

# BÁO CÁO BÀI TẬP

Môn học: **Lập trình hệ thống (NT209) Thực hành**

**Lab 3 – Kỹ thuật dịch ngược cơ bản**

GVHD: *Đỗ Thị Hương Lan*

## 1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT209.N21.ANTT

STT	Họ và tên	MSSV	Email
1	Nguyễn Thị Minh Châu	21520645	21520645@gm.uit.edu.vn

## 2. NỘI DUNG THỰC HIỆN:<sup>1</sup>

STT	Công việc	Kết quả tự đánh giá
1	Câu 1	Tốt
2	Câu 2	Tốt
3	Câu 3	Tốt

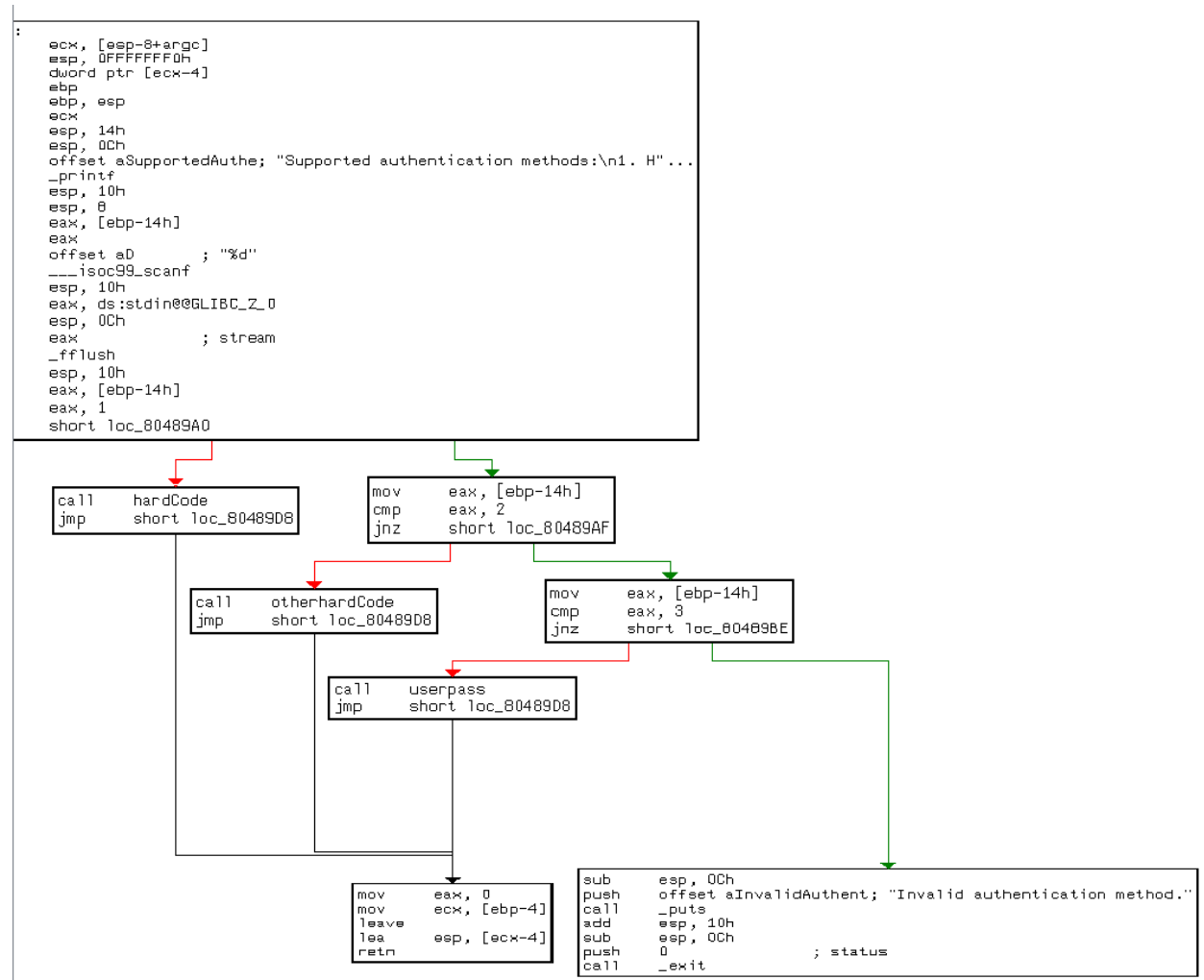
Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.

