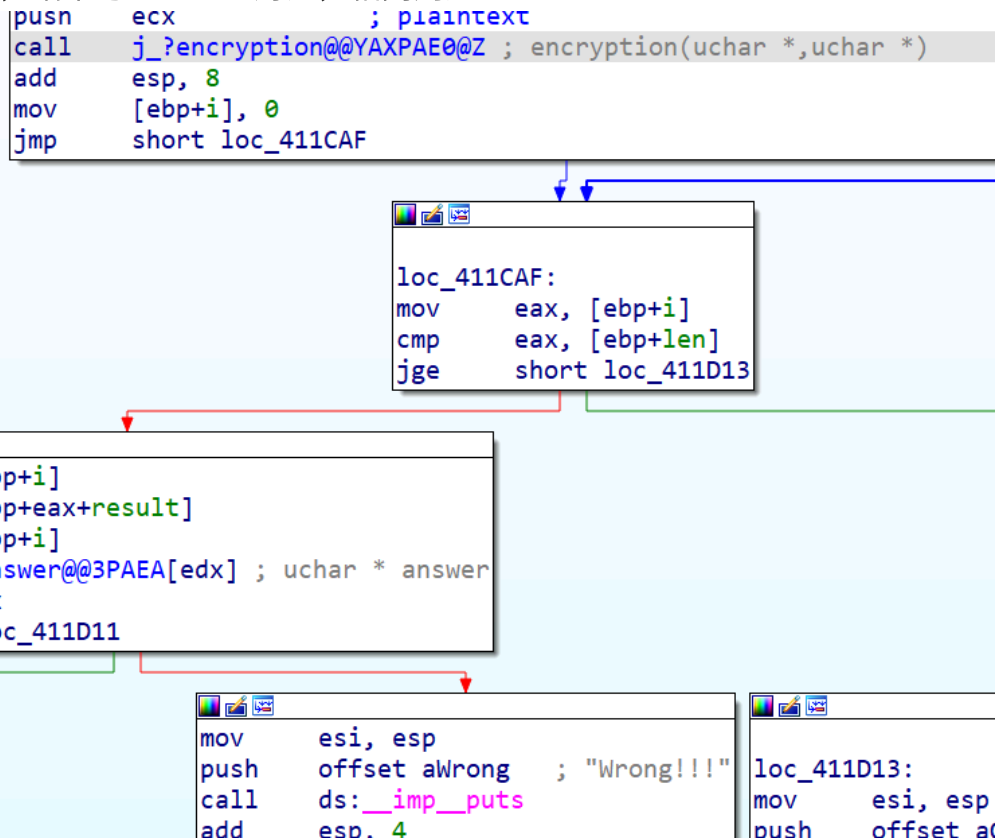


RC4逆向

主逻辑

将明文进行加密（encryption()），结果与 answer 对比，相同则通过。



encryption() 分析

1. 定义 i、j，初始化 sbbox（init_sbbox()）；

```
mov     ?pos_i@@3HA, 0 ; int pos_i
mov     ?pos_j@@3HA, 0 ; int pos_j
mov     eax, [ebp+plaintext]
push    eax                ; Str
call    j__strlen
add     esp, 4
mov     [ebp+len], eax
mov     [ebp+i], 0
call    j_?init_sbbox@@YAXXZ ; init_sbbox(void)
nop
mov     [ebp+i], 0
jmp     short loc_411838
```

2. 根据 generate_key() 结果，逐字节异或加密，结果存到 result 里。

```

loc_411838:
mov     eax, [ebp+i]
cmp     eax, [ebp+len]
jge     short loc_41185D

```

```

mov     eax, [ebp+plaintext]
add     eax, [ebp+i]
movzx   ebx, byte ptr [eax]
call    j_?generate_key@@YAEXZ ; generate_key(void)
movzx   ecx, al
xor     ebx, ecx
mov     edx, [ebp+result]
add     edx, [ebp+i]
mov     [edx], bl
jmp     short loc_41182F

