

Reversing.kr – Easy Crack

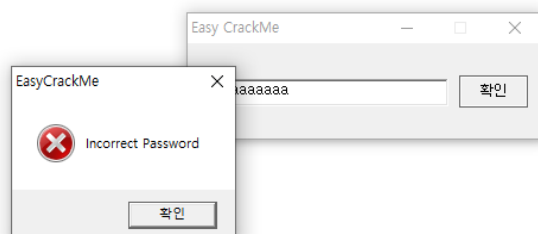
주어진 파일

Easy_CrackMe.exe	2021-01-10 오후 7:22	응용 프로그램	40KB
------------------	--------------------	---------	------

X32dbg로 주어진 실행 파일 분석

CPU	로그	메모	중단점	메모리 맵	호출 스택	SEH	스크립트	기호	소스	참조	스레드	핸들
00401188	55	push ebp							EntryPoint			
00401189	8BEC	mov ebp,esp										
0040118A	6A FF	push FFFFFFFF										
0040118B	68 B0504000	push easy_crackme.405080										
0040118C	68 541E4000	push easy_crackme.401E54										
0040118D	64:A1 00000000	mov eax,dword ptr [0]										
0040118E	50	push eax										
0040118F	64:8925 00000000	mov dword ptr [0],esp										
00401190	83EC 58	sub esp,58										
00401191	53	push ebx										
00401192	56	push esi										
00401193	57	push edi										
00401194	8965 E8	mov dword ptr ss:[ebp-18],esp										
00401195	FF15 20504000	call dword ptr ds:[<&GetVersion>]										
00401196	33D2	xor edx,edx										
00401197	8AD4	mov dl,ah										
00401198	8915 34854000	mov dword ptr ds:[408534],edx										
00401199	8BC8	mov ecx,eax										
0040119A	81E1 FF000000	and ecx,FF										
0040119B	8900 30854000	mov dword ptr ds:[408530],ecx										
0040119C	C1E1 08	shl ecx,8										
0040119D	03CA	add ecx,edx										
0040119E	8900 2C854000	mov dword ptr ds:[40852C],ecx										
0040119F	C1E8 10	shr eax,10										
004011A0	43 28854000	mov dword ptr ds:[408538],eax										

실행시킨 후 임의의 문자열 입력한 후 결과화면



잘못된 입력 값의 결과로 출력된 메시지를 문자열 참조에서 검색 후 해당 문자열을 출력하는 코드에서 bp 설정

주소	디스어셈블리	문자열
0040113C	push easy_crackme.406030	"Incorrect Password"
697C26AE	mov dword ptr ds:[ebx+18],dui70.6972A1EA	"incorrect header check"
697C27D7	mov dword ptr ds:[ebx+18],dui70.6972A212	"incorrect data check"
75C28AC8	push combase.75A9D9DC	"The async operation object that was used incorrectly."
75C28B18	push combase.75A9D9DC	"The async operation object that was used incorrectly."

Bp 위치 근처에 "congratulation"이라는 문자열 출력 코드 확인

00401109	85C0	test eax, eax	
00401108	75 28	jne easy_crackme.401135	
0040110D	807C24 04 45	cmp byte ptr ss:[esp+4], 45	45: 'E'
00401112	75 21	jne easy_crackme.401135	
00401114	6A 40	push 40	
00401116	68 58604000	push easy_crackme.406058	406058: "EasyCrackMe"
00401118	68 44604000	push easy_crackme.406044	406044: "Congratulation !!"
00401120	S7	push edi	
00401121	FF15 A0504000	call dword ptr ds:[<&MessageBoxA>]	
00401127	6A 00	push 0	
00401129	S7	push edi	
0040112A	FF15 A4504000	call dword ptr ds:[<&EndDialog>]	
00401130	5F	pop edi	
00401131	83C4 64	add esp, 64	
00401134	C3	ret	
00401135	6A 10	push 10	
00401137	68 58604000	push easy_crackme.406058	406058: "EasyCrackMe"
0040113C	68 30604000	push easy_crackme.406030	406030: "Incorrect Password"
00401141	S7	push edi	
00401142	FF15 A0504000	call dword ptr ds:[<&MessageBoxA>]	