---

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

# BÁO CÁO CHI TIẾT

## Hàm main của chương trình:



```

View-A Pseudocode-A Hex View-1 Structures Enums Imports
.text:0804895F      push    offset aSupportedAuthe ; "Supported authentication methods:\n1. H"..
.text:08048964      call    _printf
.text:08048969      add     esp, 10h
.text:0804896C      sub     esp, 8
.text:0804896F      lea     eax, [ebp-14h]
.text:08048972      push    eax
.text:08048973      push    offset aD ; "%d"
.text:08048978      call    ___isoc99_scanf
.text:0804897D      add     esp, 10h
.text:08048980      mov     eax, ds:stdin@GLIBC_2_0
.text:08048985      sub     esp, 0Ch
.text:08048988      push    eax ; stream
.text:08048989      call    _fflush
.text:0804898E      add     esp, 10h
.text:08048991      mov     eax, [ebp-14h]
.text:08048994      cmp     eax, 1
.text:08048997      jnz     short loc_80489A0
.text:08048999      call    hardCode
.text:0804899E      jmp     short loc_80489D8
.text:080489A0 ; -----
.text:080489A0      loc_80489A0: ; CODE XREF: main+4C↑j
.text:080489A0      mov     eax, [ebp-14h]
.text:080489A3      cmp     eax, 2
.text:080489A6      jnz     short loc_80489AF
.text:080489A8      call    otherhardCode
.text:080489AD      jmp     short loc_80489D8
.text:080489AF ; -----
.text:080489AF      loc_80489AF: ; CODE XREF: main+5B↑j
.text:080489AF      mov     eax, [ebp-14h]
.text:080489B2      cmp     eax, 3
.text:080489B5      jnz     short loc_80489BE
.text:080489B7      call    userpass
.text:080489BC      jmp     short loc_80489D8
.text:080489BE ; -----

```

Di chuyển đến các hàm hardCode, otherhardCode, userpass tùy theo input nhập vào.

## 2.1

Hàm hardCode:

```

.text:08048690 hardCode      proc near                               ; CODE XREF: main+4E↓p
.text:08048690
.text:08048690 s1          = byte ptr -3F0h
.text:08048690
.text:08048690      push    ebp
.text:08048691      mov     ebp, esp
.text:08048693      sub     esp, 3F8h
.text:08048699      call    _getchar
.text:0804869E      sub     esp, 0Ch
.text:080486A1      push    offset aEnterTheHardCo ; "Enter the hard-coded password (option 1"...
.text:080486A6      call    _puts
.text:080486AB      add     esp, 10h
.text:080486AE      sub     esp, 8
.text:080486B1      lea     eax, [ebp+s1]
.text:080486B7      push    eax
.text:080486B8      push    offset asc_804923E ; "%[\n]"
.text:080486BD      call    ___isoc99_scanf
.text:080486C2      add     esp, 10h
.text:080486C5      sub     esp, 8
.text:080486C8      lea     eax, [ebp+s1]
.text:080486CE      push    eax
.text:080486CF      push    offset format ; "Your input hard-coded password: %s\n"
.text:080486D4      call    _printf
.text:080486D9      add     esp, 10h
.text:080486DC      sub     esp, 8
.text:080486DF      push    offset s2 ; "Don't wish it were easier. Wish you wer"...
.text:080486E4      lea     eax, [ebp+s1]
.text:080486EA      push    eax ; s1
.text:080486EB      call    _strcmp
.text:080486F0      add     esp, 10h
.text:080486F3      test    eax, eax
.text:080486F5      jnz     short loc_80486FE
.text:080486F7      call    success_1
.text:080486FC      jmp     short loc_8048703

```

Biến 's1' được tạo và cấp phát cho một vùng nhớ. Lệnh push 'aEnterTheHardCo' lưu dòng "Enter the hard-coded password (option 1)..." để xuất ra màn hình. Sau đó cho người dùng nhập input bằng `__isoc99_scanf`. Push 'format' và gọi `printf` để xuất ra màn hình: "Your input hard-coded password: " và password nhập vào được lưu ở biến 's1'. Biến 's2' có lưu chuỗi: "Don't wish it were easier. Wish you were better".

```

.rodata:08049268 ; char s2[]
.rodata:08049268 s2      db 'Don',27h,'t wish it were easier. Wish you were better',0
.rodata:08049268                ; DATA XREF: hardCode+4F↑o

