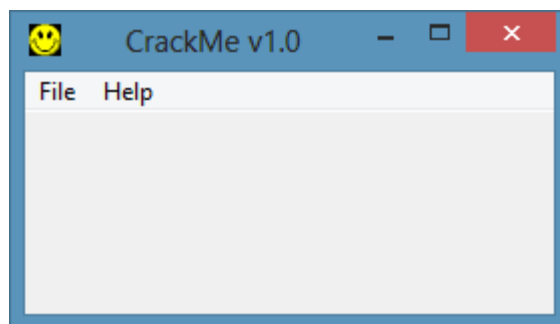


Keygen para el CrackMe#01 de Cruehead

Haciendo el KeyGen de un clásico



By deurus
08/09/2014

ÍNDICE

1.	Introducción	2
2.	El algoritmo	2
3.	KeyGen	4
4.	Enlaces	4
5.	Repositorio	4

Equipo utilizado:

S.O: Windows 7 x32 /Windows 8 x64

Depurador: Ollydbg 1.10 (32bits) con plugins

Analizador: PEiD 0.95

1. Introducción

Esta es la primera entrega de tres en las que vamos a ver tres crackmes que todo reverser debería hacer. Son la serie del autor Cruuehead. Aunque los hice hace ya muchos años, he decidido documentarlos para que el lector que empieza pueda deleitarse. En este caso se trata del típico Nombre / Serial.

2. El algoritmo

El algoritmo de este crackme es lo más sencillo que nos podemos encontrar.

Abrimos el crackme con **Oll**y y buscamos en las “string references” el mensaje de error. Pinchamos sobre él y en la parte superior enseguida vemos 2 calls muy interesantes.

00401226	74 BE	JE SHORT 004011E6	
00401228	68 8E214000	PUSH OFFSET 0040218E	ASCII "DEURUS"
0040122D	E8 4C010000	CALL 0040137E	; GENERA EL SUM DEL NOMBRE
00401232	50	PUSH EAX	
00401233	68 7E214000	PUSH OFFSET 0040217E	ASCII "12345"
00401238	E8 9B010000	CALL 00401308	; GENERA EL SUM DEL SERIAL
0040123D	83C4 04	ADD ESP,4	
00401240	58	POP EAX	
00401241	3BC3	CMP EAX,EBX	
00401243	74 07	JE SHORT 0040124C	; COMPARA SUM NOMBRE CON SUM SERIAL

Veamos que hace con el **nombre**.

0040137E	8B7424 04	MOV ESI,DWORD PTR SS:[ARG.1]	
00401382	56	PUSH ESI	
00401383	8A06	MOV AL,BYTE PTR DS:[ESI]	
00401385	84C0	TEST AL,AL	
00401387	74 13	JZ SHORT 0040139C	
00401389	3C 41	CMP AL,41	
0040138B	72 1F	JB SHORT 004013AC	
0040138D	3C 5A	CMP AL,5A	
0040138F	73 03	JAE SHORT 00401394	
00401391	46	INC ESI	
00401392	EB EF	JMP SHORT 00401388	
00401394	E8 39000000	CALL 00401302	
00401399	46	INC ESI	
0040139A	EB E7	JMP SHORT 00401388	
0040139C	5E	POP ESI	ASCII "DEURUS"
0040139D	E8 20000000	CALL 004013C2	
004013A2	81F7 78560000	XOR EDI,00005678	
004013A8	8BC7	MOV EAX,EDI	
004013AA	EB 15	JMP SHORT 004013C1	
004013AD	5E	POP ESI	
004013AE	6A 30	PUSH 30	
004013AF	68 60214000	PUSH OFFSET 00402160	
004013B4	68 69214000	PUSH OFFSET 00402169	
004013B9	FF75 08	PUSH DWORD PTR SS:[EBP+08]	
004013BC	E8 79000000	CALL <JMP.&USER32.MessageBoxA>	
004013C1	C3	RETN	
004013C2	33FF	XOR EDI,EDI	
004013C4	33DB	XOR EBX,EBX	
004013C6	8A1E	MOV BL,BYTE PTR DS:[ESI]	
004013C8	84DB	TEST BL,BL	
004013CA	74 05	JZ SHORT 004013D1	
004013CC	03FB	ADD EDI,EBX	
004013CE	46	INC ESI	
004013CF	EB F5	JMP SHORT 004013C6	
004013D1	C3	RETN	
004013D2	2C 20	SUB AL,20	
004013D4	8806	MOV BYTE PTR DS:[ESI],AL	
004013D6	C3	RETN	
004013D7	C3	RETN	

