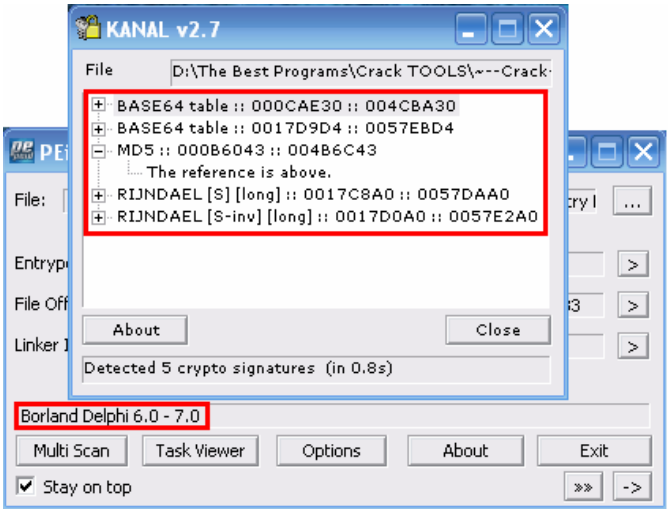
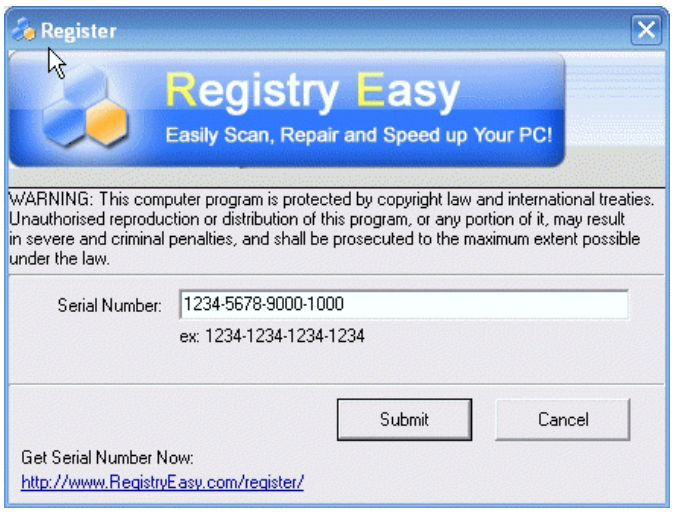
مبحث :	چگونگی تحلیل و پیدا کردن الگوریتم برنامه Registry Easy 4.0 و نوشتن Keygen برای آن .
هدف :	Registry Easy 4.0 http://www.registryeasy.com
ابزار :	PEiD – Olly DBG

شروع کار :

ابتدا برنامه را با PEiD چک می کنیم و Hash های احتمالی رو در نظر می گیریم :

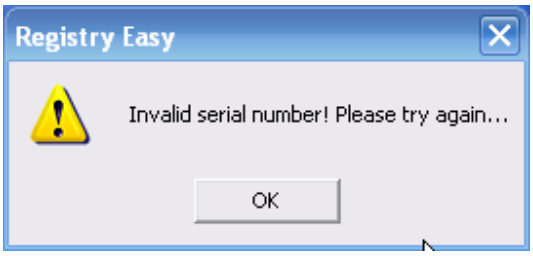


برنامه را اجرا می کنیم و یک راست می رویم سر قسمت رجیستر برنامه که به شکل زیر است :



اینجا فرمت وارد کردن سریال نامبر رو می بینید.
حالا بهتره بریم سراغ تابعی که سریال نامبر رو چک میکنه!
برای این منظور یک سریال به دلخواه خودتون بدید و دکمه Submit

رو بزنید.پیغام زیر نمایش داده می شود :



