

Target : NBS C++ Crackme #1
 Lang. : c++
 Tools : OllyDBG , PEiD
 Cracker: allko

Learn Reverse Engineering Series- lesson five

Introduction :

مرحبا...من جديد أطل عليكم في درس جديد لتعلم الهندسة العكسية...درس اليوم يختلف قليلا عن سابقه فالبرنامج مكتوب بلغة C++ وليس بالاسمبلي...وكما ذكرت سابقا فان الاسمبلي أسهل لنا...لان الاسمبلي تعتبر من لغات (LLL : Low Level Language) أي انه لعمل برنامج ما باستخدام الاسمبلي فانك بحاجة لأسطر برمجية اقل مما ستحتاجه لتنفيذ نفس البرنامج بال (HLL) ..شيئ آخر اود ذكره هو ان طريقة التشفير لا تختلف كثيرا رغم انها اطول...قد يكون هذا الدرس الاخير من هذه النوعية لانه اعتقد بانتهاك من هذا الدرس ستصبح جاهزا لكسر هذا النوع من الحماية بنفسك...

: 000

Microsoft Visual C++ 6.0

كالعادة تفحص البرنامج باستخدام PEiD والنتيجة

: 001

شغل البرنامج...من قائمة file اختر register...حسنا هناك message بـ username و s/n اكتب أي شيء لتظهر لك شاشة الـ error...من القائمة السابقة اختر about...كما ترى فالمبرمج سمح لك باستخدام أي طريقة (هل تفكر في الـ patching ؟ لا تفرح كثيرا فلن اسمح لك بذلك!!) نريد عمل keygenerater وليس فقط تغيير هذا الامر او ذلك...

: 010

والان حمل البرنامج الى olly...لاحظ ان البرنامج مختلف عما عهدناه مع برامج asm...لا بأس...search for ثم all referenced text strings وابحث عن رسالة الخطأ...لاحظ ان هناك 3 رسائل...اختر الاولى...

00401B9D	. 6A 10	PUSH 10	Style = MB_OK+MB_ICONHAND+MB_APPLMODAL
00401B9F	. 68 80824000	PUSH CrackMe.00408280	Title = "Error"
00401BA4	. 68 88894000	PUSH CrackMe.00408988	Text = "Invalid name/key, please try again."
00401BA9	. 50	PUSH EAX	hOwner
00401BAA	. FF15 F0704000	CALL DWORD PTR DS:[<&USER32.Message]	MessageBoxA

ملاحظة : لجعل الـ calls والـ jumps تظهر بلونين ازرقي واصفر كما بالصورة :
 Right click → appearance → highlighting → jumps 'n' calls

والآن ابحث عما قد يكون مفيدا...ان كنت قد قرأت وفهمت الدروس السابقة فانت تعلم ما نبحت عنه...نريد الدالة التي تستقبل الـ user والـ pass...بالنظر للعلاى قليلا ستجد اثنتين : GetWindowTextA

00401B19	. 68 FF000000	PUSH 0FF	Count = FF (255.)
00401B1E	. 51	PUSH ECX	Buffer
00401B1F	. 52	PUSH EDX	hWnd => NULL
00401B20	. FFD6	CALL ESI	GetWindowTextA
00401B22	. 8B0D B8924000	MOV ECX,DWORD PTR DS:[408B8924000]	
00401B28	. 8D8424 1C0400	LEA EAX,DWORD PTR SS:[ESP+41C]	
00401B2F	. 68 FF000000	PUSH 0FF	Count = FF (255.)
00401B34	. 50	PUSH EAX	Buffer
00401B35	. 51	PUSH ECX	hWnd => NULL
00401B36	. FFD6	CALL ESI	GetWindowTextA

ضع BreakPoint كما هو واضح ثم شغل البرنامج F9 والان ادخل الـ yوزنيم allko والباسوورد 123456 ثم اضغط register . ها قد عدنا...تتبع (trace over) باستخدام F8 مرة واحدة...لاحظ بارامتر الـ buffer يبدو ان الـ yوزنيم يخزن في الـ stack على عنوان 0012F7F8 والان تتبع حتى تنتهي من الدالة الاولى...لاحظ ان EAX يحوي عدد characters أي 5 .تتبع مرة اخرى حتى تنتهي من الدالة الثانية ولاحظ بارامتر الـ buffer وعنوان السيريال : 0012F9F8 وألان لاحظ الأوامر التالية (تذكر ان تنتبه الى قسم التعليقات بالاسفل)