```

Sau đó call hàm `_strcmp` để so sánh s1 và s2

```

.plt:08048410 _strcmp      proc near                               ; CODE XREF: is_equal+F↓p
.plt:08048410                                     ; hardCode+5B↓p ...
.plt:08048410      jmp     ds:off_804B00C
.plt:08048410 _strcmp      endp

```

Nếu s1 và s2 không giống nhau nghĩa là password đã nhập sai => nhảy đến `loc_80486FE` để call `failed`. Ngược lại sẽ call `success_1` và in ra màn hình bạn đã nhập đúng

```

.text:080486FE
.text:080486FE loc_80486FE:                                     ; CODE XREF: hardCode+65↑j
.text:080486FE      call    failed
.text:08048703

```

```

.text:080485CC      public success_1
.text:080485CC      success_1      proc near                ; CODE XREF: hardCode+67↓p
.text:080485CC      push     ebp
.text:080485CD      mov     ebp, esp
.text:080485CF      sub     esp, 8
.text:080485D2      sub     esp, 0Ch
.text:080485D5      push     offset s                ; "Congrats! You found the hard-coded secr"...
.text:080485DA      call    _puts
.text:080485DF      add     esp, 10h
.text:080485E2      sub     esp, 0Ch
.text:080485E5      push     offset aHandInThisToYo ; "Hand in this to your instructor as a pr"...
.text:080485EA      call    _puts
.text:080485EF      add     esp, 10h
.text:080485F2      sub     esp, 0Ch
.text:080485F5      push     offset aStayHomeForThe ; "\\Stay home for the safety of yourself "...
.text:080485FA      call    _puts
.text:080485FF      add     esp, 10h
.text:08048602      nop
.text:08048603      leave

```

Dưới đây là hình ảnh sau khi nhập đúng password “Don’t wish it were easier. Wish you were better”

```

(zeri@kali)-[~/Desktop/Lab3]
$ ./basic-reverse
Supported authentication methods:
1. Hard-coded password
2. Another hard-coded password
3. Username/password
Enter your choice: 1
Enter the hard-coded password (option 1):
Don't wish it were easier. Wish you were better
Your input hard-coded password: Don't wish it were easier. Wish you were better
Congrats! You found the hard-coded secret, good job :).
Hand in this to your instructor as a proof:
"Stay home for the safety of yourself and others."

```

## 2.2

Hàm otherhardCode:

```

.text:08048706      push     ebp
.text:08048707      mov      ebp, esp
.text:08048709      sub      esp, 3F8h
.text:0804870F      call     _getchar
.text:08048714      sub      esp, 0Ch
.text:08048717      push     offset aEnterTheHardCo ; "Enter the hard-coded password (option 2)...
.text:0804871C      call     _puts
.text:08048721      add      esp, 10h
.text:08048724      sub      esp, 8
.text:08048727      lea      eax, [ebp+s1]
.text:0804872D      push     eax
.text:0804872E      push     offset asc_804923E ; "%[^\\n]"
.text:08048733      call     ___isoc99_scanf
.text:08048738      add      esp, 10h
.text:0804873B      sub      esp, 8
.text:0804873E      lea      eax, [ebp+s1]
.text:08048744      push     eax
.text:08048745      push     offset format ; "Your input hard-coded password: %s\\n"
.text:0804874A      call     _printf
.text:0804874F      add      esp, 10h
.text:08048752      mov      [ebp+var_C], 0Ah
.text:08048759      mov      eax, [ebp+var_C]
.text:0804875C      mov      eax, WHAT_THAT[eax*4]
.text:08048763      mov      [ebp+s2], eax
.text:08048766      sub      esp, 8
.text:08048769      push     [ebp+s2] ; s2
.text:0804876C      lea      eax, [ebp+s1]
.text:08048772      push     eax ; s1
.text:08048773      call     _strcmp
.text:08048778      add      esp, 10h
.text:0804877B      test     eax, eax
.text:0804877D      jnz      short loc_8048786
.text:0804877F      call     success_2
.text:08048784      jmp      short loc_804878B

