Target : Pulsar crackme
Level : 0/10
Lang. : asm32
Tools : OllyDBG
Cracker: allko

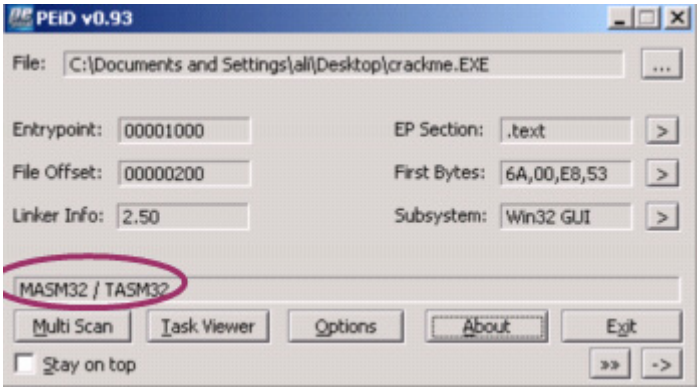
Introduction :

هذا اول درس لي لذلك ارجوا ان ينال رضاكم...مع ملاحظة ان الشرح سيكون بالتفصيل الممل..واعذروني ان كان هناك أي تقصير فانا مازلت مبتدئا (في الواقع انا اقل من مبتدئ)...يرجى ملاحظة ان هذه الوثيقة تتبع الـ open resources وانها تنشر تحت اتفاقية GNU . فلننطلق!!

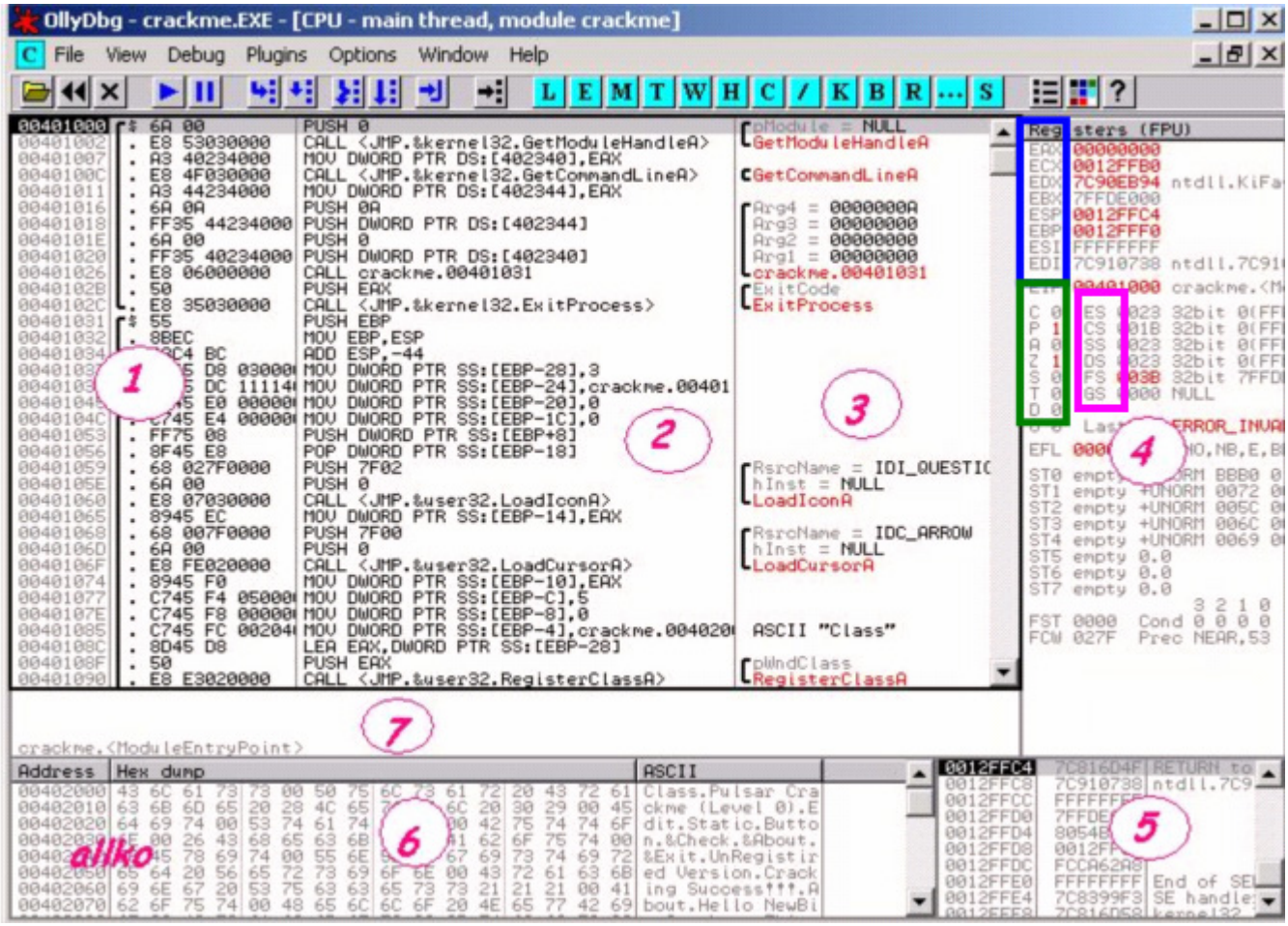
Here we go :

