
abex crackme2 패치해보기

사이버보안학과 정보보안 학술소학회

신동석
20180128

CONTENTS

abex crackme2 소개 03

원하는 패치 설명 04

패치 과정 및 결과 05

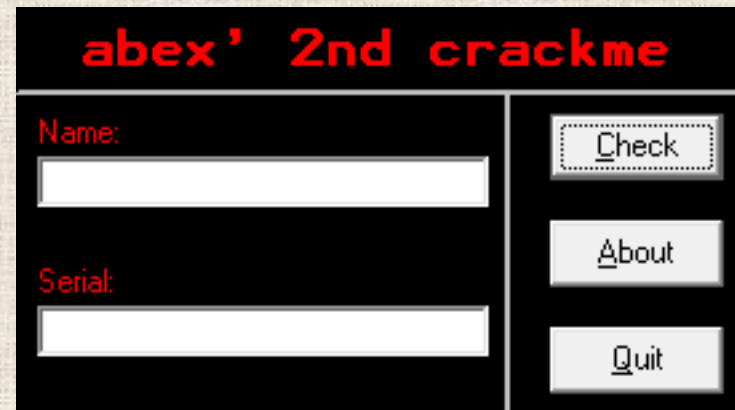
abex crackme2

- Crackme 란?

크랙 연습을 위해 만들어진 파일

- abex crackme2

입력한 NAME에 따라 생성되는 Serial의 값을 입력해야 한다. NAME은 4글자 이상이어야 하는 조건이 있다.



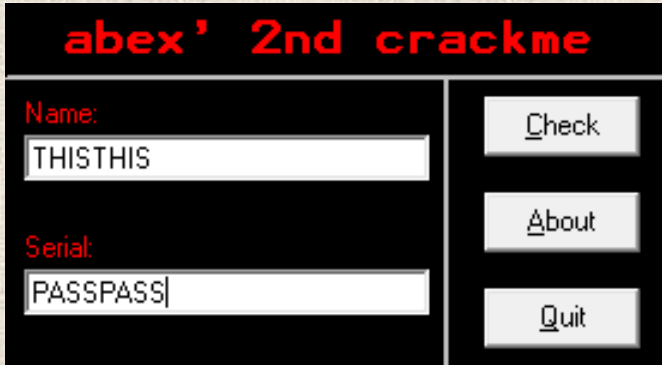
abex' 2nd crackme	
Name: <input type="text"/>	<input type="button" value="Check"/>
Serial: <input type="text"/>	<input type="button" value="About"/> <input type="button" value="Quit"/>

원하는 패치

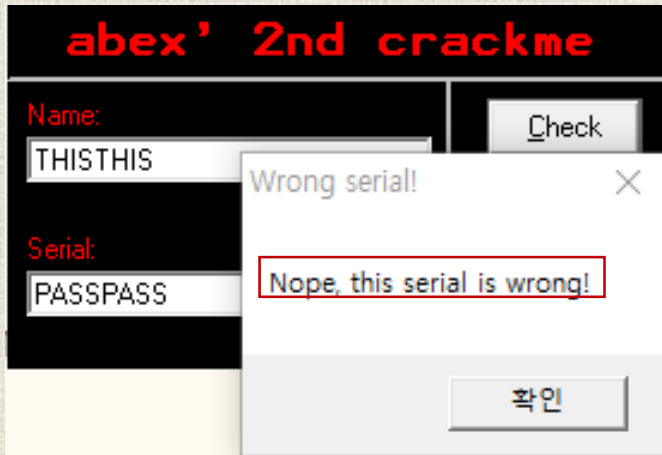


1. NAME이 4글자 이상일 때 Serial에 어떤 값을 입력하더라도 통과하기.
2. 1번에 글자 제한 조건을 없애서 아무런 값 입력 없이도 통과하기.

패치1 과정



- 자신이 입력한 Name과 Serial의 값을 찾아내기 위해서 알아보기 쉬운 값으로 입력해준다.



