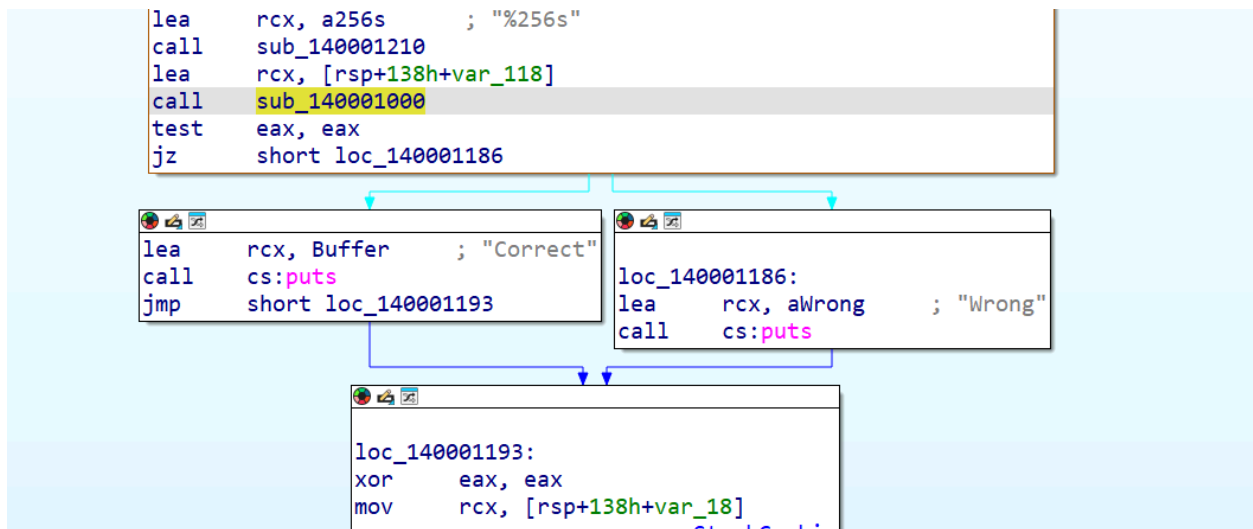
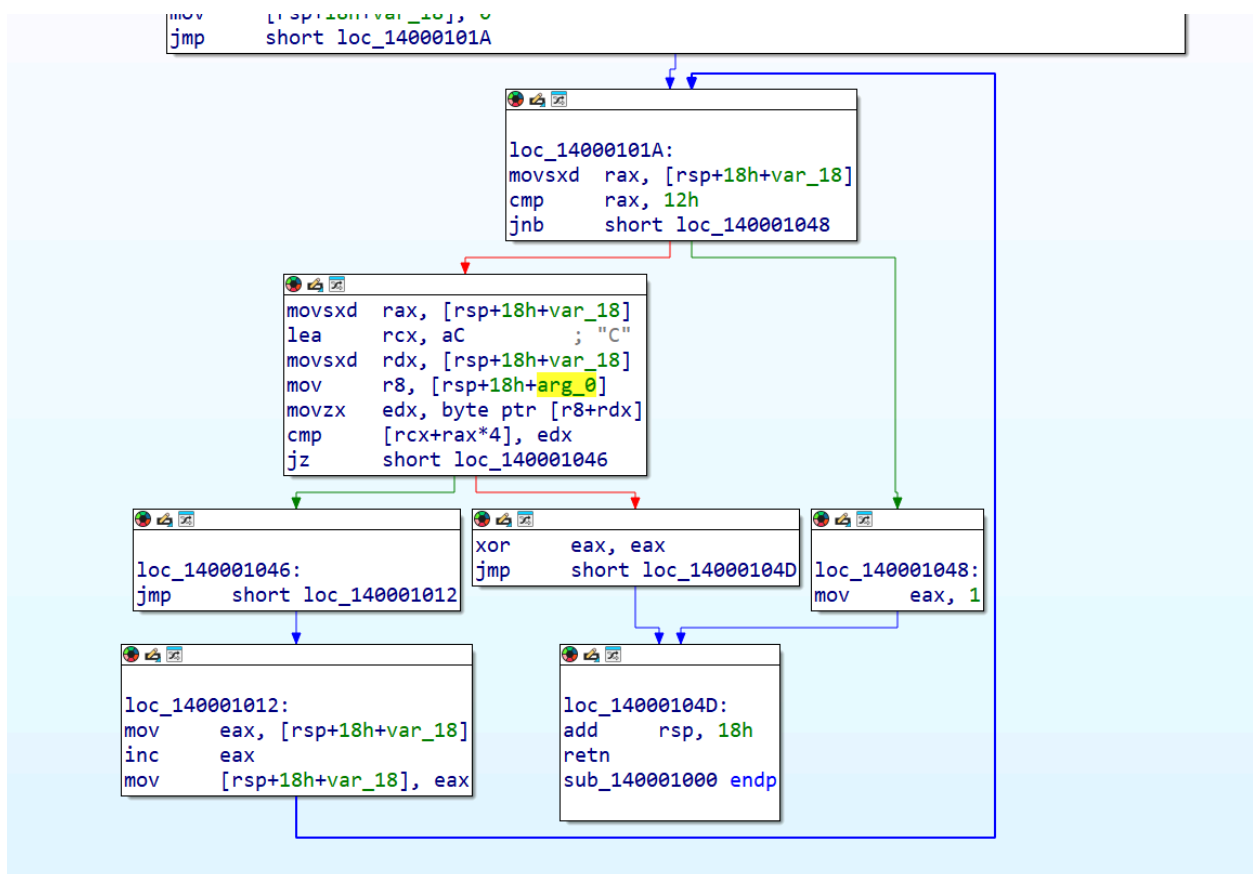


