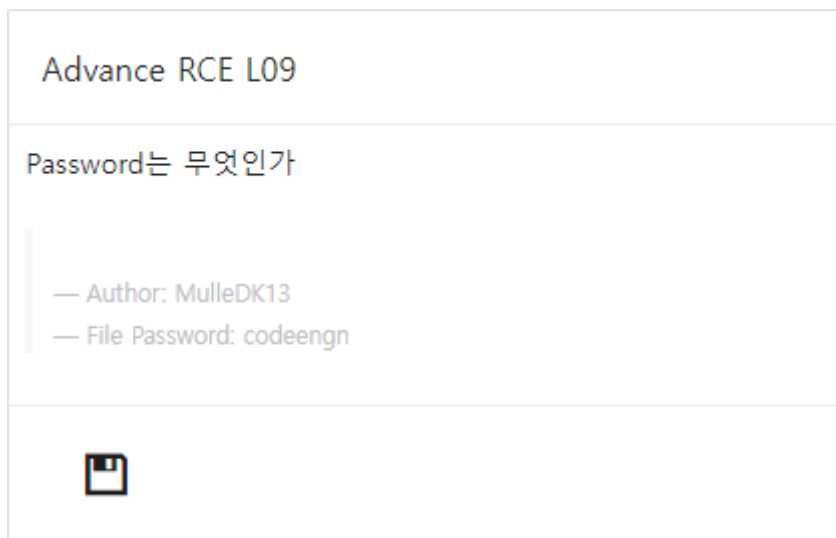


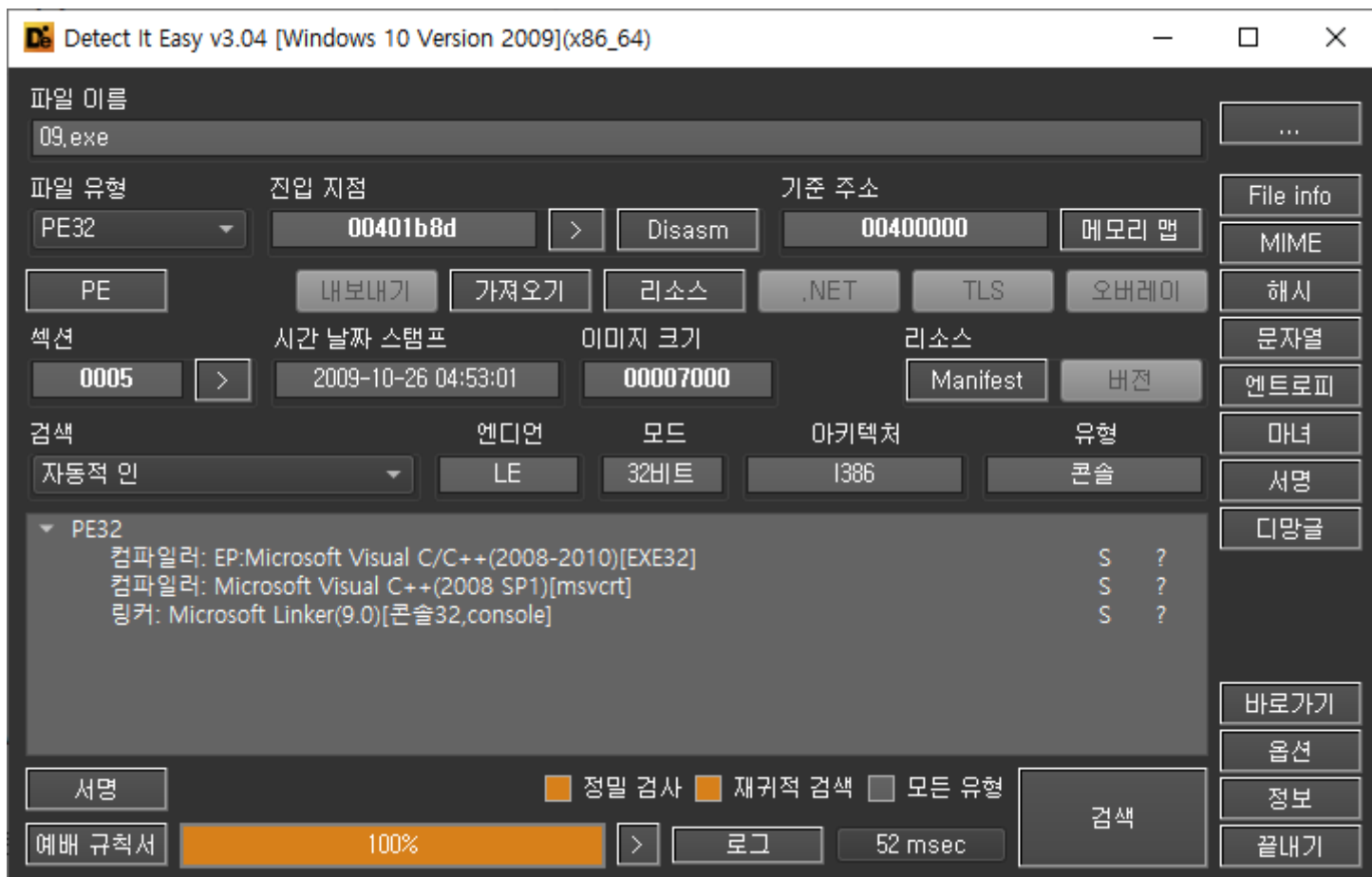
# codeengn-advance-L09 