- 저런식으로 입력하면 잘못된 serial 값 이라는 메시지 창이 출력된다.
- 메시지 창에 포함되어 있는 문자열들을 이용해서 이 크랙미의 중요한 분기점을 찾아낸다.

패치1 과정

00403329	. FF15 58104000	CALL DWORD PTR DS:[&MSUBUM60.__vbaVarT	MSUBUM60.__vbaVarTstEq
0040332F	. 66:85C0	TEST AX,AX	
00403332	. 0F84 D0000000	JE abexcm2.00403408	
00403408	> 98B3D B0104000	MOV EDI,DWORD PTR DS:[&MSUBUM60.__vbaU	MSUBUM60.__vbaVarDup
0040340E	. BB 0A000000	MOV EBX,0A	
00403413	> 8D45 BC	LEA EAX,DWORD PTR SS:[EBP-44]	
00403416	. 8D4D CC	LEA ECX,DWORD PTR SS:[EBP-34]	
00403419	. 50	PUSH EAX	
0040341A	. 51	PUSH ECX	
0040341B	. FF15 A4104000	CALL DWORD PTR DS:[&MSUBUM60.__vbaVarT	MSUBUM60.__vbaVarTstNe
00403421	. 66:85C0	TEST AX,AX	
00403424	. 0F84 C7000000	JE abexcm2.004034F1	
0040342A	. 899D 34FFFFFF	MOV DWORD PTR SS:[EBP-CC],EBX	
00403430	. 899D 44FFFFFF	MOV DWORD PTR SS:[EBP-BC],EBX	
00403436	. B8 04000280	MOV EAX,80020004	
0040343B	. BB 08000000	MOV EBX,8	
00403440	. 8D95 14FFFFFF	LEA EDX,DWORD PTR SS:[EBP-EC]	
00403446	. 8D8D 54FFFFFF	LEA ECX,DWORD PTR SS:[EBP-AC]	
0040344C	. 8985 3CFFFFFF	MOV DWORD PTR SS:[EBP-C4],EAX	
00403452	. 8985 4CFFFFFF	MOV DWORD PTR SS:[EBP-B4],EAX	
00403458	. C785 1CFFFFFF	MOV DWORD PTR SS:[EBP-E4],abexcm2.00402	UNICODE "Wrong serial!"
00403462	. 899D 14FFFFFF	MOV DWORD PTR SS:[EBP-EC],EBX	
00403468	. FFD7	CALL EDI	
0040346A	. 8D95 24FFFFFF	LEA EDX,DWORD PTR SS:[EBP-DC]	
00403470	. 8D8D 64FFFFFF	LEA ECX,DWORD PTR SS:[EBP-9C]	
00403476	. C785 2CFFFFFF	MOV DWORD PTR SS:[EBP-D4],abexcm2.00402	UNICODE "Nope, this serial is wrong!"

00403329 에서 __vbaVarTstEq라는 함수 호출 후, 나온 리턴값을 이용해서 성공과 실패로 분기되므로, __vbaVarTstEq는 생성된 serial과 입력한 serial을 비교하는 함수 일 것이다. 00403329에 BP를 만들어 준다.

패치1 과정

00403321	. 8055 BC	LEA EDX, DWORD PTR SS:[EBP-44]	0019F250
00403324	. 8045 CC	LEA EAX, DWORD PTR SS:[EBP-34]	0019F260
00403327	. 52	PUSH EDX	
00403328	. 50	PUSH EAX	
00403329	. FF15 58104000	CALL DWORD PTR DS:[<&MSVBVM60.__vbaVarT	MSVBVM60.__vbaVarTstEq

