Lab
4

BÁO CÁO BÀI THỰC HÀNH SỐ 4

Tìm hiểu cơ bản về kỹ thuật dịch ngược

Basic Reverse

Môn học: Lập trình hệ thống

Giảng viên hướng dẫn	ThS. Đỗ Thị Hương Lan
Sinh viên thực hiện	Nguyễn Hồ Nhật Khoa (22520677) Lê Quốc Ngô (22520951) Trần Tiễn Nhật (22521030)
Mức độ hoàn thành	Hoàn thành
Thời gian thực hiện	17/5/2024
Tự chấm điểm	10/10

Yêu cầu 1.

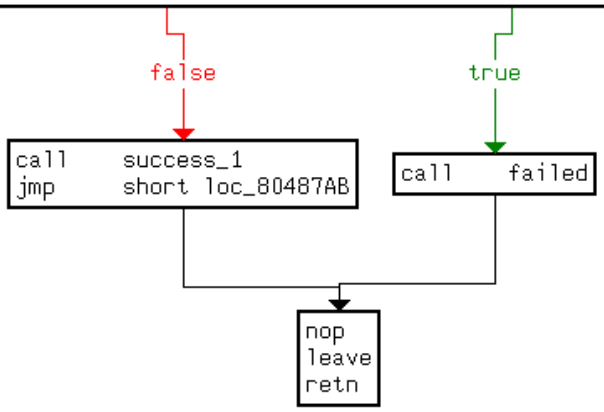
```
push    ebp                     (Lưu giá trị của ebp trên đỉnh stack)
mov     ebp, esp                (Con trỏ stack esp trỏ vào đỉnh stack ebp)
sub     esp, 3F8h               (Cấp phát 1016 bytes)
call    _getchar                (Loại bỏ khoảng trắng)
sub     esp, 0Ch                (Cấp phát thêm 12 bytes)
push    offset aEnterTheHardCo ; "Enter the hard-coded password (option 1"...
call    _puts                   (Đưa địa chỉ chuỗi "Enter the hard-coded password
                                (option 1)" lên đỉnh stack => call _puts in ra màn hình)
add     esp, 10h                (thêm 10 vào esp)
sub     esp, 8
lea     eax, [ebp+s1]           (tính toán địa chỉ s1 trong stack => lưu vào eax)
push    eax
push    offset asc_804926A ; "%[^\n]"
call    ___isoc99_scanf         (xuống dòng , đọc 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                 (tương tự, in ra "Your input hard-coded
                                password:" + chuỗi s1 đã nhập)
add     esp, 10h
sub     esp, 8
push    offset s2       ; "Never offer to teach fish to swim"
                                (đưa chuỗi s2 " Never offer to teach fish to swim" vào
                                stack)
lea     eax, [ebp+s1]           (Đưa s1 vào eax)
push    eax             ; s1    (đưa eax vào stack)
call    _strcmp                 (so sánh s1, s2)
add     esp, 10h
test    eax, eax                (and eax với chính nó => 0 thì giống call success_1)
jnz     short loc_80487A6
call    success_1
jmp     short loc_80487AB
```

Chuỗi được lưu tại biến s2:

```
.rodata:08048C70 ; char s2[]
.rodata:08048C70 s2              db 'Never offer to teach fish to swim',0
```

Kết quả trả về ở eax, nếu eax = 0 thì 2 chuỗi bằng nhau thì thực hiện tiếp đến hàm success_1 và in ra thông điệp báo thành công, còn nếu không bằng thì nhảy đến short loc_80487A6 và thông báo không phê duyệt.

```
hardCode:
push    ebp
mov     ebp, esp
sub     esp, 3F8h
call    _getchar
sub     esp, 0Ch
push    offset aEnterTheHardCo; "Enter the hard-coded password (option 1"...
call    _puts
add     esp, 10h
sub     esp, 8
lea     eax, [ebp+s1]
push    eax
push    offset asc_804926A; "%[^\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
sub     esp, 8
push    offset s2       ; "Never offer to teach fish to swim"
lea     eax, [ebp+s1]
push    eax             ; s1
call    _strcmp
add     esp, 10h
test    eax, eax
jnz     short loc_80487A6
```



