

BÁO CÁO THỰC HÀNH

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

Lab 3 – Kỹ thuật dịch ngược – Cơ bản

GVHD: *Đỗ Thị Thu Hiền*

1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT209.N22.ATCL.1

STT	Họ và tên	MSSV	Email
1	Võ Sỹ Minh	21521146	21521146@uit.edu.vn
2	Lê Huy Hùng	21520888	21520888@uit.edu.vn
3			

2. NỘI DUNG THỰC HIỆN:¹

STT	Công việc	Kết quả tự đánh giá
1	Yêu cầu 2.1	100%
2	Yêu cầu 2.2	100%
3	Yêu cầu 2.3	100%

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

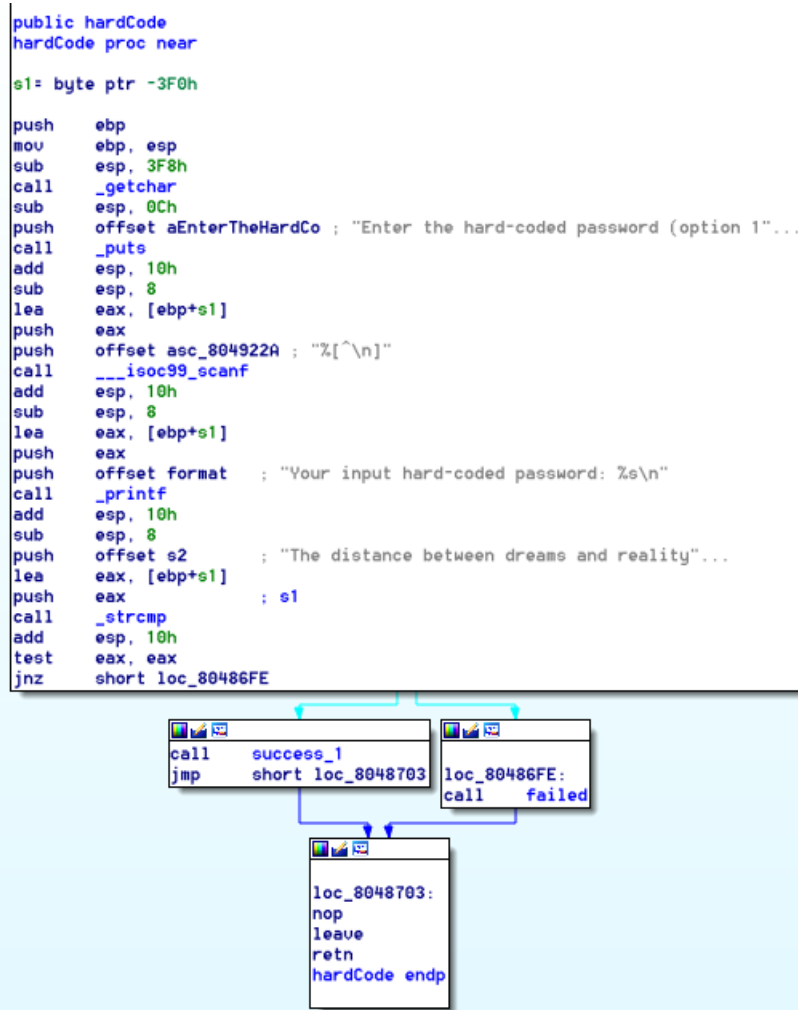
¹ Ghi nội dung công việc, yêu cầu trong bài Thực hành

BÁO CÁO CHI TIẾT

1. Yêu cầu 2.1: The distance between dreams and reality is called action

Yêu cầu 2.1. Phân tích và tìm **passphrase cố định (option 1)** của **basic-reverse** với phương pháp chứng thực 1. Báo cáo phương pháp phân tích, input tìm được và hình ảnh minh chứng chạy file.

Xem xét hàm `hardCode()`.



