

Target : crackme v2.0 by Cruehead/MiB
Lang. : masm32
Tools : olly , PEiD
Cracker : allko

Learn Reverse Engineering Series- lesson six Introduction to code injection

Introduction:

بدءاً من هذا الدرس، سيتغير مجرى السلسلة نوعاً ما... سنبدأ اليوم في قسم جديد من أقسام الهندسة العكسية... حتى الآن اطلعنا على عدة أساليب للهندسة العكسية كال patching و keygening واعني بالثانية اكتشاف خوارزمية التشفير بغض النظر عن عمل keygen فهذا ليس بالأمر الصعب ويعتمد على إتقانك للغة برمجة... اليوم سنبدأ بما يسمى بال code-injection أو حقن الكود... باختصار هذا يعني إضافة كود إلى البرنامج الضحية لجعله يقوم بأمر ما نريده... هذا الأمر قد يكون إظهار message box معينة... أو self-keygenning أي جعل البرنامج يقوم بإظهار السيريال الصحيح بنفسه... وهناك أمور أخرى ستعرفها بالدروس القادمة... والآن دعنا نبدأ... المثال الأول...

Let's crack it !!

لن أقوم بشرح كيفية فك الحماية بالتفصيل فهو بسيط للغاية (أكثر مما تتصور)... تفحص البرنامج لتجد أنه مكتوب بالاسمبلي، كما أنه يحتاج إلى كلمة السر الصحيحة فقط أي أننا لن ندخل username مما يزيد في سهولته، افتحه ب olly وضع breakpoint عند GetDlgItemTextA والآن تفحص البرنامج كما كنا نفعل في الأمثلة الخمسة السابقة... ملاحظة : أثناء تتبعك للبرنامج باستخدام F8 ستصل إلى تعليمة RETN 10 عند العنوان 401286 ستأخذك إلى ال kernel أي ستجد العناوين أصبحت تبدأ ب 77... لا تقلق أكمل تتبعك إلى أن ترجع إلى البرنامج... أو يمكنك وضع نقط توقف عند 401223 ثم اضغط F9...
لعلك توصلت إلى التالي: يوجد في البرنامج strings هي : s1 و s2... نقوم نحن بإدخال السيريال أي s3 (هذه الأسماء للإيضاح فقط)
وعملية التشفير هي كالتالي
(correct code) = s1 xor s2
our crypeted string = s4 xor s2
ال correct code :

00402130	6D 61 74 65 21 0D 4E 6F 77 20 74 72 79 20 74 68	mate!.Now try th
00402140	65 20 6E 65 78 74 20 43 72 61 63 6B 4D 65 21 00	e next CrackMe!
00402150	1F 2C 37 36 3B 3D 28 19 3D 26 1A 31 2D 3B 37 3E	7,76;=(+=%*1-;7>
00402160	4E 6F 20 6C 75 63 6B 21 00 4E 6F 20 6C 75 63 6B	No luck!.No luck
00402170	20 74 68 65 72 6F 3C 20 6D 61 74 65 21 00 21 22	there mate! 13

: S2

00402180	33 34 35 36 00 00 00 00 00 00 00 00 00 00 00	3456.....
00402190	00 00 54 72 79 20 74 6F 20 63 72 61 63 6B 20 6D	..Try to crack m
004021A0	65 21 00 4D 65 73 73 69 6E 67 5F 69 6E 5F 62 79	e!.Messing_in_by
004021B0	74 65 73 00 00 00 00 00 00 00 00 00 00 00 00	tes.....
004021C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	

(لا يهم أيهما s1 وأيهما s2... فهذه مجرد أسماء من عندي!!)
هناك ثلاث إجراءات procedures مهمة في البرنامج الأولى عند العنوان 401365 وظيفتها فحص ال char المدخل إذا كان رقماً أم لا.

```

00401365 $ C605 1821400 MOV BYTE PTR DS:[4021181],0
0040136C . 8B7424 04 MOV ESI,DWORD PTR SS:[ESP+4]
00401370 . 56 PUSH ESI
00401371 > 8A06 MOV AL,BYTE PTR DS:[ESI]
00401373 . 84C0 TEST AL,AL
00401375 ~ 74 19 JE SHORT CRACKME2.00401390
00401377 . FE05 1821400 INC BYTE PTR DS:[4021181]
0040137D . 3C 41 CMP AL,41
0040137F ~ 72 04 JE SHORT CRACKME2.00401385
00401381 ~ 3C 5A CMP AL,5A
00401383 ~ 73 03 JNB SHORT CRACKME2.00401388
00401385 > 46 INC ESI
00401386 ^ EB E9 JMP SHORT CRACKME2.00401371
00401388 > E8 25000000 CALL CRACKME2.00401382
0040138D . 46 INC ESI
0040138E ^ EB E1 JMP SHORT CRACKME2.00401371
00401390 > 5E POP ESI
00401391 . E8 03000000 CALL CRACKME2.00401399
00401396 ~ EB 00 JMP SHORT CRACKME2.00401398
00401398 > C3 RETN

