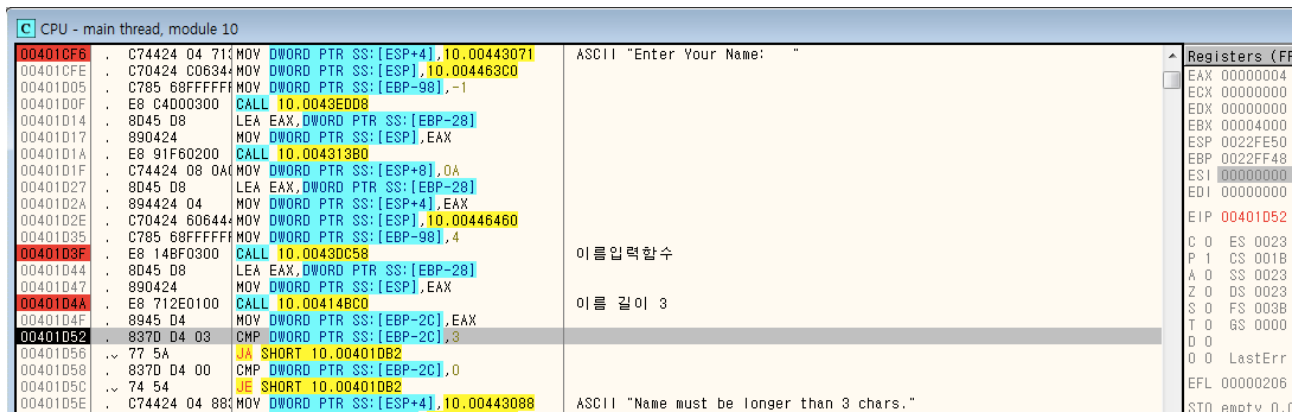


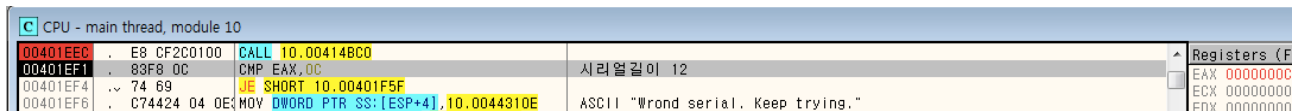
이름과 시리얼을 입력받고 인증하는 프로그램입니다.

패킹되어있지 않고, Dev C++로 개발되었음을 확인했고 분석을 진행합니다.



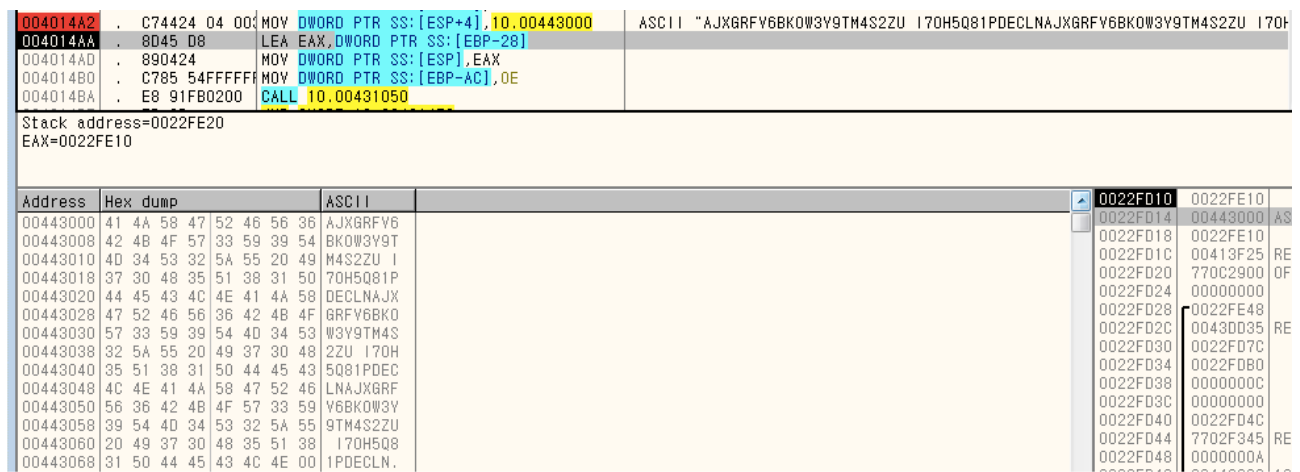
이름을 입력받는데 그 길이는 3글자 이상이어야 합니다.