```

Push `aEnterTheHardCo` vào stack sau đó sử dụng lệnh `_puts` để in ra màn hình dòng chữ “Enter the hard-coded password (option 2): “. Sau đó cho người dùng nhập password bằng cách gọi hàm `__isoc99_scanf`. Dữ liệu người dùng nhập vào sẽ được lưu trong `[ebp+s1]`. Push format rồi gọi hàm `printf` để in ra dòng chữ “Your input hard-coded password: “ để xác nhận dữ liệu người dùng đã nhập.

```

.text:08048733      call     ___isoc99_scanf
.text:08048738      add      esp, 10h
.text:0804873B      sub      esp, 8
.text:0804873E      lea      eax, [ebp+s1]
.text:08048744      push     eax
.text:08048745      push     offset format ; "Your input hard-coded password: %s\\n"
.text:0804874A      call     _printf

```

Sau đó gán giá trị 10 vào `%eax`. Lấy phần tử thứ 11 của mảng `WHAT_THAT` gán vào `[esp+s2]`. Chuỗi `s2` được gán bằng chuỗi cách chuỗi đầu tiên 10 chuỗi – “As strong as a horse”

```

.text:08048752      mov     [ebp+var_C], 0Ah
.text:08048759      mov     eax, [ebp+var_C]
.text:0804875C      mov     eax, WHAT_THAT[eax*4]
.text:08048763      mov     [ebp+s2], eax
.text:08048766      sub     esp, 8
.text:08048769      push    [ebp+s2]          ; s2
.text:0804876C      lea     eax, [ebp+s1]
.text:08048772      push    eax              ; s1
.text:08048773      call    _strcmp
.text:08048778      add     esp, 10h
.text:0804877B      test    eax, eax
.text:0804877D      jnz     short loc_8048786
.text:0804877F      call    success_2
.text:08048784      jmp     short loc_804878B

.rodata:08048BF0     db     41h ; A
.rodata:08048BF1     db     73h ; s
.rodata:08048BF2     db     20h
.rodata:08048BF3     db     73h ; s
.rodata:08048BF4     db     74h ; t
.rodata:08048BF5     db     72h ; r
.rodata:08048BF6     db     6Fh ; o
.rodata:08048BF7     db     6Eh ; n
.rodata:08048BF8     db     67h ; g
.rodata:08048BF9     db     20h
.rodata:08048BFA     db     61h ; a
.rodata:08048BFB     db     73h ; s
.rodata:08048BFC     db     20h
.rodata:08048BFD     db     61h ; a
.rodata:08048BFE     db     20h
.rodata:08048BFF     db     68h ; h
.rodata:08048C00     db     6Fh ; o
.rodata:08048C01     db     72h ; r
.rodata:08048C02     db     73h ; s
.rodata:08048C03     db     65h ; e

```

Sau đó so sánh s1 và s2. Ta thấy kết quả so sánh trả về thanh ghi %eax, nếu %eax bằng 0 ('s1' và 's2' giống nhau) thì gọi 'success\_2'. Còn nếu %eax khác 0 ('s1' và 's2' khác nhau) (nhánh true) thì gọi nhảy đến loc\_8048786 và gọi 'failed'. Password là "As strong as a horse".

```

(zeri@kali)-[~/Desktop/Lab3]
$ ./basic-reverse
Supported authentication methods:
1. Hard-coded password
2. Another hard-coded password
3. Username/password
Enter your choice: 2
Enter the hard-coded password (option 2):
As strong as a horse
Your input hard-coded password: As strong as a horse
Congrats! You defeated a harder level of finding hard-coded secret :).
Hand in this to your instructor as a proof:
"Stay positive during the COVID-19 pandemic."

```

## 2.3

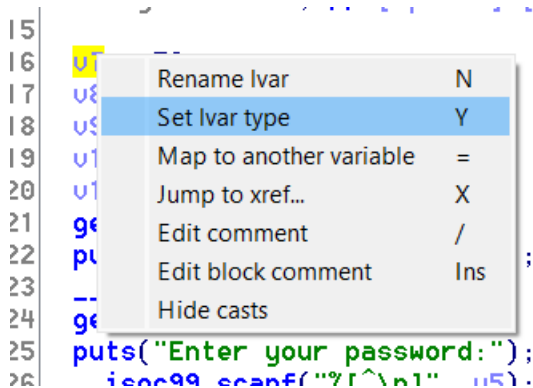
Xét mã giả của hàm userpass:

```

v7 = 58;
v8 = 96;
v9 = 115;
v10 = 85;
v11 = 82;
getchar();
puts("Enter your username:");
__isoc99_scanf("%[^\n]", s);
getchar();
puts("Enter your password:");
__isoc99_scanf("%[^\n]", v5);
printf("Your input username: %s and password: %s\n", s, v5);

