KEYGENING X3CHUN's Crypto KeygenMe #2

by bLaCk-eye

No intro this time coz we all know how much I suck in making intro's :)

Target: Crypto KeygenMe #2 by x3chun
Dificulty: 1-2 of 10
Requirements: Keygen
Protection: serial
Date (of this tutorial) : 18.11.2003, 18:00
Link: www.crackmes.de

This target is compiled in visual c and it's not protected so we only need Olly and Ida Disassemble the target with Ida.
Start the target with olly and put a breakpoint on GetDlgItemtextA (search on the web iyou are a complete newbie).This is the code:
-for getting the user name

```
.text:004010FB 6A 64                    push    64h             ; nMaxCount
.text:004010FD 50                       push    eax             ; lpString
.text:004010FE 68 E9+                   push    3E9h            ; nIDDlgItem
.text:00401103 56                       push    esi             ; hDlg
.text:00401104 FF D7                    call    edi ; GetDlgItemTextA
.text:00401106 8B D8                    mov     ebx, eax
.text:00401108 83 FB+                   cmp     ebx, 3
.text:0040110B 7D 1C                    jge     short loc_401129
```

so the user name must be greater then 3 chars

-for getting the serial

```
.text:0040112F 6A 64                    push    64h             ; nMaxCoun t
.text:00401131 51                       push    ecx             ; lpString
.text:00401132 68 E8+                   push    3E8h            ; nIDDlgItem
.text:00401137 56                       push    esi             ; hDlg
.text:00401138 FF D7                    cal l    edi ; GetDlgItemTextA
.text:0040113A 83 F8+                   cmp     eax, 10h
.text:0040113D 75 CE                    jnz     short loc_40110D
```

so the serial must 16 chars long.
Enter a name and a 16 chars long. Now comes this

```
.text:00401145 52                                push    edx
.text:00401146 E8 B5 FE FF FF                    call    sub_401000
.text:0040114B 89 85 B0 FD FF FF                 mov     [ebp+var_250], eax
.text:00401151 8D 85 28 FC FF FF                 lea     eax, [ebp+var_3D8]
.text:00401157 50                                push    eax
.text:00401158 E8 A3 FE FF FF                    call    sub_401000
.text:0040115D 89 85 B4 FD FF FF                 mov     [ebp+var_24C], eax
```

These two calls get the serial in hex format, or you like it better they are something like in ascii 2 text and the result are two dwords.

Next:

```
.text:00401163 8D 8D 40 FF FF FF        lea     ecx, [ebp+var_C0]
.text:00401169 51                       push    ecx
.text:0040116A E8 31 08 00 00           call    sub_4019A0
.text:0040116F 8D 55 9C                 lea     edx, [ebp+String]
.text:00401172 53                       push    ebx
.text:00401173 8D 85 40 FF FF FF        lea     eax, [ebp+var_C0]
.text:00401179 52                       push    edx
.text:0040117A 50                       push    eax
.text:0040117B E8 60 08 00 00           call    sub_4019E0
.text:00401180 8D 8D 40 FF FF FF        lea     ecx, [ebp +var_C0]
.text:00401186 8D 95 78 FE FF FF        lea     edx, [ebp+var_188]
.text:0040118C 51                       push    ecx
.text:0040118D 52                       push    edx
.text:0040118E E8 3D 09 00 00           call    sub_401AD0
.text:00401193 83 C4 20                 add     esp, 20h
```

This code uses the name to get a message digest of 128 bits. At first I thought it's md5 but it turned out it isn't md2,md4 or md5. If you know the algo please mail me.

The message digest (md) is saved in var_188.