00401B38	MOV BL,BYTE PTR SS:[ESP+21C]	يتم نقل أول حرف من اليوزر "a" إلى u[1] إلى BL
00401B3F	XOR EDX,EDX	صفر EDX
00401B41	MOV AL,BYTE PTR SS:[ESP+41C]	يتم نقل أول حرف من السيرال s[1] إلى AL
00401B48	TEST AL,AL	هل AL يساوي صفر؟
00401B4A	SETL DL	إذا كان مساويا للصفر فاجعل DL يساوي 1 (SET if Equal)
00401B4D	XOR EAX,EAX	صفر EAX
00401B4F	TEST BL,BL	هل BL يساوي صفر
00401B51	SETL AL	إذا كان مساويا للصفر فاجعل AL يساوي 1 (SET if Equal)
00401B54	OR EDX,EAX	نفذ عملية OR بين EDX,EAX
00401B56	JE SHORT CrackMe.00401B81	إذا كانت النتيجة 0 فاقفز.

كما ترى إذا لم تدخل أي اسم أو لم تدخل أي سيرال او لم تدخل الاثنين فستظهر رسالة خطأ والا ستقفز الى 00401B81 كما في حالتنا... ملاحظة : 1 OR 1 =1 , 1 OR 0=1 , 0 OR 1=1 , 0 OR 0=0

00401B81	MOV EDI,DWORD PTR DS:[<&KERNEL32.lstrlenA>]	يتم نقل عنوان إحدى دوال API (lstrlenA) إلى EDI
00401B87	LEA EDX,DWORD PTR SS:[ESP+21C]	الآن EDX يشير إلى عنوان اليوزرنيم
00401B8E	PUSH EDX ; /String	يتم دفع اليوزرنيم كبارامتر للدالة إلى الـ stack
00401B8F	CALL EDI ; \lstrlenA	يتم استدعاء الدالة لحساب طول اليوزرنيم
00401B91	CMP EAX,3	النتيجة خزنت في EAX وستقارن مع القيمة 3
00401B94	JGE SHORT CrackMe.00401BBF	إذا كان اليوزرنيم < 3 فاقفز إلى 00401BBF

فعلا فان allko اكبر من 3 © إذا سنتخطى رسالة الخطأ

00401BBF	LEA ECX,DWORD PTR SS:[ESP+41C]	الآن ECX يشير إلى عنوان السيرال
00401BC6	PUSH ECX	يتم دفع السيرال كبارامتر للدالة إلى الـ stack
00401BC7	CALL EDI	استدعي الدالة التي عنوانها موجود في EDI (نفس الدالة في الأعلى)
00401BC9	CMP EAX,22	النتيجة خزنت في EAX وستقارن مع القيمة 22 بالـ hex أي 34 عشري
00401BCC	JE SHORT CrackMe.00401BF7	إذا كان الـ basورد = 22 فاقفز إلى 00401BF7

حسنا ضع نقطة توقف عند امر القفز ثم اعد تشغيل البرنامج ctrl + F2 وشغل البرنامج واكتب التالي allko و 123456789012345678901234567890abcd هل لاحظت ذلك؟ لقد تحولت الاحرف abcd الى ABCD مباشرة فور كتابتها في المكان المخصص...تذكر هذا فقد ينفعنا...نحن الان عند امر القفز اضغط F8 لتنتقل الى 00401BF7

011 :

والآن سوف نبدأ بتفحص كود الحماية خطوة خطوة.