حالا متن این پیغام رو جستجو کنید می رسید به شکل زیر :

```
004E062B  . E8 30D9FAFF    CALL Registry.0048DF60
004E0630  .v EB 4E         JMP SHORT Registry.004E0680
004E0632  > 6A 30          PUSH 30
004E0634  . B9 EC064E00    MOV ECX,Registry.004E06EC                ASCII "Registry Easy"
004E0639  . BA FC064E00    MOV EDX,Registry.004E06FC                ASCII "Invalid serial number! Please try again..."
004E063E  . A1 00F55700    MOV EAX,DWORD PTR DS:[57F500]
004E0643  . 8B00           MOV EAX,DWORD PTR DS:[EAX]
004E0645  . E8 7612FBFF    CALL Registry.004918C0
004E064A  . 8B83 10030000  MOV EAX,DWORD PTR DS:[EBX+310]
004E0650  . 8B10           MOV EDX,DWORD PTR DS:[EAX]
004E0652  . FF92 C4000000  CALL DWORD PTR DS:[EDX+C4]
004E0658  .v EB 26         JMP SHORT Registry.004E0680
004E065A  > 6A 30          PUSH 30
004E065C  . B9 EC064E00    MOV ECX,Registry.004E06EC                ASCII "Registry Easy"
004E0661  . BA FC064E00    MOV EDX,Registry.004E06FC                ASCII "Invalid serial number! Please try again..."
004E0666  . A1 00F55700    MOV EAX,DWORD PTR DS:[57F500]
004E066B  . 8B00           MOV EAX,DWORD PTR DS:[EAX]
004E066D  . E8 4E12FBFF    CALL Registry.004918C0
004E0672  . 8B83 10030000  MOV EAX,DWORD PTR DS:[EBX+310]
004E0678  . 8B10           MOV EDX,DWORD PTR DS:[EAX]
004E067A  . FF92 C4000000  CALL DWORD PTR DS:[EDX+C4]
004E0680  > 33C0           XOR EAX,EAX
004E0682  . 5A             POP EDX
```

حالا انقدر بالا بروید تا به اول تابع برسید یعنی اینجا :

```
004E0450  $ 55             PUSH EBP
004E0451  . 8BEC           MOV EBP,ESP
004E0453  . B9 0A000000    MOV ECX,0A
004E0458  > 6A 00          PUSH 0
004E045A  . 6A 00          PUSH 0
004E045C  . 49             DEC ECX
004E045D  .^75 F9          JNZ SHORT Registry.004E0458
```

برگردید به برنامه و یک سریال به صورت زیر بدهید :
(1234-5678-9000-1000)
حالا رو آدرس زیر (004E04C7) یک BP بزارید و دکمه Submit رو فشار بدهید .
در این مکان برنامه شما متوقف می شود!

```
004E04C7  . 8B45 FC        MOV EAX,DWORD PTR SS:[EBP-4]
004E04CA  . E8 F54AF2FF    CALL Registry.00404FC4
004E04CF  . 8B45 E0        MOV EAX,DWORD PTR SS:[EBP-20]
004E04D2  . 8D55 F8        LEA EDX,DWORD PTR SS:[EBP-8]
004E04D5  . E8 128AF2FF    CALL Registry.00408EEC
Stack SS:[0012F1F4]=0138F730, (ASCII "1234567890001000")
EAX=0012F1D8
```

همان طور که می بینید سریال شما به صورت (1234567890001000) بهینه شده .
دوبار F8 بزنید تا به شکل زیر برسید :

```
004E04C2  . BA 09000000    MOV EDX,9
004E04C7  . 8B45 FC        MOV EAX,DWORD PTR SS:[EBP-4]
004E04CA  . E8 F54AF2FF    CALL Registry.00404FC4
004E04CF  . 8B45 E0        MOV EAX,DWORD PTR SS:[EBP-20]
004E04D2  . 8D55 F8        LEA EDX,DWORD PTR SS:[EBP-8]
Stack SS:[0012F1D8]=01371824, (ASCII "90001000")
EAX=0012F1D8
```