풀이

리버싱 문제풀이 / Wonlf / 2022. 5. 16. 17:43



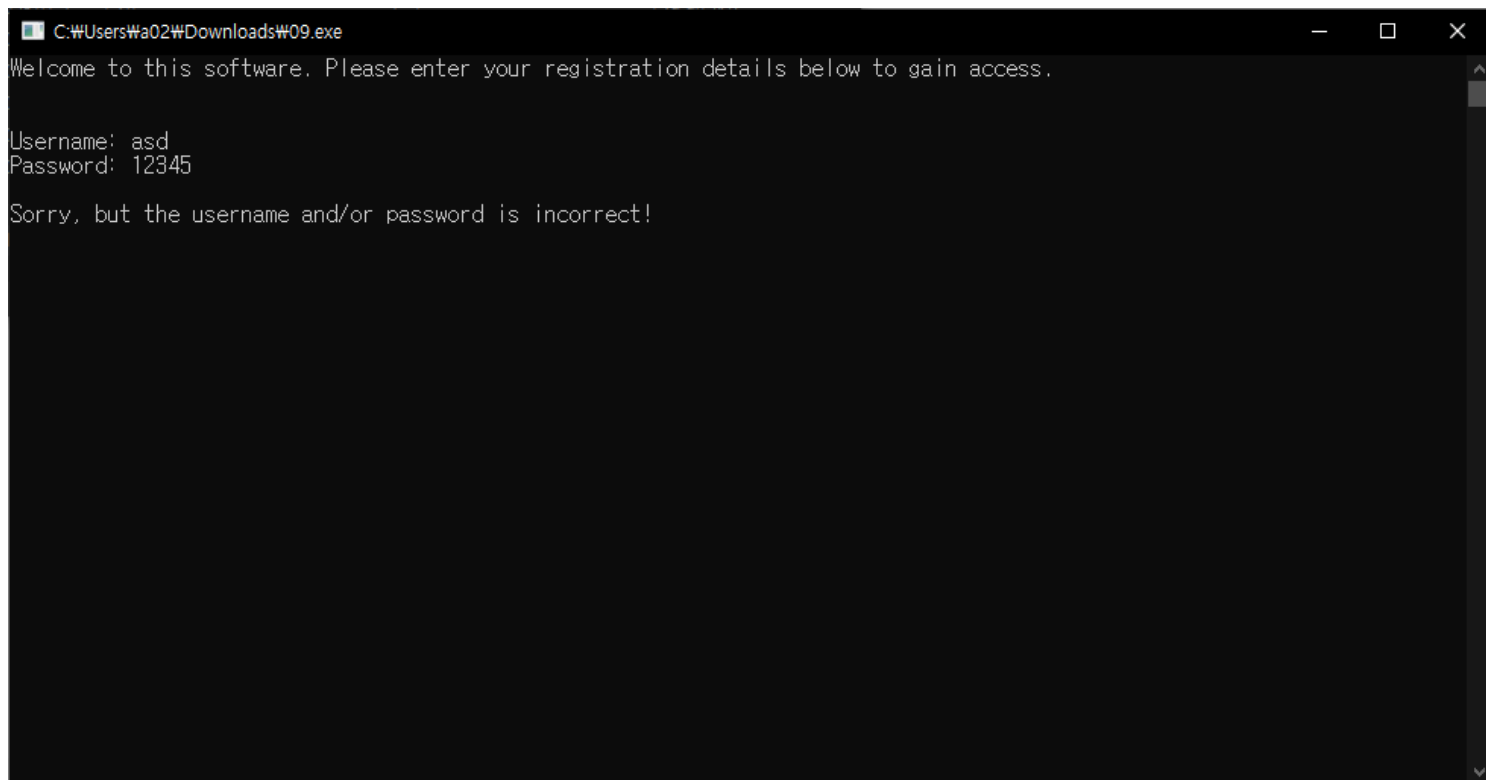
문제는 Password를 원하고 있다.