Convierte minúsculas a mayúsculas y descarta números

Suma los valores ASCII del nombre

ASCII "No luck!"
ASCII "No luck there, mate!"

Para “deurus” pondría todo en mayúsculas, sumaría su valor ascii y le haría **XOR 0x5678**.

Ejemplo:

deurus → DEURUS → $0x44+0x45+0x55+0x52+0x55+0x53 = 0x1D8$ XOR $0x5678 = 0x57A0$

Veamos que hace con el **serial** introducido.

```

004013D8 33C0 XOR EAX,EAX
004013DA 33FF XOR EDI,EDI
004013DC 33DB XOR EBX,EBX
004013DE 8B7424 04 MOV ESI,DWORD PTR SS:[ARG.1]
004013E2 B0 0A MOV AL,0A
004013E4 8A1E MOV BL,BYTE PTR DS:[ESI]
004013E6 84DB TEST BL,BL
004013E8 74 0B JZ SHORT 004013F5
004013EA 80EB 30 SUB BL,30
004013ED 0FAFF8 IMUL EDI,EAX
004013F0 03FB ADD EDI,EBX
004013F2 46 INC ESI
004013F3 EB ED JMP SHORT 004013E2
004013F5 81F7 34120000 XOR EDI,00001234
004013F8 8BDF MOV EBX,EDI
004013FD C3 RETN
  
```

Pasa nuestro serial a hexadecimal

HexSerial XOR 1234

Convierte nuestro **serial** a hexadecimal y le hace **XOR 0x1234**.

Ejemplo:

Serial = 12345 → $0x3039$ XOR $0x1234 = 0x220D$

Una vez que tenemos el **SUMNombre** y el **SUMSerial** los compara. Lo vemos en **CMP EAX, EBX**.

```

00401226 74 BE JE SHORT 004011E6
00401228 68 8E214000 PUSH OFFSET 0040218E
0040122D E8 4C010000 CALL 0040137E
00401232 50 PUSH EAX
00401233 68 7E214000 PUSH OFFSET 0040217E
00401238 E8 9B010000 CALL 004013D8
0040123D 83C4 04 ADD ESP,4
00401240 58 POP EAX
00401241 3BC3 CMP EAX,EBX
00401243 74 07 JE SHORT 0040124C
  
```

ASCII "DEURUS"
; GENERA EL SUM DEL NOMBRE

ASCII "12345"
; GENERA EL SUM DEL SERIAL

; COMPARA SUM NOMBRE CON SUM SERIAL

En resumen, si a nuestro **SUMNombre** le hacemos **XOR 0x5678** y **XOR 0x1234** ya tenemos el serial bueno.

Ejemplo:

deurus → DEURUS → $0x44+0x45+0x55+0x52+0x55+0x53 = 0x1D8$ **XOR 0x5678 =**
0x57A0 XOR 0x1234 = 0x4594

$0x4594 = 17812$ = Nuestro serial bueno.

3. KeyGen

```
1 char Nombre[20];
2 GetWindowText(hwndEdit1, Nombre, 20);
3 char Serial[20];
4 int len = strlen(Nombre);
5 int suma = 0;
6 boolean error = false;
7 for(int i = 0; i <= len; i = i + 1)
8 {
9     suma += toupper(Nombre[i]);
10 }
11 suma = suma^0x444C; //444C == 5678 xor 1234
12 wsprintf(Serial, "%d", suma);
13 SetWindowText(hwndEdit2, TEXT(Serial));
```

4. Enlaces

- Crackme
- Keygen

5. Repositorio

- https://deurus.info/archivos/crackmes/Crackme_cruehead_1.0.rar