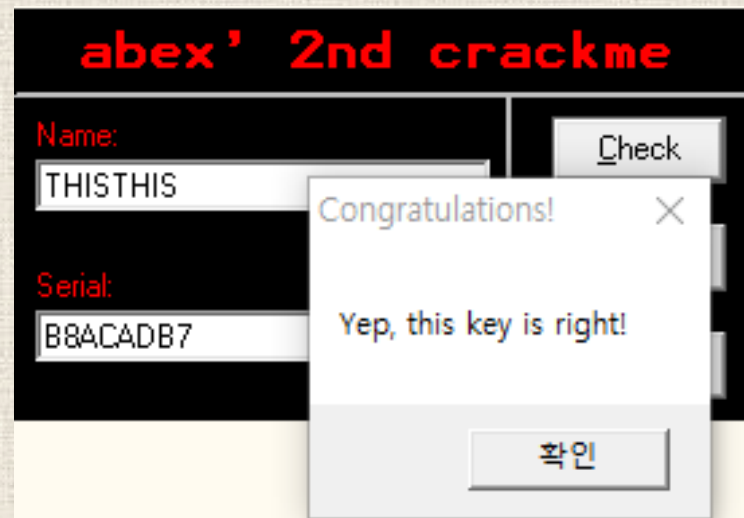
- `_vbaVarTstEq` 함수를 호출하기 전, EDX와 EAX를 가지고 함수에 들어가는것을 보아, EDX와 EAX에는 serial값이 저장되어 있을것이다.

0019F250	00000008	
0019F254	0019F240	
0019F258	026B4A2C	UNICODE "B8ACADB7"
0019F25C	0019F1F8	
0019F260	00000008	
0019F264	0019F240	
0019F268	026AE57C	UNICODE "PASSPASS"
0019F26C	0019F1F8	
0019F270	00000002	

- 실제로 EDX에는 생성된 serial, EAX에는 입력한 serial이 저장되어 있다.

패치1 과정

- EDX에는 생성된 serial이 저장되어 있다는 것을 확인
- 이 점을 이용해서 EAX가 가리키는 값을 변경하고자 한다.

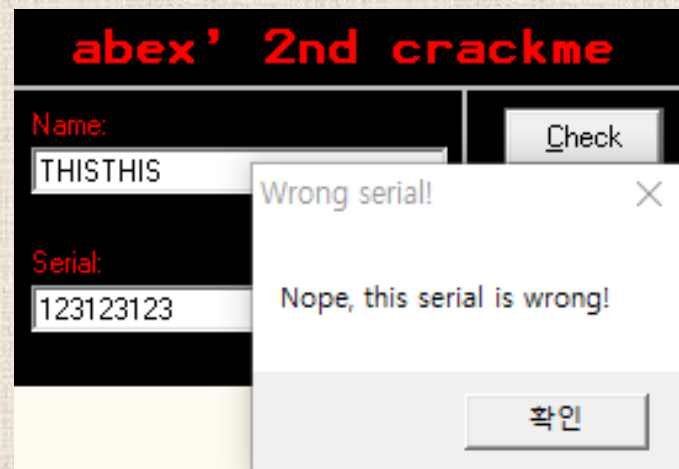
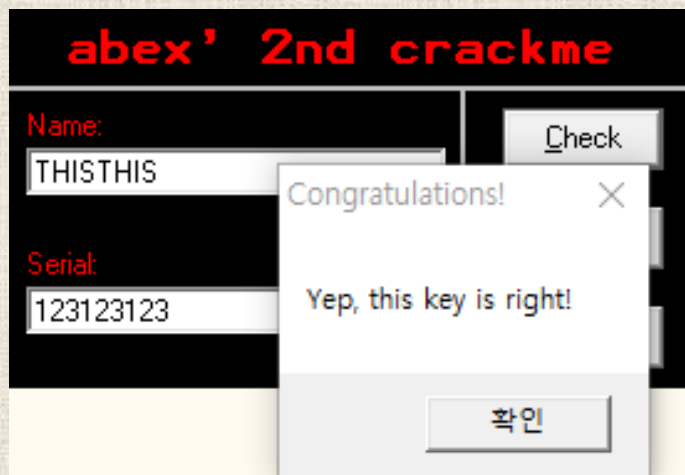