Run:

```
tnh47@Nhat: ~
tnh47@Nhat:~$ ./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):
Never offer to teach fish to swim
Your input hard-coded password: Never offer to teach fish to swim
Congrats! You found the hard-coded secret, good job :).
tnh47@Nhat:~$
```

Vậy password là: "**Never offer to teach fish to swim**"

Yêu cầu 2.

```
.text:080487AE                push    ebp
.text:080487AF                mov     ebp, esp
.text:080487B1                sub     esp, 18h
.text:080487B4                call    _getchar
.text:080487B9                sub     esp, 0Ch
.text:080487BC                push    offset aEnterYour2Numb ; "Enter your 2 numbers (separated by spac"...
.text:080487C1                call    _puts
.text:080487C6                add     esp, 10h
.text:080487C9                sub     esp, 4
.text:080487CC                lea     eax, [ebp+var_14]
.text:080487CF                push    eax
.text:080487D0                lea     eax, [ebp+var_10]
.text:080487D3                push    eax
.text:080487D4                push    offset aDD      ; "%d %d"
.text:080487D9                call    ___isoc99_scanf
.text:080487DE                add     esp, 10h
.text:080487E1                mov     edx, [ebp+var_14]
.text:080487E4                mov     eax, [ebp+var_10]
.text:080487E7                sub     esp, 4
.text:080487EA                push    edx
.text:080487EB                push    eax
.text:080487EC                push    offset aYourInputDD ; "Your input: %d %d\n"
.text:080487F1                call    _printf
.text:080487F6                add     esp, 16
.text:080487F9                mov     dword ptr [ebp-14o], 8
.text:08048800                mov     eax, [ebp+var_10]
.text:08048803                cmp     eax, [ebp+var_C]
.text:08048806                jnz     short loc_8048839
.text:08048808                mov     eax, [ebp+var_10]
.text:0804880B                mov     edx, ds:funny_seq[eax*4]
.text:08048812                mov     eax, [ebp+var_10]
.text:08048815                sub     esp, 8
.text:08048818                push    edx
.text:08048819                push    eax
.text:0804881A                call    funny_func
.text:0804881F                add     esp, 16
.text:08048822                mov     edx, eax
.text:08048824                mov     eax, [ebp+var_14]
.text:08048827                cmp     edx, eax
.text:08048829                jnz     short loc_8048832
.text:0804882B                call    success_2
.text:08048830                jmp     short loc_804883E
```

Tương tự, yêu cầu 1 sử dụng _getchar() bỏ khoảng trắng, _puts() in ra thông điệp ” Enter your 2 numbers (separated by space) (option 2):”, _isoc99_scanf (người dùng nhập 2 số), và _printf() (in ra 2 số vừa nhập)

mov dword ptr [ebp-12], 8	*(Gán giá trị 8 vào biến cục bộ tại ebp-14)*
mov eax, [ebp+var_10]	*(Lấy giá trị của số thứ nhất và lưu vào eax)*
cmp eax, [ebp-12]	*(So sánh giá trị của số thứ nhất với 8.)*
jnz short loc_8048839	*(Nếu không bằng, nhảy đến loc_8048839)*
mov eax, [ebp+var_10]	*(Lấy giá trị của số thứ nhất và lưu vào eax)*
mov edx, ds:funny_seq[eax*4]	*(Lấy giá trị từ mảng funny_seq tại vị trí eax*4 và lưu vào edx)*

```
.rodata:08048B60                public funny_seq
.rodata:08048B60 funny_seq      dd 10                   ; DATA XREF: otherhardCode+5D↑r
.rodata:08048B64                db   3
.rodata:08048B65                db   0
.rodata:08048B66                db   0
.rodata:08048B67                db   0
.rodata:08048B68                db   6
.rodata:08048B69                db   0
.rodata:08048B6A                db   0
.rodata:08048B6B                db   0
.rodata:08048B6C                db   9
.rodata:08048B6D                db   0
.rodata:08048B6E                db   0
.rodata:08048B6F                db   0
.rodata:08048B70                db   1
.rodata:08048B71                db   0
.rodata:08048B72                db   0
.rodata:08048B73                db   0
.rodata:08048B74                db   4
.rodata:08048B75                db   0
.rodata:08048B76                db   0
.rodata:08048B77                db   0
.rodata:08048B78                db   7
.rodata:08048B79                db   0
.rodata:08048B7A                db   0
.rodata:08048B7B                db   0
.rodata:08048B7C                db   2
.rodata:08048B7D                db   0
.rodata:08048B7E                db   0
.rodata:08048B7F                db   0
.rodata:08048B80                db   5
.rodata:08048B81                db   0
.rodata:08048B82                db   0
.rodata:08048B83                db   0
.rodata:08048B84                db   8
.rodata:08048B85                db   0
.rodata:08048B86                db   0
.rodata:08048B87                db   0
```

