

10.Continued reversing techniques in VB, use of decompilers and a basic anti-anti-trick

2012년 1월 29일 일요일

오후 4:00

Hello everybody.

모두들 안녕.

Welcome to this Part 10 in my series about reversing for newbies/beginners.

나의 초보자 reversing series Part 10에 온 것을 환영해.

This "saga" is intended for complete starters in reversing, also for those without any programming experience at all.

이 "saga"는 완벽히 reversing 초보자를 맞춰서 만들어졌다. 또한 어떠한 programming 경험이 없어도 된다.

Lena151 (2006)

Set your screen resolution to 1152*864 and press F11 to see the movie full screen !!!

Again, I have made this movie interactive.

You screen 해상도를 1152*864로 설정해 그리고 full screen으로 movie를 보기 위해 F11를 눌러

So, if you are a fast reader and you want to continue to the next screen, just click here on this invisible hotspot. You don't see it, but it IS there on text screens.

그래서, 네가 이것을 빨리 읽고 다음 screen을 보고 싶다면, 보이는 hotspot 여기를 눌러. 보고 싶지 않을 때는 여기에 두자마.

Then the movie will skip the text and continue with the next screen.

Movie는 text와 다음 screen을 skip 할 수 있다.

If something is not clear or goes too fast, you can always use the control buttons and the slider below on this screen.

무언가 명확하지 않거나 빨리 넘기고자 할 때, 항상 control button과 이 screen 밑에 있는 slider 바를 사용해.

He, try it out and click on the hotspot to skip this text and to go to the next screen now!!!

도전해봐. 그리고 이 text와 다음 screen을 보기 위해 hotspot을 click 해.

During the whole movie you can click this spot to leave immediately

이 movie 어디에서나 즉시 떠나기 위해 이 spot을 click 할 수 있다.

1. Abstract

In this Part 10, we will reverse a freeware application and two ReverseMe's to learn something about Visual Basic (VB) programs, while studying the reversing of a registration scheme.

이번 Part 10 에서, 우리는 freeware application 을 registration scheme 의 reversing 을 공부할 때 two ReverseMe's 를 Visual Basic program 을 배우고 reverse 할 것이다.

For better comprehension and if you are a newbie, I advise you to first see the previous parts in this series before seeing this movie.

The goal of this tutorial is to teach you something about a program's behaviour.

이 tutorial 의 목표는 너에게 program's 의 behaviour 를 가르치는 것이다.

In my search not to harm authors, we will study two VB ReverseMe's and a freeware VB application "protected" by a registration scheme.

나의 연구는 제작자에게 해를 주지 않는다. 우리는 registration scheme 에 의하여 two VB ReverseMe 와 보호된 freeware VB application 을 공부할 것이다.

Here, these applications are only chosen because it is ideal for this reversing tutorial and they are targeted for educational purposes only.

헤헤, 이 applications 은 한 가지 이유로 선택됐다. 그것은 reversing tutorial 에 이상적이고 교육적인 목적에 딱 맞는 target 이다.

I hope you will exploit your newly acquired knowledge in a positive way.

네가 새로 얻은 지식을 긍정적인 방향으로 사용하기를 바란다.

In this matter, I also want to refer to Part 1.

문제가 있으면 Part1 을 참조해.

As always, the material used here is own work or collected and reworked by me.

항상, 사용된 구성요소들은 나의 일에서 또는 수집된 것이다. 나에 의해서 다시 탄생했다.

Special thanks to Eternal Bliss for his material

외적인 Bliss 구성요소에 대해서 특별히 감사한다.

2. Tools and Target

이것도 똑같음

The tools for today are : Ollydebug, SmartCheck, VB Decompiler Lite, Veoveo and... your brain.

오늘의 tools 은 : Ollydebug, SmartCheck, VB Decompiler Lite, Veoveo 그리고 너의 두뇌다.

The first can be obtained for free at

먼저 이곳에서 무료로 얻을 수 있다.

<http://www.ollydbg.de>

Numega Technologies' SmartCheck was a software company that was acquired by Compuware in 1997.

Numega Technologies' SmartCheck 는 Compuware 에 의하여 1997 년에 획득한 software company 였다.

Compuware only continued development of SmartCheck till late 2001. Then it was discontinued.

Compuware 는 SmartCheck 개발이 2001년까지 계속된다. 그 이후는 계속되지 않았다.

SmartCheck was old as shareware.

SmartCheck 는 오래된 shareware 다.

However now, it can still be found on internet as freeware. Just google for it.

I'm using version 6.20 here.

그러나 이제, 여전히 internet에서 무료로 찾을 수 있다. Google에서 찾아봐.

나는 6.20 version 을 사용 중이다.

VB Decompiler lite is freeware and can be downloaded here :

VB Decompiler lite 는 freeware이고 여기에서 download 받을 수 있다 :

<http://www.vb-decompiler.org/files/lite.rar>

Spanish version of VeoVeo can be downloaded from author's site, sources included, at:

Spanish version VeoVeo 는 제작자의 site에서 download 받을 수 있다. Sources도 포함되어 있다.

<http://www.terra.es/personal/guillett/archivos/veovo34.zip>

(but I have included the English version)

As usual, the brain is your responsibility ;)

그러나 나는 English version 을 포함했다.

보통, 두뇌는 너의 책임감이다.

Todays targets are two ReverseMe's and a free software called CrackersConvert v1.0

오늘 타겟들은 two ReverseMe's이고 free software이며 CrackersConver v1.0으로 불린다.

3. Behaviour of the program

Because you know meanwhile about the importance of decent study of your target and because you've seen how to do it in the previous Parts in this series, I will leave this task (for all three) to you today.

왜냐하면 너는 중요하고 괜찮은 너의 target 공부를 알고 있다. 왜냐하면 너는 이 series의 이전 Parts에서 어떻게 하는지 봤다. 나는 이 task를 너에게 넘기고 떠날 것이다.(3개의 모든것을)

Let's reverse a program and 2 ReverseMe's, so, I have cut their behavior's study from this movie to reduce the size.

ReverseMe's 2개를 reverse하자. 그들의 behavior's 공부를 이 movie의 size를 줄이기 위해 삭제했다.

Please, go ahead on your own, the executables are all included in this package.

제발, 너의 것들 중에서 앞의 것으로 가라. 이 package에서 Executable은 모든 것을 포함했다.

As you can see, I have already loaded the application in NuMega's SmartCheck.

네가 볼 때, 나는 이미 NuMega's SmartCheck에서 application을 load 했다.

This means that we will deal with VB targets of course.

우리는 VB target 과 거래할 수 있다.

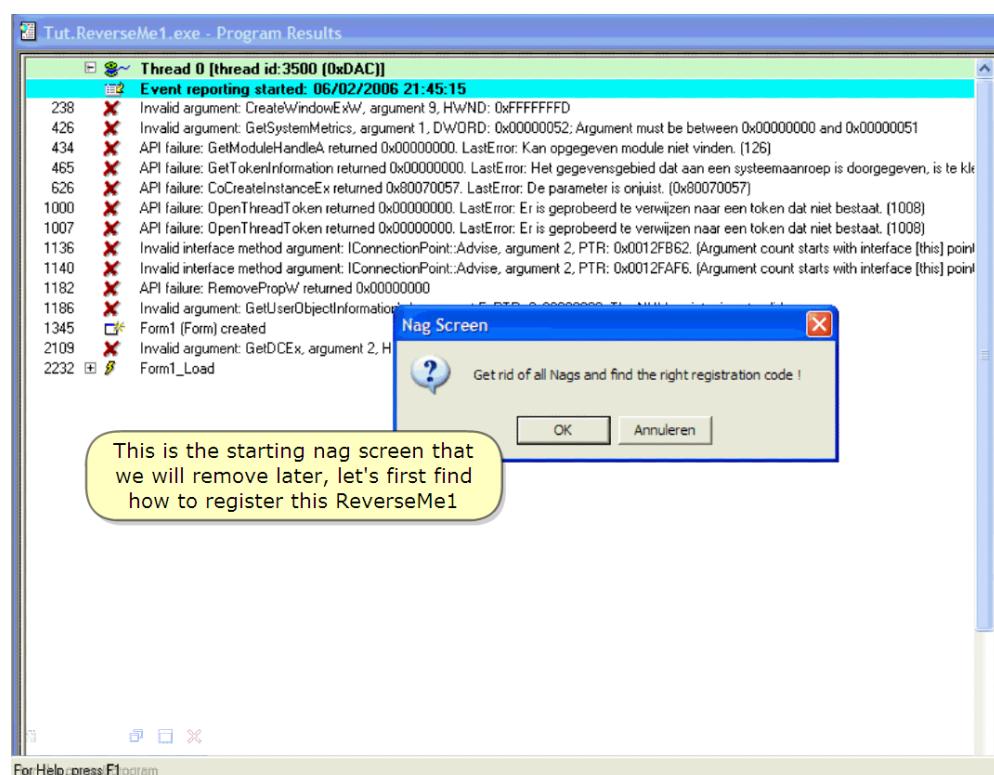
For VB and the use of SmartCheck, I want to refer you to Part09 in this series.

VB 와 SmartCheck 를 사용하는 것은, 나는 이 series 의 Part09 를 참조하기를 원한다.

But there is also a html page included herein, please take a look in upfront, I will use it later. Let's finally start the RverseMe1 ...