- Biến `s1` và được cấp phát.
- `___isoc99_scanf` để cho người dùng nhập input. Dùng hàm `_printf` để in ra thông báo “Your input hard-coded password: %s” được lưu trong biến `format` và thay thế chuỗi “%s” với password chúng ta nhập tại `s1`.

```

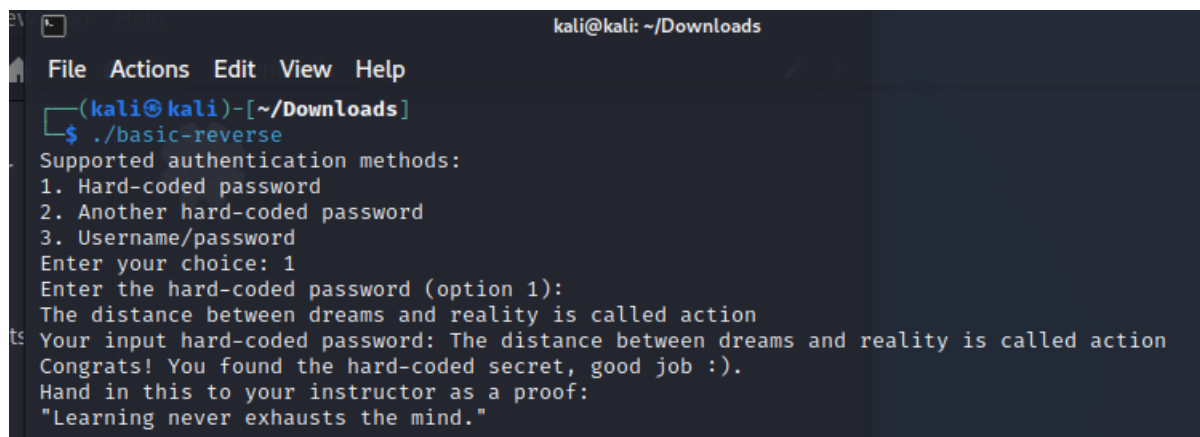
add     esp, 10h
sub     esp, 8
lea     eax, [ebp+s1]
push    eax
push    offset format      ; "Your input hard-coded password: %s\n"
call    _printf
add     esp, 10h
sub     esp, 8
push    offset s2          ; "The distance between dreams and reality"...
lea     eax, [ebp+s1]
push    eax                ; s1
call    _strcmp
add     esp, 10h
test    eax, eax
jnz     short loc_80486FE

```

- 2 dòng push các tham số vào s1 (input), s2 (chuỗi có sẵn). Xét thấy chuỗi được lưu tại biến s2 -> ***The distance between dreams and reality is called action***
- Với 2 tham số trên lệnh _strcmp trả về kết quả ở eax, nếu eax = 0 thì 2 chuỗi bằng nhau thì thực hiện tiếp hàm success_1 và in ra thông điệp báo thành công, ngược lại nếu không bằng (eax != 0) thì nhảy đến short loc_80486FE với thông điệp thất bại.

Vậy hard-code ở option 1 là The distance between dreams and reality is called action

Thử nghiệm kết quả:



```

kali@kali: ~/Downloads
File Actions Edit View Help
(kali@kali)-[~/Downloads]
└─$ ./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):
The distance between dreams and reality is called action
Your input hard-coded password: The distance between dreams and reality is called action
Congrats! You found the hard-coded secret, good job :).
Hand in this to your instructor as a proof:
"Learning never exhausts the mind."

```

2. Yêu cầu 2.2: Good watch prevents misfortune

Yêu cầu 2.2. Phân tích và tìm **passphrase cố định (option 2)** của **basic-reverse** với phương pháp chứng thực 2. Báo cáo phương pháp phân tích, input tìm được và hình ảnh minh chứng chạy file.

Xem xét hàm otherhardCode()

```

public otherhardCode
otherhardCode proc near

s1= byte ptr -3F8h
s2= dword ptr -10h
var_C= dword ptr -0Ch

push    ebp
mov     ebp, esp
sub     esp, 3F8h
call    _getchar
sub     esp, 0Ch
push    offset aEnterTheHard_0 ; "Enter the hard-coded password (option 2"...
call    _puts
add     esp, 10h
sub     esp, 8
lea     eax, [ebp+s1]
push    eax
push    offset asc_804922A ; "[%i^\\n]"
call    ___isoc99_scanf
add     esp, 10h
sub     esp, 8
lea     eax, [ebp+s1]
push    eax
push    offset format ; "Your input hard-coded password: %s\\n"
call    _printf
add     esp, 10h
mov     [ebp+var_C], 2Dh
mov     eax, [ebp+var_C]
mov     eax, WHAT_THAT[eax*4]
mov     [ebp+s2], eax
sub     esp, 8
push    [ebp+s2] ; s2
lea     eax, [ebp+s1]
push    eax ; s1
call    _strcmp
add     esp, 10h
test    eax, eax
jnz     short loc_8048786

call    success_2
jmp     short loc_804878B

loc_8048786:
call    failed

```

Tương tự option 1, các dòng đầu thực hiện các thao tác cơ bản. `_getchar` (loại bỏ khoảng trắng), `_puts` (in ra thông điệp), `__isoc99_scanf` (cho người dùng nhập hard-code), `_printf` (in ra hard-code vừa nhập).