```

```

loc_41185D:
mov
add
mov
pop
pop
pop
add
cmp
call

```

init_sbox() 分析

1. 初始化, $sbox[i] = i$ (大小为256):

```

loc_411A28:
cmp     [ebp+i], 100h
jge     short loc_411A40

```

```

mov     eax, [ebp+i]
mov     ecx, [ebp+i]
mov     ?sbox@@3PAHA[eax*4], ecx ; int * sbox
jmp     short loc_411A1F

```

2. 特定算法处理 (先置 i 、 j 都为0, i 从0到255循环256轮):

- $j = j + sbox[i]$: 实现时, $sbox$ 为一个指针, 所以索引为 $i*4$ 实现 int 型数组 (4个字节);

```

mov     eax, [ebp+i]
mov     ecx, [ebp+j]
add     ecx, ?sbox@@3PAHA[eax*4] ; int * sbox

```

- $j = j + \text{key}[i \% \text{key_len}]$ ，存入 ecx；

```

mov     eax, [ebp+i]
cdq
idiv    [ebp+key_len]
movzx   edx, byte ptr ?key@@3PAEA[edx] ; "RC4key"
add     ecx, edx

```

- 扩展（8位扩展至32位）：先取符号位和低8位，正数直接进行下一步加密，若为负数，则先自减（计算反码），然后把除了低8位的其他bit置为1，再自加（计算补码），将8位补码扩展成32位补码，用于实现模256；

```

and     ecx, 800000FFh
jns     short loc_411A8F

```

```

dec     ecx
or      ecx, 0FFFFFF0h
inc     ecx

```

- 交换 sbox[i] 和 sbox[j]。

```

loc_411A8F:
mov     [ebp+j], ecx
mov     eax, [ebp+i]
mov     cl, byte ptr ?sbox@@3PAHA[eax*4] ; int * sbox
mov     [ebp+temp], cl
mov     eax, [ebp+i]
mov     ecx, [ebp+j]
mov     edx, ?sbox@@3PAHA[ecx*4] ; int * sbox
mov     ?sbox@@3PAHA[eax*4], edx ; int * sbox
movzx   eax, [ebp+temp]
mov     ecx, [ebp+j]
mov     ?sbox@@3PAHA[ecx*4], eax ; int * sbox
jmp     short loc_411A50

```

generate_key() 分析

1. 计算 $i += 1$ ，然后模256（过程同上）；
2. 计算 $j = (j + \text{sbox}[i]) \bmod 256$ ；

```
mov     ?pos_i@@3HA, eax ; int pos_i
mov     eax, ?pos_i@@3HA ; int pos_i
mov     ecx, ?pos_j@@3HA ; int pos_j
add     ecx, ?sbox@@3PAHA[eax*4] ; int * sbox
and     ecx, 800000FFh
jns     short loc_411913
```

```
dec     ecx
or      ecx, 0FFFFFF00h
inc     ecx
```



3. 交换 $\text{sbox}[i]$ 和 $\text{sbox}[j]$ （过程初始S盒第二步中的交换）；
4. 计算 $\text{sbox}[(\text{sbox}[i] + \text{sbox}[j]) \bmod 256]$ ，作为函数返回值，其中 t 做为索引暂存变量。

```
mov     eax, ?pos_i@@3HA ; int pos_i
mov     ecx, ?sbox@@3PAHA[eax*4] ; int * sbox
mov     edx, ?pos_j@@3HA ; int pos_j
add     ecx, ?sbox@@3PAHA[edx*4] ; int * sbox
and     ecx, 800000FFh
jns     short loc_41197B
```

```
dec     ecx
or      ecx, 0FFFFFF00h
inc     ecx
```



```
loc_41197B:
mov     [ebp+t], ecx
mov     eax, [ebp+t]
movzx   eax, byte ptr ?sbox@@3PAHA[eax*4] ; int * sbox
```