3글자보다 적을 경우 에러메세지가 출력됩니다.



패스워드도 길이의 제한이 있습니다. 길이가 12(0x0C)여야 합니다.

만일 길이가 12가 아니면 에러메세지가 출력됩니다.



진행하다 보면 엄청 긴 문자열을 저장합니다.

이 문자열은 특정 문자열이 반복되는 형태입니다.

CPU - main thread, module 10			
0040138B	. E8 D0060300	CALL 10.00431A90	이름을 가져온다!
004013C0	. 89C3	MOV EBX,EAX	
004013C2	. 8B45 F8	MOV EAX,DWORD PTR SS:[EBP-8]	
004013C5	. 894424 04	MOV DWORD PTR SS:[ESP+4],EAX	
004013C9	. 8B45 0C	MOV EAX,DWORD PTR SS:[EBP+C]	
004013CC	. 890424	MOV DWORD PTR SS:[ESP],EAX	
004013CF	. E8 BC060300	CALL 10.00431A90	
004013D4	. 0FB00	MOVSX EAX,BYTE PTR DS:[EAX]	
004013D7	. 890424	MOV DWORD PTR SS:[ESP],EAX	
004013DA	. E8 91030100	CALL <JMP.&msvcrt.toupper>	toupper
004013DF	. 8803	MOV BYTE PTR DS:[EBX],AL	
004013E1	. 8D45 F8	LEA EAX,DWORD PTR SS:[EBP-8]	
004013E4	. FF00	INC DWORD PTR DS:[EAX]	
004013E6	. EB B6	JMP SHORT 10.0040139E	
004013E8	. 8B45 0C	MOV EAX,DWORD PTR SS:[EBP+C]	
004013EB	. 894424 04	MOV DWORD PTR SS:[ESP+4],EAX	
004013EF	. 8B45 08	MOV EAX,DWORD PTR SS:[EBP+8]	
004013F2	. 890424	MOV DWORD PTR SS:[ESP],EAX	
004013F5	. E8 06FD0200	CALL 10.00431100	
004013FA	. 8B45 08	MOV EAX,DWORD PTR SS:[EBP+8]	
004013FD	. 83C4 14	ADD ESP,14	
00401400	. 5B	POP EBX	
00401401	. 5D	POP EBP	
00401402	. C2 0400	RETN 4	

진행하다보면 다음과 같은 루틴이 나옵니다.

BreakPoint가 설정되어 있는 중간 부분을 보시면 'toupper'라는 함수 콜이 있습니다.

CPU - main thread, module 10			
0040138B	. E8 D0060300	CALL 10.00431A90	이름을 가져온다!
004013C0	. 89C3	MOV EBX,EAX	
004013C2	. 8B45 F8	MOV EAX,DWORD PTR SS:[EBP-8]	
004013C5	. 894424 04	MOV DWORD PTR SS:[ESP+4],EAX	
004013C9	. 8B45 0C	MOV EAX,DWORD PTR SS:[EBP+C]	
004013CC	. 890424	MOV DWORD PTR SS:[ESP],EAX	
004013CF	. E8 BC060300	CALL 10.00431A90	
004013D4	. 0FB00	MOVSX EAX,BYTE PTR DS:[EAX]	한글자만선택
004013D7	. 890424	MOV DWORD PTR SS:[ESP],EAX	
004013DA	. E8 91030100	CALL <JMP.&msvcrt.toupper>	toupper
004013DF	. 8803	MOV BYTE PTR DS:[EBX],AL	
004013E1	. 8D45 F8	LEA EAX,DWORD PTR SS:[EBP-8]	
004013E4	. FF00	INC DWORD PTR DS:[EAX]	
004013E6	. EB B6	JMP SHORT 10.0040139E	
004013E8	. 8B45 0C	MOV EAX,DWORD PTR SS:[EBP+C]	
004013EB	. 894424 04	MOV DWORD PTR SS:[ESP+4],EAX	
004013EF	. 8B45 08	MOV EAX,DWORD PTR SS:[EBP+8]	
004013F2	. 890424	MOV DWORD PTR SS:[ESP],EAX	
004013F5	. E8 06FD0200	CALL 10.00431100	
004013FA	. 8B45 08	MOV EAX,DWORD PTR SS:[EBP+8]	
004013FD	. 83C4 14	ADD ESP,14	
00401400	. 5B	POP EBX	
00401401	. 5D	POP EBP	
00401402	. C2 0400	RETN 4	
00401405	. 90	NOP	
00401406	. 55	PUSH EBP	
00401407	. 89E5	MOV EBP,ESP	