00401BF7	LEA EAX,DWORD PTR SS:[ESP+41C]	الآن EAX يشير إلى عنوان السيرال
00401BFE	XOR EBP,EBP	صفر EBP
00401C00	PUSH EAX	ادفع قيمة EAX (أي عنوان السيرال) الى المكس (stack)
00401C01	XOR BL,BL	صفر BL
00401C03	MOV DWORD PTR SS:[ESP+1C],EBP	انقل قيمة EBP (أي 0) الى الـ stack على عنوان 0012F5F4
00401C07	CALL EDI	استدعي الدالة التي عنوانها موجود في EDI (نفس الدالة في الأعلى)
00401C09	TEST EAX,EAX	هل EAX يساوي صفر؟ (لا فهو يحوي طول السيرال أي 0x22)
00401C0B	JLE SHORT CrackMe.00401C52	اقفز إذا كان صفرا
00401C0D	XOR ESI,ESI	صفر ESI
00401C0F	MOV AL,BYTE PTR SS:[ESP+ESI+41C]	يتم نقل حرف من السيرال s[1] إلى AL @
00401C16	INC BL	زود قيمة BL
00401C18	CMP AL,2D	قارن AL مع 0x2d (لو نظرت إلى جدول ASCII لعرفت أنها الشرطة -)
00401C1A	JE SHORT CrackMe.00401C39	اقفز إذا تساوا الى 00401C39
00401C1C	MOVSX ECX,WORD PTR SS:[ESP+18]	انقل القيمة SS:[0012F5F4]=0000 الى ECX
00401C21	PUSH EAX	ادفع قيمة EAX (أي s[1] = "1") الى المكس (stack)
00401C22	MOV BYTE PTR SS:[ESP+ECX+20],AL	انقل قيمة AL الى Stack SS:[0012F5F8]
00401C26	CALL CrackMe.00402510	استدعي العنوان 00402510 (اضغط F7 للذهاب الى هناك)

To be continue

@
انظر الى العنوان 401C47 هناك امر INC EBP يليه MOV ESI, BP أي : يستخدم EBP كعداد ويتم نقل قيمه كل مرة الى ESI أي ان ESI بدوره اصبح عددا...ولذلك فان القيمة [ESP+ESI+41C] تشير بداية الى اول حرف ، ثم مع ازدياد ESI ستشير الى الحروف التالية ...
والان تتبع trace into باستخدام F7 لنذهب الى ذلك العنوان

MOV AL, BYTE PTR SS:[ESP+4]	يتم نقل أول خانة في السيريال الى AL (انظر الى قسم التعليقات بالأسفل)
PUSH ESI	يتم دفع قيمة ESI الى ال stack
XOR ESI, ESI	مفر ESI
XOR ECX, ECX	وكذلك ECX
CMP AL, 5A	قارن AL مع القيمة 5A (هذه تكافئ Z أي أكبر قيمة لل capital letters) @@
SETG CL	إذا كان AL أكبر ، فاجعل CL يساوي 1 (SET if Greater)
XOR EDX, EDX	مفر EDX
CMP AL, 41	قارن AL مع القيمة 41 (هذه تكافئ A أي اصغر قيمة لل capital letters)
SETL DL	إذا كان AL اصغر ، فاجعل DL يساوي 1 (SET if Lower)
OR ECX, EDX	نفذ عملية OR بين ECX, EDX (أي: تحقق ما إذا كان حرفا كبيرا أم لا)
JE SHORT CrackMe.0040252E	إذا كانت النتيجة 1 (أي انه أكبر أو اصغر ، وبالتالي فهو ليس حرفا كبيرا) فاقفز و إلا ، إذا كانت النتيجة 0 فزود ESI
MOV ESI, 1	مفر ECX
XOR ECX, ECX	قارن AL مع 39 (هذه تكافئ 9 أي أكبر قيمة لرقم)
CMP AL, 39	إذا كان AL أكبر ، فاجعل CL يساوي 1 (SET if Greater)
SETG CL	مفر EDX
XOR EDX, EDX	قارن AL مع 30 (هذه تكافئ 0 أي اصغر قيمة لرقم)
CMP AL, 30	إذا كان AL اصغر ، فاجعل DL يساوي 1 (SET if Lower)
SETL DL	نفذ عملية OR بين ECX, EDX (أي: تحقق ما إذا كان رقما أم لا)
OR ECX, EDX	إذا كانت النتيجة 1 (أي انه أكبر أو اصغر ، وبالتالي فهو ليس رقما) فاقفز و إلا ، إذا كانت النتيجة 0 فزود ESI
JE SHORT CrackMe.00402541	قارن ESI مع 2 (إذا لم يكن حرفا أو رقما في ESI=2)
INC ESI	ارجع قيمة ESI الأصلية من ال stack
CMP ESI, 2	إذا تساوت ESI مع 2 فاجعل AL مساويا ل 1
POP ESI	عد من حيث بدأت
SETB AL	
RETN 4	

@@
تذكر ان الحروف الصغيرة تحول تلقائيا إلى كبيرة ، وبالتالي عندما نختبر الخانة الأولى (أو أي خانة أخرى) ما إذا كانت حرفا كبيرا أم لا فهذا يعني اننا نختبرها اذا كانت حرفا (بصورة عامة) ام لا.