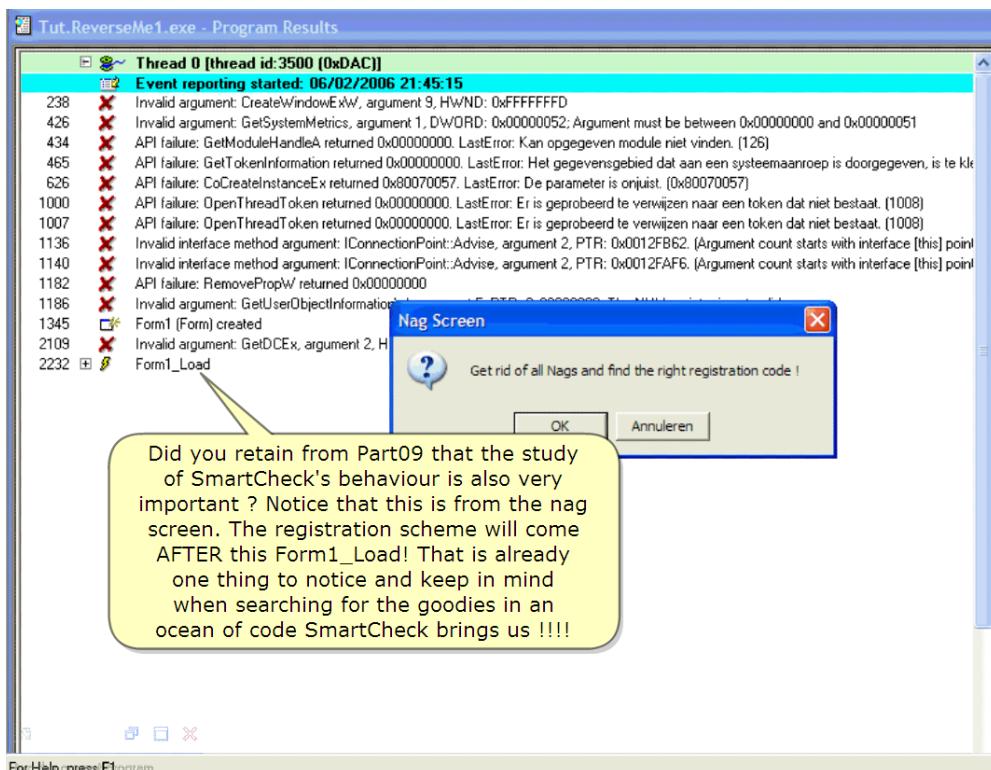
또한 html page 를 첨부했다. 확인해 봐. 나는 나중에 사용할 것이다. 마지막으로 ReverseMe1 을 시작하자.

4. ReverseMe1



This is the starting nag screen that we will remove later, let's first find how to register this ReverseMe1

이것은 starting nag screen 이다. 우리는 나중에 삭제할 것이다. 처음에는 어떻게 ReverseMe1 을 등록하는지 찾자.



Did you retain from Part09 that the study of SmartCheck's behaviour is also very important?

너는 Part09에서 SmartCheck's behavior 를 어떻게 유지할 거야? 이것은 매우 중요한 거야.

Notice that this is from the nag screen. The registration scheme will come AFTER this Form1_Load!

이것은 Nag screen에서 왔다. OI registration scheme 는 Form1_Load 후에 온다.

That is already one thing to notice and keep in mind when searching for the goodies in an ocean of code SmartCheck brings us !!!

우리가 좋은 것을 찾을 때 수 많은 code 에서 SmartCheck 는 우리에게 한 가지를 가져 옵니다.

번역 주)goodies 는 맛있는 regcode 를 말하는듯.

Let's try our luck

우리의 운을 믿고 도전해.

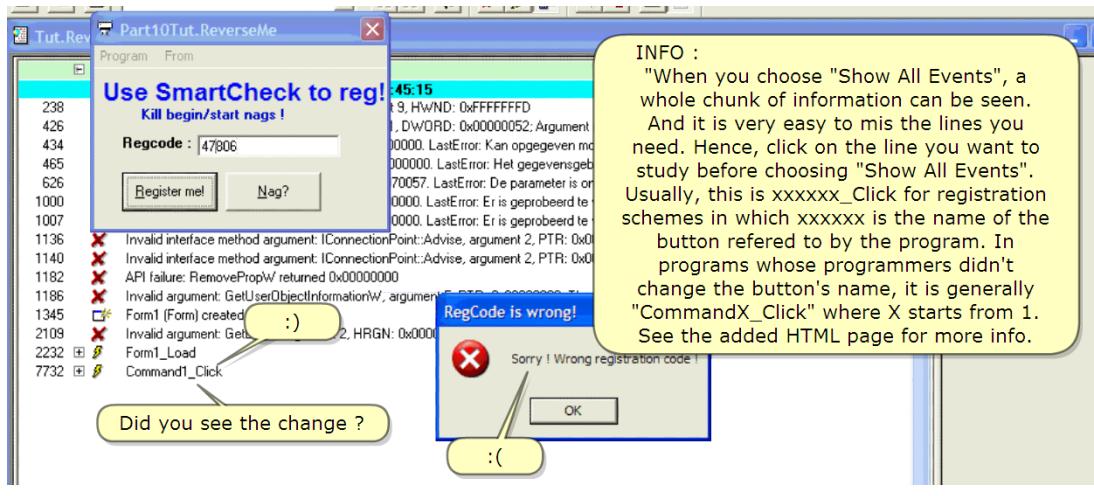
INFO :

Keep an eye on the events here.

Events 창에 눈을 유지해.

It is imperative because this is a very good indication where to look for the goodies ...

이것은 중요하다. 왜냐하면 이것은 우리가 goodies 를 찾기 위한 매우 좋은 방향을 나타낸다.



:)

Did you see the change?

바뀐걸 봤어?

:)

INFO :

"When you choose "Show all Events", a whole chunk of information can be seen.

And it is very easy to miss the lines you need.

"네가 "Show all Events"를 선택할 때, 모든 information 덩어리들은 보였다.

이것은 네가 필요한 lines 을 놓칠 확률이 높다.

Hence, click on the line you want to study before choosing "Show All Events".

그리하여, 네가 공부하고자 원할 때 이 line 을 click 해. 그리고 나서 "Show All Events"를 선택해.

Usually, this is xxxxx_Click for registration schemes in which xxxxx is the name of the button

referred to by the program.

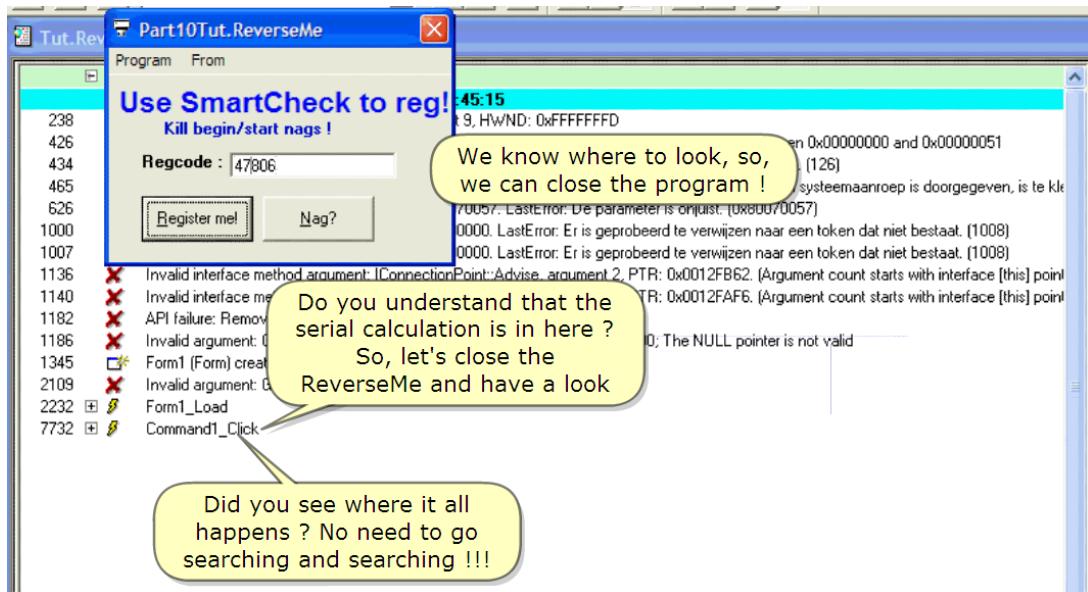
보통, registration scheme 위한 xxxxx_Click 이다. Registration scheme 는 program 에 의하여 참조된 button 의 name 이다.

In programs whose programmers didn't change the button's name, it is generally "CommandX_Click" where X starts from 1.

이 program 에서 programmer 는 button's name 을 바꾸지 않았다. 이것은 보통 "CommandX_Click"은 1 에서 부터 시작한다.

See the added HTML page for more info.

좀 더 많은 정보를 위해 HTML page 에 추가했다.



Did you see where it all happens? No need to go searching and searching !!!

그곳에서 무슨 일이 일어났는지 봤어? 찾는 것이 필요없다.

Do you understand that the serial calculation is in here?

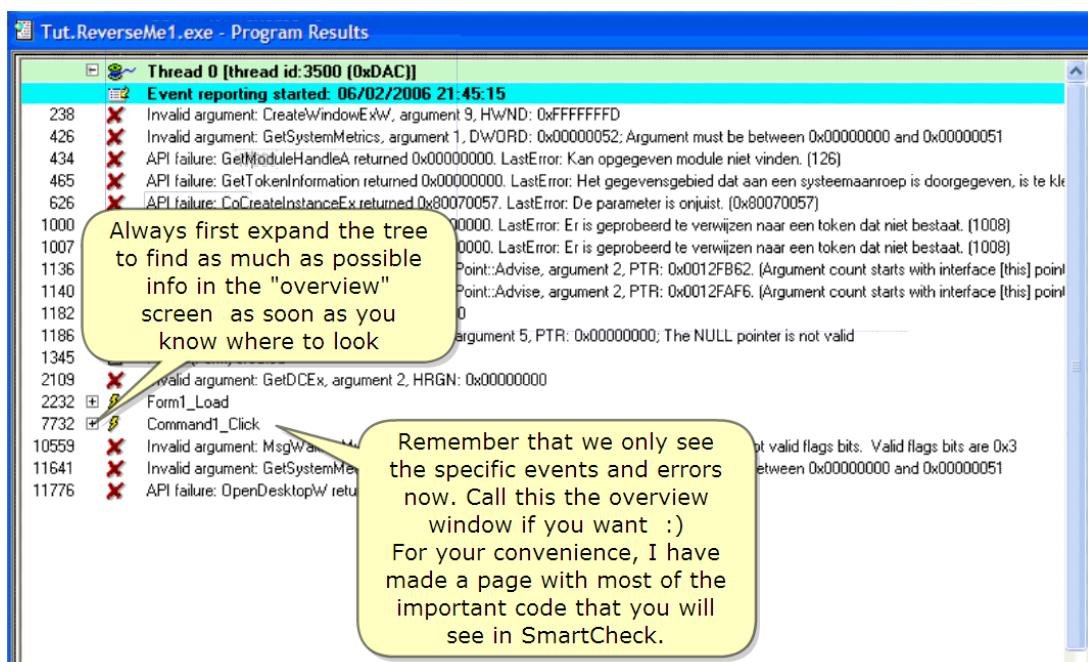
이곳에서 serial 이 계산되는 것을 이해했어?