همان طور که می بینید دو قسمت آخر جدا شده ! (90001000)
کمی به پایین تر می رویم تا به شکل زیر برسیم :

```
004E0514  . 50            PUSH EAX
004E0515  . 8D55 D8       LEA EDX,DWORD PTR SS:[EBP-28]
004E0518  . B8 E0064E00   MOV EAX,Registry.004E06E0        ASCII "Easy Gang"
004E051D  . E8 3A70FDFF   CALL Registry.004B755C
004E0522  . 8B45 D8       MOV EAX,DWORD PTR SS:[EBP-28]
004E0525  . B9 0A000000   MOV ECX,0A
004E052A  . 8BD6          MOV EDX,ESI
```

دو تا BP مطابق شکل می زاریم
با F8 میریم پایین تا به BP دومی در آدرس (004E0522) برسیم !
مطابق شکل زیر خواهیم داشت :

```
004E0514  . 50            PUSH EAX
004E0515  . 8D55 D8       LEA EDX,DWORD PTR SS:[EBP-28]
004E0518  . B8 E0064E00   MOV EAX,Registry.004E06E0        ASCII "Easy Gang"
004E051D  . E8 3A70FDFF   CALL Registry.004B755C
004E0522  . 8B45 D8       MOV EAX,DWORD PTR SS:[EBP-28]
004E0525  . B9 0A000000   MOV ECX,0A
004E052A  . 8BD6          MOV EDX,ESI
Stack SS:[0012F1D0]=0138F768, (ASCII "ec7235e71546232da9f7cc02f32c9d0e")
EAX=0012F184
```

بله خودشه یک کد که با (MD5) Hash شده !
به نظر شما این کد از کجا آمده ؟
همان طور که دیدید درBP اولی کلمه (Easy Gang) ثبت شده بود .

این کلمه رو وقتی با (MD5) Hash کنید کد زیر به دست می اید:
(ec7235e71546232da9f7cc02f32c9d0e)
تا به اینجا اولین قدم برای رمز نگاری برنامه رو دیدیم.
BP بعدی را مطابق شکل زیر روی آدرس (004E056B) می زارید :

```
004E0568  . 8D55 D0       LEA EDX,DWORD PTR SS:[EBP-30]
004E056B  . 8B45 FC       MOV EAX,DWORD PTR SS:[EBP-4]
004E056E  . E8 C98BF2FF   CALL Registry.0040913C
004E0573  . 837D D0 00    CMP DWORD PTR SS:[EBP-30],0
004E0577  .v0F84 DD000000 JE Registry.004E065A
Stack SS:[0012F1F4]=0137CFC0, (ASCII "45678")
EAX=0012F1F4
```

همان طور که می بینید اعداد (45678) از کل سریال ما جدا می شود اینجا نتیجه می گیریم
که فرمت (1234-5678-9000-1000) معنای خاصی نداره همچنین میشه فهمید
این قسمت دارای یک ویژگی هست که جدا شده که در جلوتر به آن می پردازیم ! (در آینده مقایسه مي شود)

BP بعدی رو مثل شکل زیر می زارید و روش توقف می کنید :

```
004E057D  . 8D55 CC       LEA EDX,DWORD PTR SS:[EBP-34]
004E0580  . 8B45 F0       MOV EAX,DWORD PTR SS:[EBP-10]
004E0583  . E8 B48BF2FF   CALL Registry.0040913C
004E0588  . 837D CC 00    CMP DWORD PTR SS:[EBP-34],0
004E058C  .v0F84 C8000000 JE Registry.004E065A
Stack SS:[0012F1E8]=0138F798, (ASCII "c7235e7154")
EAX=0012F1C8
```

همان طور که پیداست یک کد (10) رقمی وجود داره .
کد آشنا به نظر میرسه شک نکنید که (MD5) هست !
به کد های قبلی نگاهی می ا ندازیم . مشخصه که این (10) رقم جدا شده از

کد Hash شده کلمه (Easy Gang) است یعنی :
(ec7235e71546232da9f7cc02f32c9d0e)
این کد رو هم در نظر مي گيريم و ادامه می دهیم .
دوتا BP پشت سر هم مانند شکل زیر می زاریم :

```
004E05BF  . 50          PUSH EAX
004E05C0  . FF75 F8     PUSH DWORD PTR SS:[EBP-8]
004E05C3  . FF75 F0     PUSH DWORD PTR SS:[EBP-10]
004E05C6  . FF75 F4     PUSH DWORD PTR SS:[EBP-C]
004E05C9  . 8D45 B8     LEA EAX,DWORD PTR SS:[EBP-48]
Stack SS:[0012F1F0]=0138F750, (ASCII "90001000")
```