000 - قم بتشغيل الملف (الـ crackme) ولاحظه جيدا...هناك جملة unregistered version...ادخل أي رقم واضغط على check...لاحظ انه لا توجد أي نافذة message تخبرك بان السيريال خطأ...

001- قم بفحص الملف باستخدام PEiD لمعرفة ما اذا كان مضغوطا packed ...لحسن الحظ فانه ليس مضغوط ولحسن الحظ اكثر فكما يظهر في الصورة فالبرنامج مكتوب بالـ assembly مما يجعل الملف اسهل واكثر وضوحا...



010 - اطلق الوحش الصغير OllyDBG وافتح الـ crackme ...ستشاهد الصورة التالية...



كما تلاحظ فقد قمت بتقسيم الشاشة إلى 7 أقسام كالتالي :
1: قسم العناوين addresses وهو يقسم إلى قسمين ايمن وأيسر...العنوان موجود في القسم الأيسر وكمثال عليه : 40150E
2- قسم الأوامر instructions وكمثال الأمر PUSH EAX
3- قسم التعليقات comments وهنا تعليقات على الأوامر المختلفة وهذا القسم ينشئه olly بنفسه والأقسام الثلاثة هذه تشكل قسم الـ cpu
4- قسم المسجلات registers وهنا تجد
كل مسجل والقيمة التي يحتويها (المستطيل الأزرق)
كل مسجل من النوع segment registers مع قيمها (المستطيل الوردي)
كما تجد الرايات flags مع قيمها (المستطيل الأخضر)
ولن يهمنا سوى القسم الأول...وأحيانا الثالث...
5- المكدس stuck هنا تجد كل ما يخزن في الـ stuck بواسطة أوامر الـ push
6 – dumped memory قليلا ما ننظر إليه
7- هذا القسم التوضيحي مفيد ، فعندما نتتبع البرنامج فانه يخبرك ما إذا كانت التعليمة instructions الحالية لها references أي هناك قفزاتjumps أو اتصالات calls تؤدي اليها . أيضا في كثير من الأوامر مثل mov فانه يخبرك بمحتويات كل operand.

011- والآن دعنا نبدأ...
في البداية ابدأ بإلقاء نظرة سريعة على البرنامج من أعلى إلى أسفل...وركز انتباهك على قسم التعليقات (3) وابحث عن أي دالة قد تبدوا مثيرة للاهتمام...
ها قد وجدنا ما قد يكون مفيدا...الدالة SetWindowTextAالتي تعرض جملة unregistered version ...لاحظ ان هذه الدالة لها بارامترات 2parameters (فمثلا احدهما هو البارامتر text ويحتوي على النص الذي ستعرضه الدالة)

```
00401285  . 68 46204000    PUSH crackme.00402046          Text = "UnRegistired Version"
0040128A  . FF35 4C254000  PUSH DWORD PTR DS:[40254C]     hWnd = NULL
00401290  . E8 13010000    CALL <JMP.&user32.SetWindowTextA>  SetWindowTextA
```