So, let's close the ReverseMe and have a look

ReverseMe 를 닫고 이걸 봐.

We know where to look, so, we can close the program!

우리는 보기 위해 어디를 봐야 하는지 알고 있다. 그래서, Program 을 닫을 수 있다.



Remember that we only see the specific events and errors now.

기억해. 우리는 이제 오직 명확한 events 와 error 를 볼 수 있다.

Call this the overview windows if you want :)

네가 원한다면 overview window 를 부른다.

For your convenience, I have made a page with most of the important code that you will see in SmartCheck.

너의 편리함을 위해, 나는 대부분 중요한 code page 를 만들었다. 그것을 SmartCheck 에서 볼 수 있다.

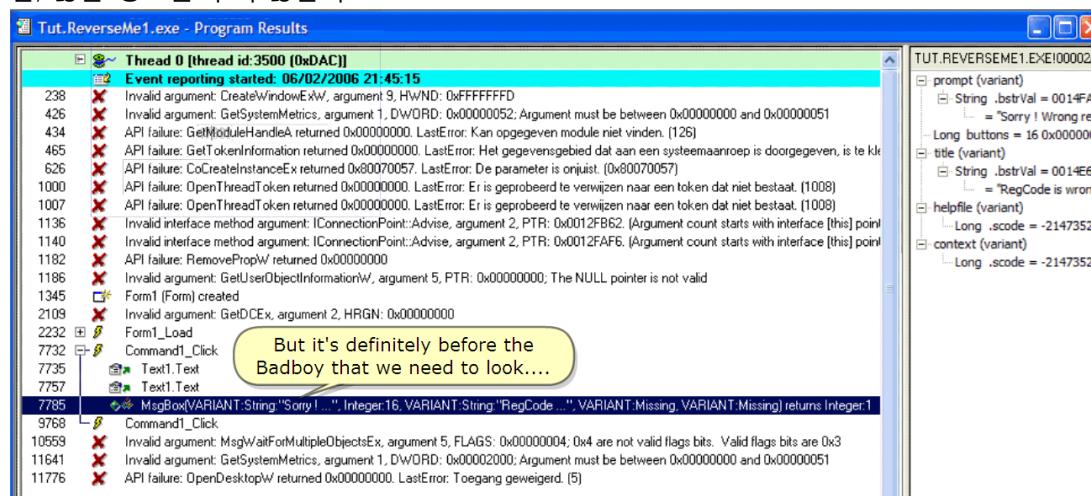
Always first expand the tree to find as much as possible info in the "overview" screen as soon as you know where to look

항상 첫번째로 "overview" screen 에서 네가 볼 수 있는 장소를 알았을 때 가능한 많은 정보를 찾기 위해 tree 를 펼쳐봐라.

번역 주)의심 가는 곳을 펼쳐보라는 뜻

Mmmm, that doesn't give a lot of info !

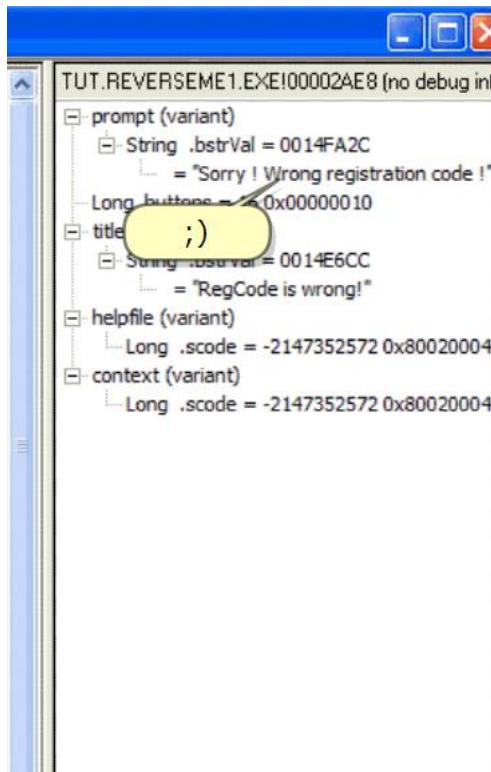
음, 많은 정보를 주지 않는다.



But it's definitely before the Badboy that we need to look....

But it's definitely before the Badboy that we need to look

그러나 이것은 분명하다. 그리고나서 우리가 badboy 를 보기 위해 필요하다.



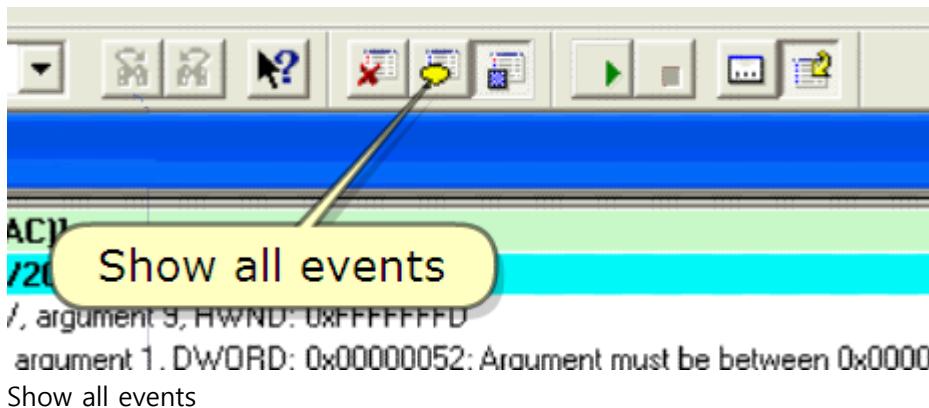
;)

So, let's start looking for the registration in both Text1.Text, starting in the first.

그래서, registration 을 찾기 위해 먼저 Text1.Text 에서 시작하자.

This means, that with this line highlighted we can click "Show all events" and we will probably find the compare after the highlighted line and before the badboy

이 뜻은, highlighted 된 line 과 함께한다. 우리는 highlight 된 line 후에 "Show all events"를 click 할 수 있고 우리는 아마 compare 된 것을 찾을 수 있다. 그리고나서 badboy 가 나타난다.



Study the code

Code 를 공부하자.

BTW, notice that this way of thinking is necessary not to drown in the ocean of code SmartCheck offers us!

By the way, 이 생각하는 방법은 엄청 많은 code 에서 해매지 않도록 SmartCheck 는 우리에게 제공한다.

So, let's study the code here.

여기 Code 를 공부하자.

Tut.ReverseMe1.exe - Program Results

7732	Command1_Click
7733	OLE _TextBox::AddRef(LPINTERFACE:0103BCDC) returns DWORD:1
7734	__vbaObjSet(LPINTERFACE *:0012F484, LPINTERFACE:0103BCDC) returns LPVOID:103BCDC
7735	Text1.Text
7746	__vbaStrCmp(String:"I'mlena1...", String:"47806") returns DWORD:FFFFFF
7747	__vbaFreeStr(LPBSTR:0012F488) returns DWORD:10
7752	__vbaFreeObj(LPINTERFACE *:0012F484)
7755	OLE _TextBox::AddRef(LPINTERFACE:0103BCDC) returns DWORD:1
7756	__vbaObjSet(LPINTERFACE *:0012F484, LPINTERFACE:0103BCDC) returns LPVOID:103BCDC
7757	Text1.Text
7768	__vbaStrCmp(String:"I'mlena1...", String:"47806") returns DWORD:FFFFFF
7769	__vbaFreeStr(LPBSTR:0012F488) returns DWORD:10

In fact, it is not difficult at all. We almost immediately see

In fact, it is not difficult at all. We almost immediately see

사실, 이것은 중요하지 않다. 우리는 대부분 즉시 볼 수 있다.

Tut.ReverseMe1.exe - Program Results

7732	Command1_Click
7733	OLE _TextBox::AddRef(LPINTERFACE:0103BCDC) returns DWORD:1
7734	__vbaObjSet(LPINTERFACE *:0012F484, LPINTERFACE:0103BCDC) returns LPVOID:103BCDC
7735	Text1.Text
7746	__vbaStrCmp(String:"I'mlena1...", String:"47806") returns DWORD:FFFFFF
7747	__vbaFreeStr(LPBSTR:0012F488) returns DWORD:10
7752	__vbaFreeObj(LPINTERFACE *:0012F484)
7755	OLE _TextBox::AddRef(LPINTERFACE:0103BCDC) returns DWORD:1
7756	__vbaStrCmp(String:"zzzzz", String:"yyyyy") returns DWORD:0
7757	__vbaFreeStr(LPBSTR:0012F488) returns DWORD:10
7768	__vbaFreeObj(LPINTERFACE *:0012F484) returns DWORD:1
7769	Text1.Text
7774	__vbaStrCmp(String:"zzzzz", String:"yyyyy") returns DWORD:0
7777	Explanation:
7780	__vbaStrCmp is used for comparing Strings.

