

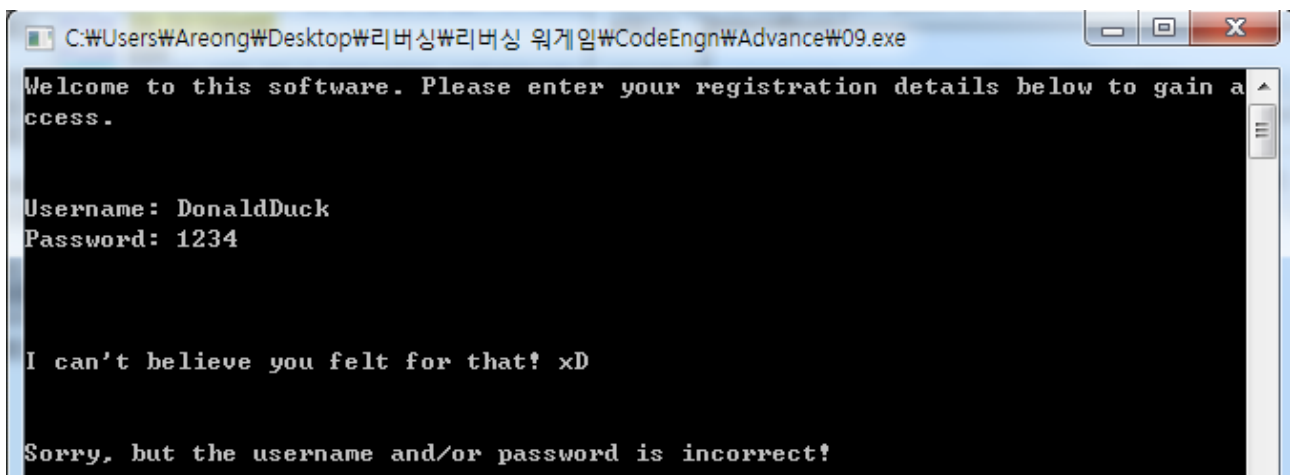
이름과 패스워드를 입력해서 인증하는 프로그램입니다.

그런데 문제는 패스워드만 찾으라고 했는데 왜 이름과 패스워드를 입력하는 기능이 있습니다.

Address	Hex dump	ASCII
01194080	54 68 69 73 49 73 54 68 65 55 73 65 72 6E 61 6D	ThisIsTheUsernam
01194090	65 3A 20 44 6F 6E 61 6C 64 44 75 63 68 00 00 00	e: <u>DonaldDuck...</u>

프로그램을 실행하고 Hex dump에 몇가지 문자열중에 찾아낸 문자열입니다.

ThisIsTheUsername: DonaldDuck 라고 되어 있습니다.



살짝 달라진 프로그램의 출력을 볼 수 있습니다.

DonaldDuck이 Username인 것이 확실히 되었습니다.

마저 분석을 하도록 하겠습니다.

CPU - main thread, module 09			
01191320	55	PUSH EBP	
01191321	8BEC	MOV EBP,ESP	
01191323	83E4 F8	AND ESP,FFFFFFF8	
01191326	51	PUSH ECX	
01191327	56	PUSH ESI	
01191328	6A 04	PUSH 4	
0119132A	68 00300000	PUSH 3000	
0119132F	68 00010000	PUSH 100	
01191334	6A 00	PUSH 0	
01191336	FF15 0030190	CALL DWORD PTR DS:[<&KERNEL32.VirtualAlloc	Protect = PAGE_READWRITE AllocationType = MEM_COMMIT MEM_RESERVE Size = 100 (256.) Address = NULL VirtualAlloc
0119133C	33C9	XOR ECX,ECX	
0119133E	8BFF	MOV EDI,EDI	
01191340	C70488 8F228F	MOV DWORD PTR DS:[EAX+ECX*4],88228F	
01191347	83C1 04	ADD ECX,4	
0119134A	81F9 00010000	CMP ECX,100	
01191350	72 EE	JB SHORT 09.01191340	
01191352	05 A0020000	ADD EAX,2A0	
01191357	A3 04441901	MOV DWORD PTR DS:[1194404],EAX	
0119135C	0FB605 383119	MOVZX EAX,BYTE PTR DS:[1193138]	
01191363	84C0	TEST AL,AL	
01191365	74 20	JE SHORT 09.01191387	
01191367	BE 38311901	MOV ESI,09.01193138	ASCII "Welcome to this software. Please enter y
0119136C	8D6424 00	LEA ESP,DWORD PTR SS:[ESP]	
01191370	50	PUSH EAX	
01191371	A1 78301901	MOV EAX,DWORD PTR DS:[<&MSVCP90.?cout@	
01191376	50	PUSH EAX	
01191377	E8 04020000	CALL 09.01191580	

Jump is taken
01191340=09.01191340

256 byte의 공간을 확보하고 0x0088228F의 값을 특정주소에 넣는 것을 알 수 있습니다.