```

الثانية تليها مباشرة..عند العنوان 401399 وهذه مسؤولة عن عملية التشفير الفعلية...

```

00401399 $ 33DB XOR EBX,EBX
0040139B . 33FF XOR EDI,EDI
0040139D > 8A8F A321400 MOV CL,BYTE PTR DS:[EDI+4021A3]
004013A3 . 8A1E MOV BL,BYTE PTR DS:[ESI]
004013A5 . 84DB TEST BL,BL
004013A7 ~ 74 08 JE SHORT CRACKME2.004013B1
004013A9 . 33C0 XOR BL,CL
004013AB . 8B7424 04 MOV BYTE PTR DS:[ESI],BL
004013AD . 46 INC ESI
004013AE . 47 INC EDI
004013AF ^ EB EC JMP SHORT CRACKME2.0040139D
004013B1 > C3 RETN

```

الثالثة عند العنوان 4013B8 وهي مسؤولة عن المقارنة بين الكود الصحيح والخطأ...

```

004013B8 $ 33FF XOR EDI,EDI
004013BA . 33C9 XOR ECX,ECX
004013BC . 8A0D 1821400 MOV CL,BYTE PTR DS:[4021181]
004013C2 . 8B7424 04 MOV ESI,DWORD PTR SS:[ESP+4]
004013C4 . BF 50214000 MOV EDI,CRACKME2.00402150
004013C6 . F3:A6 REPE CMPS BYTE PTR ES:[EDI],BYTE PTR DS
004013C8 . C3 RETN

```

كما ترى في الصورة الثانية فان ال char الأول في السيريال يذهب إلى bl وال char الأول من s2 يذهب إلى cl ثم هناك عملية xor

إذن ال correct code هي ناتج عملية xor بين s1 و s2 أي :

```

00402140 65 20 6E 65 78 74 20 43 72 61 63 6B 4D 65 21 00 e next CrackMe!
00402150 1F 2C 37 36 3B 3D 28 19 3D 26 1A 31 2D 3B 37 3E 76;=(1-;1-;7>
00402150 4E 6F 20 6C 75 63 6B 21 00 4E 6F 20 6C 75 63 6B No luck!..No luck

```

نعرف انه إذا كان $s1 = \text{correct code} \text{ xor } s2$ فان $\text{correct code} = s1 \text{ xor } s2$ أي :

Correct code	1F-2C-37-36-3B-3D-28-19-3D-26-1A-31-2D 3B 37 3E
S2	4D-65-73-73-69-6E-67-5F-69-6E-5F-62-79-74-65-73
S1	52 49 44 45 52 53 4F 46 54 48 45 53 54 4F 52 4D

إذن الكلمة السرية الصحيحة هي s1 وعند تحويلها إلى ASCII تصبح RIDERSOFTHESTORM حماية تافهة اليس كذلك؟ بلى!!!

ما نريده هنا هو كتابة الkeygen في نفس البرنامج...أي باستخدام Code Injection لذا هناك عدة خطوات يجب عملها :

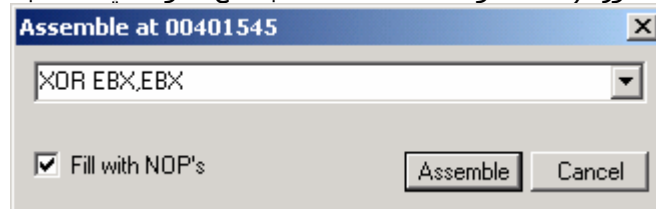
- 1- أن تجد مكان لتغيير مجرى (سريان)الأوامر أي تغيير ال code-flow
- 2- أن تجد مكان فارغا لحقن الكود الخاص بنا فيه (يسمى cave)
- 3- بعد كتابة ال code يجب إعادة السيطرة (التحكم)إلى البرنامج

