

Anti_Baek

주어진 프로그램을 실행하면



다음과 같은 버튼이 나온다 'NO'를 누르면 프로그램이 종료되고 YES를 누르면



합격이라는 사진이 등장한다.

이제 디버거로 실행을 해서 YES를 누르면



영역을 전개한 백종원이 나온다. 우리는 영역에 당한것이다...

디버거로 실행하면 다른 사진이 등장하므로 안티 디버깅 기법이 있다는 것을 알 수 있다.

힌트로 주어진 IsDebuggerPresent 함수를 찾아보면

| | | | |
|------------------|------------------------|---|------------------------------|
| 00007FF66C061B38 | E9 A7060000 | jmp anti_baek.7FF66C0621E4 | Anti_Baek.cpp:133 |
| 00007FF66C061B3D | 48:88424 B0020000 | mov rax,qword ptr ss:[rsp+2B0] | |
| 00007FF66C061B45 | 48:25 FFFF0000 | and rax,FFFF | |
| 00007FF66C061B48 | 0FB7C0 | movzx eax,ax | |
| 00007FF66C061B4E | 894424 7C | mov dword ptr ss:[rsp+7C],eax | Anti_Baek.cpp:134 |
| 00007FF66C061B52 | 8B4424 7C | mov eax,dword ptr ss:[rsp+7C] | |
| 00007FF66C061B56 | 894424 38 | mov dword ptr ss:[rsp+38],eax | |
| 00007FF66C061B5A | 817C24 38 E9030000 | cmp dword ptr ss:[rsp+38],3E9 | |
| 00007FF66C061B62 | 74 13 | je anti_baek.7FF66C061B77 | |
| 00007FF66C061B64 | 817C24 38 EA030000 | cmp dword ptr ss:[rsp+38],3EA | |
| 00007FF66C061B6C | 0F84 4F030000 | je anti_baek.7FF66C061EC1 | |
| 00007FF66C061B72 | E9 55030000 | jmp anti_baek.7FF66C061EC0 | |
| 00007FF66C061B77 | FF15 98240000 | call qword ptr ds:[<IsDebuggerPresent>] | Anti_Baek.cpp:137 |
| 00007FF66C061B7D | 8905 194C0000 | mov dword ptr ds:[<int is_debug>],eax | |
| 00007FF66C061B83 | 833D 124C0000 00 | cmp dword ptr ds:[<int is_debug>],0 | |
| 00007FF66C061B8A | 0F84 81000000 | je anti_baek.7FF66C061C11 | |
| 00007FF66C061B90 | 33D2 | xor edx,edx | Anti_Baek.cpp:138 |
| 00007FF66C061B92 | 48:8D0D 7F2D0000 | lea rcx,qword ptr ds:[7FF66C064918] | 00007FF66C064918:L"baek.png" |
| 00007FF66C061B99 | E8 C2F8FFFF | call <anti_baek.public: static class Gdiplus:: | |
| 00007FF66C061B9E | 48:8905 C34C0000 | mov qword ptr ds:[<class Gdiplus::Image *gpImag | |
| 00007FF66C061BA5 | C705 E9480000 01000000 | mov dword ptr ds:[<int is_button_click>],1 | Anti_Baek.cpp:139 |
| 00007FF66C061BAF | BA E9030000 | mov edx,3E9 | Anti_Baek.cpp:140 |
| 00007FF66C061BB4 | 48:8B8C24 A0020000 | mov rcx,qword ptr ss:[rsp+2A0] | |
| 00007FF66C061BBC | FF15 56250000 | call qword ptr ds:[<GetDlgItem>] | |
| 00007FF66C061BC2 | 48:8BC8 | mov rcx,rcx | |
| 00007FF66C061BC5 | FF15 F5240000 | call qword ptr ds:[<DestroyWindow>] | |
| 00007FF66C061BCB | BA EA030000 | mov edx,3EA | Anti_Baek.cpp:141 |

해당 위치에 존재하고 이 결과값(eax)을 1 → 0으로 바꿔주면 안티디버깅이 우회가 된다.