و با F8 آن هارو Trace می کنیم .

```
004E05BC  . 8D45 C0     LEA EAX,DWORD PTR SS:[EBP-40]
004E05BF  . 50          PUSH EAX
004E05C0  . FF75 F8     PUSH DWORD PTR SS:[EBP-8]
004E05C3  . FF75 F0     PUSH DWORD PTR SS:[EBP-10]
004E05C6  . FF75 F4     PUSH DWORD PTR SS:[EBP-C]
004E05C9  . 8D45 B8     LEA EAX,DWORD PTR SS:[EBP-48]
Stack SS:[0012F1E8]=0138F798, (ASCII "c7235e7154")
```

به کد ها توجه کنید دو قسمت آخر از سریال نامبر (90001000) و قسمت
جدا شده 10 رقمی (c7235e7154) رو می بینید !

BP بعدی رو 5 خط پایین تر مي زاریم و بعد از رسیدن به BP کد زیر رو خواهیم داشت :

```
004E05C6  . FF75 F4     PUSH DWORD PTR SS:[EBP-C]
004E05C9  . 8D45 B8     LEA EAX,DWORD PTR SS:[EBP-48]
004E05CC  . BA 03000000 MOV EDX,3
004E05D1  . E8 4E48F2FF CALL Registry.00404E24
004E05D6  . 8B45 B8     MOV EAX,DWORD PTR SS:[EBP-48]
004E05D9  . 8D55 BC     LEA EDX,DWORD PTR SS:[EBP-44]
004E05DC  . E8 7B6FFDFF CALL Registry.004B755C
004E05E1  . 8B45 BC     MOV EAX,DWORD PTR SS:[EBP-44]
Stack SS:[0012F1B0]=0138F7E0, (ASCII "90001000c7235e71543")
EAX=004E05D6 (Registry.004E05D6)
```

درسته اینجا مشخص میشه کد های قسمت سوم و چهارم (90001000) با کد
10رقمی جداشده ترکیب میشه و کد زیر رو ایجاد میکنه :
(90001000 c7235e7154 3)
اینجا مشخص شد یک عدد دیگه هم اضافه شده (3)
در جلوتر خواهیم گفت این عدد از کجا آمده !
سه خط پایین تر BP بعدی رو بزارید و روش توقف کنید .
مانند شکل زیر :

```
004E05D9  . 8D55 BC     LEA EDX,DWORD PTR SS:[EBP-44]
004E05DC  . E8 7B6FFDFF CALL Registry.004B755C
004E05E1  . 8B45 BC     MOV EAX,DWORD PTR SS:[EBP-44]
004E05E4  . B9 05000000 MOV ECX,5
004E05E9  . 8BD6        MOV EDX,ESI
Stack SS:[0012F1B4]=0138F800, (ASCII "a5c514ce566dcfc8856f757c45998c28")
EAX=0012F184
```

حالا یک کد جدید داریم .
(a5c514ce566dcfc8856f757c45998c28)
کاملا واضح هست این کد (MD5) هست ولی از کجا به دست آمده ؟

به کد های قبلی رجوع مي کنیم قبل از این کد جدید ,ما کد (3 c7235e7154 90001000) رو داشتیم.
وقتی این کد رو به (MD5) تبدیل کنید کد جدیدی که حالا داریم به دست می اید تا اینجای کار ساده پیش رفتیم .
دو تا BP آخر رو هم مثل شکل زیر می زارید. پشت سر هم !
کامپایل مي کنید تا روی اولین BP متوقف شوید .
کد 5 رقمی دیده می شود :

```
004E05EB  . E8 D449F2FF    CALL Registry.00404FC4
004E05F0  . 8B55 C0        MOV EDX,DWORD PTR SS:[EBP-40]
004E05F3  . 8B45 FC        MOV EAX,DWORD PTR SS:[EBP-4]
004E05F6  . E8 B548F2FF    CALL Registry.00404EB0
004E05FB  .v75 35          JNZ SHORT Registry.004E0632
004E05FD  . 8D55 B0        LEA EDX,DWORD PTR SS:[EBP-50]
Stack SS:[0012F1B8]=01367C3C, (ASCII "c514c")
EDX=01367C3C, (ASCII "c514c")
```

