HOW TO CRACK *Net.Medic 1.2.2*

Tutorial by UmE

Introduction: this time we will crack Net.Medic 1.2.2, a great program to mantain under control you internet connection. We will re-enable some functions that we can not use in demo version.

Necessary tools: Wdasm 8.9.

Program description: Net.Medic 1.2.2, netMedic.exe, 1.030.144 bytes.

PARENTAL ADVISORY: this tutorial is cracking oriented!!!

Step 1: when you run the program the main screen shows you a window with some parameters that the application will monitorize during an Internet session. If you want to see more parameters you can go to the “View” -> “Details” menu and choose an item….when you select one a message box appear and tells you:

“The feature you have requested is available only in the retail version of Net.Medic. If you would like a temporary license to try all the advanced features of Net.Medic for the next 30 days, click OK ”

Now if you push the OK button the program will start with a time trial period of 30 days after that you got to buy the full version of the program. Push the “Cancel” buttom and let’s try to re-enabled the crippled features. Notice: also other features in the “Window” menù are disabled.

Step 2: open W32Dasm and dissasemble the netMedic.exe file. Go to the “Refs” -> “String Data Reference…” menu and search for the string “The feature you have…..”. When you find it double clik on it and W32Dasm will bring you to the piece of code where the string is referenced. You have:

```
:0043F1FE 7403                  je 0043F203
:0043F200 8B7020                mov esi, dword ptr [eax+20]

* Referenced by a (U)nconditional or (C)onditional Jump at Address:
|:0043F1FE(C)
|
:0043F203 8D4DEC                lea ecx, dword ptr [ebp-14]
:0043F206 BB01000000            mov ebx, 00000001

* Reference To: MFC40.Ordinal:01E6, Ord:01E6h
                                |
:0043F20B E888A80400            Call 00489A98
:0043F210 8B45E4                mov eax, dword ptr [ebp-1C]
:0043F213 885DFC                mov byte ptr [ebp-04], bl
:0043F216 83B8C800000005        cmp dword ptr [eax+000000C8], 00000005
```

```
:0043F21D 754E                    jne 0043F26D
```

* Possible Reference to String Resource ID=57882: "The feature you have requested is available only in the reta"

```
                                  |
:0043F21F 681AE20000              push 0000E21A              <- This is the string!!!
:0043F224 8D4DF0                  lea ecx, dword ptr [ebp-10]

* Reference To: MFC40.Ordinal:0E48, Ord:0E48h
                                  |
:0043F227 E878A80400              Call 00489AA4
```

If you look a little above you can notice that this code is referenced by a conditiona jump at the address 0043F1FE. Let's go at 0043F1FE address and we have:

```
:0043F1F4 1BFF                    sbb edi, edi
:0043F1F6 83E7E2                  and edi, FFFFFFE2
:0043F1F9 83C73C                  add edi, 0000003C
:0043F1FC 3BC6                    cmp eax, esi
:0043F1FE 7403                    je 0043F203
:0043F200 8B7020                  mov esi, dword ptr [eax+20]
```

Move upward the code and you'll find that this piece of code is called from the 0043F1AA address. Go to the 0043F1AA address and you find:

```
* Referenced by a CALL at Addresses:
|:004091ED   , :0040922D   , :0040926D   , :004092CD   , :0040932D
|:0040938D   , :004093ED   , :0040944D   , :004094AD   , :0040950D
|:00409BF4   , :00409C41   , :00409C8E   , :00409CDB   , :00409D28
|:00409D9F   , :0040A38E   , :0042703D   , :0042829E   , :0042C18E
|:00437D9D
|
:0043F1A0 8B442404                mov eax, dword ptr [esp+04]
:0043F1A4 B998704A00              mov ecx, 004A7098
:0043F1A9 50                      push eax
:0043F1AA E801000000              call 0043F1B0
:0043F1AF C3                      ret
```

You can notice that this piece of code is referenced from a lot of call. Every call rapresents a disabled function. If you go to analyze the caller code at the various addresses you can notice a similar way to proceed for all the calls; for example at 004091ED (the first call in the list above) you have:

```
:004091ED E8AE5F0300              call 0043F1A0          <- the fatidical call!!
:004091F2 83C404                  add esp, 00000004      <-updates the stack
:004091F5 85C0                    test eax, eax          <-test eax
:004091F7 740D                    je 00409206            <-jump if eax=1 (func. enable)
```

Idem for the others callers.
Now we must do that the function at 0043F1A0 retrives always eax=1…how can we do this? In this way…..just patch the instruction at 0043F1A0 with this instructions:

```
push   00000001              <- push 1 in the stack
pop    eax                   <- load the value you have pushed in eax
ret                          <- return to the caller
```

In exadecimal is: 6A 01 58 C3

The function will retrive now always eax=1 and all the functions will be enable when you'll go to use them.

Ok guys that's all for now. I hope this tutorial could be useful for someone.

Greetings to Volatility and all the Immortal Descendants.

Contact me at: ume15@hotmail.com