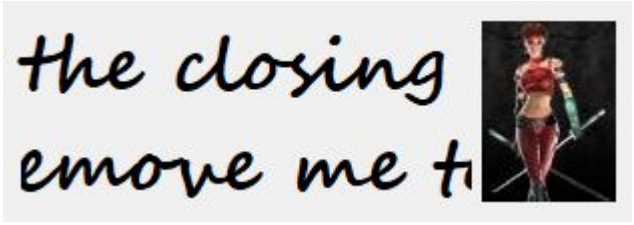
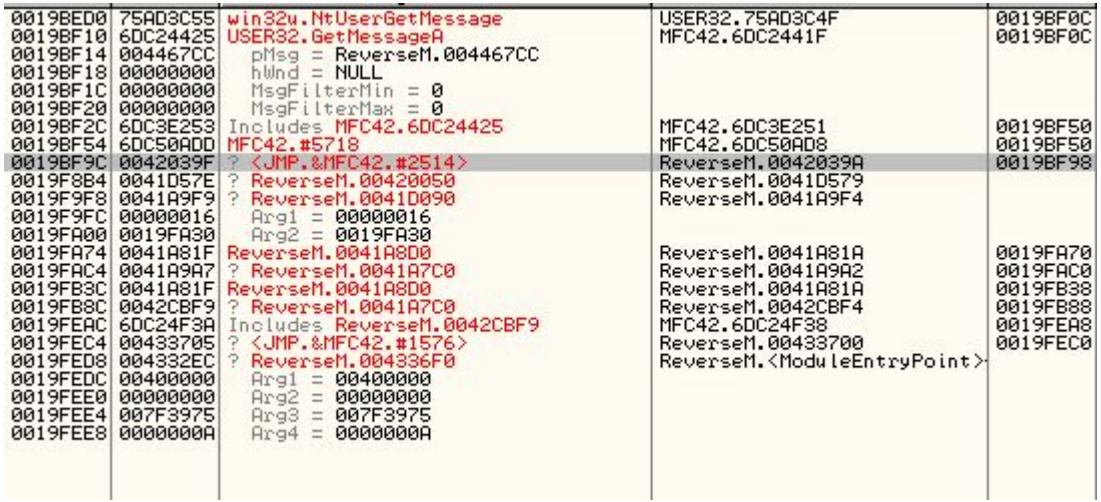
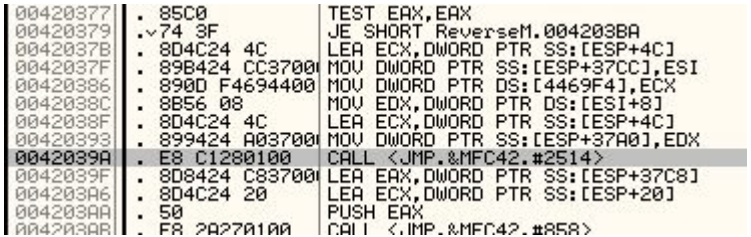


프로그램 실행 시 나타나는 Nag 창이다. 그리고 메인 프로그램이 실행되고, Exit 버튼을 누르면 또 다른 Nag 창이 나온다.

메시지를 해석해보면 둘다 삭제해달라고 한다.

디버깅을 통해 삭제하면 될것같다.