해당 위치 주변에서 입력한 문자열을 검사하는 루틴 발견

00401098	8B7C24 70	mov edi, dword ptr ss:[esp+70]	
0040109F	8D4424 08	lea eax, dword ptr ss:[esp+8]	
004010A3	50	push eax	
004010A4	68 E8030000	push SE8	
004010A9	S7	push edi	
004010AA	FF15 9C504000	call dword ptr ds:[<&GetDlgItemTextA>]	
004010B0	807C24 05 61	cmp byte ptr ss:[esp+5], 61	61: 'a'
004010B5	75 7E	jne easy_crackme.401135	
004010B7	6A 02	push 2	
004010B9	8D4C24 0A	lea ecx, dword ptr ss:[esp+A]	
004010BD	68 78604000	push easy_crackme.406078	406078: "sy"
004010C2	51	push ecx	
004010C3	E8 88000000	call easy_crackme.401150	
004010C8	83C4 0C	add esp, C	
004010CD	85C0	test eax, eax	
004010CD	75 66	jne easy_crackme.401135	
004010CF	53	push ebx	
004010D0	56	push esi	
004010D1	BE 6C604000	mov esi, easy_crackme.40606C	40606C: "R3versing"
004010D6	8D4424 10	lea eax, dword ptr ss:[esp+10]	
004010DA	8A10	mov dl, byte ptr ds:[eax]	
004010DC	8A1E	mov bl, byte ptr ds:[esi]	
004010DE	8ACA	mov cl, dl	
004010E0	3AD3	cmp dl, bl	
004010E2	75 1E	jne easy_crackme.401102	
004010E4	84C9	test cl, cl	
004010E6	74 16	ja easy_crackme.4010FE	
004010E8	8A50 01	mov dl, byte ptr ds:[eax+1]	
004010EB	8A5E 01	mov bl, byte ptr ds:[esi+1]	
004010EE	8ACA	mov cl, dl	
004010F0	3AD3	cmp dl, bl	
004010F2	75 0E	jne easy_crackme.401102	
004010F4	83C0 02	add eax, 2	
004010F7	83C6 02	add esi, 2	
004010FA	84C9	test cl, cl	
004010FC	75 DC	jne easy_crackme.40100A	
004010FE	33C0	xor eax, eax	
00401100	EB 05	jmp easy_crackme.401107	
00401102	1BC0	sbb eax, eax	
00401104	83D8 FF	sbb eax, FFFFFFFF	
00401107	5E	pop esi	
00401108	58	pop ebx	
00401109	85C0	test eax, eax	
0040110B	75 28	jne easy_crackme.401135	
0040110D	807C24 04 45	cmp byte ptr ss:[esp+4], 45	45: 'E'
00401112	75 21	jne easy_crackme.401135	