DS:[00561E14]=61 ('a')  
EAX=00561E14, (ASCII "abcd")

이름을 받아오고 한 글자만 EAX에 저장합니다.

CPU - main thread, module 10			
0040138B	. E8 D0060300	CALL 10.00431A90	이름을 가져온다!
004013C0	. 89C3	MOV EBX,EAX	
004013C2	. 8B45 F8	MOV EAX,DWORD PTR SS:[EBP-8]	
004013C5	. 894424 04	MOV DWORD PTR SS:[ESP+4],EAX	
004013C9	. 8B45 0C	MOV EAX,DWORD PTR SS:[EBP+C]	
004013CC	. 890424	MOV DWORD PTR SS:[ESP],EAX	
004013CF	. E8 BC060300	CALL 10.00431A90	
004013D4	. 0FB00	MOVSX EAX,BYTE PTR DS:[EAX]	한글자만선택
004013D7	. 890424	MOV DWORD PTR SS:[ESP],EAX	
004013DA	. E8 91030100	CALL <JMP.&msvcrt.toupper>	toupper
004013DF	. 8803	MOV BYTE PTR DS:[EBX],AL	
004013E1	. 8D45 F8	LEA EAX,DWORD PTR SS:[EBP-8]	

Registers (FPU)	
EAX	00000041
ECX	00000000
EDX	00561E14 ASCII "Abcd"
EBX	00561E14 ASCII "abcd"
ESP	0022FCF0
EBP	0022FD08
ESI	00000000
EDI	00000000
EIP	004013E1 10.004013E1
C 1 ES 0023 32bit 0(FFFF)	

toupper는 소문자를 대문자로 바꿔주는 함수로 'abcd'가 'Abcd'로 바뀐 것을 확인할 수 있습니다.

이름에 존재하는 소문자를 전부 대문자로 바꾸고 다음 연산을 진행하는 것으로 보아

대, 소문자의 구분은 없음을 알 수 있습니다.

CPU - main thread, module 10				Registers (FPU)
00401306	. E8 D060300	CALL 10.00431A90	이름을 가져온다!	EAX 0000004A
00401307	. 89C3	MOV EBX,EAX		ECX 00000009
00401308	. 8B45 F8	MOV EAX,DWORD PTR SS:[EBP-8]		EDX 00561E14 ASCII "WWWCCGJJrr
00401309	. 894424 04	MOV DWORD PTR SS:[ESP+4],EAX		EBX 00561E1C ASCII "rrrr
0040130A	. 8B45 0C	MOV EAX,DWORD PTR SS:[EBP+C]		ESP 0022FD00
0040130B	. 890424	MOV DWORD PTR SS:[ESP],EAX		EBP 0022FD08
0040130C	. E8 BC060300	CALL 10.00431A90		ESI 00000000
0040130D	. 0FB000	MOVSB EAX, BYTE PTR DS:[EAX]	한글자만선택	EDI 00000000
0040130E	. 890424	MOV DWORD PTR SS:[ESP],EAX		EIP 0040130F 10.0040130F
0040130F	. E8 91030100	CALL <JMP.&msvcrt.toupper>	toupper	
00401310	. 8B03	MOV BYTE PTR DS:[EBX],AL		

패스워드로 입력한 값도 대문자로 바뀌는 것을 확인할 수 있습니다.