rev-basic-2



마찬가지의 구성



언뜻봐서는 루프문으로도 보인다.

```
1 int 1, // [130000] [130000]
2
3
4
5 for ( i = 0; (unsigned __int64)i < 0x12; ++i )
6 {
7     if ( *(_DWORD *)&aC[4 * i] != *(unsigned __int8 *)(a1 + i) )
8         return 0;
9 }
10 return 1;
11 }
```

맞네.... 0x12면 18이니까 18번 반복하나 보다. 플래그의 길이는 18로 추정.

```
03000 aC          db 'C',0                      ; DATA XREF: sub_140001000+2870
03002          align 4
03004 a0          db 'o',0
03006          align 8
03008 aM          db 'm',0
0300A          align 4
0300C aP          db 'p',0
0300E          align 10h
03010 a4          db '4',0
03012          align 4
03014 aR          db 'r',0
03016          align 8
03018 aE          db 'e',0
0301A          align 4
0301C          db '_',0
0301E          align 20h
03020 aT          db 't',0
03022          align 4
03024          db 'h',0
03026          align 8
03028 aE_0        db 'e',0
0302A          align 4
0302C          db '_',0
0302E          align 10h
03030 aA          db 'a',0
03032          align 4
03034 aR_0        db 'r',0
03036          align 8
03038 aR_1        db 'r',0
0303A          align 4
0303C a4_0        db '4',0
0303E          align 20h
03040 aY          db 'y',0
```

&aC 확인하려고 스택 들어가니까 바로 정답 나와있었음... 헐.....

Comp4re_the_arra4y

그런데 이대로 입력하니까 정답 아니라 그래서, 동적분석하면서 살펴보자

```
.text:00007FF6A0F4103D cmp [rcx+rax*4], edx  
.text:00007FF6A0F41040 jz short loc_7FF6A0F41046
```

일단 edx를 C와 같게 만들어야겠음.


아니 강 잘못 옮겼네

Comp4re_the_arr4y 임....

1 LEVEL 1 rev-basic-2 문제를 해결했습니다.

대단해요. 문제를 어떻게 해결하셨나요?
풀이를 작성하면 포인트까지 받을 수 있어요.

괜찮아요

 풀이 작성하기