문자열 비교 루틴에 bp 설정 후 분석 재 실행

00401080	807C24 05 61	cmp byte ptr ss:[esp+5],61	61: 'a'
00401085	75 7E	jne easy_crackme.401135	
00401087	6A 02	push 2	
00401089	8D4C24 0A	lea ecx,dword ptr ss:[esp+A]	
0040108D	68 78604000	push easy_crackme.406078	406078: "sy"
004010C2	51	push ecx	
004010C3	E8 88000000	call easy_crackme.401150	
004010C8	83C4 0C	add esp,C	
004010C8	85C0	test eax,ecx	
004010C9	75 66	jne easy_crackme.401135	
004010CF	53	push ebx	
004010D0	56	push esi	
004010D1	BE 6C604000	mov esi,easy_crackme.40606C	40606C: "R3versing"
004010D6	8D4424 10	lea eax,dword ptr ss:[esp+10]	
004010DA	8A10	mov dl,byte ptr ds:[eax]	
004010DC	8A1E	mov bl,byte ptr ds:[esi]	
004010DE	8ACA	mov cl,dl	
004010E0	3AD3	cmp dl,bl	
004010E2	75 1E	jne easy_crackme.401102	
004010E4	84C9	test cl,cl	
004010E6	74 16	je easy_crackme.4010FE	
004010E8	8A50 01	mov dl,byte ptr ds:[eax+1]	
004010EB	8A5E 01	mov bl,byte ptr ds:[esi+1]	
004010EE	8ACA	mov cl,dl	
004010F0	3AD3	cmp dl,bl	
004010F2	75 0E	jne easy_crackme.401102	
004010F4	83C0 02	add eax,2	
004010F7	83C6 02	add esi,2	
004010FA	84C9	test cl,cl	
004010FC	75 DC	jne easy_crackme.4010DA	
004010FE	33C0	xor eax,ecx	
00401100	EB 05	jmp easy_crackme.401107	
00401102	18C0	sbb eax,ecx	
00401104	83D8 FF	sbb eax,FFFFFFFF	
00401107	5E	pop esi	
00401108	58	pop ebx	
00401109	85C0	test eax,ecx	
0040110A	75 28	jne easy_crackme.401135	
0040110D	807C24 04 45	cmp byte ptr ss:[esp+4],45	45: 'E'
00401112	75 21	jne easy_crackme.401135	
00401114	6A 40	push 40	
00401116	68 58604000	push easy_crackme.406058	406058: "EasyCrackMe"
00401118	68 44604000	push easy_crackme.406044	406044: "Congratulation !!"
00401120	57	push edi	
00401121	FF15 A0504000	call dword ptr ds:[<&MessageBoxA>]	
00401127	6A 00	push 0	
00401129	57	push edi	
0040112A	FF15 A4504000	call dword ptr ds:[<&EndDialog>]	
00401130	5F	pop edi	
00401131	83C4 64	add esp,64	
00401134	C3	ret	
00401135	6A 10	push 10	
00401137	68 58604000	push easy_crackme.406058	406058: "EasyCrackMe"
00401139	68 30604000	push easy_crackme.406030	406030: "Incorrect Password"
00401141	57	push edi	
00401142	FF15 A0504000	call dword ptr ds:[<&MessageBoxA>]	

현재 esp+0x4 부터 입력한 문자열이 들어있으며 esp+0x5 의 값이 ASCII 값 61, 즉 'a' 인지 확인함 → 입력 값의 두번째 글자는 'a' 이어야함.

004010AA	FF15 9C504000	call dword ptr ds:[<&GetDlgItemTextA>]	
00401080	807C24 05 61	cmp byte ptr ss:[esp+5],61	61: 'a'
00401085	75 7E	jne easy_crackme.401135	
00401087	6A 02	push 2	

1번 루틴의 조건에 충족하는 문자열로 다시 실행 후 두번째 루틴 확인

00401087	6A 02	push 2	
00401089	8D4C24 0A	lea ecx,dword ptr ss:[esp+A]	
0040108D	68 78604000	push easy_crackme.406078	406078: "sy"
004010C2	51	push ecx	
004010C3	E8 88000000	call easy_crackme.401150	
004010C8	83C4 0C	add esp,C	
004010C8	85C0	test eax,ecx	

현재 esp+0xA에 입력한 문자열의 3번째 문자부터 저장되어 있음. 이 문자열과 미리 저장되어있는 "5y" 와 비교연산

00401150	55	push ebp	
00401151	8BEC	mov ebp,esp	
00401153	57	push edi	
00401154	56	push esi	
00401155	53	push ebx	
00401156	8B4D 10	mov ecx,dword ptr ss:[ebp+10]	
00401159	E3 26	jecxz easy_crackme.401181	
00401158	8BD9	mov ebx,ecx	ecx: "0123456789; <=>"
0040115D	8B7D 08	mov edi,dword ptr ss:[ebp+8]	
00401160	8BF7	mov esi,edi	
00401162	33C0	xor eax,eax	
00401164	F2:AE	repne scasb	
00401166	F7D9	neg ecx	ecx: "0123456789; <=>"
00401168	03CB	add ecx,ebx	ecx: "0123456789; <=>"
0040116A	8BFE	mov edi,esi	
0040116C	8B75 0C	mov esi,dword ptr ss:[ebp+C]	
0040116F	F3:A6	repe cmpsb	
00401171	8A46 FF	mov al,byte ptr ds:[esi-1]	
00401174	33C9	xor ecx,ecx	ecx: "0123456789; <=>"
00401176	3A47 FF	cmp al,byte ptr ds:[edi-1]	
00401179	77 04	ja easy_crackme.40117F	
0040117B	74 04	je easy_crackme.401181	
0040117D	49	dec ecx	ecx: "0123456789; <=>"
0040117E	49	dec ecx	ecx: "0123456789; <=>"
0040117F	F7D1	not ecx	ecx: "0123456789; <=>"
00401181	8BC1	mov eax,ecx	ecx: "0123456789; <=>"
00401183	5B	pop ebx	
00401184	5E	pop esi	
00401185	5F	pop edi	
00401186	C9	leave	
00401187	C3	ret	