CPU - main thread, module 10			
0040184E	. E8 50D50200	CALL 10.0042E0B0	이름 가져온다!!
00401853	. 0FB000	MOVSB EAX, BYTE PTR DS:[EAX]	여기가 분영 문자 하나로 패스워드 비교하는 구분갈다
00401856	. 894424 04	MOV DWORD PTR SS:[ESP+4],EAX	
0040185A	. 8D45 C8	LEA EAX,DWORD PTR SS:[EBP-38]	
0040185D	. 890424	MOV DWORD PTR SS:[ESP],EAX	
00401860	. E8 DB010300	CALL 10.00431A40	
00401865	. 8B45 90	MOV EAX,DWORD PTR SS:[EBP-70]	
00401868	. 89C2	MOV EDX,EAX	
0040186A	. 01D2	ADD EDX,EDX	
0040186C	. 8D0402	LEA EAX,DWORD PTR DS:[EDX+EAX]	
0040186F	. 8945 8C	MOV DWORD PTR SS:[EBP-74],EAX	
00401872	. 8B45 90	MOV EAX,DWORD PTR SS:[EBP-70]	
00401875	. 89C2	MOV EDX,EAX	
00401877	. 01D2	ADD EDX,EDX	
00401879	. 01C2	ADD EDX,EAX	
0040187B	. 8D42 03	LEA EAX,DWORD PTR DS:[EDX+3]	
0040187E	. 3B45 8C	CMP EAX,DWORD PTR SS:[EBP-74]	
00401881	. 0F8E E8020000	JLE 10.00401B6F	
00401887	. 8B45 8C	MOV EAX,DWORD PTR SS:[EBP-74]	
0040188A	. 894424 04	MOV DWORD PTR SS:[ESP+4],EAX	
0040188E	. 8B45 0C	MOV EAX,DWORD PTR SS:[EBP+C]	
00401891	. 890424	MOV DWORD PTR SS:[ESP],EAX	
00401894	. C785 54FFFFFF	MOV DWORD PTR SS:[EBP-AC],9	
0040189E	. E8 00D50200	CALL 10.0042E0B0	패스워드 가져온다!!
004018A3	. 0FB000	MOVSB EAX, BYTE PTR DS:[EAX]	어떠한 반복문이 지나니 한 글자가 줄었다
004018A6	. 894424 04	MOV DWORD PTR SS:[ESP+4],EAX	
004018AA	. 8D45 B8	LEA EAX,DWORD PTR SS:[EBP-48]	
EAX=00000041 Stack SS:[0022FD14]=00000000			
CPU - main thread, module 10			
00431A76	. 8B10	MOV BYTE PTR DS:[EAX],DL	다른버퍼에 옮겨서 비교하는거같다
00431A78	. 8B75 FC	MOV ESI,DWORD PTR SS:[EBP-4]	
00431A7B	. 8908	MOV EAX,EBX	
00431A7D	. 8B5D F8	MOV EBX,DWORD PTR SS:[EBP-8]	
00431A80	. 89EC	MOV ESP,EBP	
00431A82	. 5D	POP EBP	
00431A83	. C3	RETN	
00431A84	. 90	NOP	
00431A85	. 90	NOP	
00431A86	. 90	NOP	
00431A87	. 90	NOP	
00431A88	. 90	NOP	
00431A89	. 90	NOP	
00431A8A	. 90	NOP	
00431A8B	. 90	NOP	
00431A8C	. 90	NOP	
00431A8D	. 90	NOP	
00431A8E	. 90	NOP	
00431A8F	. 90	NOP	
00431A90	. 55	PUSH EBP	
00431A91	. 89E5	MOV EBP,ESP	
00431A93	. 53	PUSH EBX	
00431A94	. 83EC 04	SUB ESP,4	
00431A97	. 8B5D 08	MOV EBX,DWORD PTR SS:[EBP+8]	
00431A9A	. 8B13	MOV EDX,DWORD PTR DS:[EBX]	
00431A9C	. 8B42 FC	MOV EAX,DWORD PTR DS:[EDX-4]	
00431A9F	. C1E8 1F	SHR EAX,1F	
DL=41 ('A') DS:[00561E14]=EE			
CPU - main thread, module 10			
00431A76	. 8B10	MOV BYTE PTR DS:[EAX],DL	다른버퍼에 옮겨서 비교하는거같다
00431A78	. 8B75 FC	MOV ESI,DWORD PTR SS:[EBP-4]	
00431A7B	. 8908	MOV EAX,EBX	
00431A7D	. 8B5D F8	MOV EBX,DWORD PTR SS:[EBP-8]	
00431A80	. 89EC	MOV ESP,EBP	
00431A82	. 5D	POP EBP	
00431A83	. C3	RETN	
00431A84	. 90	NOP	
00431A85	. 90	NOP	
00431A86	. 90	NOP	
00431A87	. 90	NOP	
00431A88	. 90	NOP	
00431A89	. 90	NOP	
00431A8A	. 90	NOP	
00431A8B	. 90	NOP	
00431A8C	. 90	NOP	
00431A8D	. 90	NOP	
00431A8E	. 90	NOP	
00431A8F	. 90	NOP	
00431A90	. 55	PUSH EBP	
00431A91	. 89E5	MOV EBP,ESP	
00431A93	. 53	PUSH EBX	
00431A94	. 83EC 04	SUB ESP,4	
00431A97	. 8B5D 08	MOV EBX,DWORD PTR SS:[EBP+8]	
00431A9A	. 8B13	MOV EDX,DWORD PTR DS:[EBX]	
00431A9C	. 8B42 FC	MOV EAX,DWORD PTR DS:[EDX-4]	
00431A9F	. C1E8 1F	SHR EAX,1F	
DL=57 ('W') DS:[00563B94]=EE			