이는 Heap Spray라는 기법으로 CodeEngn의 아카이브에 검색하면 많은 자료를 볼 수 있습니다.

Heap Spray기법은 아무 의미없는 명령어나 노슬레이드를 힙에 꽂 채우는 것입니다.

그 후 중간에 셸코드를 삽입하고 취약점이 있는 코드를 통해 힙으로 실행을 옮겨오도록 하는 기법입니다.

간단하게 말하면 실행흐름을 바꿔 원하는 데로 동작하게 하는 기법입니다.

C:\Users\Areong\Desktop\리버싱\리버싱 워크\임CodeEngn\Advance\09.exe			
Welcome to this software. Please enter your registration details below to gain access.			
Username: DonaldDuck			
Password: 1234			
013B1048	3B01	CMP EAX,DWORD PTR DS:[ECX]	
013B104D	8B0D 78303B0	MOV ECX,DWORD PTR DS:[<&MSVCP90.?cout@std@@	MSVCP90.?cout@std@@
013B1053	52	PUSH EDX	
013B1054	0F944424 0F	SETB BYTE PTR SS:[ESP+4]	
013B1059	FF15 44303B0	CALL DWORD PTR DS:[<&MSVCP90.??6?\$basic_	MSVCP90.??6?\$basic_
013B105F	B9 18323B01	MOV ECX,09.013B3218	ASCII "DonaldDuck"
013B1064	B8 0C443B01	MOV EAX,09.013B440C	ASCII "DonaldDuck"
013B1069	8D4424 000000	LEA ESP,DWORD PTR SS:[ESP]	
013B1070	8A10	MOV DL,BYTE PTR DS:[EAX]	
013B1072	3A11	CMP DL,BYTE PTR DS:[ECX]	
013B1074	75 1A	JNZ SHORT 09.013B1090	

DS:[003F02A0]=0088228F
EAX=000004D2

패스워드를 '1234'로 입력한 뒤 그 HEX값인 4D2와 0x0088228F를 비교합니다.

위에서 Heap Spray를 통해 삽입한 값을 패스워드로 사용하는 것을 알 수 있습니다.

지금은 패스워드 인증으로만 사용되었지만 그 지점에 셸코드가 존재하게 되면 이를 이용한 공격이 가능합니다.

CPU - main thread, module 09			
0119105F	. B9 18321901	MOV ECX,09.01193218	ASCII "DonaldDuck"
01191064	. B8 0C441901	MOV EAX,09.0119440C	ASCII "DonaldDuck"
01191069	. 8DA424 000000	LEA ESP,DWORD PTR SS:[ESP]	
01191070	> 8A10	MOV DL,BYTE PTR DS:[EAX]	
01191072	. 3A11	CMP DL,BYTE PTR DS:[ECX]	
01191074	. 75 1A	JNZ SHORT 09.01191090	
01191076	. 84D2	TEST DL,DL	
01191078	. 74 12	JE SHORT 09.0119108C	
0119107A	. 8A50 01	MOV DL,BYTE PTR DS:[EAX+1]	
0119107D	. 3A51 01	CMP DL,BYTE PTR DS:[ECX+1]	
01191080	. 75 0E	JNZ SHORT 09.01191090	
01191082	. 83C0 02	ADD EAX,2	
01191085	. 83C1 02	ADD ECX,2	
01191088	. 84D2	TEST DL,DL	
0119108A	. 75 E4	JNZ SHORT 09.01191070	

여기는 입력한 이름이 'DonaldDuck' 인지 확인하는 루틴입니다.

1대1 비교를 통해 확인합니다.

013B12AD	. 74 30	JE SHORT 09.013B12DF	
013B12AF	. 807C24 0B 00	CMP BYTE PTR SS:[ESP+B],0	
013B12B4	. 74 29	JE SHORT 09.013B12DF	
013B12B6	. 0FB605 C83138	MOVZX EAX,BYTE PTR DS:[13B31C8]	
013B12BD	. 84C0	TEST AL,AL	
013B12BF	. 74 47	JE SHORT 09.013B1308	
013B12C1	. BE C8313B01	MOV ESI,09.013B31C8	ASCII "Welcome, TheRightName. You've gained access!"
013B12C6	. 50	PUSH EAX	
013B12C7	. A1 78303B01	MOV EAX,DWORD PTR DS:[<&MSVCP90.?cout@	
013B12CC	. 50	PUSH EAX	
013B12CD	. E8 AE020000	CALL 09.013B1580	
013B12D2	. 8A46 01	MOV AL,BYTE PTR DS:[ESI+1]	
013B12D5	. 46	INC ESI	
013B12D6	. 83C4 08	ADD ESP,8	
013B12D9	. 84C0	TEST AL,AL	
013B12DB	. 75 E9	JNZ SHORT 09.013B12C6	
013B12DD	. EB 29	JMP SHORT 09.013B1308	
013B12DF	. 0FB605 903138	MOVZX EAX,BYTE PTR DS:[13B3190]	