ضع نقطة توقف breakpoint وذلك بالضغط على F2 (لست متاكدا من ان هذا هو الطريق الأصح لكنه صحيح بلا شك) . انزل قليلا لترى دالة اكثر اثارة للانتباه...انها GetWindowTextLengthA وكما هو واضح من الاسم فإنها مسئولة عن حساب طول الـ string التي ندخلها ...ومن المهم معرفة ان هذا الطول يخزن في EAX...

```
0040129B  . FF35 48254000  PUSH DWORD PTR DS:[402548]     hWnd = NULL
0040129B  . E8 0E010000    CALL <JMP.&user32.GetWindowTextLengthA>  GetWindowTextLengthA
004012A0  . 83F8 0B        CMP EAX,0B
004012A3  .v0F85 93000000  JNZ crackme.0040133C
004012A9  . 68 00020000    PUSH 200                       Count = 200 (512.)
004012AE  . 68 48234000    PUSH crackme.00402348          Buffer = crackme.00402348
004012B3  . FF35 48254000  PUSH DWORD PTR DS:[402548]     hWnd = NULL
004012B9  . E8 F6000000    CALL <JMP.&user32.GetWindowTextA>  GetWindowTextA
```

أسفل منها هناك امر مقارنة يقارن طول الـ s/n بـ B (decimal 11) وبالتالي نكون قد عرفنا ان طول الـ s/n 11 خانة وهذه معلومة مهمة. أسفل منه هناك قفزة ...اذا لم يكون طول السيريال 11 فلن نحصل على شاشة good cracker ثم هناك دالة أخرى مهمة أيضا وهي GetWindowTextA وكما هو واضح فإنها تاخذ ما كتبناه في المربع المخصوص...
لكن إلى أين تأخذه ؟ انظر إلى البارامتر الثاني buffer ...واضح ان السيريال المدخل سيخزن في 402348
جيد والآن للنظر ما يوجد بعد هذه الدالة...

```
004012BE  . 803D 4B234000  CMP BYTE PTR DS:[40234B],2D
004012C5  .v75 75          JNZ SHORT crackme.0040133C
004012C7  . C605 4B234000  MOV BYTE PTR DS:[40234B],41
004012CE  . 803D 4F234000  CMP BYTE PTR DS:[40234F],2D
004012D5  .v75 65          JNZ SHORT crackme.0040133C
004012D7  . C605 4F234000  MOV BYTE PTR DS:[40234F],42
004012DE  . B9 0A000000    MOV ECX,0A
004012E3  > FE81 48234000  INC BYTE PTR DS:[ECX+402348]
004012E9  .^E2 F8          LOOPD SHORT crackme.004012E3
004012EB  . BE 48234000    MOV ESI,crackme.00402348
004012F0  . BF F3204000    MOV EDI,crackme.004020F3       ASCII "PvMBtBsC112"
004012F5  . B9 0B000000    MOV ECX,0B
004012FA  . FC             CLD
004012FB  . F3:A6          REPE CMPS BYTE PTR ES:[EDI],BYTE PTR DS
004012FD  .v75 3D          JNZ SHORT crackme.0040133C
004012FF  . 68 5B204000    PUSH crackme.0040205B          Text = "Cracking Success!!!"
00401304  . FF35 4C254000  PUSH DWORD PTR DS:[40254C]     hWnd = NULL
0040130A  . E8 99000000    CALL <JMP.&user32.SetWindowTextA>  SetWindowTextA
```

شغل البرنامج بالضغط على F9 سيخرج لك الـ crackme ادخل أي سيريال مكون من 11 خانة وليكن 12345678912 (ملاحظة...حتى الآن لا نعلم ما اذا كان السيريال أرقام أو حروف أو الاثنين معا)
اضغط على check وفورا ستعود إلى olly ...نحن الآن عند نقطة التوقف التي انشاناها...سنبدأ بعملية التتبع...