패치1 과정

00403321	.	8D55 BC	LEA EDX,DWORD PTR SS:[EBP-44]	0019F250
00403324	.	8D45 BC	LEA EAX,DWORD PTR SS:[EBP-44]	0019F260
00403327	.	52	PUSH EDX	
00403328	.	50	PUSH EAX	
00403329	.	FF15 58104000	CALL DWORD PTR DS:[<&MSUBUM60.__vbaVarT	MSUBUM60.__vbaVarTstEq

- EAX가 가리키는 값을 EDX가 가리키는 값과 같게 만들면 NAME이 무엇이든 __vbaVarTstEq함수에는 같은 값이 들어가서 무조건 성공 메시지 출력으로 넘어갈 것이다.

패치1 과정



- 성공메시지가 출력되며, 확인을 누르면 바로 실패 메시지 또한 출력된다.

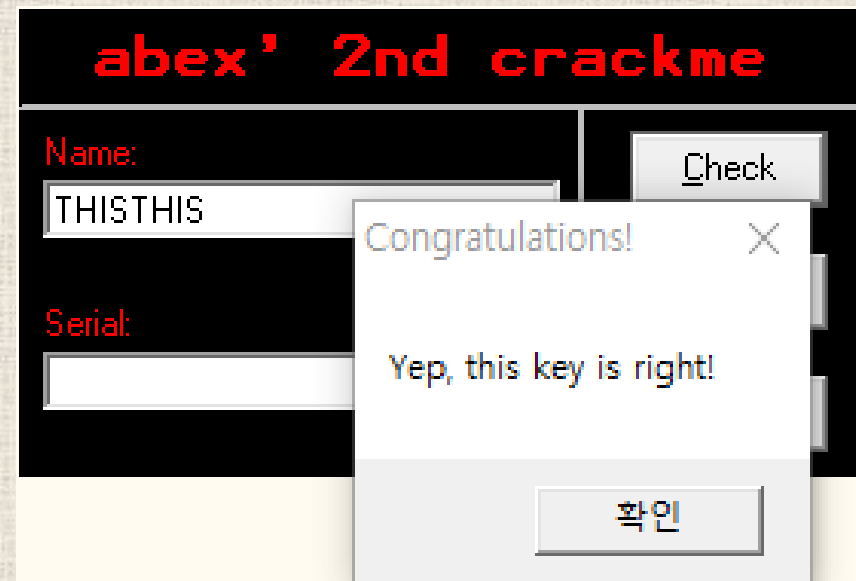
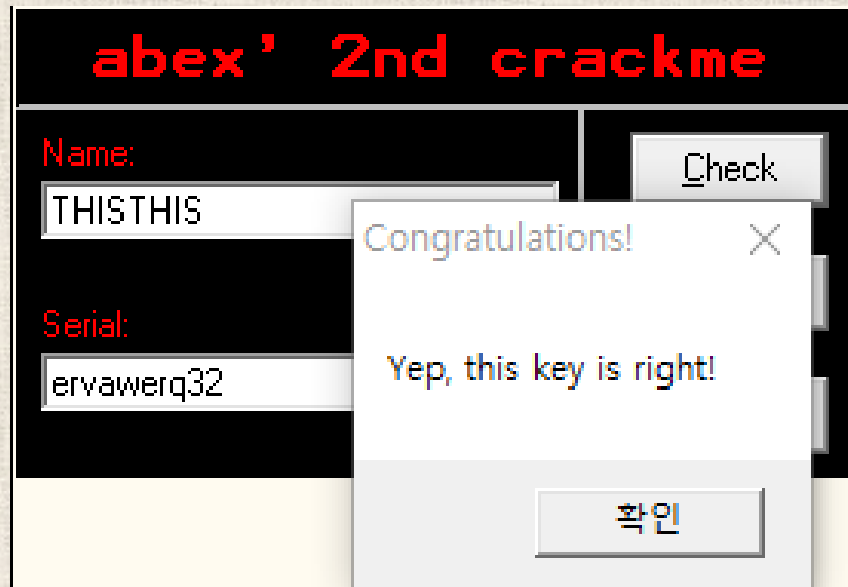
패치1 과정