Nhìn vào mảng ta thấy được funny_seq[eax*4] bằng 5.

```
mov     eax, [ebp+var_10]
sub     esp, 8
push    edx
push    eax
call    funny_func
```

(Gọi hàm funny_func với tham số thứ nhất là eax bằng 8, tham số thứ 2 là edx bằng 5)

```
.text:0804868B                public funny_func
.text:0804868B funny_func     proc near               ; CODE XREF: otherhardCode+6C↓p
.text:0804868B
.text:0804868B var_4          = dword ptr -4
.text:0804868B arg_0          = dword ptr  8
.text:0804868B arg_4          = dword ptr  0Ch
.text:0804868B
.text:0804868B                push    ebp
.text:0804868C                mov     ebp, esp
.text:0804868E                sub     esp, 16
.text:08048691                mov     [ebp+var_4], 0
.text:08048698                mov     edx, [ebp+arg_0]
.text:0804869B                mov     eax, [ebp+arg_4]
.text:0804869E                add     eax, edx
.text:080486A0                imul    eax, [ebp+arg_0]
.text:080486A4                mov     edx, eax
.text:080486A6                mov     eax, [ebp+arg_4]
.text:080486A9                add     eax, edx
.text:080486AB                mov     [ebp+var_4], eax
.text:080486AE                mov     eax, [ebp+var_4]
.text:080486B1                leave
.text:080486B2                retn
.text:080486B2 funny_func     endp
```

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

Giá trị trả về bằng 8 * (8 + 5) + 5 = 109

```
add     esp, 16
mov     edx, eax                 (Lấy giá trị trả về từ funny_func và lưu vào edx)
mov     eax, [ebp+var_14]        (Lấy giá trị của số thứ hai và lưu vào eax)
cmp     edx, eax                 (So sánh giá trị trả về từ funny_func với số thứ
                                  hai, nếu bằng => nhảy đến success_2)
jnz     short loc_8048832        (Nếu không bằng, nhảy đến loc_8048832)
call    success_2
jmp     short loc_804883E
```

⇨ **2 số là 8 và 109**

Run:

```
tnh47@Nhat: ~
tnh47@Nhat:~$ ./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):
8 109
Your input: 8 109
Congrats! You found a secret pair of numbers :).
tnh47@Nhat:~$
```

Yêu cầu 3.

Đoạn đầu là khai báo các biến:

```
size_t v0; // ebx@2
int result; // eax@3
long double v2; // fst7@13
size_t v3; // eax@15
size_t v4; // edx@16
char v5[9]; // [sp+1Ah] [bp-2Eh]@6
char pass[10]; // [sp+23h] [bp-25h]@1
char s[10]; // [sp+2Dh] [bp-1Bh]@1
char v8; // [sp+37h] [bp-11h]@1
char v9; // [sp+38h] [bp-10h]@1
char v10; // [sp+39h] [bp-Fh]@1
char v11; // [sp+3Ah] [bp-Eh]@1
char v12; // [sp+3Bh] [bp-Dh]@1
unsigned int i; // [sp+3Ch] [bp-Ch]@4
```

Ta thấy có 5 biến kiểu char nằm ở các vị trí liên tục từ (sp+0x37) đến (sp+0x3B), hay có thể nói có 5 ký tự nằm liên tục nhau (cách nhau 1 byte) từ địa chỉ của biến v8. Sau đó, trong đoạn mã giả có đoạn truy xuất các giá trị dựa trên địa chỉ của v8.

```
for ( i = 0; (signed int)i <= 8; ++i )
{
  if ( (signed int)i > 1 )
  {
    if ( (signed int)i > 3 )
      v5[i] = *(&v8 + 8 - i);
    else
      v5[i] = s[i + 2];
  }
  else
  {
    v5[i] = s[i + 5];
  }
```

Do đó có thể thấy đây là mảng gồm 5 ký tự.