00 : تفحص الصورة الأولى ...تجد انه عند العنوان 401391 يتم استدعاء ال carpeting procedure التي عنوانها ... 401399

حسنا فكر بالأمر كالتالي...سنغير هذا العنوان إلى عنوان ال cave ومن ثم فان البرنامج عندما يصل إلى هذه الخطوة سيذهب إلى الكود الذي أدخلناه بدل تلك الإجرائية ومن ثم في نهاية الكود سنعيد السيطرة للبرنامج... 01 : في البرامج المكتوب بالاسميلي لعلك لاحظت وجود مكان فارغ كبيرة جدا في نهاية البرنامج...حسنا هذا هو المكان الصحيح...لكن لا تفرح كثيرا ففي البرامج الأخرى ستكون العملية أكثر تعقيدا...

10 - الكود الذي سنكتبه هو نفسه الموجود في الصورة الثانية... لماذا؟ لأنه ينتج الشفرة الصحيحة وهذا ما نريده...

إذن اذهب للعنوان 401391 واضغط space ثم غير الأمر إلى التالي : PUSH 401545 (لاحظ أن الفراغ كبير ويمكنك وضع الكود أينما تريد... أنا اخترت العنوان 401545) والآن حدد الإجراء الثانية (الظاهرة في الصورة الثانية) ثم انسخها إلى أي محرر كال notepad... والآن "نسق" هذه الأوامر أي احذف جزء العنوان منها (هناك plug ins تقوم بنفس الخطوة لكن الأمر لن يأخذ أكثر من دقيقة) والآن اذهب إلى 401545 واضغط space وادخل أول أمر كما بالصورة (انسخه والصقه... لذلك قمنا بنسخ الكود كي لا نتعب أنفسنا بكتابة الأوامر من جديد)



واستمر إلى أن تنتهي من جميع الأسطر البرمجية... لاحظ عند إدخال jump أو call أن تحذف اسم البرنامج... فعند نسخه يكون هكذا

JMP SHORT CRACKME2.00401371 احذف الاسم ليصبح هكذا JMP SHORT 00401371 و غير العنوان أيضا إلى 401545

وأكمل جميع الأوامر... كان من المفروض أن نغير عنوان كل jump إلى العنوان الصحيح فور إدخالها لكنني أفضل عمل ذلك بعد الانتهاء من كتابة كل العناوين... وحيث أن أمر القفز يحتوي على short فلن يسمح لكolly بإبقائه كما هو... كلمة short تعني أن يقفز إلى عنوان قريب لذا تجدolly يقترح عليك إبقاء العنوان كما هو مع تغيير الصيغة إلى long... دعك من هذا وافعل كما قلت وبعد أن تصبح خبيراً اتبع الأسلوب الذي تريده...

الآن يجب أن نقوم بتعديل العناوين... ارجع إلى العنوان 401399 واضغط على كل قفزة لتر إلى أين تقفز (ملاحظة : ستصبح العملية أسهل أن كان هناك أسهم تشير إلى اتجاه القفزة... راجع الدروس السابقة لتعرف كيف تظهر هذه الأسهم) والآن ارجع إلى الكود الذي أدخلناه للتو وغير القفزات (أي اضغط space) ...ستصل إلى النتيجة التالية :

