

Reverse2 L09 Report by vivaman

1. Level9

00401340 > C70488 8F228800 /MOV DWORD PTR DS:[EAX+ECX*4],88228F	→	시리얼부분,초기화(003F0000 부터 003F03F0 까지) 시리얼일력(88228Fh=8921743)
00401347 . 83C1 04 ADD ECX,4		
0040134A . 81F9 00010000 CMP ECX,100		
00401350 .^ 72 EE JNB SHORT ED48143D.00401340		
00401352 . 05 A0020000 ADD EAX,2A0	→	나중에 시리얼 값이 될 곳 :003F02A0 00404404 에 저장 DS:[00403138]=57 ('W') : 아래 "Welcome"의 "W"랑 비교해서 맞으면 "Welcome to this software.~~~" 출력 틀리면 그냥 통과
00401357 . A3 04444000 MOV DWORD PTR DS:[404404],EAX		
0040135C . 0FB605 38314000 MOVZX EAX,BYTE PTR DS:[403138]		
00401363 . 84C0 TEST AL,AL	→	Please enter your registration details below to gain access."
00401365 . 74 20 JE SHORT ED48143D.00401387		
00401367 . BE 38314000 MOV ESI,ED48143D.00403138 ; ASCII "Welcome to this software.		
0040136C . 8D6424 00 LEA ESP,DWORD PTR SS:[ESP]	→	"Welcome" 출력부분
00401370 > 50 /PUSH EAX		
00401371 . A1 78304000 JMOV EAX,DWORD PTR DS:[<&MSVCP90.?cout@std@@@3V?>		
00401376 . 50 /PUSH EAX		
00401377 . E8 04020000 JCALL ED48143D.00401580		
0040137C . 8A46 01 JMOV AL,BYTE PTR DS:[ESI+1]		
0040137F . 46 JINC ESI		
00401380 . 83C4 08 ADD ESP,8		
00401383 . 84C0 JTEST AL,AL		
00401385 .^ 75 E9 JNZ SHORT ED48143D.00401370		
00401387 > 8B0D 50304000 MOV ECX,DWORD PTR DS:[<&MSVCP90.?endl@std@@@YAAA>		
0040138D . 51 PUSH ECX ; /Arg1 => 7848BF50		
0040138E . 8B0D 78304000 MOV ECX,DWORD PTR DS:[<&MSVCP90.?cout@std@@@3V?>		
00401394 . FF15 44304000 CALL DWORD PTR DS:[<&MSVCP90.??6?basic_ostringstream>		
0040139A . 8B15 50304000 MOV EDX,DWORD PTR DS:[<&MSVCP90.?endl@std@@@YAAA>		
004013A0 . 8B0D 78304000 MOV ECX,DWORD PTR DS:[<&MSVCP90.?cout@std@@@3V?>		
004013A6 . 52 PUSH EDX ; /Arg1		
004013A7 . 8BC2 MOV EAX,EDX ;		
004013A9 . 50 PUSH EAX ; /Arg1 => 7848BF50		
004013AA . FF15 44304000 CALL DWORD PTR DS:[<&MSVCP90.??6?basic_ostringstream>		
004013B0 . 8BC8 MOV ECX,EAX ;		
004013B2 . FF15 44304000 CALL DWORD PTR DS:[<&MSVCP90.??6?basic_ostringstream>		
004013B8 . 8B0D 78304000 MOV ECX,DWORD PTR DS:[<&MSVCP90.?cout@std@@@3V?>		
004013BE . 68 24324000 PUSH ED48143D.00403224 ; ASCII "Username: "		
004013C3 . 51 PUSH ECX		
004013C4 . E8 57030000 CALL ED48143D.00401720		
004013C9 . 8B15 58304000 MOV EDX,DWORD PTR DS:[<&MSVCP90.?cin@std@@@3V?>b>		
004013CF . 68 0C444000 PUSH ED48143D.0040440C	→	0040440C 에 "Username: " 입력 받음
004013D4 . 52 PUSH EDX	→	"Username: " 실제로 입력 받는 CALL