Die로 열어본다.



특이 사항은 보이지 않는다.

프로그램을 실행시켜 본다.



문제에서의 Password는 이것을 원하는 것 같고, Password를 알려면 Username까지 알아야 할 것이다.

디버거로 열어본다.

|          |                 |   |   |
|----------|-----------------|---|---|
| 00FB12A5 | FF15 4430FB00   | call dword ptr ds:[<??6?\$basic_ostream@GU?\$char...  |   |
| 00FB12AB | 84DB            | test bl,bl  |   |
| 00FB12AD | 74 30           | je 09.FB12DF  |   |
| 00FB12AF | 807C24 0B 00    | cmp byte ptr ss:[esp+8],0                             |   |
| 00FB12B4 | 74 29           | je 09.FB12DF  |   |
| 00FB12B6 | 0FB605 C831FB00 | movzx eax,byte ptr ds:[FB31C8]                        | 00FB31C8:"Welcome, TheRightName. You've gained access!"     |
| 00FB12BD | 84C0            | test al,al  |   |
| 00FB12BF | 74 47           | je 09.FB1308  |   |
| 00FB12C1 | BE C831FB00     | mov esi,09.FB31C8                                     | FB31C8:"Welcome, TheRightName. You've gained access!"       |
| 00FB12C6 | 50              | push eax  |   |
| 00FB12C7 | A1 7830FB00     | mov eax,dword ptr ds:[<??cout@std@@3V?\$basic_ostr... |   |
| 00FB12CC | 50              | push eax  |   |
| 00FB12CD | E8 AE020000     | call 09.FB1580  |   |
| 00FB12D2 | 8A46 01         | mov al,byte ptr ds:[esi+1]                            |   |
| 00FB12D5 | 46              | inc esi   |   |
| 00FB12D6 | 83C4 08         | add esp,8   |   |
| 00FB12D9 | 84C0            | test al,al  |   |
| 00FB12DB | 75 E9           | jne 09.FB12C6   |   |
| 00FB12DD | EB 29           | jmp 09.FB1308   |   |
| 00FB12DF | 0FB605 9031FB00 | movzx eax,byte ptr ds:[FB3190]                        | 00FB3190:"Sorry, but the username and/or password is incorr |
| 00FB12E6 | 84C0            | test al,al  |   |
| 00FB12E8 | 74 1E           | je 09.FB1308  |   |
| 00FB12EA | BE 9031FB00     | mov esi,09.FB3190                                     | FB3190:"Sorry, but the username and/or password is incorrec |

실패 했을 때의 문자열을 통해 성공과 실패로 가는 분기를 찾았다.

test bl, bl 구문과, cmp 구문을 통과해야 실패로 점프를 뛰지 않고 성공을 출력 하는 것 같다.