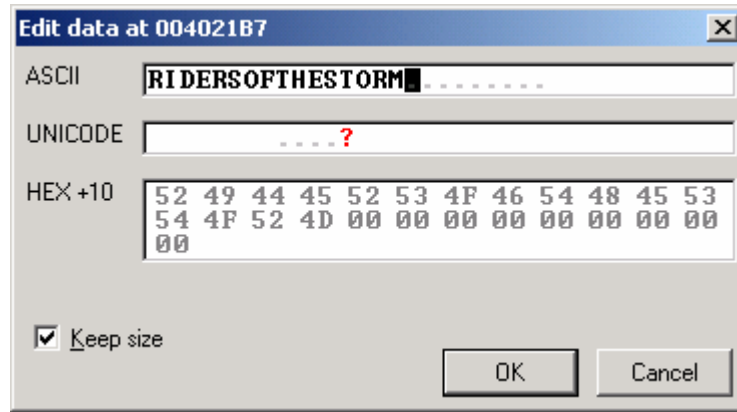
00401545	33DB	XOR EBX,EBX
00401547	33FF	XOR EDI,EDI
00401549	8A8F A3214001	MOV CL,BYTE PTR DS:[EDI+4021A3]
0040154F	8A1E	MOV BL,BYTE PTR DS:[ESI]
00401551	84DB	TEST BL,BL
00401553	74 0C	JE SHORT patched.00401561
00401555	32D9	XOR BL,CL
00401557	8B1E	MOV BYTE PTR DS:[ESI],BL
00401559	46	INC ESI
0040155A	47	INC EDI
0040155B	EB EC	JMP SHORT patched.00401549
0040155D	C3	RETN

بالنسبة لعملية إرجاع السيطرة للبرنامج... فانظر إلى أمر RETN انه يتكفل بالمهمة. والآن بعد أن قمت بفعل ذلك يدويا... جرب ال plug in المرفقة... حدد الكود الأصلي ثم right click → ASM2Clipboard → copy fixed assembly code to clipboard ستجد النتيجة أفضل بكثير... لكن يجب أن تتدرب على عمل ذلك "يدويا"...

والآن قم بحفظ التغييرات... right → copy all → all modifications → copy to executable → right click → save file

اعد تشغيل البرنامج المعدل... الآن اذهب إلى بداية الكود المعدل (401545) وضع نقطة توقف... شغل... وتتبع البرنامج...

ماذا تلاحظ؟ لقد فشلنا أليس كذلك؟ نعم فنحن نسخنا الكود as is أي كما هو... ما حصل أننا غيرنا مكانه فقط!! الكود يعمل xor بين s2 و السيريال المدخل... إذن الحل أن نجعله يعمل xor بين s2 , Correct code , لذا أولا علينا إدخال الكلمة السرية RIDERSOFTHESTORM في مكان ما... وليكن 4021B7 إذن اذهب إلى هناك وحدد جميع الأسفار (النقاط على اليمين) واضغط space



لتحصل على التالي

004021A0	65	21	00	4D	65	73	73	69	6E	67	5F	69	6E	5F	62	79	et.Messing_in_by
004021B0	74	65	73	00	00	00	00	52	49	44	45	52	53	4F	46	54	tes....RIDERSOFT
004021C0	48	45	53	54	4F	52	4D	00	00	00	00	00	00	00	00	00	HESTORM.....
004021D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	

غير أمر MOV BL, BYTE PTR DS:[ESI] إلى
 MOV BL, BYTE PTR [4021B7+EDI] حيث 4021B7 هي عنوان ما كتبناه للتو.. وEDI يعمل كعداد يغير مكان
 المؤشر (بداية edi=0 والمؤشر يشير إلى أول حرف... ومع تزايد المؤشر يصبح المؤشر يشير إلى الأحرف التالية). احفظ
 التغييرات لتحصل واعد التشغيل...
 رائع!! نجحنا!!

Do you know:

الماسونية:
 أصل التسمية بالانجليزية free-masonry أي البناء الحر ، وبالفرنسية franc-macon أي البناء الصادق وقد
 ترجمت الكلمتان الى اللغة العربية بكلمة : الماسونية.
 وفي حقيقة الأمر فالماسونية هي منظمة يهودية إرهابية مغلقة غامضة محكمة التنظيم تهدف إلى سيطرة
 اليهود على العالم عن طريق تقويض الأديان – غير اليهودي – والأخلاق ، وإشاعة الإلحاد والإباحية والفساد ،
 واستخدام الشخصيات المرموقة في العالم ، يوسقهم عهد متين بحفظ الأسرار وتنفيذ ما يطلب منهم.
 بتصرف من كتاب "واقعا المعاصر والغزو الفكري" للدكتور صالح الرقب.

Notes:

Maybe this example is not the best one to illustrate code injection, but at least it gives you a clear image on what code injection is. Anyway, the next example shall be excellent. I often hear the word "inline patching" and I saw that it's almost as the code injection; well I really don't know the differences, if there are, between them. If you know, please tell me!

THX.

DiDi...you are da best!!

20/2/2006

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