__vbaStrCmp(String:"zzzzz", String:"yyyyy") returns DWORD:0

Explanation:

설명:

__vbaStrCmp is used for comparing Strings.

__vbaStrCmp 는 String 을 비교하는데 사용됐다.

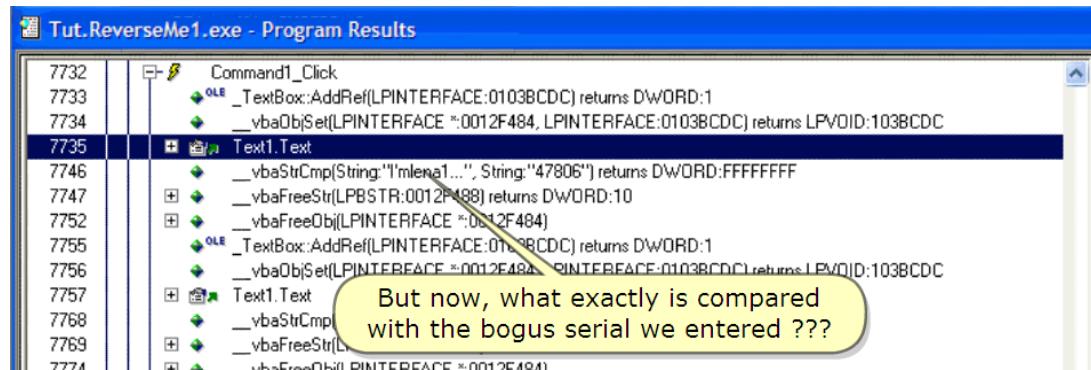
Tut.ReverseMe1.exe - Program Results

7732	Command1_Click
7733	OLE _TextBox::AddRef(LPINTERFACE:0103BCDC) returns DWORD:1
7734	__vbaObjSet(LPINTERFACE *:0012F484, LPINTERFACE:0103BCDC) returns LPVOID:103BCDC
7735	Text1.Text
7746	__vbaStrCmp(String:"I'mlena1...", String:"47806") returns DWORD:FFFFFF
7747	__vbaFreeStr(LPBSTR:0012F488) returns DWORD:10
7752	__vbaFreeObj(LPINTERFACE *:0012F484)
7755	OLE _TextBox::AddRef(LPINTERFACE:0103BCDC) returns DWORD:1
7756	__vbaObjSet(LPINTERFACE *:0012F484, LPINTERFACE:0103BCDC) returns LPVOID:103BCDC
7757	Text1.Text

Here, result is NOT EQUAL because
DWORD == FFFFFFFF

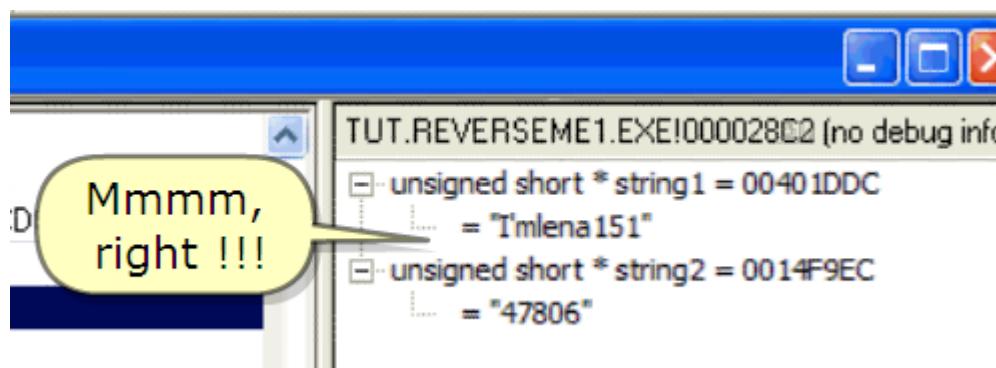
Here, result is NOT EQUAL because DWORD == FFFFFFFF

여기, result 는 같지 않다. 왜냐하면 DWORD == FFFFFFFF



But now, what exactly is compared with the bogus serial we entered ???

그러나 이제, 정확히 무엇을 비교했는지 serial 과 함께 보자. 들어가자???



Mmmm, right !!!

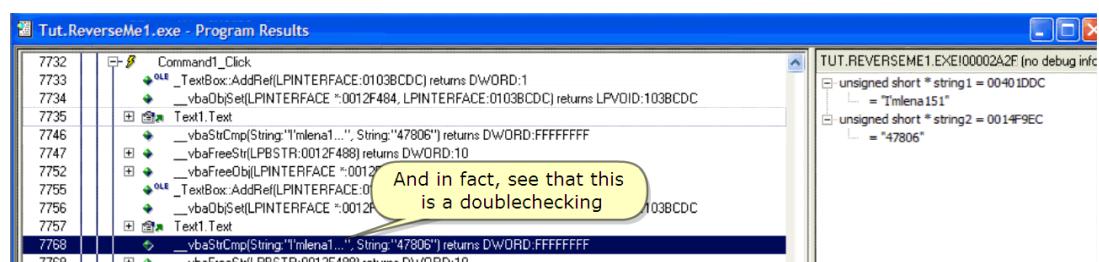
음, 좋아 !!!

:)

INFO :

The HTML page included in this package contains most of the important code used in SmartCheck. Take a look!

이 package 에 포함된 HTML page 는 SmartCheck 에 대부분 중요한 code 가 들어있다. 살펴보자!



And in fact, see that this is a doublechecking

사실, 우리는 doubleclick 할 수 있다.

The screenshot shows the assembly view and memory dump of the application. The assembly view highlights a call to `MessageBox` at address 0012F49C. The memory dump window shows the contents of memory at that address, which is a message box with the text "Sorry ! Wrong registration code !" and "RegCode is wrong!". A yellow callout box points to this message box.

... before the Badboy is displayed.

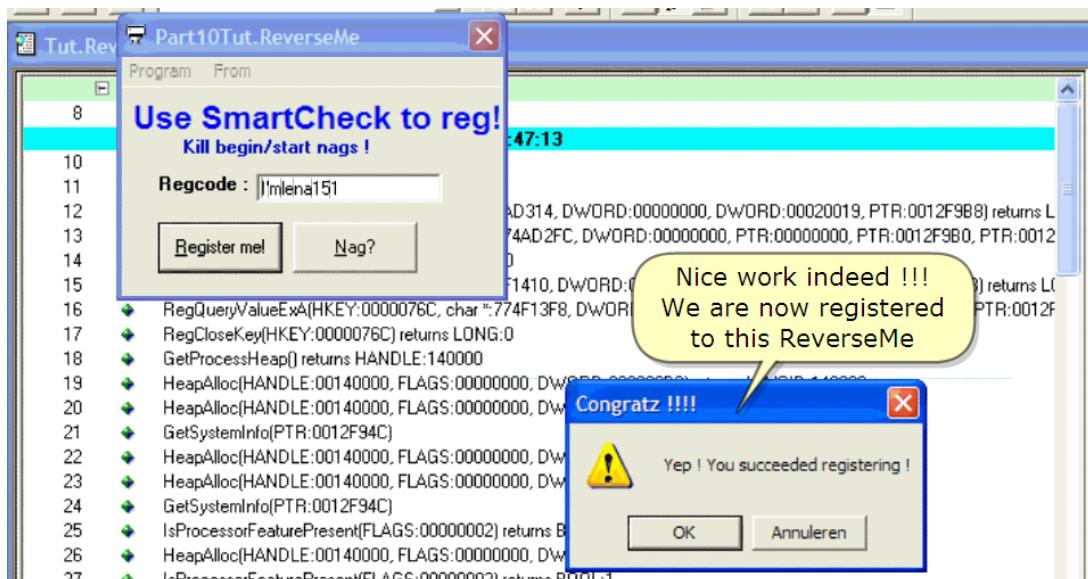
그러고나서 Badboy 는 display 됐다.

Ok. We know what must be the serial, so let's try it out.

Ok. 우리는 serial 이 무엇인지 알고 있다. 도전해 보자.

BTW, is it also clear that this is a hardcoded serial ? There is no calculation with the serial whatsoever ---> hardcoded

By the way, 이것 또한 명확하다. 그것은 hardcoded 된 serial 이야? 어떤 serial 이든지 계산하는 곳이 없다 ---> hardcoded



Nice work indeed!!!

정말 잘했어!!!

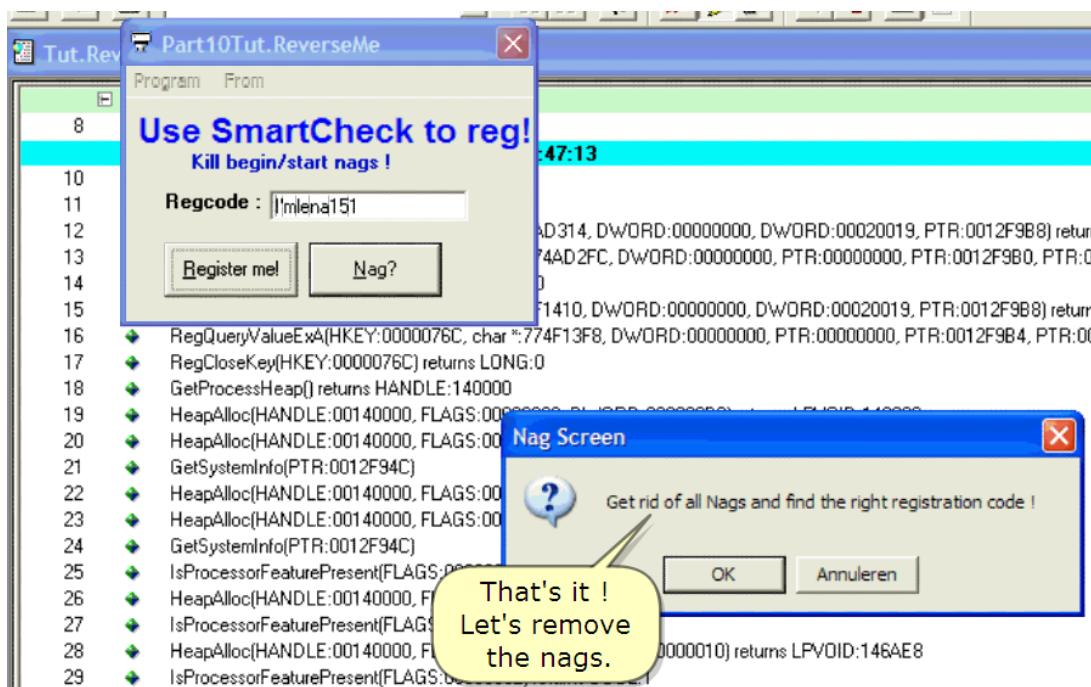
We are now registered to this ReverseMe

우리는 이미 ReverseMe 에 등록됐다.