100 – تتبع البرنامج ...
اضغط على F8 لتنتقل إلى الأمر التالي...استمر في الضغط حتى تصل إلى امر المقارنة أسفل دالة GetWindowTextLengthA مباشرة... انظر إلى قسم الـ registers وبالتحديد إلى EAX...ما هي قيمته؟

```
0040129B  . E8 0E010000   CALL <JMP.&user32.GetWindowTextLengthA>      hWnd = 002B01CH (class=
004012A0  . 83F8 0B       CMP EAX,0B                                   GetWindowTextLengthA
```

كما تلاحظ فان طول السيريال تم تخزينه في EAX أي 0xB (11) ...فلنتابع...استمر بالضغط على F8 حتى تصل إلى الدالة GetWindowTextA ...والان اضغط F8 لتحصل على الصورة التالية

```
004012BE  . 803D 4B234000  CMP BYTE PTR DS:[40234B],2D
004012C5  . 75 75          JNZ SHORT crackme.0040133C
004012C7  . C605 4B234000  MOV BYTE PTR DS:[40234B],41
004012CE  . 803D 4F234000  CMP BYTE PTR DS:[40234F],2D
004012D5  . 75 65          JNZ SHORT crackme.0040133C
004012D7  . C605 4F234000  MOV BYTE PTR DS:[40234F],42
004012DE  . B9 0A000000    MOV ECX,0A
004012E3  > FE81 48234000  INC BYTE PTR DS:[ECX+402348]
004012E9  .^E2 F8          LOOPD SHORT crackme.004012E3
004012EB  . BE 48234000    MOV ESI,crackme.00402348                ASCII "12345678912"
004012F0  . BF F3204000    MOV EDI,crackme.004020F3                ASCII "PvMBtBsC112"
```

ماذا تلاحظ؟ تم تخزين السيريال في العنوان 4012EB أي بالـ buffer رقم 402348 وأسفل منه هناك string تثير الانتباه...هل يعقل ان تكون هي السيريال الصحيح؟ اذهب لموقع البرنامج وانسخه وشغل النسخة وادخل هذا السيريال...ماذا تلاحظ؟
محاولة فاشلة!!! يبدوا ان المبرمج اذكى مما توقعنا...لا باس...انظر إلى الأمر مرة اخرى انه مقارنة بين القيمة 0x2D والقيمة 4...كيف عرفت انها 4؟ انظر إلى القسم 7 لتجد كل ما تحتاجه...وكما تلاحظ القيمة 4 تمثل الخانة الرابعة من السيريال...يمكننا ان نستنتج من ذلك ان الخانة الرابعة للسيريال يجب ان تكون 0x2D وباستخدام جدول ASCII تعرف انها علامة الشرطة " – "

مرمن علمنا أن السيريال يخزن في 402348 وان طوله 11 فيمكننا وضع التخطيط التالي (انظر إلى اليسار).
وبالتالي فعندما نشاهد CMP BYTE PTR DS:[40234B],2D
فهذا يعني مقارنة 0x2D مع محتوى الذاكرة الذي عنوانه مخزن بال segment DS والـ offset له 40234B أي انه الرقم 4...