이후 그대로 실행을 해보면

| | | | |
|------------------|-----------------------|--|----------------------------|
| 00007FF66C061E02 | 48:899424 A0000000 | mov qword ptr ss:[rsp+A0],rcx | [rsp+A0]: "SECRET" |
| 00007FF66C061E03 | 48:C74424 50 FFFFFFFF | mov qword ptr ss:[rsp+50],FFFFFFFFFFFFFFFF | |
| 00007FF66C061E08 | 48:FF4424 50 | inc qword ptr ss:[rsp+50] | |
| 00007FF66C061E10 | 48:889424 A0000000 | mov rdx,qword ptr ss:[rsp+A0] | |
| 00007FF66C061E15 | 48:8B7C24 50 | mov rdi,qword ptr ss:[rsp+50] | |
| 00007FF66C061E19 | 803C3A 00 | cmp byte ptr ds:[rdi+rdi],0 | |
| 00007FF66C061E1B | 75 E8 | jne anti_baek.7FF66C061E03 | |
| 00007FF66C061E18 | 48:885424 50 | mov rdx,qword ptr ss:[rsp+50] | |
| 00007FF66C061E20 | 48:899424 A8000000 | mov qword ptr ss:[rsp+A8],rdx | |
| 00007FF66C061E28 | 33D2 | xor edx,edx | |
| 00007FF66C061E2A | 48:88C1 | mov rax,rcx | |
| 00007FF66C061E2D | 48:888C24 A8000000 | mov rcx,qword ptr ss:[rsp+A8] | |
| 00007FF66C061E35 | 48:F7F1 | div rcx | |
| 00007FF66C061E38 | 48:88C2 | mov rax,rdx | |
| 00007FF66C061E38 | 48:8D0D 72420000 | lea rcx,qword ptr ds:[<char "key">] | 00007FF66C0660B4: "SECRET" |
| 00007FF66C061E42 | 0FB80401 | movsx eax,byte ptr ds:[rcx+rax] | |
| 00007FF66C061E46 | 888C24 80000000 | mov ecx,dword ptr ss:[rsp+80] | |
| 00007FF66C061E4D | 33C8 | xor ecx,eax | |
| 00007FF66C061E4F | 88C1 | mov eax,ecx | |
| 00007FF66C061E51 | 48:634C24 30 | movsxd rcx,dword ptr ss:[rsp+30] | |
| 00007FF66C061E56 | 88840C 80010000 | mov byte ptr ss:[rsp+rcx+180],al | |
| 00007FF66C061E5D | E9 22FFFFFF | jmp anti_baek.7FF66C061D84 | Anti_Baek.cpp:153 |
| 00007FF66C061E62 | BA E9030000 | mov edx,3E9 | Anti_Baek.cpp:154 |
| 00007FF66C061E67 | 48:8B8C24 A0020000 | mov rcx,qword ptr ss:[rsp+2A0] | |
| 00007FF66C061E6F | FF15 A3220000 | call qword ptr ds:[<&GetDlgItem>] | |
| 00007FF66C061E75 | 48:8BC8 | mov rcx,rax | |
| 00007FF66C061E78 | FF15 42220000 | call qword ptr ds:[<&DestroyWindow>] | |
| 00007FF66C061E7E | BA EA030000 | mov edx,3EA | |
| 00007FF66C061E83 | 48:8B8C24 A0020000 | mov rcx,qword ptr ss:[rsp+2A0] | |
| 00007FF66C061E8B | FF15 87220000 | call qword ptr ds:[<&GetDlgItem>] | |
| 00007FF66C061E91 | 48:8BC8 | mov rcx,rax | |
| 00007FF66C061E94 | FF15 26220000 | call qword ptr ds:[<&DestroyWindow>] | |
| 00007FF66C061E9A | 41:B8 01000000 | mov r8d,1 | |
| 00007FF66C061EA0 | 33D2 | xor edx,edx | |
| 00007FF66C061EA2 | 48:8B8C24 A0020000 | mov rcx,qword ptr ss:[rsp+2A0] | |
| 00007FF66C061EAA | FF15 80220000 | call qword ptr ds:[<&invalidateRect>] | |
| 00007FF66C061EB0 | 48:8B8C24 A0020000 | mov rcx,qword ptr ss:[rsp+2A0] | |
| 00007FF66C061EB8 | FF15 6A220000 | call qword ptr ds:[<&invalidateWindow>] | |