هذه ال snippet of code وظيفتها بكل بساطة تفحص ما اذا كان الذي ادخلته رمزا ام لا...فهو يختبره بداية هل هو حرف ؟ ثم : هل هو رقم؟ فاذا لم يكن حرفا او رقما فهو رمز (مثلا ! او @ او # ...) وهذا غير مقبول.
والآن دعنا نكمل الإجرائية Procedure السابقة...

continued	
00401C2B TEST AL, AL	هل AL يساوي صفر ؟(انظر إلى امر SETB AL في الجدول السابق لتعرف المقصود)
00401C2D JNZ CrackMe.00401B96	تفحص راية الصفر Z اذا لم يكن صفرا (أي انه رمزا) فاقفز الى شاشة الخطأ
00401C33 INC DWORD PTR SS:[ESP+18]	زود قيمة [0012F5F4] SS: بمقدار واحد.
00401C37 JMP SHORT CrackMe.00401C40	اقفز الى 00401C40
00401C39 CMP BL, 5	سنصل الى هنا اذا كان الحرف الاول شرطة - قارن BL مع 5 @@@
00401C3C JNZ SHORT CrackMe.00401C68	اقفز الى رسالة الخطأ
00401C3E XOR BL, BL	مفر BL
00401C40 LEA EDX, DWORD PTR SS:[ESP+41C]	انقل عنوان السيريال الى EDX
00401C47 INC EBP	زود EBP
00401C48 PUSH EDX	ادفع قيمة EDX (عنوان السيريال) الى ال stack
00401C49 MOVSB ESI, BP	انقل قيمة BP الى ESI
00401C4C CALL EDI	استدعي الدالة التي عنوانها في EDI (انها دالة حساب الطول strlenA)
00401C4E CMP ESI, EAX	قارن قيمة EAX مع قيمة ESI (0x22 مع 0x1)
00401C50 JL SHORT CrackMe.00401C0F	إذا كان ESI اقل من 0x22 (أي مازال هناك خانات في S/n) فاقفز الى 00401C0F @@@@

00401C52	MOVSX EDX,WORD PTR SS:[ESP+18]	انسخ القيمة 0x1C الى EDX
00401C57	LEA EAX,DWORD PTR SS:[ESP+1C]	انسخ عنوان السيريال من المكسدس الى EAX (انظر الصورة بالاسفل)
00401C5B	PUSH EAX	ادفع قيمة EAX الى ال stack (المكدس)
00401C5C	MOV BYTE PTR SS:[ESP+EDX+20],0	انسخ القيمة 0 الى العنوان 12F614
00401C61	CALL EDI	استدعي دالة طول السترنج
00401C63	CMP EAX,1C	قارن قيمة EAX مع 0x1C (EAX=22)
00401C66	JE SHORT CrackMe.00401C91	إذا تساوى افقر الى 00401C91

0012F5EC	00000000
0012F5F0	00000000
0012F5F4	0000001C
0012F5F8	34333231
0012F5FC	44434241
0012F600	48474645
0012F604	4C4B4A49
0012F608	504F4E4D
0012F60C	54535251
0012F610	58575655

1	31	I	49	V	56
2	32	J	4A	W	57
3	33	K	4B	X	58
4	34	L	4C		
5	35	M	4D		
A	41	N	4E		
B	42	O	4F		
C	43	P	50		
D	44	Q	51		
E	45	R	52		
F	46	S	53		
G	47	T	54		
H	48	U	55		