0019F7D0	00000064	
0019F7D4	0019F884	
0019F7D8	00401080	easy_crackme.00401080로 변환 (출발: ???)
0019F7DC	004010C8	easy_crackme.004010C8로 변환 (출발: easy_
0019F7E0	0019F7F2	"0123456789; <=>"
0019F7E4	00406078	"5y"
0019F7E8	00000002	
0019F7EC	00000111	
0019F7F0	31306161	

따라서 3, 4 번째 문자 = "5y" 이므로 두번째 루틴도 충족하는 문자열 다시 입력.

004010D1	BE 6C604000	mov esi,easy_crackme.40606C	esi: "R3versing", 40606C: "R3versing"
004010D6	8D4424 10	lea eax,dword ptr ss:[esp+10]	
004010DA	8A10	mov dl,byte ptr ds:[eax]	
004010DC	8A1E	mov bl,byte ptr ds:[esi]	esi: "R3versing"

이후 3번째 검사 루틴에서 5번째 문자부터 미리 저장된 "R3versing" 과 비교

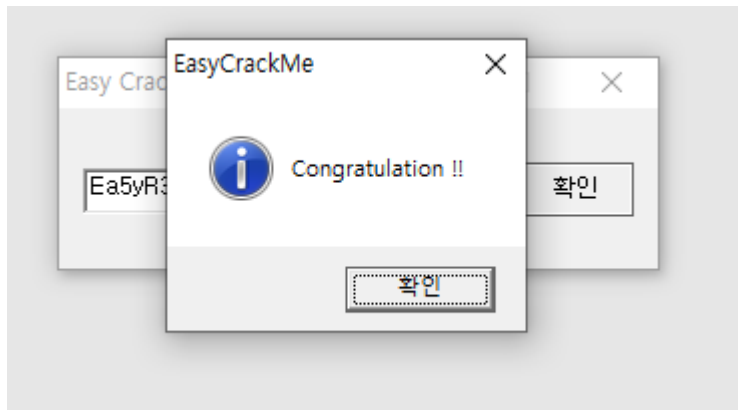
004010D1	BE 6C604000	mov esi,easy_crackme.40606C	
004010D6	8D4424 10	lea eax,dword ptr ss:[esp+10]	
004010DA	8A10	mov dl,byte ptr ds:[eax]	EAX 0019F7F4 "0123456789"
004010DC	8A1E	mov bl,byte ptr ds:[esi]	EBX 00000052 'R'
004010DE	8ACA	mov cl,dl	ECX 00000030 '0'
004010E0	3A03	cmp dl,bl	EDX 00000030 '0'
004010E2	75 1E	jne easy_crackme.401102	EBP 0019F884
			ESP 0019F7E4 "h\n*"
			ESI 0040606C "R3versing"
			EDI 002A0A68

5번째 문자부터 "R3versing"과 한 문자라도 다르면 "Incorrect Password" 출력

문자열 입력값을 "aa5yR3versing"로 다시 시도

00401108	75 28	jne easy_crackme.401135	
0040110D	807C24 04 45	cmp byte ptr ss:[esp+4],45	45: 'E'
00401112	75 21	jne easy_crackme.401135	

마지막 검사 루틴에서 esp+0x4에 들어있는 입력 문자열의 첫번째 문자와 "E"를 비교
문자열 입력값 = "Ea5yR3versing" 으로 다시 시도



Flag = "Ea5yR3versing"