일단 test bl, bl을 먼저 보자면

이 구문은 b와 b를 and연산 하는 구문이지만, 실제 뜻은 b이 0인지 아닌지를 판단하고 0이라면 ZF를 세팅하는 구문이다.

그럼 b는 0이면 안되기 때문에 b의 값을 건드리는 부분을 위로 올라가며 찾아보도록 하자

|          |                 |  |                         |
|----------|-----------------|--|-------------------------|
| 00FB1000 | \$ 51           | push ecx   | sub_FB1000              |
| 00FB1001 | . 53            | push ebx   |                         |
| 00FB1002 | . 56            | push esi   |                         |
| 00FB1003 | . B9 F831FB00   | mov ecx,09.FB31F8                                  |                         |
| 00FB1008 | . B8 0C44FB00   | mov eax,09.FB440C                                  | FB440C:"what is"        |
| 00FB100D | . 8D49 00       | lea ecx,dword ptr ds:[ecx]                         |                         |
| 00FB1010 | > 8A10          | mov dl,byte ptr ds:[eax]                           |                         |
| 00FB1012 | . 3A11          | cmp dl,byte ptr ds:[ecx]                           |                         |
| 00FB1014 | . 75 1A         | jne 09.FB1030                                      |                         |
| 00FB1016 | . 84D2          | test dl,dl   |                         |
| 00FB1018 | . 74 12         | je 09.FB102C                                       |                         |
| 00FB101A | . 8A50 01       | mov dl,byte ptr ds:[eax+1]                         |                         |
| 00FB101D | . 3A51 01       | cmp dl,byte ptr ds:[ecx+1]                         |                         |
| 00FB1020 | . 75 0E         | jne 09.FB1030                                      |                         |
| 00FB1022 | . 83C0 02       | add eax,2  |                         |
| 00FB1025 | . 83C1 02       | add ecx,2  |                         |
| 00FB1028 | . 84D2          | test dl,dl   |                         |
| 00FB102A | . 75 E4         | jne 09.FB1010                                      |                         |
| 00FB102C | > 33C0          | xor eax,eax  |                         |
| 00FB102E | . EB 05         | jmp 09.FB1035                                      |                         |
| 00FB1030 | > 1BC0          | sbb eax,eax  |                         |
| 00FB1032 | . 83D8 FF       | sbb eax,FFFFFFFF                                   |                         |
| 00FB1035 | > 8B0D 0444FB00 | mov ecx,dword ptr ds:[FB4404]                      |                         |
| 00FB1038 | . 8B15 5030FB00 | mov edx,dword ptr ds:[<&?end1@std@@YAAAV?\$basic_c |                         |
| 00FB1041 | . 85C0          | test eax,eax                                       |                         |
| 00FB1043 | . A1 2C44FB00   | mov eax,dword ptr ds:[FB442C]                      | 00FB442C:"90"           |
| 00FB1048 | . 0F94C3        | sete bl  | ZF가 세팅이 되어 있으면 bl이 1이 됨 |
| 00FB104B | . 3B01          | cmp eax,dword ptr ds:[ecx]                         |                         |

001602A0] 88228F

위로 올라가보면, 프로그램의 모든 원리를 알 수 있는데 찾았던부분부터 보게되면

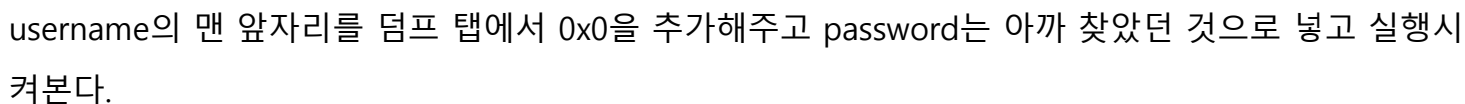
sete명령어로 b를 조작하는 구문이 보이고, 그 아래에는 내가 Password에 입력한 값과 특정 값을 비교하는 구문이 보였다.

Password는 0x88228F로 예상하고 페이지에 인증하니 실제로 통과가 되었다 하지만 여기서 멈추지 않고, 성공 구문으로 갈 수 있는 Username과 Password를 찾아보겠다.