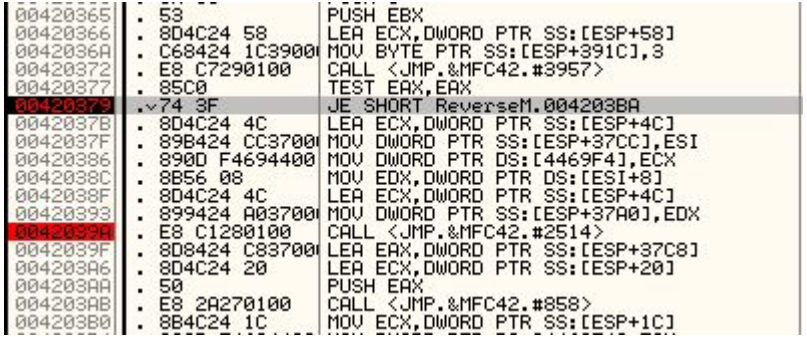




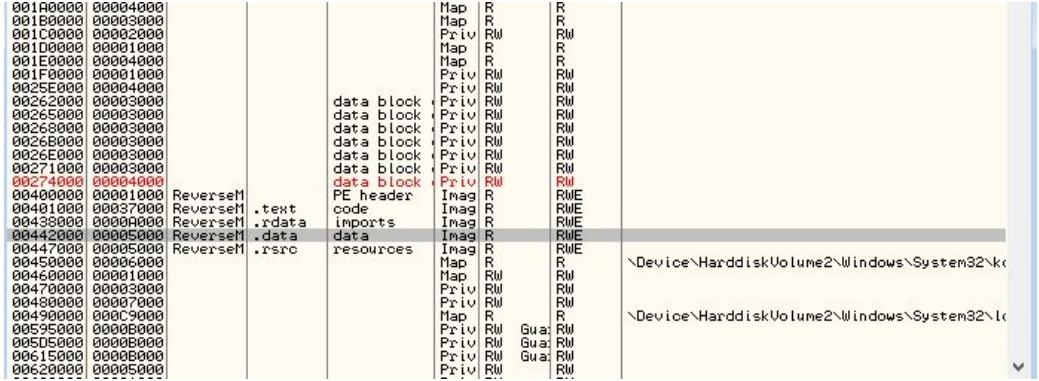
시작 Nag 창을 분석하기 위해
실행하고, 디버깅 중지를 시켜
어디서 호출되는지 살펴본다.

Alt + K를 눌러 콜스택을 살펴본 결과
JMP.&MFC42.#2514부분에서
호출되는걸 알 수 있다.

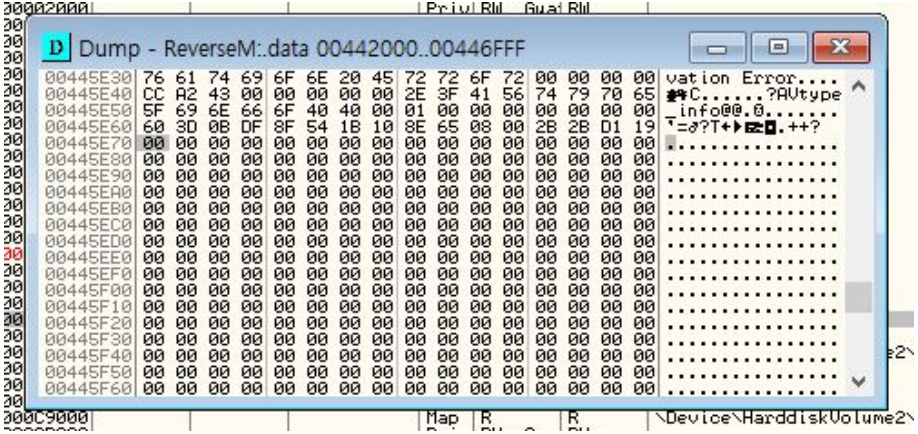
해당 영역으로 가보니
MFC42.dll에서 제공하는 함수를
호출하는 루틴이 나온다. 그리고 그
위쪽을 살펴보니 분기문이 하나있고,
EAX 값에 따라 해당 함수를
호출할지 말지를 결정한다.



함수가 호출되기 전 분기문에 bp를 걸고 실행해본다. 그리고 분기가 되지 않아 밑에 함수를 호출한다. 그리고 계속 실행해보니 시작 Nag창을 띄우고 다시 분기문으로 이동하고, 똑같은 함수에서 메인 프로그램을 실행한다. 즉, Nag 창을 삭제하기 위해서 첫 분기에는 분기되게 하고, 두 번째는 분기되지 않게 하고, 세번째에는 분기되게 패치하면 될것같다.