یک F8 می شود :

```
004E05EB  . E8 D449F2FF    CALL Registry.00404FC4
004E05F0  . 8B55 C0        MOV EDX,DWORD PTR SS:[EBP-40]
004E05F3  . 8B45 FC        MOV EAX,DWORD PTR SS:[EBP-4]
004E05F6  . E8 B548F2FF    CALL Registry.00404EB0
004E05FB  .v75 35          JNZ SHORT Registry.004E0632
004E05FD  . 8D55 B0        LEA EDX,DWORD PTR SS:[EBP-50]
Stack SS:[0012F1F4]=0137CFC0, (ASCII "45678")
EAX=0012F1B8
```

یعنی کد (c514c) با این (45678) چک مي شود اگر مانند هم بود
JNZ در آدرس (004E05FB) عمل نمی کند و پیغام رجیستر رو خواهیم دید!
اما این دوتا با هم برابر نیست برای همین پرش انجام مي شود و پیغام خطا ظاهر می شود .

حال اگر کد اولی رو (c514c) جای گزین عدد (45678) در سریال نامبری که وارد کردیم بکنیم
و سریال رو وارد کنیم پیغام رجیستر خواهید داشت !
Valid Serial =(123c-514c-9000-1000)

تا به اینجای کار فهمیدیم که :

- کلمه (Easy Gang) تبدیل به کد (MD5) می شود .
- (ec7235e71546232da9f7cc02f32c9d0e)
- 10 رقم از این کد جدا می شود (چرا این ده رقم جدا می شود توضیح داده خواهد شد) .
- (c7235e7154)
- قسمت های سوم و چهارم سریال نامبر با کد 10 رقمی جداشده بالا مخلوط می شود .
- (90001000 c7235e7154)
- عدد مخلوط شده با یک عدد دیگر جمع می شود .
- (90001000 c7235e7154 3)
  (چگونگی اضافه شدن این عدد و این که از کجا آمد توضیح داده خواهد شد) .
- کد مخلوط شده با (MD5) Hash می شود و کد جدید به دست می اید .
- (a5c514ce566dcfc8856f757c45998c28)
- 5 رقم از این کد جدا می شود .
- (c514c)
- 5 رقم جدا شده با 5 عدد (45678) مقایسه می شود .
- اگر برابر بود سریال صحیح است !

نتیجه غیر اخلاقی :

اگر توجه کرده باشید می توانید پی ببرید که کد قسمت دوم (5678) اصلاً نقشی
در ایجاد کد تا به اینجا نداشته و فقط در انتها نقش برابری رو دارد !
شاید تا به حال به این فکر کرده باشید که کد قسمت اول (1234) هم اینگونه بوده و
تنها قسمت های سوم (9000) و چهارم (1000) در تولید کد نقش داشته اند .
باید بگم این اشتباه است و در ادامه خواهیم فهمید قسمت اول کلیدی ترین قسمت برای ایجاد کد است !

تحلیل کد های به دست آمده و طریقه ایجاد کد های اصلی (Keygen) :

در نظر بگیرید کد Hash شده ی :
(a5c514ce566dcfc8856f757c45998c28)
به نظر شما چرا در مرحله جدا سازی 10 رقم (مرحله دوم) کد (c7235e7154)
جدا شد چرا کد 10 رقمی (757c45998c) جدا نشد؟
اینجاست که به نقش قسمت اول سریال (1234) پی میبریم !
اصول نوشتن Keygen برای این برنامه دانستن این رابطه ها است.