위 그림을 보면 이름으로 입력한 값의 한 글자를 가져와서 다른 버퍼에 옮깁니다.

패스워드 역시 같은 동작을 합니다.

CPU - main thread, module 10			Registers (FPU)
00414AC2	77 2C	JN SHORT 10.00414AF0	EAX 00000001
00414AC4	FC	CLD	ECX 00000001
00414AC5	807426 00	LEA ESI,DWORD PTR DS:[ESI]	EDX 00563894
00414AC9	80BC27 000000	LEA EDI,DWORD PTR DS:[EDI]	EBX 00000001
00414AD0	8B4D 10	MOV ECX,DWORD PTR SS:[EBP+10]	ESP 0022FCB4
00414AD3	8B75 EC	MOV ESI,DWORD PTR SS:[EBP+14]	EBP 0022FCB8
00414AD6	8B7D 0C	MOV EDI,DWORD PTR SS:[EBP+C]	ESI 00563A94 ASCII "AJXGRFVBK"
00414AD9	01CE	ADD ESI,ECX	EDI 00563894
00414ADB	3B08	CMP EBX,EBX	EIP 00414ADF 10.00414ADF
00414ADD	89D9	MOV ECX,EBX	C 0 ES 0023 32bit 0(FFFFFFFF)
00414ADF	F3:A6	REPE CMPS BYTE PTR ES:[EDI],BYTE PTR DS:[ESI] 저장한A를 꺼내온다	P 1 CS 001B 32bit 0(FFFFFFFF)
00414AE1	74 1A	JE SHORT 10.00414AF0	A 0 SS 0023 32bit 0(FFFFFFFF)
00414AE3	FF45 10	INC DWORD PTR SS:[EBP+10]	Z 1 DS 0023 32bit 0(FFFFFFFF)
00414AE6	8B45 10	MOV EAX,DWORD PTR SS:[EBP+10]	S 0 FS 003B 32bit 7FFDF000(FF)
00414AE9	01D8	ADD EAX,EBX	T 0 GS 0000 NULL
00414AEB	3B45 F0	CMP EAX,DWORD PTR SS:[EBP-10]	D 0
00414AEE	76 E0	JBE SHORT 10.00414ADD	0 0 LastErr ERROR_SUCCESS (00)
00414AF0	83C4 08	ADD ESP,8	EFL 00000246 (NO,NB,E,BE,NS,PE)
00414AF3	B8 FFFFFFFF	MOV EAX,-1	ST0 empty 0.0
00414AF8	5B	POP EBX	ST1 empty 0.0
00414AF9	5E	POP ESI	ST2 empty 0.0
00414AFA	5F	POP EDI	ST3 empty 0.0
00414AFB	5D	POP EBP	ST4 empty 0.0
00414AFC	C3	RETN	ST5 empty 0.0
00414AFD	8B45 10	MOV EAX,DWORD PTR SS:[EBP+10]	ST6 empty 1.000000000000000000
00414B00	83C4 08	ADD ESP,8	ST7 empty 0.0
00414B03	5B	POP EBX	3 2 1 0 E