402348 =1
402349 =2
40234A =3
40234B =4
40234C =5
40234D =6
40234E =7
40234F =8
402350 =9
402351 =1
402352 =2

ثم اضغط...انتظر!!هناك قفزة...وبما ان المقارنة سلبية أي ان ما أدخلناه لا يساوي الـ – فان القفزة ستنفذ ولن نتمكن من اكمال تتبع البرنامج...امامك حلان...اما ان تتبعه نظريا أي دون تطبيق باستخدام F8 أو الحل الآخر – وهو ما افضله – ان تقوم بتعديل القفزة من JNZ إلى JZ ...اضغط عليها right click ثم اختر assemble ومن النافذة الجديدة غيرها واضغط انتر ثم escape (لاننا نريد تغيير امر واحد فقط) تلاحظ ان الأمر تغير لونه إلى الاحمر...بامكانك الآن الضغط على F8...لاحظ في قسم التعليقات بالأسفل(القسم 7) انه يخبرك ما اذا كانت القفزة ستتم ام لا وهنا يخبرك بانه : jump is NOT taken حسنا اضغط F8 مرة أخرى...هناك امر mov حيث ستنقل القيمة 41 (ASCII : A) إلى الخانة الرابعة...
الأوامر الثلاث التالية ينطبق عليها نفس الكلام...المقارنة للخانة 8 مع 0x2D أي – وبعدها نغير القفزة ثم نرى امر mov حيث تنتقل القيمة 42 (ASCII : B) إلى الخانة الثامنة...
الأمر التالي ينقل القيمة 0xA (10) إلى ECX وأنت تعلم ان هذا المسجل يستخدم كعداد لذلك نستنتج ان هناك loop لعشر مرات...
وفعلا الأمران التاليان هما الـloop المقصودة (لاحظ ان olly يضع على يسار كل loop خطا ليدلك عليها...انظر الي يسار هذين الأمرين) . أول أمر في الـ loop هو INC BYTE PTR DS:[ECX+402348] أي انه سيزود اخر خانة بمقدار واحد و الآن قم بالضغط على F8 مرتين وراقب السيريال الخاص بنا...هل شاهدت؟ لقد زادت اخر خانة بمقدار واحد...اكمل الـ loop لتجد بالنهاية ان جميع الخانات تزيد بمقدار واحد عدا الخانة الأولى.
ها قد خرجنا من loop واصبح السيريال كالتالي 134B678C:23
الأمران التاليان ينقلنا السيريال المعدل إلى ESI والسيريال الذي اثار انتباهنا إلى EDI وانظر إلى قسم المسجلات لتتاكد بنفسك . الأمران التالي يضع القيم 11 في ECX اذا هناك loop تكرر 11 مرة...
تجاهل الأمر التالي وانتقل إلى الذي بعده...كما توقعنا هناك loop مقارنة كالتالي
REPE CMPS BYTE PTR ES:[EDI],BYTE PTR DS:[ESI]

أي ان اول بايت في ESI يقارن مع اول بايت فيEDI (وهذا معنى BYTE PTR) وذلك ل 11 مرة أي ان كل خانة تقارن مع نظيرتها في الـ PvMBtBsC112 : string...فاذا كانت نتيجة المقارنة ايجابية فان القفزة التالية لن تتم وسينتقل إلى جملة good boy

101 - الخطوة الأخيرة
المعادلة التي حصلنا عليها هي كالتالي(الأرقام في الصف السفلي تجدها في جدول ASCII)

P	v	M	B	t	B	s	C	1	1	2
50	76	4D	42	74	42	73	43	31	32	32

والان اطرح 1 من كل خانة عدا الأولى مع العلم بان الخانتين 4 و8 يجب ان تكونا –

P	u	L	-	s	A	r	-	0	0	1

إذن السيريال نمبر الصحيح هو PuL-sAr-001
وعند ادخال هذا السيريال يقارن البرنامج الخانتين 4 و8 ب – فيجد أنهما متطابقتان ثم يستبدلهما ب A و B على الترتيب ثم يزود كل خانة ما عدا الأولى بمقدار واحد ويقارن النتيجة ب PvMBtBsC112 فإذا حصل التطابق فالسيريال صحيح...

Notes :

You can easily patch the jump instruction exactly before "cracking success" message , but the goal is to find the real serial
If you want to change the jump instruction and make this change permanent ;
Right click ➔ assemble ➔ (change it) ➔ assemble ➔ escape
Then
Right click ➔ copy to executable ➔ all modifications ➔ copy all
And in the new window
Right click ➔ save

Outroduction :

It was really easy although it takes me hours to find the serial and 3.5 hours to write this tutorial !! So if you are a newbie , just like me , this tutorial is good for you…
If u have any comments , suggestions or useful notes , feel free to mail me .

Greetings :

Forums : Coding and protection forum , arabteam's coding forum
Friends : DiDi