خوب در قسمت اول داشتیم (1234) .
به شکل زیر توجه کنید :

```
1234-5678-9000-1000
MD5 : ec7235e71546232da9f7cc02f32c9d0e
```

یعنی یکی رفته جلو بعد 10 تا جدا شده .
اینجا مشخص هست به مقدار عدد اول جلو رفته .
مثال : اگر قسمت اول ما به صورت زیر باشه :
(5235)
5 تا از کد Hash جلو میره بعد 10 تا عدد جدا میکنه می شود شکل زیر :

```
5234-5678-9000-1000
MD5 : ec7235e71546232da9f7cc02f32c9d0e
```

حالا این عدد را با یک عدد دیگه جمع میکنه که من تا به حال هیچ توضیحی قبلش نداده بودم .
شکل زیر رو در نظر بگیرید :

```
1234-5678-9000-1000
MD5 : ec7235e71546232da9f7cc02f32c9d0e
c7235e7154 + 3
```

آن عدد سوم ما هست که با این کد جمع می شود !
قبلش هم که کد های قسمت سوم و چهارم بود یعنی می شود :
(90001000 c7235e7154 3)
حالا این کد دوباره Hash می شود و کد جدید به دست می اید .
حالا نوبت 5 رقم بعدی می شود .
به شکل زیر دقت کنید :

1234-5678-9000-1000
MD5 : a5c514ce566dcfc8856f757c45998c28

به تعداد عدد دوم میره جلو بعد 5 تا عدد جدا میکنه !
این همان عددی هست که باید مقایسه بشود .
مثلا اگه عدد دوم 7 باشه اول 7 تا میریم جلو بعد 5 تا می شمریم و جدا می کنیم می شود : (e566d)

خوب بلاخره تحلیل کد انجام شد .
اکنون شما به رابطه های بین سریال وارد شده اشنا شده اید .
تنها با در نظر گرفتن این رابطه ها شروع به نوشتن Keygen کنید .
من این کار رو هم کرده ام و با زبان دلفی کد رو نوشته ام.
به کد دقت کنید تنها از رابطه ها استفاده شده !

SP tnx to :

All Persian cracker – Shabgard.org , Unreal-RCE.net & Security team Deltahacking.net
Snaker, Qwerton, Jibz for PEiD – Oleh Yuschuk for OllyDBG
All my friend ,All deltahacking ,Unreal & shabgard members… and You !

```
function Random_Serial(PLen: Integer): string;
var char: string;
begin
  Randomize;
  char    := '1234657890';
  Result  := '';
  repeat
    Result := Result + Char[Random(Length(Char)) + 1];
  until (Length(Result) = PLen)
end;

function Generate_Serial : string;
var r1,{r2,}r3,r4,R_Total,n1,n2,n3,r_g,r_g2,r_g3,Part1,Part2,MD5_H,Final_Serial: string;
    i,j: integer;
begin
  result := 'Error : contact impostor_76171@yahoo.com';
  r1 := Random_Serial(4);
  //r2 := Random_Serial(4);//we dont need it !
  r3 := Random_Serial(4);
  r4 := Random_Serial(4);
  n1:='ec7235e71546232da9f7cc02f32c9d0e';

  r_g := copy(r1,0,1);//get the firs char of random serial1 (r1)
  for i := strtoint(r_g)+1 to length(n1) do n2 := n2 + (n1[i]);
  n2 := copy(n2,0,10);
  r_g2 := copy(r1,3,1);
  R_Total := (r3+r4) + (n2) + (r_g2);

  MD5_H := StrMD5(R_Total);

  r_g3 := copy(r1,2,1);
  for j := strtoint(r_g3)+1 to length(MD5_H) do n3 := n3 + (MD5_H[j]);
  n3 := copy(n3,0,5);

  Part1 := copy(r1,0,3) + copy(n3,0,1);
  Part2 := copy(n3,2,4);
  Final_Serial := Part1 + '-' + Part2 + '-' + r3 + '-' + r4;

  Result := Final_Serial;
end;
```

موفق باشید

مهدی هزاوه ای

(c) 2007

Feedback : impostor_76171@yahoo.com - www.impostor.blogfa.com