그렇게 따로 버퍼에 저장한 값들을 위에서 저장한 특정 문자열의 반복과 비교하게 됩니다.

이전 그림의 패스워드가 저장된 버퍼와 밑줄 친 버퍼의 주소가 같은 것을 확인할 수 있습니다.

00414ADF	F3:A6	REPE CMPS BYTE PTR ES:[EDI],BYTE PTR DS:[ESI] 저장한A를 꺼내온다	00414AE1	74 1A	JE SHORT 10.00414AFD
00414AE1	74 1A	JE SHORT 10.00414AFD	00414AE3	FF45 10	INC DWORD PTR SS:[EBP+10]
00414AE3	FF45 10	INC DWORD PTR SS:[EBP+10]	00414AE6	8B45 10	MOV EAX,DWORD PTR SS:[EBP+10]
00414AE6	8B45 10	MOV EAX,DWORD PTR SS:[EBP+10]	00414AE9	01D8	ADD EAX,EBX
00414AE9	01D8	ADD EAX,EBX	00414AEB	3B45 F0	CMP EAX,DWORD PTR SS:[EBP-10]
00414AEB	3B45 F0	CMP EAX,DWORD PTR SS:[EBP-10]	00414AEE	76 E0	JBE SHORT 10.00414ADD
00414AEE	76 E0	JBE SHORT 10.00414ADD	00414AF0	83C4 08	ADD ESP,8
00414AF0	83C4 08	ADD ESP,8	00414AF3	B8 FFFFFFFF	MOV EAX,-1
00414AF3	B8 FFFFFFFF	MOV EAX,-1	00414AF8	5B	POP EBX
00414AF8	5B	POP EBX	00414AF9	5E	POP ESI
00414AF9	5E	POP ESI	00414AFA	5F	POP EDI
00414AFA	5F	POP EDI	00414AFB	5D	POP EBP
00414AFB	5D	POP EBP	00414AFC	C3	RETN
00414AFC	C3	RETN	00414AFD	8B45 10	MOV EAX,DWORD PTR SS:[EBP+10]
00414AFD	8B45 10	MOV EAX,DWORD PTR SS:[EBP+10]	00414B00	83C4 08	ADD ESP,8
00414B00	83C4 08	ADD ESP,8	00414B03	5B	POP EBX
00414B03	5B	POP EBX	Jump is taken 00414AFD=10.00414AFD		
ECX=00000001 (decimal 1.) DS:[ESI]=[00563A9F]=57 ('W') ES:[EDI]=[00563B94]=57 ('W')					

문자열에서 입력한 값과 같은 값을 찾을 때까지 반복하다가 찾으면 루틴을 빠져나옵니다.

루틴 도중에는 카운트를 증가시켜 입력한 값과 같은 값이 문자열 내에서 어디에 있는지 체크합니다.

그리고 루틴을 빠져나올 때 카운트 값을 버퍼에 저장합니다.