Xét các dòng sau:

```

.text:0804874F      add     esp, 10h
.text:08048752      mov     [ebp+var_C], 2Dh
.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
.text:08048786 ; -----
.text:08048786      loc_8048786:              ; CODE XREF: otherhardCode+77fj
.text:08048786      call    failed

```

- Chương trình sẽ lưu giá trị 2D (hexadecimal) vào ô nhớ có địa chỉ `ebp + var_C`
- Sau đó sẽ đưa giá trị này vào thanh ghi `eax`. Tiếp theo, chương trình sẽ tìm giá trị tại địa chỉ **WHAT_THAT** + `eax * 4` và lưu nó vào `s2`.
- Sau đó, chương trình sẽ so sánh chuỗi được nhập vào bởi người dùng (`s1`) với giá trị trong `s2`.
- Chúng ta cần tìm giá trị tại địa chỉ **WHAT_THAT** được sử dụng như một tham số truyền vào cho phép tính toán giá trị của `s2` thông qua việc lấy giá trị tại **WHAT_THAT** + `eax * 4`.

Địa chỉ của **WHAT_THAT** là **0804B060**

```

.data:0804B060      public WHAT_THAT
.data:0804B060      dd offset aYouScratchMyBa ; DATA XREF: otherhardCode+56f8
.data:0804B060      ; "You scratch my back and I'll scratch
.data:0804B064      dd offset aNewOneInOldOne ; "New one in, old one out"
.data:0804B068      dd offset aItTooLateToLoc ; "It' too late to lock the stable whe
.data:0804B06C      dd offset aWithAgeComesWi ; "With age comes wisdom"
.data:0804B070      dd offset aNothingIsMoreP ; "Nothing is more precious than inde
.data:0804B074      dd offset aHandsomeIsAsHa ; "Handsome is as handsome does"
.data:0804B078      dd offset aNeverOfferToTe ; "Never offer to teach fish to swim"
.data:0804B07C      dd offset aToTryToRunBeFo ; "To try to run before the one can w
.data:0804B080      dd offset aNobodyHasEverS ; "Nobody has ever shed tears without
.data:0804B084      dd offset aYouGetWhatYouP ; "You get what you pay for"
.data:0804B088      dd offset aAsStrongAsAHor ; "As strong as a horse"
.data:0804B08C      dd offset aAllRoadsLeadTo ; "All roads lead to Rome"

```

Vậy ta tìm chuỗi cần tìm bằng cách tính $0804B060 + \text{eax} * 4 = 0804B060 + (0x2D * 4) = 0804B060 + B4 = 0804B114$. Và đây là chuỗi tại địa chỉ **0804B114**

```

.data:0804B0F8      dd offset aNothingVenture ; "Nothing venture nothing gains"
.data:0804B0FC      dd offset aOtherTimesOthe ; "Other times other ways"
.data:0804B100      dd offset aWhileThereSLif ; "While there's life, there's hope"
.data:0804B104      dd offset aTheEmptyVessel ; "The empty vessel makes greatest sound"
.data:0804B108      dd offset aHeWhoExcusesHi ; "He who excuses himself, accuses himself"
.data:0804B10C      dd offset aBeautyIsInTheE ; "Beauty is in the eye of the beholder"
.data:0804B110      dd offset aBloodIsThicker ; "Blood is thicker than water"
.data:0804B114      dd offset aGoodWatchPreve ; "Good watch prevents misfortune"
.data:0804B118      dd offset aGreatMindsThin ; "Great minds think alike"
.data:0804B11C      dd offset aHeThatKnowsNot ; "He that knows nothing doubts nothing"
.data:0804B120      dd offset aHisEyesAreBigg ; "His eyes are bigger than his belly"
.data:0804B124      dd offset aItSTheFirstSte ; "It's the first step that counts"
.data:0804B124      _data      ends