Jump is taken
013B12DF=09.013B12DF

앞서 분석한 것을 토대로 올바른 이름과 패스워드를 입력 해보았습니다.

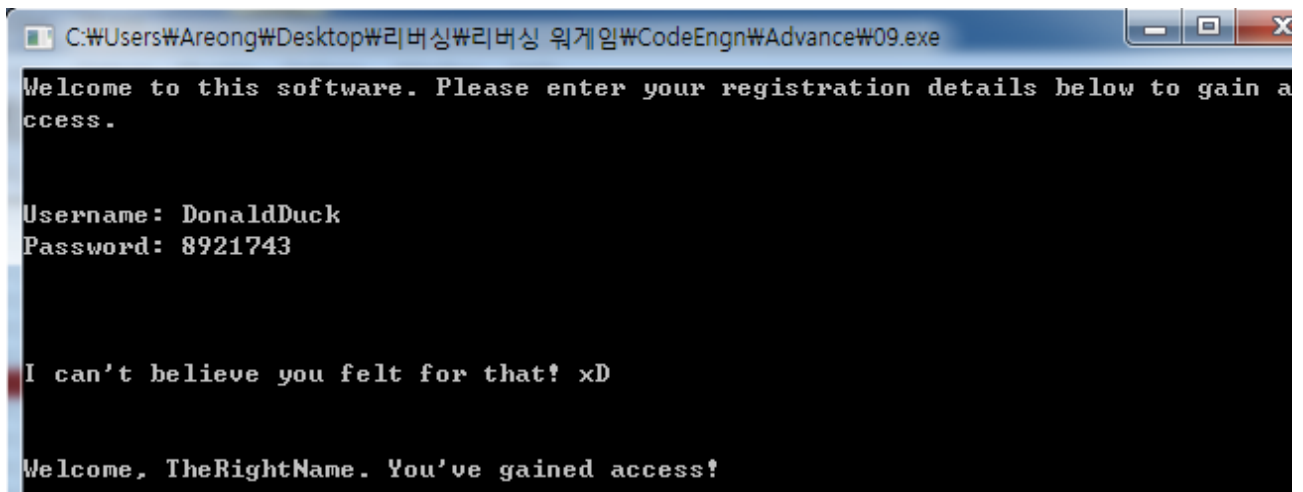
하지만 위의 그림처럼 분기가 엉뚱한 곳으로 진행되는 것을 알 수 있습니다.

TEST BL, BL을 통해 분기하게 되는데 BL은 프로그램이 진행되면서 0의 값을 가지고 바뀌질 않습니다.

그 이유로 항상 오류 메시지를 향해 분기를 하는 것입니다.

013B12A5	. FF15 44303B01	CALL DWORD PTR DS:[<&MSVCP90.??6?basic_ostream@_WU?\$char_traits@_W@std@@@	MSVCP90.??6?basic_ostream@_WU?\$char_traits@_W@std@@@
013B12AB	> 84DB	TEST BL,BL	
013B12AD	. 90	NOP	
013B12AE	. 90	NOP	
013B12AF	. 807C24 0B 00	CMP BYTE PTR SS:[ESP+B],0	
013B12B4	. 74 29	JE SHORT 09.013B12DF	
013B12B6	. 0FB605 C83138	MOVZX EAX,BYTE PTR DS:[13B31C8]	
013B12BD	. 84C0	TEST AL,AL	
013B12BF	. 74 47	JE SHORT 09.013B1308	
013B12C1	. BE C8313B01	MOV ESI,09.013B31C8	ASCII "Welcome, TheRightName. You've gained access!"

그래서 분기하는 부분을 NOP으로 채우고 실행해보도록 하겠습니다.



The screenshot shows a Windows command prompt window with the title bar "C:\Users\Areong\Desktop\리버싱 워게임\CodeEngn\Advance\09.exe". The text inside the window is as follows:

```
Welcome to this software. Please enter your registration details below to gain access.

Username: DonaldDuck
Password: 8921743

I can't believe you felt for that! xD

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

점프문을 NOP으로 하고 진행한 것이지만 인증하는데 문제는 없습니다.