But remember the nag (at startup and if we click this button).

Let's see it.

그러나 nag 을 기억해(startup 할 때 그리고 이 button 을 click 할 때)
봐라.



That's it !

Let's remove the nags.

이거다.

Nag 을 삭제하자.

SmartCheck is a good tool for helping to fish serials in VB. However, for nag removal, we have better tools at our disposal.

SmartCheck 는 VB 에서 serial 을 낚는데 도움을 주는 좋은 tool 이다. 하지만, nag 제거 할 때 좀 더 좋은 tool 이 있다.

There exist also VB Decompiler tools. At our disposal.

VB Decompiler tools 또한 존재한다. 우리가 제거할 때.

For example VB Decompile Lite or Pro. The version I'm using here is the Lite (free) version.

이 예제에서는 VB Decompiler Lite 를 Pro. 여기에서 사용하고 있는 버전은 Lite(무료) version 이다.

Let's try it, so close and decompile this ReverseMe.

도전해봐, 우리는 단을 수 있고 ReverseMe 를 decompile 한다.

;)

I opened the ReverseMe1.exe already

나는 이미 ReverseMe1.exe 를 열었다.

INFO :

A compiler creates executable code from source code while a Decompiler attempts to reverse this process.

Decompiler 는 이 process 를 reverse 하기 위해 시도할 때 Compiler 는 source code 에서 executable code 를 만들었다.

A Decompiler is a program which takes executable code and produces source code from it.

Decompiler 는 executable code 를 가지고 있고 이것에서 source code 를 생산하는 program 이다.

A Decompiler is a specialized version of a disassembler.

Decompiler 는 disassembler 의 특화된 version 이다.

While a disassembler converts executable code to assembly language, a Decompiler attempts to go further and convert the executable code to source code in a higher level language, such as C or C++ (or VB like in our case here).

Disassembler 가 executable code 를 assembly language 로 변환할 때, Decompiler 는 더 멀리 가기 위해 시도한다. 그리고 higher level language 인 C 나 C++처럼 executable code 에서 source code 로 변환한다.(or VB 도 이 경우에는 같다)

And it seems VB Decompiler Lite has done a good job indeed :

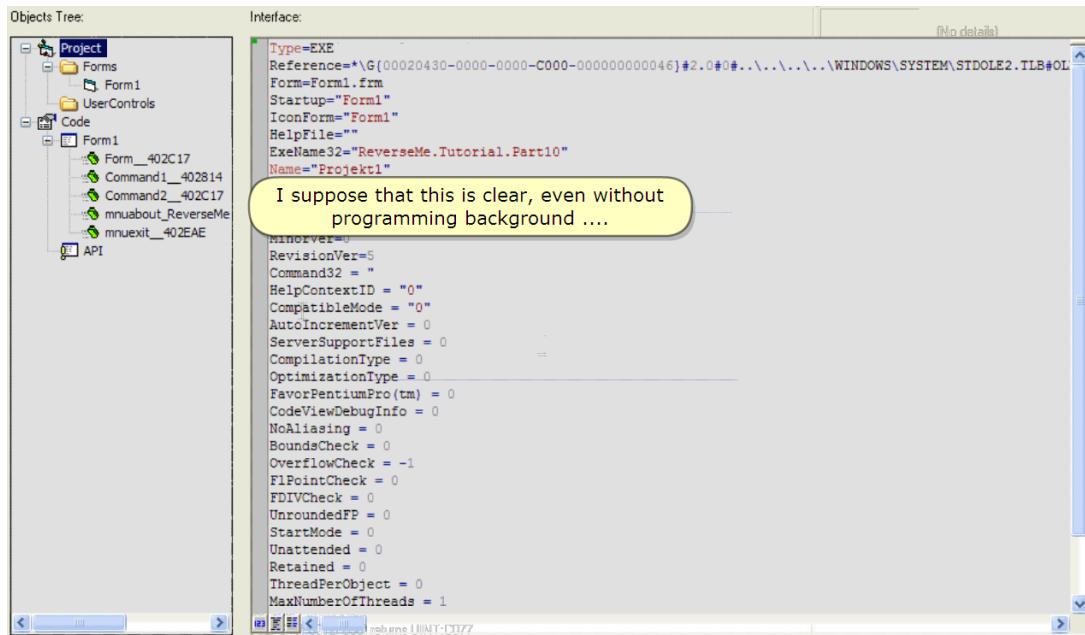
VB Decompiler Lite 는 좋은 job 을 가지고 있다고 볼 수 있다.

The screenshot shows the VB Decompiler Lite interface. On the left, the 'Objects Tree' pane displays a project structure with nodes for Project, Forms (containing Form1), UserControls, and Code. A callout bubble points to the 'Code' node with the text 'Let's immediately take a look in the code.'. The right pane, titled 'Interface:', shows a large amount of source code for an EXE file. The code includes declarations for Form1.frm, Form1.cs, and Form1.Designer.cs, along with various properties and methods. The code is color-coded for syntax highlighting.

```
Type=EXE
Reference*: \G{00020430-0000-0000-C000-000000000046}\#2.0#..\..\..\..\WINDOWS\SYSTEM\STDOLE2.TLB#OLE
Form=Form1.frm
Startup="Form1"
IconForm="Form1"
HelpFile=""
ExeName32="ReverseMe.Tutorial.Part10"
Name="Projekt1"
Title="Tutorial.Part10"
VersionCompanyName=""
MajorVer=1
MinorVer=0
RevisionVer=5
Command32=" "
HelpContextID = "0"
CompatibleMode = "0"
AutoIncrementVer = 0
ServerSupportFiles = 0
CompilationType = 0
OptimizationType = 0
FavorPentiumPro(tm) = 0
CodeViewDebugInfo = 0
NoAliasing = 0
BoundsCheck = 0
OverflowCheck = -1
FPPointCheck = 0
FDIVCheck = 0
UnroundedFP = 0
StartMode = 0
Unattended = 0
Retained = 0
ThreadPerObject = 0
MaxNumberOfThreads = 1
```

Let's immediately take a look in the code.

Code 에서 즉시 볼 수 있다.



I suppose that this is clear, even without programming background

나는 그것은 명확하다고 programming 경험이 없어도 된다고 생각한다...



About box

Exit

Clicking the "Nag?" button

Nag

Clicking the Register button

And where does the Nag routine start in both cases? Indeed at VA 402C17.

Let's see the code by doubleclicking here

Nag routine 은 VA402C17 인 양쪽의 cases에서 실행한다.

Doubleclick 에 의하여 code 를 보자.

The screenshot shows a debugger's objects tree on the left and an assembly interface on the right. The assembly window displays the following code:

```
00402C5B: xor esi, esi
00402C5D: pop ebx
00402C5E: mov [ebp-00000084h], esi
00402C64: lea edx, [ebp-00000084h]
00402C6A: lea ecx, [ebp-54h]
00402C6D: mov [ebp-24h], esi
00402C70: mov [ebp-34h], esi
00402C73: mov [ebp-44h], esi
00402C76: mov [ebp-54h], esi
00402C79: mov [ebp-64h], esi
00402C7C: mov [ebp-74h], esi
00402C7F: mov [ebp-000000A4h], esi
00402C85: mov [ebp-7Ch], 00401EF0h ; Get rid of all Nags and find the right registration code !
00402C8C: mov [ebp-00000084h], ebx
00402C92: call 0040112Ch
00402C97: push 00000003h
00402C99: lea edx, [ebp-00000084h]
00402C9F: pop edi
00402CA0: lea ecx, [ebp-24h]
00402CA3: mov [ebp-7Ch], 00000021h
00402CAA: mov [ebp-00000084h], edi
00402CBB: call 0040112Ch
00402CBS: lea edx, [ebp-00000084h]
00402CBB: lea ecx, [ebp-34h]
00402CBE: mov [ebp-7Ch], 00401F78h ; Nag Screen
00402CCE: mov [ebp-00000084h], ebx
00402CCE: call 0040112Ch
00402CD0: push 000000Ah
00402CD2: mov ecx, 80020004h
00402CD7: pop eax
00402CD8: mov [ebp-6Ch], ecx
00402CDB: mov [ebp-74h], eax
00402CDE: mov [ebp-64h], eax
```

Annotations in the assembly window:

- !!! There is no doubt !
- The nag screen is here with text ...
- ... and title !!

!!! There is no doubt !

!!! 그곳에 대해 의심이 없다!

The nag screen is here with text ...

Nag screen 은 여기에 text 와 함께 있다.

... and title !!

그리고 title !!