함수의 맨 처음부터 보게되면,

|          |               |                            |                                 |
|----------|---------------|----------------------------|---------------------------------|
| 00FB1000 | \$ 51         | push ecx                   | sub_FB1000                      |
| 00FB1001 | . 53          | push ebx                   |                                 |
| 00FB1002 | . 56          | push esi                   |                                 |
| 00FB1003 | . B9 F831FB00 | mov ecx,09.FB31F8          |                                 |
| 00FB1008 | . B8 0C44FB00 | mov eax,09.FB440C          | eax:"what is", FB440C:"what is" |
| 00FB100D | . 8D49 00     | lea ecx,dword ptr ds:[ecx] |                                 |
| 00FB1010 | > 8A10        | mov dl,byte ptr ds:[eax]   | eax:"what is"                   |
| 00FB1012 | . 3A11        | cmp dl,byte ptr ds:[ecx]   |                                 |
| 00FB1014 | . 75 1A       | jne 09.FB1030              |                                 |
| 00FB1016 | . 84D2        | test dl,dl                 |                                 |
| 00FB1018 | . 74 12       | je 09.FB102C               |                                 |
| 00FB101A | . 8A50 01     | mov dl,byte ptr ds:[eax+1] | eax+1:"hat is"                  |
| 00FB101D | . 3A51 01     | cmp dl,byte ptr ds:[ecx+1] |                                 |
| 00FB1020 | . 75 0E       | jne 09.FB1030              |                                 |
| 00FB1022 | . 83C0 02     | add eax,2                  | eax:"what is"                   |
| 00FB1025 | . 83C1 02     | add ecx,2                  |                                 |
| 00FB1028 | . 84D2        | test dl,dl                 |                                 |
| 00FB102A | . 75 E4       | jne 09.FB1010              |                                 |
| 00FB102C | > 33C0        | xor eax,eax                | eax:"what is"                   |
| 00FB102E | . EB 05       | jmp 09.FB1035              |                                 |
| 00FB1030 | > 1BC0        | sbb eax,eax                | eax:"what is"                   |

만약 다르다면 sbb구문으로 점프를 하게 된다.



| Address  | Disassembly   | Comment   |
|----------|---------------|---|
| 00FB1016 | 84D2          | test dl,dl  |
| 00FB1018 | 74 12         | je 09.FB102C  |
| 00FB101A | 8A50 01       | mov dl,byte ptr ds:[eax+1]                          |
| 00FB101D | 3A51 01       | cmp dl,byte ptr ds:[ecx+1]                          |
| 00FB1020 | 75 0E         | jne 09.FB1030                                       |
| 00FB1022 | 83C0 02       | add eax,2   |
| 00FB1025 | 83C1 02       | add ecx,2   |
| 00FB1028 | 84D2          | test dl,dl  |
| 00FB102A | 75 E4         | jne 09.FB1010                                       |
| 00FB102C | 33C0          | xor eax,eax   |
| 00FB102E | EB 05         | jmp 09.FB1035                                       |
| 00FB1030 | 1BC0          | sbb eax,eax   |
| 00FB1032 | 83D8 FF       | sbb eax,FFFFFFFF                                    |
| 00FB1035 | 8B0D 0444FB00 | mov ecx,dword ptr ds:[FB4404]                       |
| 00FB103B | 8B15 5030FB00 | mov edx,dword ptr ds:[<?end1@std@YAAAV?\$basic_c... |