```

Vậy hard-code cần tìm là **Good watch prevents misfortune**

Thử nghiệm kết quả:

```
File Actions Edit View Help
(kali@kali)-[~/Downloads]
$ ./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):
Good watch prevents misfortune
Your input hard-coded password: Good watch prevents misfortune
Good job! You defeated a harder level of finding hard-coded secret :).
Hand in this to your instructor as a proof:
"Don't let what you cannot do interfere what you can do."
```

3. Yêu cầu 2.3: username: 1146-0888, password: 2367)UY[I

Yêu cầu 2.3. Phân tích, tìm **username/password** phù hợp của **basic-reverse** với phương pháp chứng thực 3. Báo cáo phương pháp và input tìm được.

Lưu ý bắt buộc: **username** được tạo từ MSSV của các thành viên trong nhóm.

- Nhóm **3 sinh viên**, lấy 3 số cuối nối nhau. Ví dụ 21520013, 21520123 và 21521021 sẽ có username là **013123021**.
- Nhóm **2 sinh viên**, lấy 4 số cuối nối nhau bằng dấu "-". Ví dụ 21520013, 21520123 sẽ có username là 0013-0123.
- Nhóm có **1 sinh viên** có MSSV là 2152xxxx thì username là **2152-xxxx**.

SV1: 21521146

SV2: 21520888

➔ Username: 1146-0888

Sử dụng mã giả (pseudo-code) của hàm userpass()

Đoạn đầu tiên là phân khai báo các biến.

Sau đó, cấp vùng nhớ và gán giá trị `"&z{~["` cho biến v7, và tiếp tục thực hiện các bước lấy dữ liệu đầu vào và hiển thị thông tin với hai trường "username" và "password" bằng cách sử dụng các hàm getchar(), __isoc99_scanf(), puts() và printf().

Sau khi lấy được dữ liệu đầu vào, chương trình thực hiện kiểm tra độ dài của "username" và "password" bằng cách so sánh độ dài của chuỗi đầu vào với giá trị 9. Chương trình sẽ kiểm tra xem len(username) == 9 và len(password) == 9 hay không.

```

size_t v0; // ebx@2
int result; // eax@3
size_t v2; // eax@15
size_t v3; // edx@16
char v4[9]; // [sp+Ah] [bp-2Eh]@6
char v5[10]; // [sp+13h] [bp-25h]@1
char s[10]; // [sp+1Dh] [bp-1Bh]@1
char v7; // [sp+27h] [bp-11h]@1
char v8; // [sp+28h] [bp-10h]@1
char v9; // [sp+29h] [bp-Fh]@1
char v10; // [sp+2Ah] [bp-Eh]@1
char v11; // [sp+2Bh] [bp-Dh]@1
unsigned int i; // [sp+2Ch] [bp-Ch]@4

v7 = 38;
v8 = 122;
v9 = 123;
v10 = 126;
v11 = 91;

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);
if ( strlen(s) == 9 && (v0 = strlen(s), v0 == strlen(v5)) )

```

Sau đây là phần chính của chương trình:

Với s (username) = "1146-0888". Chạy i -> [0; 8] (9 lần):

2 ký tự đầu của **v4** là ký tự tại vị trí 2,3 của **s**

2 ký tự tiếp của **v4** là ký tự tại vị trí 7,8 của **s**

4 ký tự còn lại của **v4** là chuỗi **v7** "&5p~D"

→ **v4** = "4688&z{~["

```

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];
    }
}

```

Tiếp đến là phần hình thành mật khẩu:

```
for ( i = 0; ; ++i )
{
    v2 = strlen(s);
    if ( v2 <= i || (s[i] + v4[i]) / 2 != v5[i] )
        break;
}
v3 = strlen(s);
if ( v3 == i )
    result = success_3();
else
    result = failed();
}
else
{
    result = failed();
}
return result;
,
```

Biến $v2 = \text{strlen}(s) = 8$, dùng để làm điều kiện dừng vòng lặp, nếu thỏa:

`if (v2 <= i || (s[i] + v4[i]) / 2 != v5[i])`

Vậy để chạy vòng lặp để tìm password thì phải luôn để cho điều kiện $(s[i] + v4[i]) / 2 == v5[i]$ thỏa thì vòng lặp sẽ không bị break. (Hay đây là phép tính nhằm tìm password (v5))

Sau đó xét $v3 == i$ hay không, nếu có thì `success_3()` được gọi hoặc sẽ rẽ nhánh qua `failed()` nếu sai các điều kiện đã đi qua.