00401B4C	. 8B55 84	MOV EDX,DWORD PTR SS:[EBP-7C]
00401B4F	. 8B45 88	MOV EAX,DWORD PTR SS:[EBP-78]
00401B52	. 29D0	SUB EAX,EDX
00401B54	. 890424	MOV DWORD PTR SS:[ESP],EAX
00401B57	. E8 14C00300	CALL 10.00430B70
00401B5C	. 83F8 05	CMP EAX,5
00401B5F	. 7E 04	JLE SHORT 10.00401B65
00401B61	. C645 97 00	MOV BYTE PTR SS:[EBP-69],0
00401B65	> 8D45 8C	LEA EAX,DWORD PTR SS:[EBP-74]
00401B68	. FF00	INC DWORD PTR DS:[EAX]
00401B6A	. ^ E9 03FDFFFF	JMP 10.00401B72
00401B6F	> 8D45 90	LEA EAX,DWORD PTR SS:[EBP-70]
00401B72	. FF00	INC DWORD PTR DS:[EAX]
00401B74	. ^ E9 3BFCFFFF	JMP 10.004017B4
00401B79	> 0FB645 97	MOVZX EAX,BYTE PTR SS:[EBP-69]
00401B7D	. 8985 F4FEFFFF	MOV DWORD PTR SS:[EBP-10C],EAX
00401B83	. 8D45 B8	LEA EAX,DWORD PTR SS:[EBP-48]
00401B86	. 890424	MOV DWORD PTR SS:[ESP],EAX
00401B89	. C785 54FFFFFF	MOV DWORD PTR SS:[EBP-AC],0B
00401B93	. E8 D8FC0200	CALL 10.00431870
00401B98	. 8D45 C8	LEA EAX,DWORD PTR SS:[EBP-38]
EDX=00000026		
EAX=00000026		

EDX와 EAX는 이름과 패스워드로 입력한 값의 한 글자가 특정 문자열에서 어디에 있는지 index값을 가집니다.

EAX와 EDX를 빼고 그 값을 5와 비교합니다.

00401B32	. ^ E9 9C000000	JMP 10.00401B03		EAX 00000000
00401B37	> 8D45 98	LEA EAX,DWORD PTR SS:[EBP-68]		ECX 00000000
00401B3A	. 890424	MOV DWORD PTR SS:[ESP],EAX		EDX 00000000
00401B3D	. C785 54FFFFFF	MOV DWORD PTR SS:[EBP-AC],9		EBX 00004000
00401B47	. E8 24FD0200	CALL 10.00431870		ESP 0022FD10
00401B4C	. 8B55 84	MOV EDX,DWORD PTR SS:[EBP-7C]		EBP 0022FE48
00401B4F	. 8B45 88	MOV EAX,DWORD PTR SS:[EBP-78]		ESI 00000000
00401B52	. 29D0	SUB EAX,EDX		EDI 00000000
00401B54	. 890424	MOV DWORD PTR SS:[ESP],EAX	eax=b	EIP 00401B65
00401B57	. E8 14C00300	CALL 10.00430B70		C 1 ES 0023
00401B5C	. 83F8 05	CMP EAX,5		P 0 CS 001B
00401B5F	. 7E 04	JLE SHORT 10.00401B65		A 1 SS 0023
00401B61	. C645 97 00	MOV BYTE PTR SS:[EBP-69],0		Z 0 DS 0023
00401B65	> 8D45 8C	LEA EAX,DWORD PTR SS:[EBP-74]		S 1 FS 003B
00401B68	. FF00	INC DWORD PTR DS:[EAX]		T 0 GS 0000
00401B6A	. ^ E9 03FDFFFF	JMP 10.00401B72		D 0
00401B6F	> 8D45 90	LEA EAX,DWORD PTR SS:[EBP-70]		

현재 이름으로 WCJR을 입력했고 시리얼은 WWWCCCJJJRRR로 입력했습니다.

문자열에서 W의 위치는 변하지 않기 때문에 EAX는 0의 값을 가지게 되고 분기문에서 점프하게 됩니다.