004013D5 . FF15 54304000 CALL DWORD PTR DS:[<&MSVCP90.??5DU?char_trai>		
004013DB . A1 78304000 MOV EAX,DWORD PTR DS:[<&MSVCP90.?cout@std@@@3V?>;	→	0040442C 에 "Password: " 입력 받음
004013E0 . 68 30324000 PUSH ED48143D.00403230 ; Arg2 = 00403230 ASCII "Password: "		
004013E5 . 50 PUSH EAX ; Arg1 => 78505AC8		
004013E6 . E8 35030000 CALL ED48143D.00401720 ; WED48143D.00401720		
004013EB . 8B0D 58304000 MOV ECX,DWORD PTR DS:[<&MSVCP90.?cin@std@@@3V?>b>		
004013F1 . 83C4 18 ADD ESP,18	→	"Password: " 실제로 입력 받는 CALL
004013F4 . 68 2C444000 PUSH ED48143D.0040442C ; /Arg1 = 0040442C	→	"Username: ", "Password: " 비교 CALL문
004013F9 . FF15 3C304000 CALL DWORD PTR DS:[<&MSVCP90.??5?basic_istream>		
004013FF . E8 FCFBFFFF CALL ED48143D.00401000		
00401000 /\$ 51 PUSH ECX ; MSVCP90.784AD1D7		
00401001 . 53 PUSH EBX		
00401002 . 56 PUSH ESI		
00401003 . B9 F8314000 MOV ECX,ED48143D.004031F8	→	정답 "Username: " 이 입력되어 있는 곳
00401008 . B8 0C444000 MOV EAX,ED48143D.0040440C ; ASCII "vivaman"	→	[004031F8]=00 과 DL=76 ('v') 과 비교 하는 곳 그런데....비교할 이름이 없다.. 원래는 이름부분이 있었는데, "00"으로 바꾸신 듯.. 입력한 "vivaman" 부분의 첫 글자만 "00"으로 바꾸면 바로 통과 됩니다.
0040100D . 8D49 00 LEA ECX,DWORD PTR DS:[ECX]		
00401010 > 8A10 /MOV DL,BYTE PTR DS:[EAX]		
00401012 . 3A11 CMP DL,BYTE PTR DS:[ECX]		
00401014 . 75 1A JNZ SHORT ED48143D.00401030		
00401016 . 84D2 JTEST DL,DL		
00401018 . 74 12 JJE SHORT ED48143D.0040102C		
0040101A . 8A50 01 JMOV DL,BYTE PTR DS:[EAX+1]		
0040101D . 3A51 01 CMP DL,BYTE PTR DS:[ECX+1]		
00401020 . 75 0E JNZ SHORT ED48143D.00401030		
00401022 . 83C0 02 ADD EAX,2		
00401025 . 83C1 02 ADD ECX,2		
00401028 . 84D2 JTEST DL,DL		
0040102A .^ 75 E4 JNZ SHORT ED48143D.00401010		
0040102C > 33C0 XOR EAX,EAX		
0040102E . EB 05 JMP SHORT ED48143D.00401035		
00401030 > 1BC0 SBB EAX,EAX		
00401032 . 83D8 FF SBB EAX,-1		
00401035 > 8B0D 04444000 MOV ECX,DWORD PTR DS:[404404]	→	저장했던 정답 "Password"를 복사
0040103B . 8B15 50304000 MOV EDX,DWORD PTR DS:[<&MSVCP90.?endl@std@@@YAAA>		
00401041 . 85C0 TEST EAX,EAX	→	입력한 이름의 비교가 "참"이었으므로, Condition is TRUE BL=00 을 BL=01 로 셋팅
00401043 . A1 2C444000 MOV EAX,DWORD PTR DS:[40442C]	→	입력한 BABEh(47806)와 정답 88228Fh=8921743 비교
00401048 . 0F94C3 SETB BL		
0040104B . 3B01 CMP EAX,DWORD PTR DS:[ECX]	→	Condition is TRUE ;Stack SS:[0012FF6B]=78 ('x')
0040104D . 8B0D 78304000 MOV ECX,DWORD PTR DS:[<&MSVCP90.?cout@std@@@3V?>		
00401053 . 52 PUSH EDX ; /Arg1 => 7848BF50		
00401054 . 0F944424 0F SETB BYTE PTR SS:[ESP+F] ;		