00403413	>	8045 BC	LEA EAX,DWORD PTR SS:[EBP-44]	
00403416		804D CC	LEA ECX,DWORD PTR SS:[EBP-34]	
00403419		50	PUSH EAX	
0040341A		51	PUSH ECX	
0040341B		FF15 A4104000	CALL DWORD PTR DS:[&MSUBUM60.__vbaVarT	MSUBUM60.__vbaVarTstNe
00403421	.	66:85C0	TEST AX,AX	
00403424	✓	0F84 C7000000	JE abexcm2.004034F1	
0040342A	.	899D 34FFFFFF	MOV DWORD PTR SS:[EBP-CC],EBX	
00403430	.	899D 44FFFFFF	MOV DWORD PTR SS:[EBP-BC],EBX	
00403436	.	B8 04000280	MOV EAX,80020004	
0040343B	.	BB 08000000	MOV EBX,8	
00403440	.	8D95 14FFFFFF	LEA EDX,DWORD PTR SS:[EBP-EC]	
00403446	.	8D8D 54FFFFFF	LEA ECX,DWORD PTR SS:[EBP-AC]	
0040344C	.	8985 3CFFFFFF	MOV DWORD PTR SS:[EBP-C4],EAX	
00403452	.	8985 4CFFFFFF	MOV DWORD PTR SS:[EBP-B4],EAX	
00403458	.	C785 1CFFFFFF	MOV DWORD PTR SS:[EBP-E4],abexcm2.00402	UNICODE "Wrong serial!"
00403462	.	899D 14FFFFFF	MOV DWORD PTR SS:[EBP-EC],EBX	
00403468	.	FFD7	CALL EDI	
0040346A	.	8D95 24FFFFFF	LEA EDX,DWORD PTR SS:[EBP-DC]	
00403470	.	8D8D 64FFFFFF	LEA ECX,DWORD PTR SS:[EBP-9C]	
00403476	.	C785 2CFFFFFF	MOV DWORD PTR SS:[EBP-D4],abexcm2.00402	UNICODE "Nope, this serial is wrong!"

- 실패 메시지 출력 전에도 다시 한번 __vbaVarTstNe라는 함수에 전에 사용되었던 두 개의 문자열을 비교하게 된다. 이 부분 또한 EAX와 ECX가 같은 값을 가리키도록 패치해준다.

00403413	>	8045 BC	LEA EAX,DWORD PTR SS:[EBP-44]	
00403416		804D BC	LEA ECX,DWORD PTR SS:[EBP-44]	
00403419		50	PUSH EAX	
0040341A		51	PUSH ECX	
0040341B		FF15 A4104000	CALL DWORD PTR DS:[&MSUBUM60.__vbaVarT	MSUBUM60.__vbaVarTstNe