انظر الى امر القفز على عنوان C1A00401... هذه
القفزة ثاني بك هنا... ماذا يعني ذلك؟ اذا كان ال char(character) الذي يتم اختياره شرطة فاقفز الى هنا... ثم سيتم
اختبار: هل ترتيب هذا ال char يساوي 5؟ اذا نستنتج ان الخانة الخامسة هي الشرطة - ... ليس هذا فقط... لاحظ امر
XOR BL,BL... اذا كانت الخانة هي الشرطة وترتيبها 5 سينم تصفير BL وبالتالي في الدورة الثانية أي عند اعادة الاختبار
من جديد للخانة التالية ستكون قيمته 1 (قيمته 0 لكن هناك امر inc bl) أي نستنتج ان كل خانة من مضاعفات ال 5 يجب
ان تكون - ... اذن ضع نقطة توقف عن العنوان 401C0F ثم اعد التشغيل ctrl+F2 والان ضع اليوزرنيك alko والسيريال
1234-abcd-efgh-ijkl-mnop-qrst-uvwx جيد... ها قد عدنا عند العنوان 401C0F تتبع trace over باستخدام F8

@@@
لاختصار الوقت تتبع باستخدام F9 (مع وجود نقطة توقف عند العنوان 401C0F) وراقب ESI وعندما تصل الى 21 فاتوقف
عن استخدام F9 واستخدم F8

@@@@
قيمة EAX تساوي 0x22 أي 34 عشري... يتم مقارنتها مع 0x1C أي 28 عشري... لاحظ العنوان 401C22 عند هذا العنوان
ينقل ال char الذي تجاوز امر cmp al,2D (أي ال char الذي ليس شرطة) الى العنوان... هذا يعني ان السيريال بدون
الشرطات يتم تخزينه هناك... لاحظ ان هناك 6 شرطات اذن طول السيريال كان 34 اصبح 34=6-34 أي 0x1c بال hex
وهذا يفسر هذا الامر...

100 :

عملية التشفير الفعلية :
الى الآن لم يكن هناك عمليات تشفير فعلية... كل ما سبق هو اجرائيات للتحقق من بعض الامور كالطول ووجود الشرطة
والخلو من الرموز وغيرها...
نحن الان في كوكب المريخ عفوا اقصد عند العنوان 401C91 . الأسطر الخمس التالية تتبع دالة lstrcpyA التي تقوم
بنسخ chars 3 (متضمنة 0 بالنهاية... أي عمليا تنسخ 2 فقط) من السيريال المدخل وتخزنهما في 12F5EC (لاحظ قيمة
EAX)

00401C95	. 6A 03	PUSH 3	[n = 3 allko String2 = "1234ABCEFGHIJKLMNOPQRSTUVWXYZ" String1 lstrcpyA
00401C97	. 8D4424 14	LEA EAX,DWORD PTR SS:[ESP+14]	
00401C9B	. 52	PUSH EDI	
00401C9C	. 50	PUSH EAX	
00401C9D	. FF15 30704000	CALL DWORD PTR DS:[<<<KERNEL32.L	

بعد هذه الدالة بسطرين هناك اتصال
402603 والتي بدورها هي اتصال لدالة أخرى هي 402578... وهذه الصورة هي للإجرائية الثانية ضمن الاجرائية
الاساسية التي تبدأ عند 402578

004025C4	>	833D 1C8D4000	CMP DWORD PTR DS:[408D1C],1
004025C8	·	7E 0C	JLE SHORT CrackMe.004025D9
004025CD	·	6A 04	PUSH 4
004025CF	·	56	PUSH ESI
004025D0	·	E8 F1010000	CALL CrackMe.004027C6
004025D5	·	59	POP ECX
004025D6	·	59	POP ECX
004025D7	·	EB 0B	JMP SHORT CrackMe.004025E4
004025D9	>	A1 108B4000	MOV EAX,DWORD PTR DS:[408B10]
004025DE	·	8A0470	MOV AL,BYTE PTR DS:[EAX+ESI*2]
004025E1	·	83E0 04	AND EAX,4
004025E4	>	85C0	TEST EAX,EAX
004025E6	·	74 00	JE SHORT CrackMe.004025F5
004025E8	·	8D049B	LEA EAX,DWORD PTR DS:[EBX+EBX*4]
004025EB	·	8D5C46 D0	LEA EBX,DWORD PTR DS:[ESI+EAX*2-30]
004025EF	·	0FB637	MOVZX ESI,BYTE PTR DS:[EDI]
004025F2	·	47	INC EDI
004025F3	·	EB CF	JMP SHORT CrackMe.004025C4
004025F5	>	83FD 2D	CMP EBP,2D
004025F8	·	8BC3	MOV EAX,EBX
004025FA	·	75 02	JNZ SHORT CrackMe.004025FE
004025FC	·	F7D8	NEG EAX
004025FE	>	5F	POP EDI
004025FF	·	5E	POP ESI
00402600	·	5D	POP EBP
00402601	·	5B	POP EBX
00402602	·	C3	RETN