Đây là code C++ nhằm tìm password cho option này.

```
string hint = "&z{~[";
string user = "1146-0888";
string pass = "";

char v4[9];
```

```
for (int i = 0; i < 9; i++)
{
    if (i > 1)
    {
        if (i > 3)
            v4[i] = hint[i - 4];
        else
            v4[i] = user[i + 5];
    }
    else
    {
        v4[i] = user[i + 2];
    }
}
```



```

for (int i = 0; i < 9; i++)
{
    cout << v4[i];
}
cout << "\n";

for (int i = 0; i < 9; i++)
{
    pass += ((user[i] + v4[i]) / 2);
}

cout << pass;

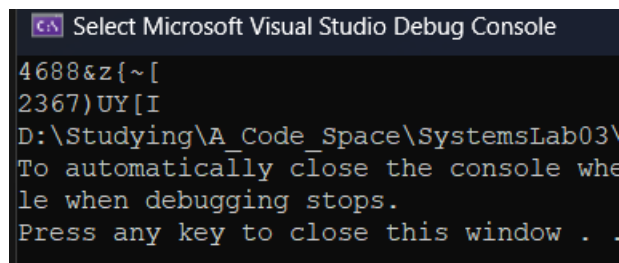
return 0;

```

Output cho thấy:

4688&z{~[là v4

2367)UY[I là password tương ứng

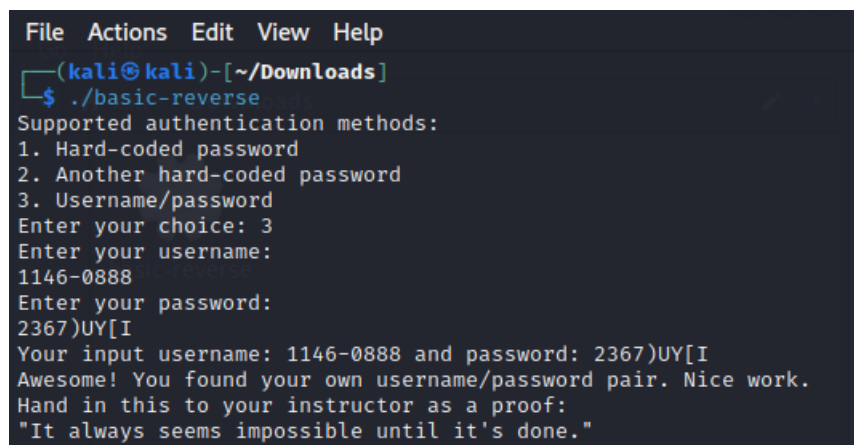


```

Select Microsoft Visual Studio Debug Console
4688&z{~[
2367)UY[I
D:\Studying\A_Code_Space\Systemslab03\
To automatically close the console when
le when debugging stops.
Press any key to close this window . .

```

Thử nghiệm kết quả:



```

File Actions Edit View Help
(kali@kali)-[~/Downloads]
$ ./basic-reverse
Supported authentication methods:
1. Hard-coded password
2. Another hard-coded password
3. Username/password
Enter your choice: 3
Enter your username:
1146-0888
Enter your password:
2367)UY[I
Your input username: 1146-0888 and password: 2367)UY[I
Awesome! You found your own username/password pair. Nice work.
Hand in this to your instructor as a proof:
"It always seems impossible until it's done."

```

YÊU CẦU CHUNG

Báo cáo:

- File **.PDF**.
- Đặt tên theo định dạng: **[Mã lớp]-Lab3_NhomX_MSSV1-MSSV2-MSSV3.pdf** (trong đó X là số thứ tự nhóm, MSSV gồm đầy đủ MSSV của tất cả các thành viên thực hiện bài thực hành).

Ví dụ: [NT209.N21.ANTT.1]-Lab3_Nhom2_21520001-21520013-21520036.pdf.

- Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

Đánh giá:

- Hoàn thành tốt yêu cầu được giao.
- Có nội dung mở rộng, ứng dụng.

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

HẾT