Đoạn dưới đây cấp giá trị cho mảng v8 bằng giá trị ASCII, đổi ra sẽ được v7 = ">:O5"

```
v8 = '>';
v9 = ':';
v10 = '~';
v11 = 'O';
v12 = '5';
```

Thực hiện tương tự công đoạn lấy dữ liệu đầu vào và hiển thị ra với 2 trường username và password: getchar() -> __isoc99_scanf -> getchar() -> puts() -> __isoc99_scanf -> printf. Xong rồi check thử xem độ dài password đầu vào của username : **len(username)== 9**? và xem **len(password) == 9**?

```
getchar();
puts("Enter your username:");
__isoc99_scanf("%[^\n]", username);
getchar();
puts("Enter your password:");
__isoc99_scanf("%[^\n]", password);
printf("Your input username: %s and password: %s\n", username, password);
if ( strlen(username) == 9 && (v0 = strlen(username), v0 == strlen(password)) )
{
```

Chương trình tạo một mảng v5 như sau:

```
for ( i = 0; (signed int)i <= 8; ++i )
{
  if ( (signed int)i > 1 )
  {
    if ( (signed int)i > 3 )
      v5[i] = *(&v8 + 8 - i);
    else
      v5[i] = username[i + 2];
  }
  else
  {
    v5[i] = username[i + 5];
  }
```

MSSV1 = "22521030"
MSSV2 = "22520951"
MSSV3 = "22520677"
username = "030951677" (3 số cuối nối nhau của 3 mã số sinh viên). Chạy i -> [0;8]:
2 ký tự đầu của v5 là ký tự tại vị trí 5,6 của username
2 ký tự tiếp của v5 là ký tự tại vị trí 4,5 của username
4 ký tự còn lại của v5 là chuỗi v8 theo thứ tự ngược lại “5O~:>”
Kết quả : v5 = “16515O~:>"

Tiếp theo hình thành mật khẩu để đối chiếu:

```
for ( i = 0; ; ++i )
{
  v3 = strlen(username);
  if ( v3 > i )
  {
    v2 = ceil((long double)((username[i] + v5[i]) / 2));
    if ( (long double)password[i] == v2 )
      continue;
  }
  break;
}
v4 = strlen(username);
if ( v4 == i )
  result = success_3();
else
  result = failed();
```

Biết v3 = strlen(username) (8) được dùng để làm giới hạn vòng lặp, nếu thỏa một trong 2 điều kiện tại if (v3 > i) thì tiếp tục vòng lặp. Sau đó xét v4 có bằng i hay không, nếu có thì success_3() được gọi.
Vậy làm thế nào để i = v4 = 8 ?
Phải cho i chạy hết vòng lặp cho đến khi v3 <= i (vì i++). Để làm được thì ta phải luôn làm cho điều kiện (username[i] + v5[i]) / 2 == password[i] đúng thì vòng lặp không break.
Dưới đây là đoạn code python được dùng để thực thi chương trình tìm password:

```python
def userpass():
    v8 = ">:~O5"
    v5 = [''] * 9
    v6 = "030951677"

    v0 = len(v6)
    for i in range(9):
        if i > 1:
            if i > 3:
                v5[i] = v8[8 - i]
            else:
                v5[i] = v6[i + 2]
        else:
            v5[i] = v6[i + 5]

    print("v5:", v5)
    res = ''
    for i in range(len(v6)):
        v2 = ((ord(v6[i]) + ord(v5[i])) // 2)
        res += chr(v2)
    print(res)
userpass()
```

Chạy để tìm ra output (password):

```
PS D:\Code\Python> python -u "d:\Code\Python\reverse-basic.py"
v5: ['1', '6', '5', '1', '5', 'O', '~', ':', '>']
04255@Z8:
PS D:\Code\Python>
```

Run:

```
tnh47@Nhat: ~
tnh47@Nhat:~$ ./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:
030951677
Enter your password:
04255@Z8:
Your input username: 030951677 and password: 04255@Z8:
Congrats! You found your own username/password pair :).
tnh47@Nhat:~$
```

Vậy password là: **04255@Z8:**