allko34@hotmail.com

حسنا لن اقوم بشرحها تفصيليا لثلاثة اسباب.اولا هذا الدرس اصبح طويلا ولا اريدك ان تصاب بالملل.ثانيا هذه الدالة او هذه الاجرائيات غير مرتبة او لنقل من الصعب تحديد الهدف من كل امر بمفرده لكن يمكنك ان ترى النتيجة التي تحصل عليها بالنهاية وبقليل من المحاولة والخطا...لاحظت ان هذه الاجرائيات بها حلقتان فقط (أي تنفذ مرتين) ولاحظت وجود القيم 1(0x31) و 2 (0x32) وهما كما تلاحظ اول خانتين في السيريال((ضع نقطة توقف هنا ثم اعد التشغيل وحاول كل مرة بسيريال مختلف لكن لا تحاول وضع احرف ك ٧4 مثلا لان العمليات الرياضية ستنتج صفرا...))أي انهما 12 لكن كما تعلم ف 12 تكافئ 0xC أي ان اول خانتين يتم تحويلها الى الصورة المكافئة بالسداسي عشري (مثلا لو كان اول خانتين 27 ستحصل على 1B بدلا من C). حسنا ها قد عدنا من هذه الدالة نحن الآن عند العنوان 401CAD

00401CAD	·	8BF8	MOV EDI,EAX
00401CAF	·	83C4 04	ADD ESP,4
00401CB2	·	33D2	XOR EDX,EDX
00401CB4	·	66:83FF 0C	CMP DI,0C
00401CB8	·	0F9FC2	SETG DL
00401CB8	·	33C0	XOR EAX,EAX
00401CBD	·	66:83FF 02	CMP DI,2
00401CC1	·	0F9CC0	SETL AL
00401CC4	·	0BD0	OR EDX,EAX
00401CC6	·	75 A0	JNZ SHORT CrackMe.00401C68

كما ترى يتم مقارنة القيمة التي حصلنا عليها للتو (0xC) مرة مع C (لا، لا تعتقد انها تقارن مع نفسها فالقيمة C موجودة ضمن بنية البرنامج ولو غيرت السيريال لن تتأثر) ومرة مع 2 وبقليل من الملاحظة نستنتج انها يجب ان تكون اصغر من C واكبر من 2 اذن لو كان اول خانتين من السيريال 19 ستحولان الى سداسي عشري أي 13 وهذه اكبر من C اذن لن يقبلها البرنامج.

00401CD2	·	68 E4924000	PUSH CrackMe.004092E4	[String2 = "" String1 IstrcpyA
00401CD7	·	52	PUSH EDX	
00401CD8	·	FFD6	CALL ESI	[String2 = "" String1 IstrcpyA
00401CDA	·	8D4424 16	LEA EAX,DWORD PTR SS:[ESP+16]	
00401CDE	·	68 E4924000	PUSH CrackMe.004092E4	
00401CE3	·	50	PUSH EAX	
00401CE4	·	FFD6	CALL ESI	

هنا يتم نسخ 0 (string 2) الى 12F5EC (string 1) في الدالة الاولى . والامر مشابه للدالة الثانية.(string 1: 12F5F2)
(
اوامر ال mov الثلاثة التالي (بدئا من 401cE9) كما بالصورة :

00401CE6	·	0FBFF7	MOVZX ESI,DI
00401CE9	·	0FBE8424 1C0	MOVZX EAX,BYTE PTR SS:[ESP+21C]
00401CF1	·	8A4C34 1C	MOV CL,BYTE PTR SS:[ESP+ESI+1C]
00401CF5	·	8A5434 1D	MOV DL,BYTE PTR SS:[ESP+ESI+1D]