```

Đầu tiên ta đổi các biến từ v7 đến v11 thành 1 mảng v7[5]



```

15
16 v7
17 v8
18 v9
19 v10
20 v11
21 getchar()
22 puts("Enter your password:");
23 __isoc99_scanf("%[^\n]", v5);
24
25
26

```

Sau đó đổi từng giá trị num sang kí tự để thu được v7 = “:sUR”

```

v7[0] = ':';
v7[1] = 's';
v7[2] = 'U';
v7[3] = 'R';
v7[4] = '\0';

```

Sau đó chúng ta nhập username (lưu vào s) và password (lưu vào v5). Tiếp theo ta kiểm tra xem username và password nhập vào đã đủ 9 kí tự chưa. Ở đây username là **2152-0645**



```

if ( strlen(s) == 9 && (v0 = strlen(s), v0 == strlen(v5)) )
{
    for ( i = 0; (signed int)i <= 8; ++i )
    {
        if ( (signed int)i > 1 )
        {
            if ( (signed int)i > 3 )
                v4[i] = v7[i - 4];
            else
                v4[i] = s[i + 5];
        }
        else
        {
            v4[i] = s[i + 2];
        }
    }
}

```

Cho chạy vòng lặp for với i từ 0 đến 8.

Ta có s = 2152-0645

- 2 ký tự đầu của v4 là ký tự thứ 2,3 trong chuỗi s
- 2 ký tự tiếp theo của v4 là ký tự thứ 7 8 trong chuỗi s
- 5 ký tự còn lại là chuỗi v7

Vậy ta được v4 = "5245: sUR"

```

for ( i = 0; ; ++i )
{
    v1 = strlen(s);
    if ( v1 <= i || (s[i] + v4[i]) / 2 != v5[i] )
        break;
}
v2 = strlen(s);
if ( v2 == i )
    success_3();
else
    failed();
}
else
{
    failed();
}
return nullsub_8();

```

Tiếp theo ta check lại password nhập vào v5 có đúng không. Biến v1 = strlen(s) được dùng làm giới hạn cho vòng lặp. Nếu thỏa một trong những điều kiện tại thì vòng lặp sẽ dừng. Sau đó xét v2 == i hay không, nếu có thì gọi success\_3(). Nếu không thì gọi failed

Cách để cho v2 == i là true thì phải cho i chạy hết vòng lặp cho đến khi v2<=i (vì i++). Phải thỏa điều kiện (s[i] + v4[i]) / 2 == v5[i] để vòng lặp không bị break.

Sau đó ta dùng chương trình C++ để giải ra password:

```

1  #include <iostream>
2  #include <string>
3
4  using namespace std;
5  int main()
6  {
7      string s = "2152-0645";
8      string v4 = "5245:~sUR";
9      string pass;
10
11     for(int i = 0; i <= 8; i++)
12     {
13         pass = pass + (char)((s[i] + v4[i])/2);
14     }
15
16     cout << pass;
17     return 0;
18 }

```

PROBLEMS   OUTPUT   TERMINAL   DEBUG CONSOLE

[Running] cd "c:\Users\admin\Downloads\Lab02\Lab02\" && g++ main.cpp -o mai  
31433HTDC

[Done] exited with code=0 in 6.545 seconds

Có thể thấy được mật khẩu là **"31433HTDC"**. Ta nhập vào chương trình thì thu được kết quả chính xác

```

(zeri@kali)-[~/Desktop/Lab3]
$ ./basic-reverse
Supported authentication methods:
1. Hard-coded password
2. Another hard-coded password
3. Username/password
Enter your choice: 3
Enter your username:
2152-0645
Enter your password:
31433HTDC
Your input username: 2152-0645 and password: 31433HTDC
Congrats! You found your own username/password pair. Nice work to receive the my message.
Hand in this to your instructor as a proof:
"Vietnam can win over SARS-CoV-2."

```