패치1 결과



패치2 과정

00403010	. FF15 44104000	CALL DWORD PTR DS:[<&MSUBUM60.__vbaVarT	MSUBUM60.__vbaVarTstLt
00403023	66:85C0	TEST AX,AX	
00403026	0F84 CD000000	JE abexcm2.004030F9	
0040302C	. B9 04000280	MOV ECX,80020004	
00403031	. B8 0A000000	MOV EAX,0A	
00403036	. 898D 3CFFFFFF	MOV DWORD PTR SS:[EBP-C4],ECX	
0040303C	. 898D 4CFFFFFF	MOV DWORD PTR SS:[EBP-B4],ECX	
00403042	. 899D 14FFFFFF	MOV DWORD PTR SS:[EBP-EC],EBX	
00403048	. 8B1D B0104000	MOV EBX,DWORD PTR DS:[<&MSUBUM60.__vbaU	MSUBUM60.__vbaVarDup
0040304E	. 8D95 14FFFFFF	LEA EDX,DWORD PTR SS:[EBP-EC]	
00403054	. 8D8D 54FFFFFF	LEA ECX,DWORD PTR SS:[EBP-AC]	
0040305A	. 8985 34FFFFFF	MOV DWORD PTR SS:[EBP-CC],EAX	
00403060	. 8985 44FFFFFF	MOV DWORD PTR SS:[EBP-BC],EAX	
00403066	. C785 1CFFFFFF	MOV DWORD PTR SS:[EBP-E4],abexcm2.00402	UNICODE "Error!"
00403070	. FFD3	CALL EBX	<&MSUBUM60.__vbaVarDup>
00403072	. 8D95 24FFFFFF	LEA EDX,DWORD PTR SS:[EBP-DC]	
00403078	. 8D8D 64FFFFFF	LEA ECX,DWORD PTR SS:[EBP-9C]	
0040307E	. C785 2CFFFFFF	MOV DWORD PTR SS:[EBP-D4],abexcm2.00402	UNICODE "Please enter at least 4 chars as name!"

- 우선 문자열의 길이 제한이 4임을 알려주는 문자열을 이용해서 분기점을 찾아낸다.

패치2 과정

- 문자열의 길이를 테스트하는 함수를 생략하기 위해서, 적절한 JMP 시작지점과 도착지점을 알아낸다.

00402FBF	. 8095 78FFFFFF	LEA EDX,DWORD PTR SS:[EBP-88]	
00402F94	. 52	PUSH EDX	
00402F95	. 56	PUSH ESI	
00402F96	. 8B0E	MOV ECX,DWORD PTR DS:[ESI]	
00402F98	. FF91 A0000000	CALL DWORD PTR DS:[ECX+A0]	
00402F9E	. 3BC7	CMP EAX,EDI	
00402FA0	. DBE2	FCLEX	
00402FA2	~ 7D 12	JGE SHORT abexcm2.00402FB6	
00402FA4	. 68 A0000000	PUSH 0A0	
00402FA9	. 68 68234000	PUSH abexcm2.00402368	
00402FAE	. 56	PUSH ESI	
00402FAF	. 50	PUSH EAX	
00402FB0	. FF15 24104000	CALL DWORD PTR DS:[<&MSUBUM60.__vbaHresu	MSUBUM60.__vbaHresultCheckObj
00402FB6	> 8B85 78FFFFFF	MOV EAX,DWORD PTR SS:[EBP-88]	
00402FBC	. 8B35 08104000	MOV ESI,DWORD PTR DS:[<&MSUBUM60.__vbaU	MSUBUM60.__vbaVarMove
00402FC2	. BB 08000000	MOV EBX,8	
00402FC7	. 8095 64FFFFFF	LEA EDX,DWORD PTR SS:[EBP-9C]	
00402FCD	. 8D4D 8C	LEA ECX,DWORD PTR SS:[EBP-74]	
00402FD0	. 89BD 78FFFFFF	MOV DWORD PTR SS:[EBP-88],EDI	
00402FD6	. 8985 6CFFFFFF	MOV DWORD PTR SS:[EBP-94],EAX	
00402FDC	. 899D 64FFFFFF	MOV DWORD PTR SS:[EBP-9C],EBX	
00402FE2	. FFD6	CALL ESI	<&MSUBUM60.__vbaVarMove>
00402FE4	. 8D8D 74FFFFFF	LEA ECX,DWORD PTR SS:[EBP-8C]	
00402FEA	. FF15 C8104000	CALL DWORD PTR DS:[<&MSUBUM60.__vbaFree	MSUBUM60.__vbaFreeObj

- 길이 테스트 위에 존재하는 구간으로, 저 부분에서 사용자가 입력한 NAME의 값을 읽어와 스택에 저장해 준다.

