

- Ezbeat -

먼저 프로그램 처음을 봐보면

Address	Hex dump	Disassembly
004AF1E0	60	PUSHAD
004AF1E1	BE 00004700	MOV ESI,Reverse2,00470000
004AF1E6	8DBE 0010F9F	LEA EDI,DWORD PTR DS:[ESI+FFF91000]
004AF1EC	57	PUSH EDI
004AF1ED	83CD FF	OR EBP,FFFFFFFF
004AF1F0	EB 10	JMP SHORT Reverse2,004AF202
004AF1F2	90	NOP

<그림 1>

위와 같이 되어있는 것을 볼 수 있습니다. PE확인 툴로 보나 소스로 보나 UPX로 패킹이 되어 있습니다. 여기 문제에서는 언패킹을 하라는 문제가 아니므로 이 부분은 건너 뛴 것입니다. 일단 모든 레지 값을 POPAD해주는 부분으로 가서 마지막 JMP문을 뛰어주면 원래 소스 부분으로 갑니다.

004AF386	61	POPAD
004AF387	8D4424 80	LEA EAX,DWORD PTR SS:[ESP-80]
004AF38B	6A 00	PUSH 0
004AF38D	39C4	CMP ESP,EAX
004AF38F	75 FA	JNZ SHORT Reverse2,004AF38B
004AF391	83EC 80	SUB ESP,-80
004AF394	E9 D783F6FF	JMP Reverse2,00417770

<그림 2>

이제 제대로 소스를 봅시다.

그 전에 제가 삽질한 것이 있는데 프로그램 종료라고 하길래 그 의미를 제대로 파악하지 못하고 계속 팝업창이 뜨면 바로 확인을 눌러서 끄는데 급급했습니다.

그래서 분석을 시작해 보면

00417770	E8 C4AF0000	CALL Reverse2,00422739
00417775	E9 79FEFFFF	JMP Reverse2,004175F3
0041777A	8BFF	MOV EDI,EDI
0041777C	EE	hlt

<그림 3>

처음 부분입니다. 아마 처음 프로그램 시작 시작을 알아내는 곳이 있으리라 생각하고 해당 함수를 들어가 보았습니다.

0042278A	33F0	XOR ESI,EAX	
0042278C	FF15 A0D34700	CALL DWORD PTR DS:[47D3A0]	kernel32.GetTickCount
00422792	33F0	XOR ESI,EAX	
00422794	8D45 F0	LEA EAX,DWORD PTR SS:[EBP-10]	

<그림 4>

그랬더니 GetTickCount라는 함수가 있더군요. 해당 함수는 운영체제 시작 후부터 시작을 Millisecond로 리턴 해 준다는 함수여서 “아! 이 부분을 잘 살펴보면 되겠구나 ..” 하고 계속 봐봤는데 끝나는 시점에도 해당 함수를 호출하지 않더군요..! 뭐 결국 해매다가 프로그램 실행 시키고 가만히 멍하니 있었는데 갑자기 혼자 꺼지더군요!!!! 그렇지!! 이제 알았습니다^^. 삽질을 끝내고 본 풀이로 가겠습니다.

Message박스를 띄우기 전에 몇 초가 지난 후에야 작동을 하더군요. 그 부분을 찾아보았습니다.

00444C3A	53	PUSH EBX	
00444C3B	55	PUSH EBP	
00444C3C	56	PUSH ESI	
00444C3D	57	PUSH EDI	
00444C3E	8B3D 58D74700	MOV EDI,DWORD PTR DS:[47D758]	WINMM,timeGetTime
00444C44	FFD7	CALL EDI	
00444C46	803D D3E84800	CMP BYTE PTR DS:[48E8D3],0	
00444C4D	8BF0	MOV ESI,EAX	
00444C4F	0F84 FF000000	JE Reverse2.00444D54	
00444C55	8B5C24 14	MOV EBX,DWORD PTR SS:[ESP+14]	
00444C59	8B2D 58D14700	MOV EBP,DWORD PTR DS:[47D158]	kernel32.Sleep
00444C5F	FFD7	CALL EDI	
00444C61	3BC6	CMP EAX,ESI	
00444C63	0F83 CF000000	JNB Reverse2.00444D38	

<그림 5>

위 부분이 이 문제의 핵심 포인트 시작 부분입니다. timeGetTime이라는 함수가 보이네요!
해당 함수는 현재 시간을 가져오면서 MilliSecond까지 표시를 해주는 함수입니다.
쪽 트레이스 해본 결과를 아래 적어보겠습니다.

00444C3A	53	PUSH EBX	
00444C3B	55	PUSH EBP	
00444C3C	56	PUSH ESI	
00444C3D	57	PUSH EDI	
00444C3E	8B3D 58D74700	MOV EDI,DWORD PTR DS:[47D758]	; WINMM.timeGetTime
00444C44	FFD7	CALL EDI	//현재 시간을 EAX로 리턴
00444C46	803D D3E84800	0>CMP BYTE PTR DS:[48E8D3],0	
00444C4D	8BF0	MOV ESI,EAX	//ESI레지에 처음 시작하는 시간을 담음
00444C4F	0F84 FF000000	JE Reverse2.00444D54	
00444C55	8B5C24 14	MOV EBX,DWORD PTR SS:[ESP+14]	
00444C59	8B2D 58D14700	MOV EBP,DWORD PTR DS:[47D158]	; kernel32.Sleep
00444C5F	FFD7	CALL EDI	//다시 한번 현재 시작을 구함
00444C61	3BC6	CMP EAX,ESI	//처음에 구한 시간과 다시 구한 시간을 비교
00444C63	0F83 CF000000	JNB Reverse2.00444D38	//나중에 구한 시간이 크면 점프(당연히 점프)
00444D38	2BC6	SUB EAX,ESI	//나중에 구한 시간에서 처음 구한 시간 뺌
00444D3A	3B43 04	CMP EAX,DWORD PTR DS:[EBX+4]	//뺀거와 [EBX+4]와 비교 [EBX+4] = 337B
00444D3D	^ 0F83 2EFFFFFF	JNB Reverse2.00444C71	//점프하면 끝나는 것...고로 차이가 337B넘으면 점프
00444D43	6A 0A	PUSH 0A	
00444D45	FFD5	CALL EBP	
00444D47	803D D3E84800	0>CMP BYTE PTR DS:[48E8D3],0	
00444D4E	^ 0F85 0BFFFFFF	JNZ Reverse2.00444C5F	

위를 분석해보면 처음 구한 시간과 방금 구한 시간의 차이가 337B가 넘으면 프로그램을 종료하는 것입니다. 그래서 이 문제의 답은

16진수 : 337B

10진수 : 13179

MD5 : DB59260CCE0B871C7B2BB780EEE305DB