The screenshot shows a debugger's objects tree on the left and an assembly interface on the right. The assembly window displays the following code, continuing from the previous snippet:

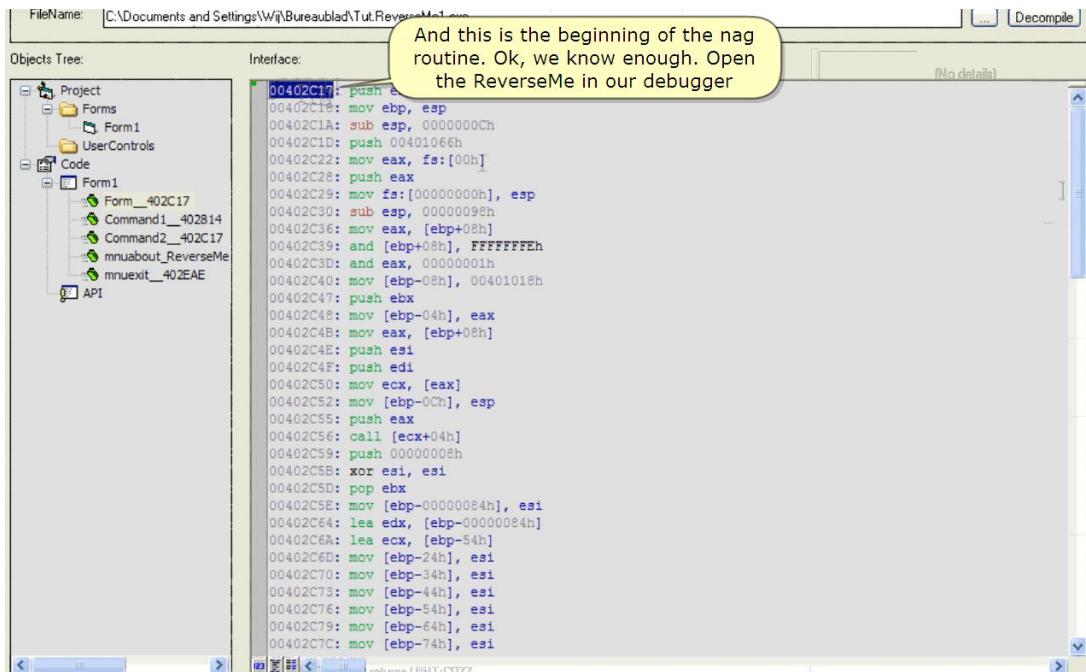
```
00402D03: lea edx, [ebp-000000A4h]
00402D09: lea ecx, [ebp-44h]
00402D0C: mov [ebp-000009Ch], eax
00402D12: mov [ebp-000000A4h], edi
00402D18: call 0040112Ch
00402D1D: lea eax, [ebp-74h]
00402D20: push eax
00402D21: lea eax, [ebp-64h]
00402D24: push eax
00402D25: push 00000002h
00402D27: call 00401114h
00402D2C: add esp, 000000Ch
00402D2F: lea eax, [ebp-44h]
00402D32: mov [ebp-7Ch], 00000001h
00402D39: mov [ebp-00000084h], 00000003h
00402D43: push eax
00402D44: lea eax, [ebp-00000084h]
00402D4A: push eax
00402D4B: call 0040110Eh
00402D50: test ax, ax
00402D53: jnz 402D5Ah
00402D55: call 00401108h
00402D5A: mov [ebp-04h], esi
00402D5D: push 00402D98h
00402D62: jmp 402D77h
00402D64: lea eax, [ebp-74h]
00402D67: push eax
00402D68: lea eax, [ebp-64h]
00402D6B: push eax
00402D6C: push 00000002h
00402D6E: call 00401114h
00402D73: add esp, 00000003h
00402D76: ret
```

Annotation in the assembly window:

End of the nag code.
Scroll up

End of the nag code. Scroll up

Nag code 의 끝이다. Scroll up



And this is the beginning of the nag routine. Ok, we know enough.

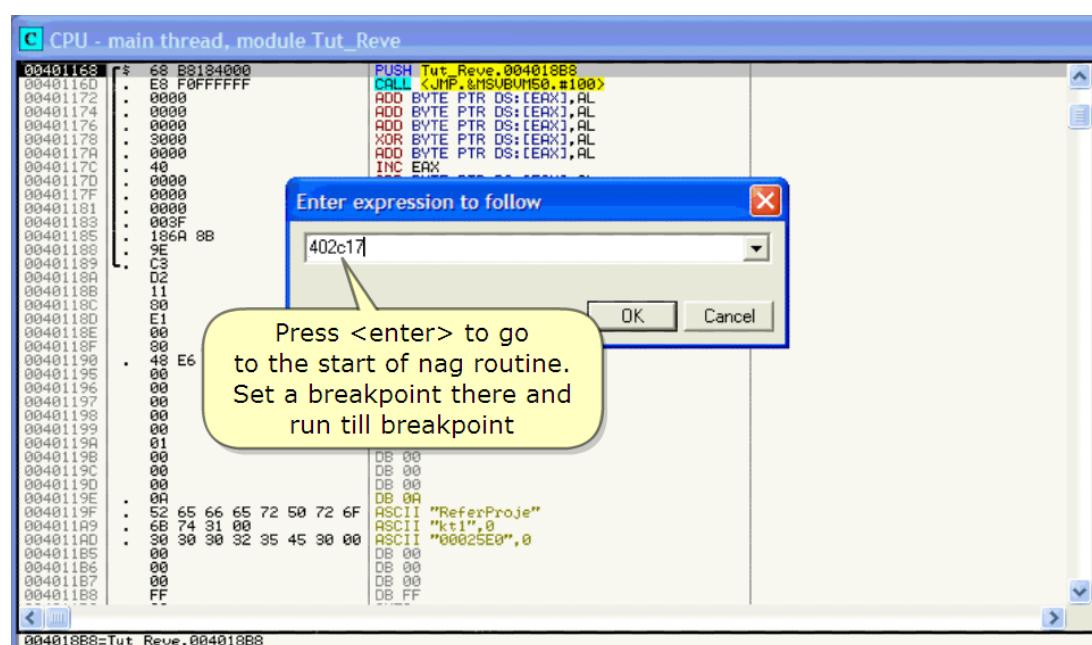
Open the ReverseMe in our debugger

그래서 이것은 nag routine 의 처음이다. Ok. 우리는 충분히 알았다.

Debugger에서 ReverseMe를 열었다.

And so we land here because I have removed it from this movie for size.

그리고 우리는 여기에 도착했다. 왜냐하면 나는 이것을 movie에서 size를 줄이기 위해 살펴봤다.



Press <enter> to go to the start of nag routine.

Set a breakpoint there and run till breakpoint

<enter>를 눌러서 nag routine의 처음으로 가자.

Breakpoint 를 그곳에 설치하고 breakpoint 까지 실행해.

Bam. This is the start of the nag routine

```
00402B65 57 PUSH EDI
00402B66 8908 00000000 MOU ECX,DWORD PTR DS:[EAX]
00402B6A 04 MOU [LOCAL_3],ESP[ERX].RL
00402B6D 50 PUSH EAX PTR DS:[ERX].RL
00402B70 FF51 04 CALL DWORD PTR DS:[ECX+4]
00402B73 E8012E5FFF CALL <JMP.&MSUBUM50..._vbaEnd>
00402B76 8965 FC 00 AND [LOCAL_11],ERX,RL
00402B79 08 MOU EAX,[ARG_11]
00402B7C 50 PUSH EAX
00402B7F 8908 MOU ECX,DWORD PTR DS:[EAX]
00402B82 FF51 08 CALL DWORD PTR DS:[ECX+8]
00402B85 8940 EC MOU ECX,[LOCAL_5]
00402B88 50 MOU EAX,[LOCAL_11]
00402B8B 8945 FC POP EDI
00402B8E 5F POP ESI
00402B91 64:8900 00000000 MOU DWORD PTR FS:[0],ECX
00402B94 5B POP EBX
00402B96 C9 LEAVE
00402B98 C2 0400 RETN 4
00402C17 > 55 PUSH EBP
00402C18 89EC MOU EBP,ESP
00402C1A 89C0 00 00 SUB ESP,00 00
00402C1D 68 66104000 PUSH <JMP.&MSUBUM50..._vbaExceptionHandler>
00402C22 40 00 00 00 ADD PTR FS:[0]
00402C28 50
00402C29 64: FS:[0],ESP
00402C2B 81 C1 00 00 00 00 PTR SS:[EBP+8]
00402C2E 8955 00 00 00 00 00 00 MOU PTR SS:[EBP+8],FFFFFFFE
00402C3D 89E5 01 AND ERX,1
00402C40 00 00 00 00 00 00 00 MOU DWORD PTR SS:[EBP-8],Tut_Reve.00401010
00402C47 69 PUSH EBX
00402C48 8945FC6 65 72 50 72 6F MOU DWORD PTR SS:[EBP-4],ERX
00402C4B 8945-08:10 00 00 00 00 00 MOU ERX,DWORD PTR SS:[EBP+8]
00402C4E 56 90 98 92 95 45 90 00 PUSH ESI,[ECX+8]
00402C4F 57 PUSH EDI
00402C50 8B08 MOU ECX,DWORD PTR DS:[EAX]
00402C52 8955 F4 MOU DWORD PTR SS:[EBP-8],ESP
00402C55 5B PUSH ERX
```

Bam. This is the start of the nag routine

음. Nag routine 의 시작이다.

Break in BP

BP 에서 멈춘다.

And see where we will return after this nag routine.
BTW, I will explain the use of the stack better in a later Part in this series.

0012FB04	7404E5A9	RETURN to MSUBUM50.7404E5A
0012FB08	0014B481	ASCII "B@"
0012FB0C	0012FB1C	Tut_Reve.00402656
0012FB10	00402656	
0012FB14	0014B4B4	
0012FB18	74031AA3	RETURN to MSUBUM50.74031AA
0012FB1C	0012FB38	
0012FB20	7404E583	RETURN to MSUBUM50.7404E58
0012FB24	00402656	Tut_Reve.00402656
0012FB28	0012FBE4	
0012FB2C	00000002	
0012FB30	00000001	

And see where we will return after this nag routine.

BTW, I will explain the use of the stack better in a later Part in this series.

Nag routine 후에 return 되는 곳을 봐.

By the way, 나는 stack 이 사용되는 것을 이 series 의 나중 Part 에서 설명할 것이다.

So, what if we completely skip this nag routine and return now already ?

완벽히 nag routine 을 skip 하고 return 을 바로 한다면?

INFO:

We can very often just put a return on the first line of a nag routine to skip it completely.

우리는 매우 자주 nag routine 의 first line 을 skip 하기 위해 그것을 완벽히 넣을 것이다.

This is because the VB code is a callback from the VB dll (remember Part 09 ???)