패치2 과정

00402FF0	8D45 8C	LEA EAX,DWORD PTR SS:[EBP-74]	
00402FF3	8D8D 64FFFFFF	LEA ECX,DWORD PTR SS:[EBP-9C]	
00402FF9	50	PUSH EAX	
00402FFA	51	PUSH ECX	
00402FFB	C785 2CFFFFFF	MOV DWORD PTR SS:[EBP-D4],4	
00403005	. C785 24FFFFFF	MOV DWORD PTR SS:[EBP-DC],8002	
0040300F	. FF15 28104000	CALL DWORD PTR DS:[&MSUBUM60.__vbaLenVar]	MSUBUM60.__vbaLenVar
00403015	8D95 24FFFFFF	LEA EDX,DWORD PTR SS:[EBP-DC]	THE LENGTH OF "NAME" = 0019F1F8
0040301B	50	PUSH EAX	
0040301C	52	PUSH EDX	
0040301D	. FF15 44104000	CALL DWORD PTR DS:[&MSUBUM60.__vbaVarTstLt]	MSUBUM60.__vbaVarTstLt

- __varVarTstLt는 길이가 4 이상인지 확인하는 함수, __vbaLenVar는 사용자가 입력한 NAME의 길이를 알아내는 함수, __vbaLenVar에서 나온 길이 값으로 __vbaVarTstLt 함수가 실행된다.

패치2 과정

```
00403197 | > 85C0 | TEST EAX,EAX  
00403199 | v 0F84 06010000 | JE abexcm2.004032A5  
  
0040329A | . | FF15 C0104000 | CALL DWORD PTR DS:[<&MSUBUM60.__vbaVarForNext  
004032A0 | .^ | E9 F2FEFFFF | JMP abexcm2.00403197
```

- 위 JMP는 올바르게 NAME이 입력되었을 시, 4번을 반복하는 구간이다. 또한 반복되는 과정마다, 스택에 Serial이 두 글자 씩 생성 된다.

패치2 과정

00403010	. FF15 44104000	CALL DWORD PTR DS:[&MSUBUM60.__vbaVarT	MSUBUM60.__vbaVarTstLt
00403023	66:85C0	TEST AX,AX	
00403026	✓ E9 60010000	JMP abexcm2.0040318B	

- Test함수 이후, 리턴 값에 상관없이 바로 serial을 생성하는 주소(0040318B)로 JMP하면 프로그램이 다운되어버린다.
- 아마 입력한 NAME의 길이가 4보다 작아서 4번의 반복을 끝내지 못해서 발생하는 현상이라고 추측.
- 따라서 JMP문의 도착지점을 serial을 생성조차 하지 않은 채 넘어가서 설정.

패치2 과정

00403010	.	FF15 44104000	CALL DWORD PTR DS:[&MSUBUM60.__vbaVarT	MSUBUM60.__vbaVarTstLt
00403023		66:85C0	TEST AX,AX	
00403026	✓	E9 F6020000	JMP abexcm2.00403321	
00403321	.	8D55 BC	LEA EDX,DWORD PTR SS:[EBP-44]	0019F250
00403324		8D45 BC	LEA EAX,DWORD PTR SS:[EBP-44]	0019F260
00403327	.	52	PUSH EDX	
00403328	.	50	PUSH EAX	
00403329	.	FF15 58104000	CALL DWORD PTR DS:[&MSUBUM60.__vbaVarT	MSUBUM60.__vbaVarTstEq

- Serial 생성 부분을 전부 뛰어넘은 상태, 패치1 에서 설정한대로 생성된 serial이 저장 되어 있을 공간과, 사용자가 입력한 serial을 비교할 것임.
- JMP문 조작으로 인해 너무 많은 중간 과정을 생략했으므로 비교함수에 들어가는 EDX값을 확인.

패치2 과정

0019F250	00000000
0019F254	000304E0
0019F258	00000098
0019F25C	00000000
0019F260	00000000
0019F264	74B433D5
0019F268	53011EA1
0019F26C	005A0049

RETURN to USER32.74B433D5 from GDI32.PolyPatBlt

- 크게 영향을 주지 않을 값으로 채워졌다고 추측.

패치2 결과