00401059 |. FF15 44304000 CALL DWORD PTR DS:[<&MSVCP90.??6?\$basic_ostream>

정답 비교가 맞으면, 0012FF68=01 로 셋팅

```
0040105F |. B9 18324000 MOV ECX,ED48143D.00403218 ; ASCII "DonaldDuck"
00401064 |. B8 0C444000 MOV EAX,ED48143D.0040440C ; ASCII "vivaman"
00401069 |. 8DA424 00000000 LEA ESP,DWORD PTR SS:[ESP]
00401070 |> 8A10 /MOV DL,BYTE PTR DS:[EAX]
00401072 |. 3A11 |CMP DL,BYTE PTR DS:[ECX]
00401074 |. 75 1A |JNZ SHORT ED48143D.00401090
00401076 |. 84D2 |TEST DL,DL
00401078 |. 74 12 |JE SHORT ED48143D.0040108C
0040107A |. 8A50 01 |MOV DL,BYTE PTR DS:[EAX+1]
0040107D |. 3A51 01 |CMP DL,BYTE PTR DS:[ECX+1]
00401080 |. 75 0E |JNZ SHORT ED48143D.00401090
```

"DonaldDuck" 속임수이며,
"I can't believe you felt for that! xD" 출력..

"DonaldDuck" 비교 부분

```
00401082 |. 83C0 02 |ADD EAX,2
00401085 |. 83C1 02 |ADD ECX,2
00401088 |. 84D2 |TEST DL,DL
0040108A |.^ 75 E4 |WJNZ SHORT ED48143D.00401070
0040108C |> 33C0 XOR EAX,EAX
0040108E |. EB 05 |JMP SHORT ED48143D.00401095
00401090 |> 18C0 SBB EAX,EAX
00401092 |. 83D8 FF |SBB EAX,-1
00401095 |> 85C0 TEST EAX,EAX
```

00401097 |. 0F85 0E020000 JNZ ED48143D.004012AB

"DonaldDuck" 아니므로 점프~

0040109D |. A1 50304000 MOV EAX,DWORD PTR DS:[<&MSVCP90.7endl@st>

"DonaldDuck" 일 때,
"I can't believe you felt for that! xD" 출력..

004012A5 |. FF15 44304000 CALL DWORD PTR DS:[<&MSVCP90.??6?\$basic_ostream>

004012AB |> 84DB TEST BL,BL

00401054 |. 0F944424 0F SETE BYTE PTR SS:[ESP+F]
에서 셋팅했던 BL(01)로 셋팅했으므로 ..노점프..

004012AD |. 74 30 |JE SHORT ED48143D.004012DF

004012AF |. 807C24 0B 00 CMP BYTE PTR SS:[ESP+B],0

00401054 |. 0F944424 0F SETE BYTE PTR SS:[ESP+F]
에서 셋팅했던 BL(01)로 셋팅했으므로 ..노점프..

004012B4 |. 74 29 |JE SHORT ED48143D.004012DF

004012B6 |. 0FB605 C8314000 MOVZX EAX,BYTE PTR DS:[4031C8]

00401308 의 "Welcome, TheRightName. You've gained access!"
중에서 "W" 인지 확인..노점프

004012BD |. 84C0 |TEST AL,AL

004012BF |. 74 47 |JE SHORT ED48143D.00401308

```
004012C1 |. BE C8314000 MOV ESI,ED48143D.004031C8; ASCII "Welcome, TheRightName. You've gained access!"
004012C6 |> 50 /PUSH EAX
004012C7 |. A1 78304000 |MOV EAX,DWORD PTR DS:[<&MSVCP90.?cout@std@@@3V?>
004012CC |. 50 |PUSH EAX
004012CD |. E8 AE020000 |CALL ED48143D.00401580
004012D2 |. 8A46 01 |MOV AL,BYTE PTR DS:[ESI+1]
004012D5 |. 46 |JINC ESI
004012D6 |. 83C4 08 |ADD ESP,8
004012D9 |. 84C0 |TEST AL,AL
004012DB |.^ 75 E9 |WJNZ SHORT ED48143D.004012C6
004012DD |. EB 29 |JMP SHORT ED48143D.00401308
```

"Welcome, TheRightName. You've gained access!"
출력

004012DF |> 0FB605 90314000 MOVZX EAX,BYTE PTR DS:[403190]

00401308 의 "Sorry, but the username and/or password is incorrect!"
중에서 "S" 인지 확인.

