Bai3

Tuesday, May 27, 2025 5:46 PM

- Tính toán giá trị với tham số được truyền vào hàm foo:
foo("qwerty0123").

```
foo(char*):
    push   rbp
    mov    rbp, rsp
    mov    QWORD PTR [rbp-24], rdi
    mov    DWORD PTR [rbp-4], 59
    mov    DWORD PTR [rbp-8], 0
    jmp    .L2
```

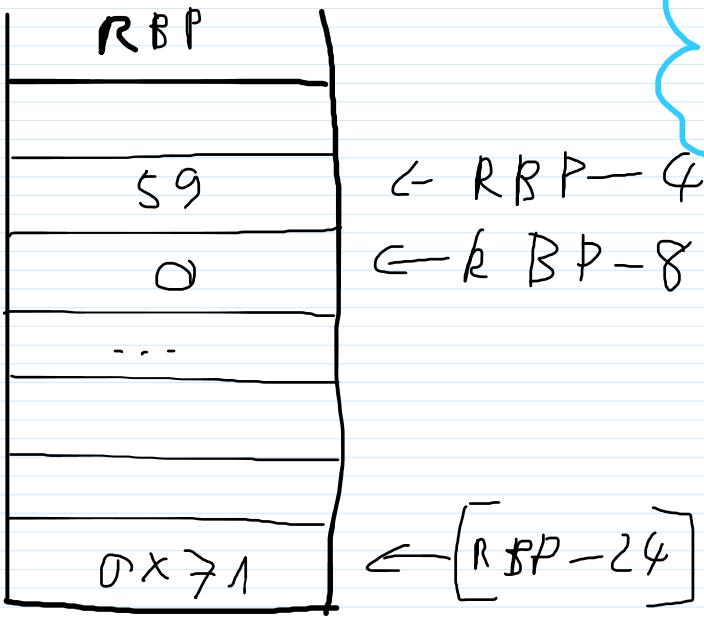
Phân tích đoạn này :

Đầu tiên thì vẫn là khởi tạo bộ nhớ bằng cách đẩy rbp (register base pointer) vào stack

Rồi cho con trỏ stack lên thực hiện chương trình thoai.

Lúc này do truyền tham số là chuỗi kí tự và thanh RDI sẽ nhận value lần lượt 0x 71 77 65 72 74 79 30 31 32 33

Sau đó gán chuỗi này vào chỗ lưu trữ cục bộ của stack và lần lượt gán value vào địa chỉ xong nhảy vào hàm .L2



Pseudo Code :

```
void foo(char *c)
{
    int tmp1 = 59;
    int tmp2 = 0;
    L2();
}
```

```
.L2:
    cmp    DWORD PTR [rbp-8], 9
    jle    .L3
    mov    eax, DWORD PTR [rbp-4]
    and    eax, 2097151
    pop    rbp
    ret
```

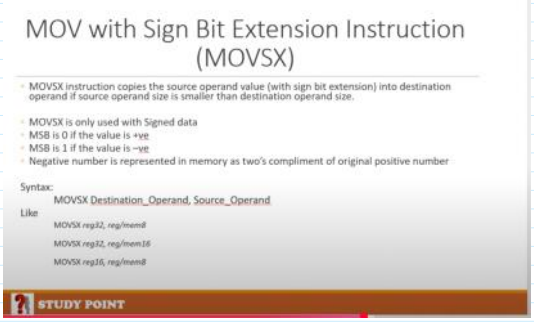
Sau khi nhảy vào L2 thì so sánh xem tại [rbp-8] có bằng 9 không?

Và dòng dưới ta thấy rằng hàm sẽ nhảy nếu nó nhỏ hơn hay bằng 9 (jle là jump if less or equal)

Và như trên thì ta đã gán nó = 0 nên chắc chắn sẽ nhảy roài.

Pseudo code .L2

```
int L2()
{
    if(tmp2 <= 9) L3();
    eax = tmp1;
   eax = eax & 2097151;
   return eax;
}
```




```
.L3:
    mov    eax, DWORD PTR [rbp-8]
```

Well, vào hàm này ta gán giá trị tại địa chỉ con trỏ tại [rbp-8]

```
.L3:
    mov    eax, DWORD PTR [rbp-8]
    movsx  rdx, eax
    mov    rax, QWORD PTR [rbp-24]
    add    rax, rdx
    movzx  eax, BYTE PTR [rax]
    movsx  edx, al
    mov    eax, DWORD PTR [rbp-4]
    imul   eax, eax, 271
    add    eax, edx
    mov    DWORD PTR [rbp-4], eax
    add    DWORD PTR [rbp-8], 1
```

Well, vào hàm này ta gán giá trị tại địa chỉ con trỏ tại [rbp-8] cho thanh eax và lúc này nó = 0 :v.

Khúc dưới là movsx (xem ảnh trên), đây là lệnh mov nhưng chỉ dùng để Copy từ thằng biến ban đầu có kích thước nhỏ hơn thằng đích

Sau đưa thanh rax vào vị trí bắt đầu của chuỗi thêm vào.
Sau đó đưa thanh rax đến vị trí (0+rdi) (tương tự s[0 + rdi])

Sau đó dùng lệnh +, rax += rdx, để dời con trỏ trỏ vào chuỗi một khoảng rdi;

Rán value dc lưu trong address của rax mà do rdi =0 ([rax] = 0x71) vào thanh eax

Do lúc này eax = 0x71 nên thanh al (chỉ có 1 byte) được gán vào edx luôn

Tiếp tục gán value trong địa chỉ của rbp-4 (= 59) vào eax.

imul tức là phép nhân á, eax *= (decimal) eax * 271, mà do eax = 59 * 271 = 15.989

Tiếp tục + edx vào eax ==> eax = 0x71 + (hex)(15.989) = 0x3ee6.

Rồi lại gán nó cho pointer tại rbp-4 = 0x3ee6.

Xong rồi + biến lên 1. rồi kết thúc hàm.

Tiếp đến vào L2 thì lại so sánh với 9 thì lúc này biến chỉ mới = 1 nên nó lại gọi vào L3 (tới đây thì chắc cũng biết đây là vòng lặp rồi hơ, nên là ngồi phân tích lại mệt lắm nên ta sẽ chuyển pseodu code rồi chạy lun cho lẹ :))).

Pseudo code :

```
#include<iostream>
using namespace std;
void L3(const int i, long long &total, const string t)
{
    char rax = t[i];    /*mov     eax, I
                        movsx   rdx, eax
                        mov     rax, QWORD PTR [rbp-24]
                        add     rax, rdx*/
    long long eax = (int)rax; /*movzx   eax, BYTE PTR [rax]
                              movsx   edx, al //lay 1 byte cuoi*/
    eax *= 271; //imul    eax, eax, 271
    eax += total; //add     eax, edx
    total = eax;

}
```

```
void foo(string t)
{
    long long total = 59; //DWORD PTR [rbp-4], 59
    int i =0; //DWORD PTR [rbp-8], 0
    //L2 chinh la vong lap for nha.
    for(int i; i <= 9;i++) //buoc nay la compare
    {
        L3(i,total,t);
    }
    total = total & 2097151;
    cout << total;
    return;
}
int main()
{
    string t = "qwerty0123";
    foo(t);

    return 0;
}
```