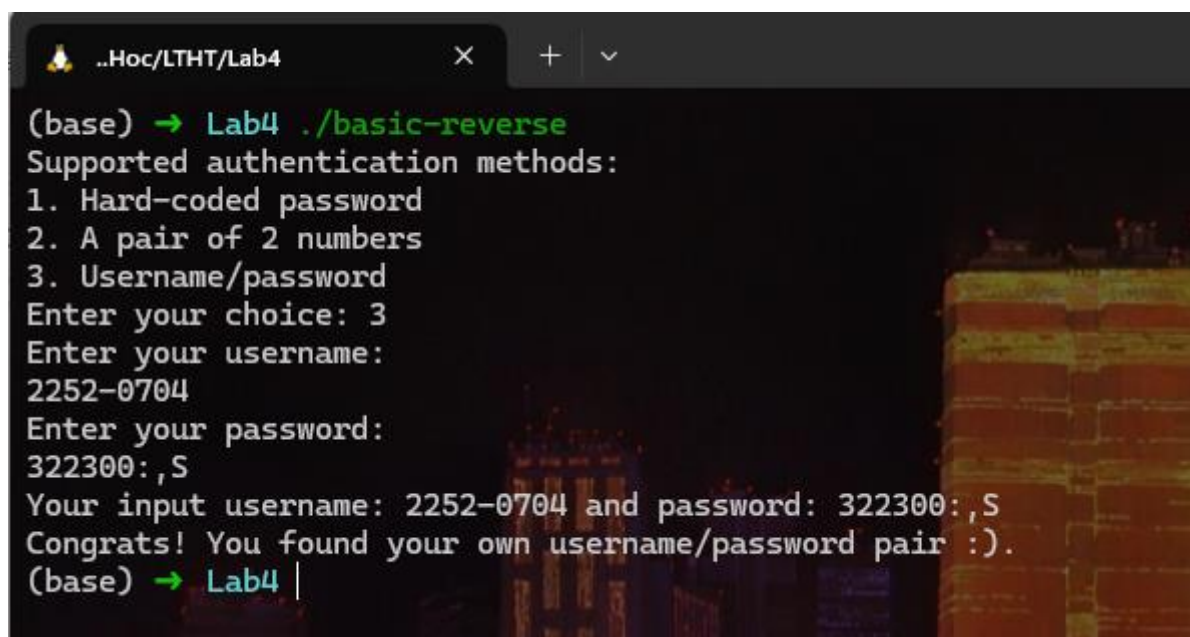
v4 = []
v5 = []

for i in range(9):
    v4.append(v7[i-4] if i > 3 else s[i+2 if i < 2 else i+5])
    v5.append((s[i]+v4[i])/2)

print((b'.'.join(i.to_bytes(1,'big') for i in v5)).decode())
```

Cho ra kết quả: 322300;,S

Vậy ta nhập cặp username/password: 2252-0704 322300;,S



```
..Hoc/LTHT/Lab4
(base) -> Lab4 ./basic-reverse
Supported authentication methods:
1. Hard-coded password
2. A pair of 2 numbers
3. Username/password
Enter your choice: 3
Enter your username:
2252-0704
Enter your password:
322300;,S
Your input username: 2252-0704 and password: 322300;,S
Congrats! You found your own username/password pair :).
(base) -> Lab4 |
```