00401B68	. FF00	INC DWORD PTR DS:[EAX]
00401B6A	. ^ E9 03FDFFFF	JMP 10.00401B72
00401B6F	> 8D45 90	LEA EAX,DWORD PTR SS:[EBP-70]
00401B72	. FF00	INC DWORD PTR DS:[EAX]
00401B74	. ^ E9 3BFCFFFF	JMP 10.004017B4
00401B79	> 0FB645 97	MOVZX EAX,BYTE PTR SS:[EBP-69]
00401B7D	. 8985 F4FEFFFF	MOV DWORD PTR SS:[EBP-10C],EAX
00401B83	. 8D45 B8	LEA EAX,DWORD PTR SS:[EBP-48]
00401B86	. 890424	MOV DWORD PTR SS:[ESP],EAX
00401B89	. C785 54FFFFFF	MOV DWORD PTR SS:[EBP-AC],0B
00401B93	. E8 D8FC0200	CALL 10.00431870
00401B98	. 8D45 C8	LEA EAX,DWORD PTR SS:[EBP-38]
Stack DS:[0022FDD4]=00000006		

이름과 패스워드의 문자열의 위치가 같거나 +5의 위치를 가지는 값의 경우 밑줄 친 부분의 값이 증가합니다.

00401B68	.	FF00	INC DWORD PTR DS:[EAX]
00401B6A	^	E9 03FDFFFF	JMP 10.00401872
00401B6F	>	8D45 90	LEA EAX,DWORD PTR SS:[EBP-70]
00401B72	.	FF00	INC DWORD PTR DS:[EAX]
00401B74	^	E9 3BFCFFFF	JMP 10.004017B4
00401B79	>	0FB645 97	MOVZX EAX,BYTE PTR SS:[EBP-69]
00401B7D	.	8985 F4FEFFFF	MOV DWORD PTR SS:[EBP-10C],EAX
00401B83	.	8D45 B8	LEA EAX,DWORD PTR SS:[EBP-48]
00401B86	.	890424	MOV DWORD PTR SS:[ESP],EAX
00401B89	.	C785 54FFFFFF	MOV DWORD PTR SS:[EBP-AC],0B
00401B93	.	E8 D8FC0200	CALL 10.00431870
00401B98	.	8D45 C8	LEA EAX,DWORD PTR SS:[EBP-38]
00401B9B	.	890424	MOV DWORD PTR SS:[ESP],EAX
00401B9E	.	C785 54FFFFFF	MOV DWORD PTR SS:[EBP-AC],0D
00401BA8	.	E8 C3FC0200	CALL 10.00431870
00401BAD	.	8D45 D8	LEA EAX,DWORD PTR SS:[EBP-28]
00401BB0	.	890424	MOV DWORD PTR SS:[ESP],EAX
00401BB3	.	C785 54FFFFFF	MOV DWORD PTR SS:[EBP-AC],-1
00401BBD	.	E8 AEFC0200	CALL 10.00431870
00401BC2	.	8B95 F4FEFFFF	MOV EDX,DWORD PTR SS:[EBP-10C]

00401872=10.00401872
----------------------

Address	Hex dump	ASCII
0022FDD4	0C 00 00 00 03 00 00 00	....L...

이름과 시리얼이 문자열 내에서 모두 같은 위치에 존재하기 때문에

버퍼의 값은 시리얼의 길이인 0x0C(12)가 되고 점프 구문으로 점프하게 됩니다.

알고리즘을 정리하도록 하겠습니다.

1. 이름과 시리얼을 입력받는다.
2. 특정 문자열 내에서 이름의 각 글자들이 어디에 위치하는지 찾아낸다.
3. 시리얼 역시 2와 같은 동작을 한다.
4. 이름 한글자당 시리얼 3글자의 위치를 비교한다.
5. 그 위치가 문자열에서 +-5 이내의 위치면 카운트를 1 증가시킨다.
6. 카운트가 시리얼의 길이와 같으면 인증을 성공시킨다.

와 같은 루틴이 존재하는 프로그램입니다.

이를 프로그램으로 만들어서 실행한 뒤 인증하면 성공입니다!

```

[areong@Areong-ui-MacBook-AirDesktop]$ gcc -o test test.c
[areong@Areong-ui-MacBook-AirDesktop]$ ./test
===== J =====
: 1
: 2
: 3
: 4
A : 5
J : 6
X : 7
G : 8
R : 9
F : 10
V : 11
===== J =====
===== R =====
: 4
A : 5
J : 6
X : 7
G : 8
R : 9
F : 10

```