الاول ينقل اول حرف من اليوزر نيم الى EAX الثاني يقوم بنسخ الحرف I الى CL...وبالنظر الى السيريال بعد رفع الشرط 1234ABCDEF GHIJ KLMNOPQRST UVWX الحرف I هو الخانة رقم 13 (عشري) . الامر التالي ينسخ الحرف I الى الخانة 14...
ان دقت النظر في الامر لوجدت ان عنوان الحرف الذي يتم نسخه هو SS:[ESP+ESI+1C] حسنا القيمة 1C ثابتة اذن اتركها اما ESP فمحتواه 22 وهو طول السيريال لكن قبل ازالة الشرط ايضا هذا لا فائدة منه اما ESI فيحتوي على C أي 12...اذن القيمة الاولى هي C+1 والثانية C+2 .
بعد ذلك بسطرين هناك دالة wsprintfA التي تقوم بتحويل char الى قيمته بال ascii ولو دقت النظر جيدا اثناء تتبعك لها باستخدام F8 لوجدتها تاخذ بارامتر عن طريق EAX الذي سبق وخزنا الحرف الاول من الاسم فيه...اذن الدالة ستحول حرف a الى ال 61 (انتبه ان اليوزر نيم بقي small letter كما هو).
اربعة اسطر اخرى وتجد دالة مقارنة تعودنا عليها. انها IstrcmpA دق اثناء تتبعك لها لتجد ان البارامتر الاول هو 61 والثاني IJ ...ماذا تستنتج؟

اول خانتين في السيريال هما 12 أي c اذن الخانتين s[c],s[c+1] يجب ان يطابقا قيمة اول حرف في اليوزرنيم أي 61. اذن غير السيريال الى الصيغة التالية 1234-abcd-efgh-61kl-mnop-qrst-uvwX المعدل بعد الدالة هناك التالي :

00401D2F	. 85C0	TEST EAX,EAX
00401D31	. 75 02	JNZ SHORT CrackMe.00401D35
00401D33	. B3 01	MOV BL,1

اعتقد ان كل شيء واضح. سنتجاوز القفز وستزيد قيمة BL الى 1.

00401D35	> 0FB8424 1D0	MOVSB EAX, BYTE PTR SS:[ESP+21D]
00401D3D	. 8A4C74 1C	MOV CL, BYTE PTR SS:[ESP+ESI*2+1C]
00401D41	. 8A5474 1D	MOV DL, BYTE PTR SS:[ESP+ESI*2+1D]

الوامر الثلاث مشابهة للوامر الثلاثة السابقة. بداية ينقل ثاني حرف في اليوزر الى EAX . لاحظ ان ESI الذي يحتوي على c مضروب ب 2. اذن القيمة الاولى هي 2C+1 والثانية 2C+2 أي U و V . الوامر التالية حتى 401D6C هي تكرار للوامر السابقة. سيتم مقارنة قيمة الحرف الثاني من اليوزر أي 6C=I مع UV لذا اعد التشغيل مستخدما السيريال التالي 1234-abcd-efgh-61kl-mnop-qrst-6cwX

ومن ثم هناك الوامر التالية :

00401D6E	. 85C0	TEST EAX,EAX
00401D70	. 75 02	JNZ SHORT CrackMe.00401D74
00401D72	. FEC3	INC BL

ولها نفس الشرح السابق.

بعد ذلك ، هناك استدعاء لدالة حساب الطول. ثم بالامر التالي

00401D84 . 8A4C14 1A | MOV CL, BYTE PTR SS:[ESP+EDX+1A] تنقل الخانة قبل الاخيرة في السيريال (في مثالنا هو الحرف W) الى cl . وهنا 00401D95 . 8A50 FF | MOV DL, BYTE PTR DS:[EAX-1] يتم نقل الخانة الاخيرة (أي X الى DL