Now the hard to understand part (may look like that):

```
.text:00401196 55                       push    ebp
.text:00401197 33 C0                    xor     eax, eax
.text:00401199 89 45 0C                 mov     [ebp+arg_4], eax
.text:0040119C 8D B5 78 FE FF FF        lea     esi, [ebp+var_188]
.text:004011A2 8B 06                    mov     eax, [esi]
.text:004011A4 8B 5E 04                 mov     ebx, [es i+4]
.text:004011A7 8B 4E 08                 mov     ecx, [esi+8]
.text:004011AA 8B 56 0C                 mov     edx, [esi+0Ch]
.text:004011AD 89 85 90 FA FF FF        mov     [ebp+var_570], eax
.text:004011B3 89 9D 94 FA FF FF        mov     [ebp+var_56C], ebx
.text:004011B9 89 8D 98 FA FF FF        mov     [ebp+var_568], ecx
.text:004011BF 89 95 9C FA FF FF        mov     [ebp+var_564], edx
.text:004011C5 8D 9D B0 FD FF FF        lea     ebx,  [ebp+var_250]
.text:004011CB BA 20 37 EF C6           mov     edx, 0C6EF3720h
.text:004011D0 8B 33                    mov     esi, [ebx]
.text:004011D2 8B 7B 04                 mov     edi, [ebx+4]
.text:004011D5 C7 45 10 20 00 00+       mov     [ebp+arg_8], 20h
.text:004011DC
.text:004011DC                  loc_4011DC:                             ; CODE
XREF: DialogFunc+1D5 j
.text:004011DC 8B C6                    mov     eax, esi
.text:004011DE 8B DE                    mov     ebx, esi
.text:004011E0 8B CE                    mov     ecx, esi
.text:004011E2 C1 E0 04                 shl     eax, 4
.text:004011E5 03 85 98 FA FF FF        add     eax, [ebp+va r_568]
.text:004011EB C1 EB 05                 shr     ebx, 5
.text:004011EE 03 9D 9C FA FF FF        add     ebx, [ebp+var_564]
.text:004011F4 03 CA                    add     ecx, edx
.text:004011F6 33 C8                    xor     ecx, eax
.text:004011F8 33 CB                    xor     ecx, ebx
.text:004011FA 2B F9                    sub     edi, ecx
.text:004011FC 8B C7                    mov     eax, edi
.text:004011FE 8B D8                    mov     ebx, eax
.text:00401200 8B C8                    mov     ecx, eax
.text:00401202 C1 E0 04                 shl     eax, 4
```

```
.text:00401205 03 85 90 FA FF FF                add     eax, [ebp+var_570]
.text:0040120B C1 EB 05                         shr     ebx, 5
.text:0040120E 03 9D 94 FA FF FF                add     ebx, [ebp+var_56C]
.text:00401214 03 CA                            add     ecx, edx
.text:00401216 33 C8                            xor     ecx, eax
.text:00401218 33 CB                            xor     ecx, ebx
.text:0040121A 2B F1                            sub     esi, ecx
.text:0040121C 81 EA B9 79 37 9E                sub     edx, 9E3779B9h
.text:00401222 FF 4D 10                         dec     [ebp+arg_8]
.text:00401225 75 B5                            jnz     short loc_4011DC
.text:00401227 33 C0                            xor     eax, eax
.text:00401229 89 85 90 FA FF FF                mov     [ebp+var_570], eax
.text:0040122F 89 85 94 FA FF FF                mov     [ebp+var_56C], eax
.text:00401235 89 85 98 FA FF FF                mov     [ebp+var_568], eax
.text:0040123B 89 85 9C FA FF FF                mov     [ebp+var_564], eax
.text:00401241 5D                               pop     ebp
.text:00401242 8D 9D B0 FD FF FF                lea     ebx, [ebp+var_250]
.text:00401248 89 33                            mov     [ebx], esi
.text:0040124A 89 7B 04                         mov     [ebx+4], edi
.text:0040124D 81 FE 68 63 33 78                cmp     esi, 'x3ch'
.text:00401253 74 04                            jz      short loc_401259
.text:00401255 83 45 0C 01                      add     [ebp+arg_4], 1
.text:00401259
.text:00401259                      loc_401259:                             ; CODE
XREF: DialogFunc+203 j
.text:00401259 81 FF 29 3A 6E 75                cmp     edi, 'un:)'
.text:0040125F 74 04                            jz      short loc_401265
.text:00401261 83 45 0C 01                      add     [ebp+arg_4], 1
.text:00401265
.text:00401265                      loc_401265:                             ; CODE
XREF: DialogFunc+20F j
.text:00401265 8B 45 0C                         mov     eax, [ebp+arg_4]
.text:00401268 85 C0                            test    eax, eax
.text:0040126A 75 1F                            jnz     short loc_40128B
.text:0040126C 8B 45 08                         mov     eax, [ebp+hDlg]
.text:0040126F 68 D8 60 40 00                   push    offset aGoo dWorks ;
lpString
.text:00401274 68 E8 03 00 00                   push    3E8h            ;
nIDDlgItem
.text:00401279 50                               push    eax             ; hDlg
.text:0040127A FF 15 A4 50 40 00                call    ds:SetDlgItemTe xtA
```

It looks like a procedure of encryption/decryption with takes as password our md and cipher text our hex serial. After looking for a few seconds at it I'm sure you'll all recognized the TEA encryption algorithm which takes a 128 bit password and encrypts an 64 bit buffer (2 dwords).The procedure used here is the decryption one. After looking into my crypto sources I'm almost sure this source is taken from WiteG//xtreeme.(the sources are identical).

After the decription it check if the buffer is now 'hc3x):nu'.

So to make a keygen we need to :

1.Get the user name

2.Hash the user name with the md of the user

3.Encrypt 'hc3x):nu' buffer with the md of the user

That's all.

The code for keygen can be easily ripped off from the crackme (like I did)

Regards

bLaCk@2002

Contact:
bLaCk-eye@post.ro

Greetz:

x3chun : nice keygenme
acid_cool: still waiting for your return
tanatos: :)
solata: where are you?
mankind: how’s school?
kRio: I’ll mail you with a nice block cipher soon.