SECRET이라는 문자열과 xor을 하는 부분이 보이고

| | | |
|------------------|--------------------|---|
| 00007FF66C061E4F | 8BC1 | mov eax,ecx |
| 00007FF66C061E51 | 48:634C24 30 | movsxd rcx,dword ptr ss:[rsp+30] |
| 00007FF66C061E56 | 88840C 80010000 | mov byte ptr ss:[rsp+rcx+180],al |
| 00007FF66C061E5D | E9 22FFFFFF | jmp anti_baek.7FF66C061D84 |
| 00007FF66C061E62 | BA E9030000 | mov edx,3E9 |
| 00007FF66C061E67 | 48:8B8C24 A0020000 | mov rcx,qword ptr ss:[rsp+2A0] |
| 00007FF66C061E6F | FF15 A3220000 | call qword ptr ds:[<&GetDlgItem>] |
| 00007FF66C061E75 | 48:8BC8 | mov rcx,rax |
| 00007FF66C061E78 | FF15 42220000 | call qword ptr ds:[<&DestroyWindow>] |
| 00007FF66C061E7E | BA EA030000 | mov edx,3EA |
| 00007FF66C061E83 | 48:8B8C24 A0020000 | mov rcx,qword ptr ss:[rsp+2A0] |
| 00007FF66C061E8B | FF15 87220000 | call qword ptr ds:[<&GetDlgItem>] |
| 00007FF66C061E91 | 48:8BC8 | mov rcx,rax |
| 00007FF66C061E94 | FF15 26220000 | call qword ptr ds:[<&DestroyWindow>] |
| 00007FF66C061E9A | 41:B8 01000000 | mov r8d,1 |
| 00007FF66C061EA0 | 33D2 | xor edx,edx |
| 00007FF66C061EA2 | 48:8B8C24 A0020000 | mov rcx,qword ptr ss:[rsp+2A0] |
| 00007FF66C061EAA | FF15 80220000 | call qword ptr ds:[<&invalidateRect>] |
| 00007FF66C061EB0 | 48:8B8C24 A0020000 | mov rcx,qword ptr ss:[rsp+2A0] |
| 00007FF66C061EB8 | FF15 6A220000 | call qword ptr ds:[<&invalidateWindow>] |

byte ptr ss:[rsp+rcx*1+180]=[00000045788FF030]=4A 'J'
al=4A 'J'

그 값을 특정 메모리에 저장한다.

| 주소 | Hex | ASCII |
|-------------------|---|--------|
| 000000045788FF030 | 4A 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | J..... |
| 000000045788FF040 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 000000045788FF050 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 000000045788FF060 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 000000045788FF070 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |

이 메모리 위치를 보면 xor 결과인 J가 들어있으므로 반복 루틴이 끝나는 지점에 break point를 걸고

| | | |
|------------------|-----------------|----------------------------------|
| 00007FF66C061E46 | 8B8C24 80000000 | mov ecx,dword ptr ss:[rsp+80] |
| 00007FF66C061E4D | 33C8 | xor ecx,eax |
| 00007FF66C061E4F | 8BC1 | mov eax,ecx |
| 중단점 설정되지 않음 | 48:634C24 30 | movsxd rcx,dword ptr ss:[rsp+30] |
| | 88840C 80010000 | mov byte ptr ss:[rsp+rcx+180],al |
| 00007FF66C061E5D | E9 22FFFFFF | jmp anti_baek.7FF66C061D84 |
| 00007FF66C061E62 | BA E9030000 | mov edx,3E9 |

실행 결과인 메모리를 확인해보면 FLAG를 얻을 수 있다.

| 주소 | Hex | ASCII |
|------------------|---|------------------|
| 00000045788FF030 | 4A 46 53 7B 44 30 6D 61 69 6E 5F 65 78 70 61 6E | JFS{D0main_expan |
| 00000045788FF040 | 35 69 6F 6E 3F 5F 4A 75 73 74 5F 40 76 6F 69 64 | sion?_Just_@void |
| 00000045788FF050 | 5F 69 74 21 7D 00 00 00 00 00 00 00 00 00 00 00 | _it!}..... |
| 00000045788FF060 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |

FLAG : JFS{D0main_expan5ion?_Just_@void_it!}