001A0000	00004000				Map	R	R	
001B0000	00003000				Map	R	R	
001C0000	00002000				Priv	RW	RW	
001D0000	00001000				Map	R	R	
001E0000	00004000				Map	R	R	
001F0000	00001000				Priv	RW	RW	
0025E000	00004000				Priv	RW	RW	
00262000	00003000			data block	Priv	RW	RW	
00265000	00003000			data block	Priv	RW	RW	
00268000	00003000			data block	Priv	RW	RW	
0026B000	00003000			data block	Priv	RW	RW	
0026E000	00003000			data block	Priv	RW	RW	
00271000	00003000			data block	Priv	RW	RW	
00274000	00004000			data block	Priv	RW	RW	
00400000	00001000	ReverseM		PE header	Imag	R	RWE	
00401000	00037000	ReverseM	.text	code	Imag	R	RWE	
00438000	0000A000	ReverseM	.rdata	imports	Imag	R	RWE	
00442000	00005000	ReverseM	.data	data	Imag	R	RWE	
00447000	00005000	ReverseM	.rsrc	resources	Imag	R	RWE	
00450000	00006000				Map	R	R	\Device\HarddiskVolume2\Windows\System32\k
00460000	00001000				Map	RW	RW	
00470000	00003000				Priv	RW	RW	
00480000	00007000				Priv	RW	RW	
00490000	000C9000				Map	R	R	\Device\HarddiskVolume2\Windows\System32\l
00595000	0000E000				Priv	RW	Gua	RW
005D5000	0000B000				Priv	RW	Gua	RW
00615000	0000B000				Priv	RW	Gua	RW
00620000	00005000				Priv	RW		RW



Dump - ReverseM:.data 00442000..00446FFF

00445E30	76	61	74	69	6F	6E	20	45	72	72	6F	72	00	00	00	00	vation Error....
00445E40	CC	A2	43	00	00	00	00	2E	3F	41	56	74	79	70	65		¢C.....?AVtype
00445E50	5F	69	6E	66	6F	40	40	00	01	00	00	00	00	00	00		_info@@.@.....
00445E60	60	3D	0B	DF	8F	54	1B	10	8E	65	08	00	2B	2B	D1	19	`=ß?T+▶◘.++?
00445E70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	▪
00445E80	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00445E90	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00445EA0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00445EB0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00445EC0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00445ED0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00445EE0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00445EF0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00445F00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00445F10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00445F20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00445F30	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00445F40	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00445F50	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00445F60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	

이번에도 전 튜토리얼과 같이 인라인 패치를 하면 될것같다. JE를 만날 때마다. 특정 규칙에 의해 분기할지 말지를 결정하는 코드를 짜면 될것같다.

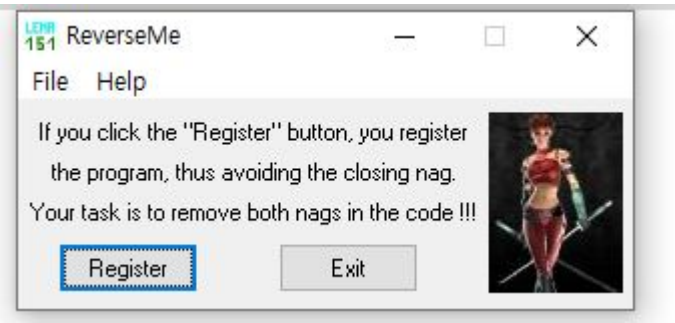
Alt + M 을 눌러 메모리 맵으로 이동해 쓸 수 있는 공간이 있는지 찾아본다. .data 영역에 RWE 권한이 있고, 지역 변수 주소를 찾기 좋을 것 같다.

445E70을 지역 변수 주소로 쓴다. 해당 주소에 있는 값과 어떤 값을 비교해 조건문에 만족할 경우에만 점프하도록 한다.(해당 주소를 어딘가에서 사용하고 있어 에러가 나서 446AF0으로 바꿨더니 됐다.)

이제 코드를 패치할 주소를 찾아야 한다. 그 전 튜토리얼과 마찬가지로 코드 맨 아래쪽으로 내리다 보면 안쓰는 공간이 있을것이다. 거기에 비교 코드를 삽입할 것이다.

이렇게 코드를 삽입한 후 원래 분기문을 인라인 패치한 부분으로 점프하도록 패치한다.

그러면 445E70부분에서 2일 때만 메인 프로그램이 실행되고, 아닌 경우에는 모두 분기 될 것이므로 Nag창이 안뜰것이다.



이렇게 프로그램을 실행하면, Nag 창이 안뜨고 프로그램이 바로 실행된다.