Codeengn Challenges Advance RCE LEVEL10 풀이

Reverse2 L10 Start

Author : qHF;

Korea :

Serial이 WWWCCCJJJRRR 일때 Name은 무엇인가

Hint 1 : 4글자임

Hint 2 : 정답으로 나올 수 있는 문자열 중 (0~9, a~z, A~Z) 순서상 가장 먼저 오는 문자열

English :

Find the Name when the Serial is WWWCCCJJJRRR

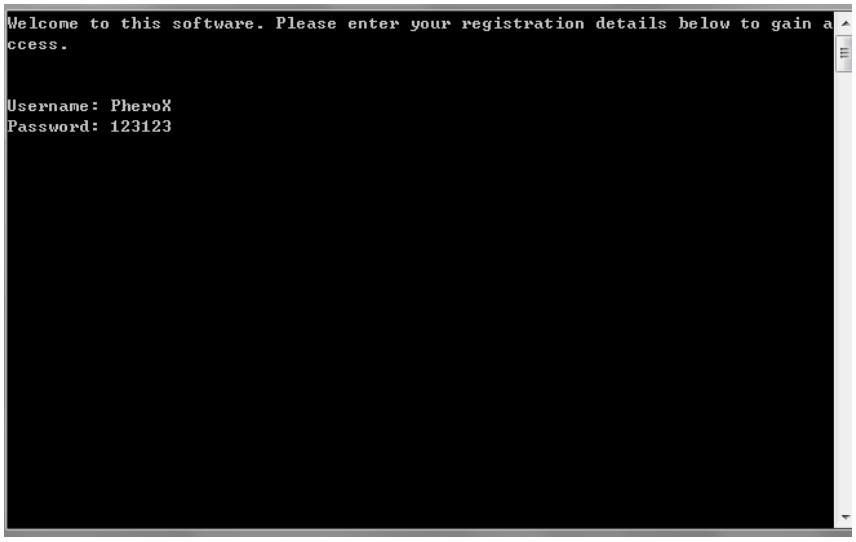
Hint 1 : Name is 4 letters

Hint 2 : Among the multiple solutions, find the one that comes first in alphabnumeric order.

Downl

총 13 분이 이 문제를 푸셨습니다. / 13 people solved this problem.

실행을 해보니 이미 정해진 Name과 Password를 입력하는 프로그램 인것 같습니다.



PEID로 보니 아무것도 나타나지 않아 바로 올리로 분석을 해보았습니다.

```
01241A1B   .  FF15 4444240 CALL DWORD PTR DS:[1244444]
01241A21   >  A1 B4402401  MOV EAX,DWORD PTR DS:[12440B4]
01241A26   .  8B0D CC3024  MOV ECX,DWORD PTR DS:[<&MSVCR90.__initenv>]
01241A2C   .  8901         MOV DWORD PTR DS:[ECX],EAX
01241A2E   .  FF35 B44024  PUSH DWORD PTR DS:[12440B4]
01241A34   .  FF35 B84024  PUSH DWORD PTR DS:[12440B8]
01241A3A   .  FF35 B04024  PUSH DWORD PTR DS:[12440B0]
01241A40   .  E8 DBF8FFFF  CALL Reverse2.01241320
01241A45   .  83C4 0C      ADD ESP,0C
01241A48   .  A3 C8402401  MOV DWORD PTR DS:[12440C8],EAX
01241A4D   .  391D BC4024  CMP DWORD PTR DS:[12440BC],EBX
```

쭉 Step Over로 진행을 하다가 다음함수에서 입력, 출력 등을 처리 한다는걸 알았습니다.