VB code 는 VB dll 에서 온 callback 이다.(Part 09 를 기억해 ???)

```
Paused CPU - main thread, module Tut_Reve
00402EE8 57 PUSH EDI
00402EE9 8B00 MOV ECX, DWORD PTR DS:[EAX]
00402EEA 8965 MOU EFL0AL_30, [ESP+1], RL
00402EEB 50 PUSH EAX PTR DS:[EAX], RL
00402EEC FF51 04 CALL DWORD PTR DS:[ECX+4]
00402EEF E8012E5FFF CALL QJMP.&MSUBUH50_<vbaEnd>
00402EF0 8B65 FC 00 AND EFL0AL_11, [EAX], RL
00402EF1 8B45 08 MOU EAX, [EAX+1]
00402EF2 50 PUSH EAX
00402EF3 8B00 MOU ECX, DWORD PTR DS:[EAX]
00402EF4 FF51 08 CALL DWORD PTR DS:[ECX+8]
00402EF5 8B4D EC MOU ECX, [EFL0AL_5]
00402EF6 8B45 FD MOU EAX, [EFL0AL_1]
00402EF7 5E POP EDI
00402EF8 5B POP ESI
00402EF9 64:8900 00000000 MOU DWORD PTR FS:[0], ECX
00402EFA 5B POP EBX
00402EFB C9 LEAVE
00402EFC C2 0400 RETN 4
00402EFD C8 RETN
00402EFE 5BEL
00402EFA 8SEC 0C9 CD 00 SUB ESP, 0C
00402EFA 68 66104000 PUSH QJMP.&MSUBUH50_<vbaExceptHandler>
00402EFA 64:A1 00000000 MOU EAX, DWORD PTR FS:[0]
00402EFA 50 PUSH EAX
00402EFA 64:8925 00000000 MOU ECX, DWORD PTR FS:[0], ESP
00402EFA 81EC 98000000 SUB ESP, 98
00402EFA 8B45 08 MOU EAX, DWORD PTR SS:[EBP+8]
00402EFA 8B65 08 FE AND DWORD PTR SS:[EBP+8], FFFFFFFE
00402EFA 8B00 01 AND EAX, 1
00402EFA C745 F8 18104000 MOU DWORD PTR SS:[EBP-8], Tut_Reve.0040101
00402EFA 50 PUSH EAX
00402EFA 8945:FC6 65 72 50 72 6F MOU DWORD PTR SS:[EBP-4], EAX
00402EFA 8B45:081 BB MOU EAX, DWORD PTR SS:[EBP+8]
00402EFA 50 39 39 32 35 45 90 09 PUSH EAX, [EAX+1]
00402EFA 50 PUSH EAX
00402EFA 8B00 MOU ECX, DWORD PTR DS:[EAX]
00402EFA 8B65 F4 MOU DWORD PTR SS:[EBP-1], ESP
00402EFA 50 PUSH EAX
```

Run

실행

Aha, the nag isn't shown ...

아하, nag 은 더 이상 보이지 않는다 ...

Let's test the changes !

바뀐 것을 test 하자.

No more nag screen !!!

더 이상 nag screen 이 없다 !!!

And also startup nag screen is gone. I've tested that by restarting.

또한 startup nag screen 은 없어졌다. Restarting 으로 test 했다.

However, I removed it from this movie to reduce the size because I want to show you

그러나, 나는 movie 의 size 를 줄이기 위해 삭제했다. 너에게 보여주고 싶다.

5. CrackersConvert

I said it already : I am assuming you have done some research before and I won't explore the behaviour of this program with you.

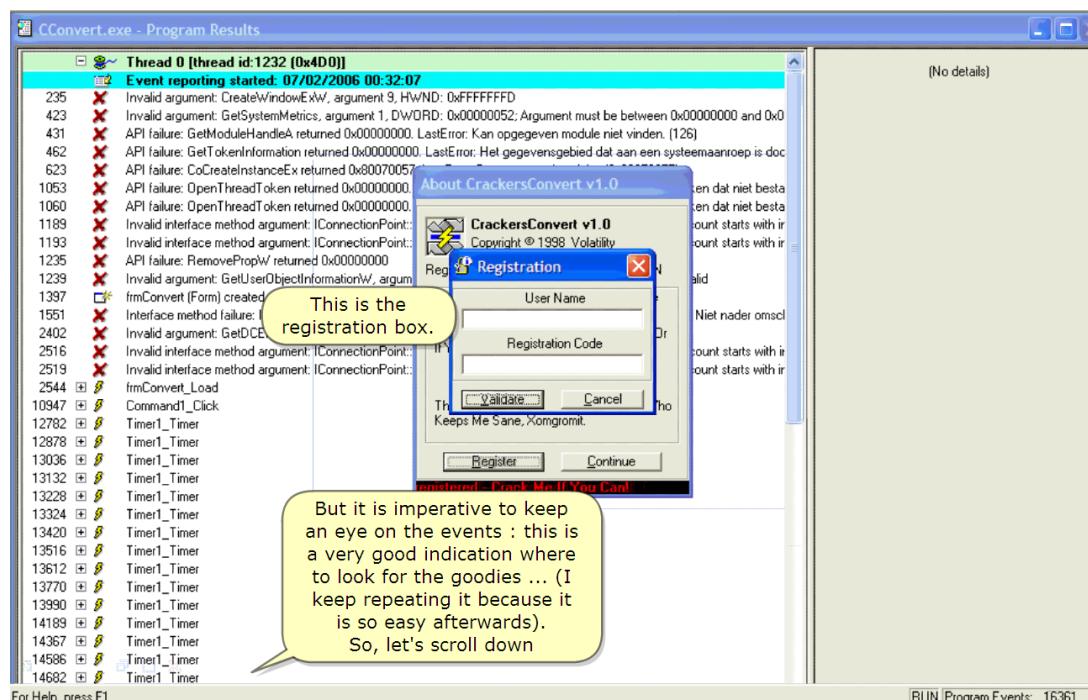
나는 이미 말했어 : 너는 이전에 약간의 연구를 끝냈다고 생각했다. 그리고 나는 너와 함께 이 program 의 behaviour 를 탐험하지 않았다.

I explored on my own and noticed that there is a possibility to register the program from the "About" box.

나는 나의 방법으로 탐험했다. 그리고 "About" box 로부터 program 을 등록하기 위한 가능성이 있다.

As you can see, I have already loaded the program in SmartCheck. So, let's go find the "About" box ...

네가 볼 때, 나는 이미 SmartCheck 에 program 을 load 해놨다. "About" box 를 찾자.



This is the registration box.

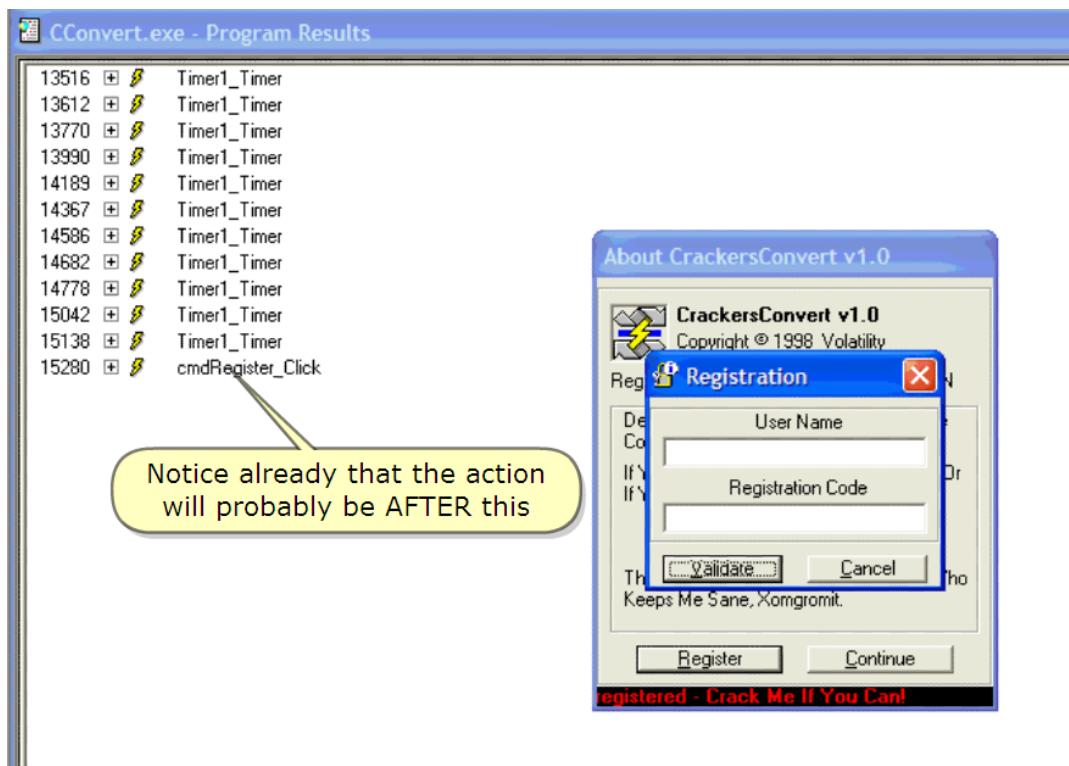
이것은 등록 box 다.

But it is imperative to keep an eye on the events : this is a very good indication where to look for the goodies ...(I keep repeating it because it is so easy afterwards).

이것은 events 를 보는데 긴요하다. : 이것은 매우 좋은 goodies 를 보기 위한 지시자다.(나는 이것을 반복한다. 왜냐하면 나중에 쉽기 때문이다.)

So, let's scroll down

이제 scroll 을 내려.



Notice already that the action will probably be AFTER this

Action 은 이전에 아마 있을 것이라고 알려준다.

INFO :

You can use whatever regcode you want of course. Perhaps you wonder why I'm using 47806 again ???

너는 regcode 를 네가 원할 때 사용할 수 있다. 아마 너는 내가 사용하고 있는 47806 을 다시 원할 것이다.

Well, sometimes, the regcode is transformed into hex before it is compared.

좋아, 언젠가 이것이 비교된 후에 regcode 는 hex 로 변환됐다.