004012E6 |. 84C0 |TEST AL,AL

004012E8 |. 74 1E |JE SHORT ED48143D.00401308

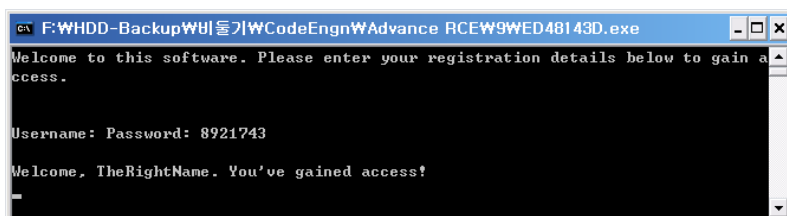
```
004012EA |. BE 90314000 MOV ESI,ED48143D.00403190; ASCII "Sorry, but the username and/or password is incorrect!"
004012EF |. 90 |NOP
004012F0 |> 8B0D 78304000 /MOV ECX,DWORD PTR DS:[<&MSVCP90.?cout@std@@@3V?>
004012F6 |. 50 |PUSH EAX
004012F7 |. 51 |PUSH ECX
004012F8 |. E8 83020000 |CALL ED48143D.00401580
004012FD |. 8A46 01 |MOV AL,BYTE PTR DS:[ESI+1]
00401300 |. 46 |JINC ESI
00401301 |. 83C4 08 |ADD ESP,8
00401304 |. 84C0 |TEST AL,AL
00401306 |.^ 75 E8 |WJNZ SHORT ED48143D.004012F0
```

"Sorry, but the username and/or password is incorrect!"
출력 부분

➤ ***** 분석은 다 끝났다. 그런데..... 새로운 문제가 생겼다..
확인해야 할 이름 부분이 "00" 이라는 것이다.

➤ 첫 번째로,
여러 가지 방법이 있을 수 있지만, 입력 되는 이름 주소가 0040440C 가 되므로,
0040440D 에 입력을 받으면 되겠다.
어차피, 0040440C 의 한글자만 "00"이면 만사 "OK" 이다.
004013CF |. 68 0C444000 PUSH ED48143D.0040440C 를
004013CF |. 68 0C444000 PUSH ED48143D.0040440D 로 바꾼다

➤ 두 번째로,
또는, 실제로 이름을 입력 받는 CALL 을 NOP 시켜 버리면.
004013D5 |. FF15 54304000 CALL DWORD PTR DS:[<&MSVCP90.??\$?5DU?\$char_traits> 를 NOP 처리..



"Username: " 입력 부분을 그냥 넘어 가고, 바로 "Password: "입력 부분으로 넘어 갈 수 있다. (화면은 좀,,,그렇다.ㅠㅠ)