Step Into로 함수 내부를 살펴보겠습니다.

```
01241320  r$  55           PUSH EBP
01241321  .   8BEC         MOV EBP,ESP
01241323  .   83E4 F8      AND ESP,FFFFFFF8
01241326  .   51           PUSH ECX
01241327  .   56           PUSH ESI
01241328  .   6A 04        PUSH 4                                  ; Protect = PAGE_READW
0124132A  .   68 00300000  PUSH 3000                               ; AllocationType = MEM
0124132F  .   68 00010000  PUSH 100                                ; Size = 100 (256.)
01241334  .   6A 00        PUSH 0                                  ; Address = NULL
01241336  .   FF15 0030240 CALL DWORD PTR DS:[<&KERNEL32.VirtualAlloc>] ; VirtualAlloc
0124133C  .   33C9         XOR ECX,ECX
0124133E  .   8BFF         MOV EDI,EDI                             ; Reverse2.01244438
01241340  >   C780 88F228  MOV DWORD PTR DS:[EAX+ECX+4],88228F
01241347  .   83C1 04      ADD ECX,4
0124134A  .   81F9 0001000 CMP ECX,100
01241350  .^  72 EE        JB SHORT Reverse2.01241340
01241352  .   05 A0020000  ADD EAX,2A0
01241357  .   A3 04442401  MOV DWORD PTR DS:[1244404],EAX
0124135C  .   0FB605 38312 MOVZX EAX,BYTE PTR DS:[1243138]
```

함수 시작후 얼마되지않아 VirtualAlloc이라는 함수를 이용해 Heap영역에 0x100사이즈만큼 88228F를 넣어주고있습니다.

그리고 EAX+240의 값을 124404에 넣어줍니다. 같은 값을 계속넣어주는게 의심스러워 일단 124404에 Hardware Break point를걸어 값을 참조하거나 변경할때 뭠추

도록 설정해주었습니다.
그리고 진행시켜보니

```
012413F9  .  FF15 3C302401 CALL DWORD PTR DS:[<&MSVCP90.??5?$basic_istream@DU?$char_traits@D@std@@@std@@QAEAAV01@AAH@Z>]
012413FF  .  E8 FCFBFFFF   CALL Reverse2.01241000
01241404  .  A1 58302401   MOV EAX,DWORD PTR DS:[<&MSVCP90.?cin@std@@3V?$basic_istream@DU?$char_traits@D@std@@@1@A>]
01241409  .  8B08          MOV ECX,DWORD PTR DS:[EAX]
```

다음 함수를 거쳐

```
01241043  .  A1 2C442401   MOV EAX,DWORD PTR DS:[124442C]
01241048  .  0F94C3        SETE BL
0124104B  .  3B01          CMP EAX,DWORD PTR DS:[ECX]
0124104D  .  8B0D 7830240  MOV ECX,DWORD PTR DS:[<&MSVCP90.?couti
01241053  .  52            PUSH EDX
01241054  .  0F944424 0F   SETE BYTE PTR SS:[ESP+F]
01241059  .  FF15 4430240  CALL DWORD PTR DS:[<&MSVCP90.??6?$bas
0124105F  .  B9 18322401   MOV ECX,Reverse2.01243218
01241064  .  B8 0C442401   MOV EAX,Reverse2.0124440C
01241069  .  8DA424 00000  LEA ESP,DWORD PTR SS:[ESP]
01241070  >  8A10          MOV DL,BYTE PTR DS:[EAX]
01241072  .  3A11          CMP DL,BYTE PTR DS:[ECX]
01241074  .v 75 1A         JNZ SHORT Reverse2.01241090
01241076  .  84D2          TEST DL,DL
01241078  .v 74 12         JE SHORT Reverse2.0124108C
0124107A  .  8A50 01       MOV DL,BYTE PTR DS:[EAX+1]
0124107D  .  3A51 01       CMP DL,BYTE PTR DS:[ECX+1]
01241080  .v 75 0E         JNZ SHORT Reverse2.01241090
01241082  .  83C0 02       ADD EAX,2
DS:[000D02A0]=0088228F
EAX=000004D2
```

제가 넣은 1234(0x4D2)와 88228F를 비교해주는 곳 위에서 뭠췄습니다.
Key는 찾은것 같네요 :D