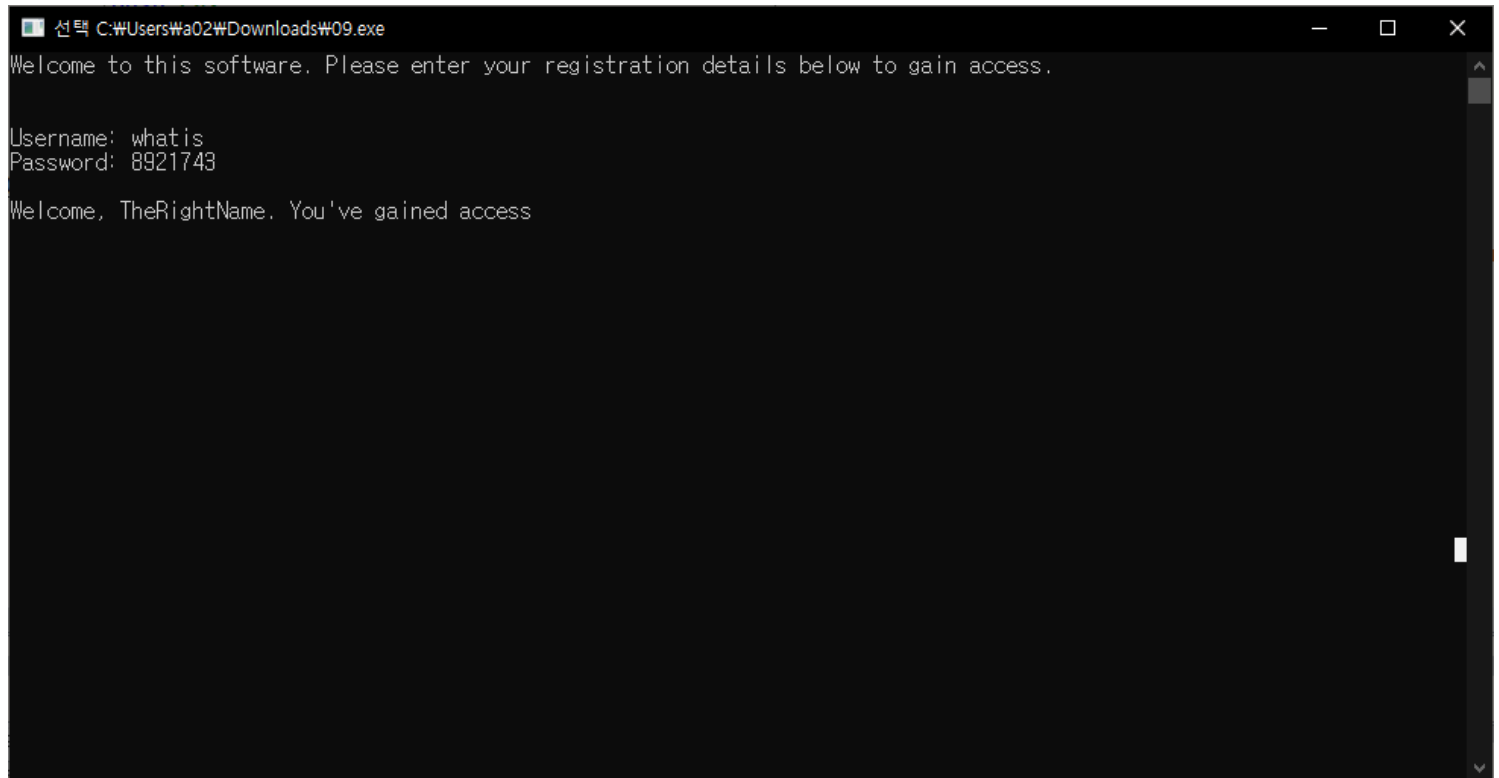
0x0이 삽입되어 있지 않다면, cmp구문으로 비교하면 다른 값이기 때문에 ZF가 0이 되고 sbb로 점프하게 되며, ZF를 세팅하는 구문이 더이상 없게 된다.

하지만

0x0이 삽입되어 있다면 xor eax,eax구문으로 점프를 하게 되고, 아래 jmp구문으로 sbb구문을 건너 뛰게 해준다.

ZF는 설정하는 구문이 없으니 계속 1로 세팅이 되어있을 것이고, 아래의 test bl, bl구문을 통과할 수 있게 된다.

이렇게 프로그램을 전부 실행하게 되면,



```
선택 C:\Users\Wa02\Downloads\W09.exe
Welcome to this software. Please enter your registration details below to gain access.

Username: whatis
Password: 8921743

Welcome, TheRightName. You've gained access
```

username의 실제 값이 무엇이던, 맨 앞에 0x0의 값만 있으면 이렇게 성공 문자열을 출력 해주게 된다.

성공!