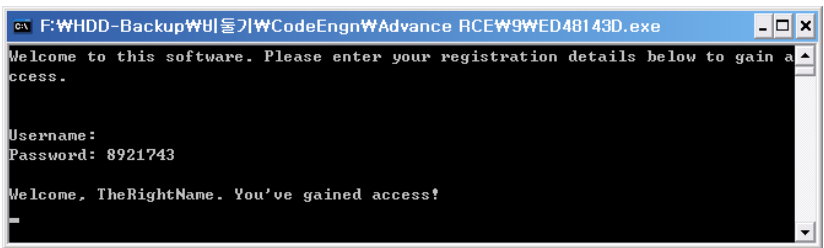
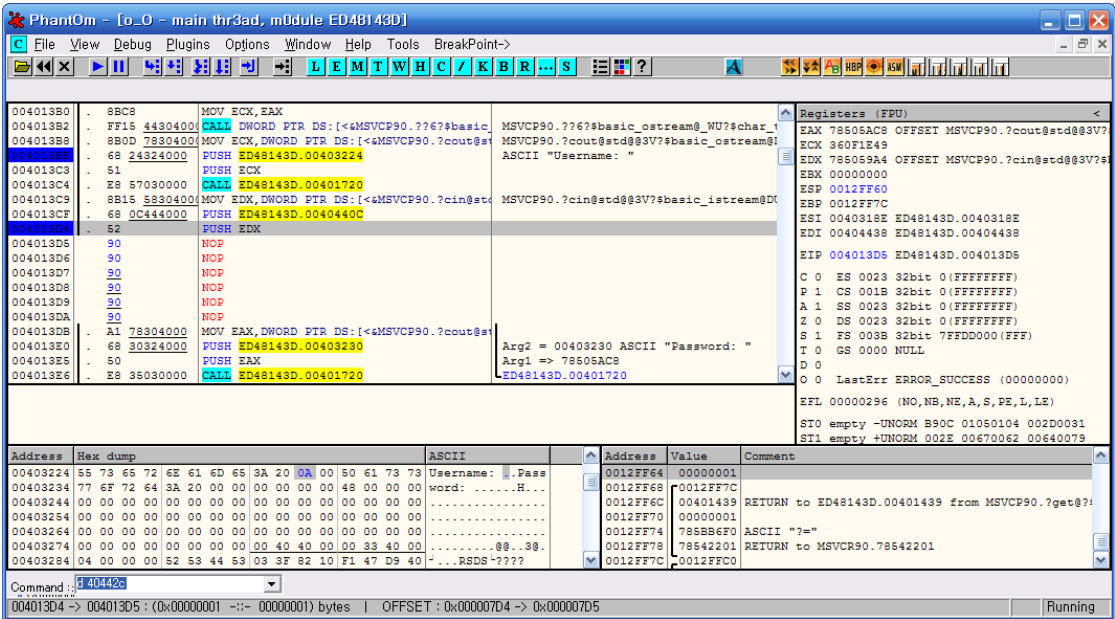
세 번째로,

"Username: ", "Password: " 입력 부분을 비슷하게 만들어 보면..

00403224 55 73 65 72 6E 61 6D 65 3A 0D 0A Username:..

또는..

00403224 55 73 65 72 6E 61 6D 65 3A 20 0A Username:.. <= 모양으로는 이 경우가 완전 똑같음.



"Username: " 뒷부분에 "0D 0A", "0A"를 붙여서 입력하면, "Username: "은 건너뛰고 바로 "Password: " 입력부분이 나타난다. 그럼, "0D", "0A"은 무엇인가?..

Dec Hex Oct Char
10 A 012 LF (NL line feed, new line)
13 D 015 CR (carriage return)

*후기
비교할 이름이 없어서 당황...

*참고
char c;
cin << c;
위 코드는 입력에서 white space (빈칸, tab, 등등) 을 무시하고
가장 처음 오는 char형태의 입력을 c에 저장합니다.

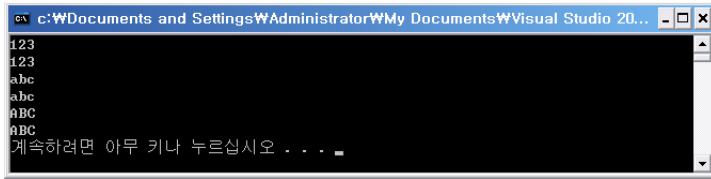
char c;
cin.get(c);
위 코드는 빈칸이 입력되면 빈칸 자체를 c에 읽어들이니다.

• "Hello world"라는 입력시...

char input[32];
cin >> input;
input에는 "Hello"만 들어갑니다. 즉 "Hello" 다음의 white space를 감지하고 뒷부분 무시.
(단, 다음에 오는 cin >> 에 밀립니다.)

char input[32];
cin.get(input,32);
input에는 "Hello world"가 모두 들어갑니다.

char s[10];
cin.getline(s, 10);
cout << s << endl;
cin.getline(s, 10);
cout << s << endl;
cin.getline(s, 10);
cout << s << endl;



```
c:\Documents and Settings\Administrator\My Documents\Visual Studio 20...
123
123
abc
abc
0BC
0BC
계속하려면 아무 키나 누르십시오 . . .
```

getline함수는 문자열을 처리하는 기본함수인데 사용법은 get함수와 완전히 동일합니다.
다만, 이 두 함수의 차이점은 단지 종료문자를 가지느냐 마느냐 하는 것 뿐입니다.
(getline는 종료문자를 읽으면 그 때의 처리에만 적용할 뿐 이 후에는 더 이상 가지지 않습니다.)

출처: 구글링..

-끝-