والان الوامر الباقية تكرر نفس الامر...هنا 00401D90 . 0FB8424 1B0 | MOVSB EDI, BYTE PTR SS:[ESP+ECX+21B] ينقل اخر حرف في اليوزرنيم (أي 0) ليتم تحويله الى القيمة المكافئة ب ascii وهي 6F لتقارن الاخيرة مع WX اذن اعد التشغيل واكتب 1234-abcd-efgh-61kl-mnop-qrst-6c6F والان نحن هن

00401DDF	. 75 02	JNZ SHORT CrackMe.00401DD3
00401DD1	. FEC3	INC BL
00401DD3	> 80FB 03	CMP BL,3

لاحظ ان قيمة BL تقارن مع 3...وقد كان هناك 3 اختبارات وفي كل مرة تزيد قيمة BL ... والان ستظهر لنا شاشة good cracker بالاضافة الى تفعيل قائمة extra...تفحصها لتجد العديد من الامور المثيرة للاهتمام...

الخلاصة :

اليوزرنيم يجب ان يكون 3 احرف على الاقل .
السيريال نمبر يجب ان يكون 34 خانة بالضبط.
الخانة الخامسة ومضاعفاتها في السيريال يجب ان تكون شرطة - (كالتالي تراها فوق زر + أقصى يمين الكيبورد)
اول خانتين في السيريال =#s[1]s[2] يجب عند تحويل قيمتهما معا الى السداسي العشري الا تزيد قيمتهما عن 12(0xC) ولا تقل عن 2.
S[#+1]s[#+2]=u[1](in hex)
S[#*2+1]s[#*2+2]=u[2](in hex)
S[0x21]S[0x22]=u[last]

فمثلا دعنا نطبق الكلام السابق على اليوزرنيم control ولنختار اول خانتين من السيريال...11 مثلا...سنحصل على #=11 ولاحظ ان اول حرف c=63 والثاني 0=6F والاخير I=6C سنحصل على 1100-0000-0006-3000-0000-006F-006C لاحظ ان الاصفار يمكن تغييرها كيفما شئت...

مثال آخر : اليوزرنيم Palestine ولنختار او خانتين 03 مثلا... سنحصل على #=03 ولاحظ ان اول حرف p=70 والثاني a=61 والاخير e=65

NB:

تنويه :

في المثال السابق حدث خطأ بسيط...في التقنية الأولى "فك الضغط" قمت باستخدام PEiD للحصول على OEP مع العلم بان برنامج فك الضغط الذي استخدمته (UN FSG 2.0) يقوم بفك الضغط دون الحاجة لـ OEP ...ما حصل هو انني معتاد على استخدام برنامج QuickUnpck الذي يلزمه OEP لذلك استخدمت PEiD ...شكرا لـ dahaka الذي نيهني على ذلك.

Do yur know?

النجوم النيوترونية والثقوب السوداء :

تتكون النجوم النيوترونية والثقوب السوداء من ألباب النجوم التي بقيت بعد انفجار على شكل مستعمرات عظمى . إذا كانت كتلة اللب المتبقي تقع بين كتلة شمسية ونصف وثلاث كتل شمسية تقريبا فإنه ينكمش ويكون نجما نيوترونيا , أما إذا كانت كتلته أكبر بكثير من ثلاث كتل شمسية فإنه ينكمش ويصبح ثقبا أسود . يبلغ قطر النجوم النيوترونية حوالي 10 كلم فقط وهي تتألف بكاملها تقريبا من نيوترونات. هذه النجوم هي من الكثافة بحيث يزن ملء ملعقة شاي من مادتها حوالي بليون طن تقريبا. ومن جهة أخرى , تتميز الثقوب السوداء بقوة جذبها التي تبلغ حدا لا يمكن معه حتى للأشعة الضوئية أن تفلت منها , لذلك تبقى الثقوب السوداء أجساما غير مرئية.

Notes :

Nothing for today.

Outroduction :

Hey buddy, it takes me about 9 hours to bring you this work , so plz if you see any mistake or miss here of there, just contact me. And as always your suggestions, comments ,criticisms and complaints are welcomed. Feel free and fly like a bee :) oh I mean feel free to mail mee @ allko34@hotmail.com . Something else, thx to dahaka for his useful note on my last lesson.

Greetings :

DiDi

Dahaka

xacker

20/2/2006

There are many crackers , but only one allko
allko34@hotmail.com