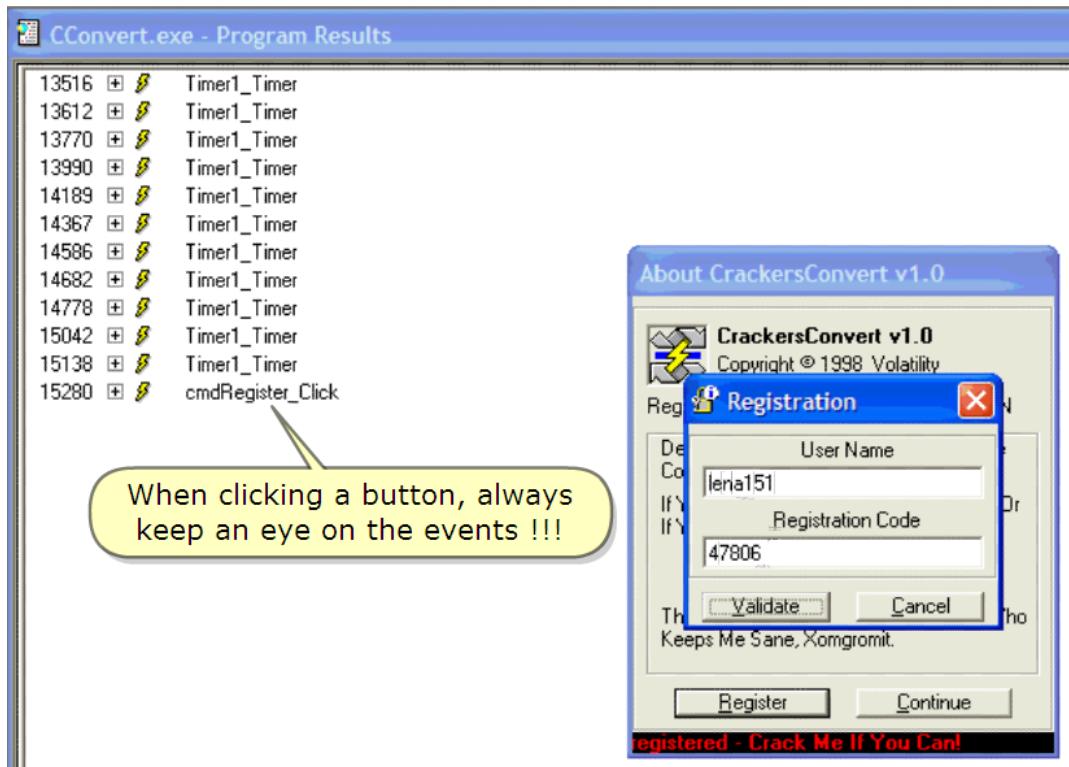
This makes it easy to follow for me, because believe me, lena151 recognizes another BABE from miles away !!!

이것을 따라가 게 만드는 것이 쉽다. 왜냐하면 나를 믿어. Lena151 는 miles away 에서 다른 BABE 를 알아봤다 !!!

47806 in dec == BABE in hex

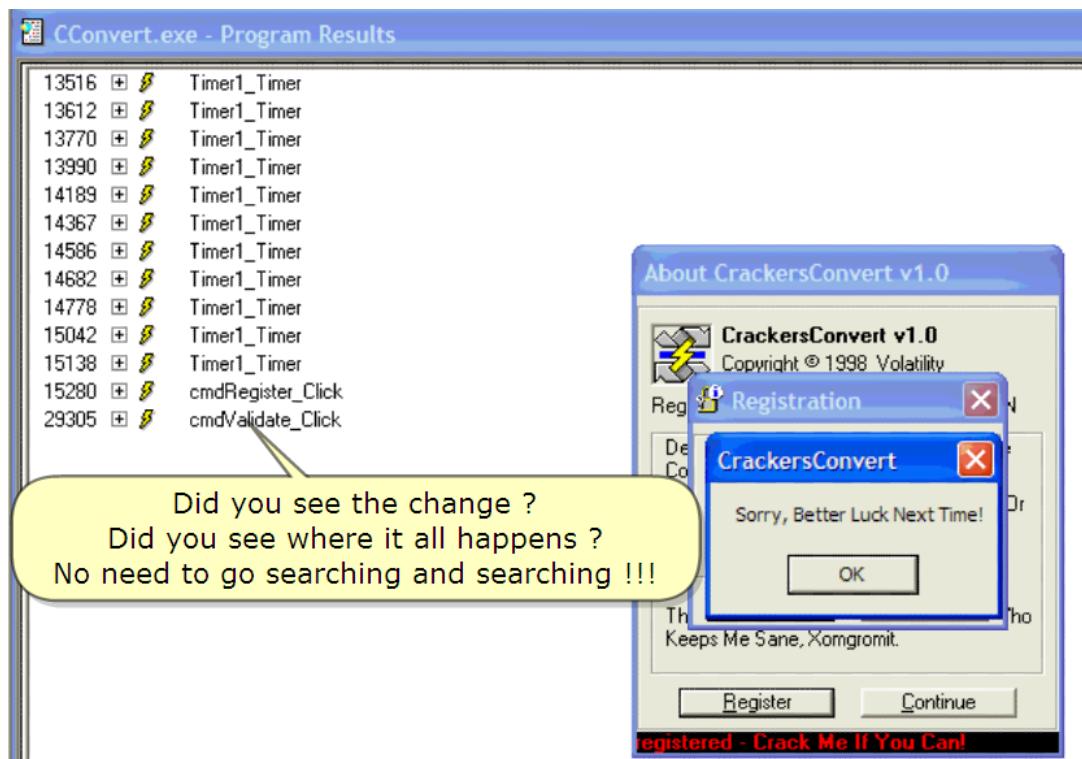
47086 은 하나씩 줄었다. == hex 에서 BABE

번역 주)47806 은 10 강 앞에서 사용했던 Registration code 다.



When clicking a button, always keep an eye on the events !!!

Button 을 click 할 때, 항상 events 를 지켜 봐 !!!



Did you see the change?

Did you see where it all happens?

No need to go searching and searching !!!

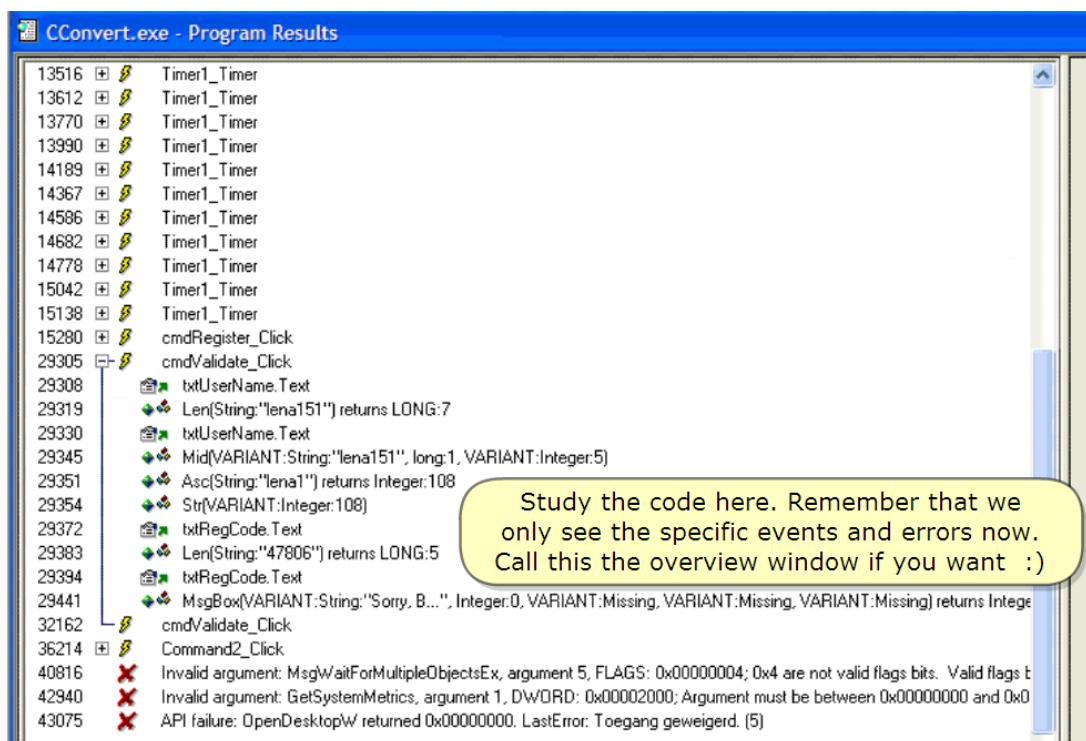
바뀐 것을 봤어?

어떤 일이 일어나는지 봤어?

더 이상 찾지 않아도 된다.

We know where to look, so, we can close the program!

우리는 어디를 봐야 되는지 안다, 그래서, 우리는 program 을 닫을 수 있다!



The screenshot shows the 'Program Results' window for the CConvert.exe application. It lists numerous events and errors. Most events are related to 'Timer1_Timer' and involve operations like Len, Mid, Asc, Str, and MsgBox. Several errors are listed at the bottom, such as invalid arguments for MsgWaitForMultipleObjectsEx, GetSystemMetrics, and OpenDesktopW. A yellow callout bubble highlights the error messages, with the text: 'Study the code here. Remember that we only see the specific events and errors now. Call this the overview window if you want :)'.

Event ID	Event Description
13516	Timer1_Timer
13612	Timer1_Timer
13770	Timer1_Timer
13990	Timer1_Timer
14189	Timer1_Timer
14367	Timer1_Timer
14586	Timer1_Timer
14682	Timer1_Timer
14778	Timer1_Timer
15042	Timer1_Timer
15138	Timer1_Timer
15280	cmdRegister_Click
29305	cmdValidate_Click
29308	txtUserName.Text
29319	Len(String:"lena151") returns LONG:7
29330	txtUserName.Text
29345	Mid(VARIANT:String:"lena151", long:1, VARIANT:Integer:5)
29351	Asc(String:"lena1") returns Integer:108
29354	Str(VARIANT:Integer:108)
29372	txtRegCode.Text
29383	Len(String:"47906") returns LONG:5
29394	txtRegCode.Text
29441	MsgBox(VARIANT:String:"Sorry, B...", Integer:0, VARIANT:Missing, VARIANT:Missing, VARIANT:Missing) returns Integer
32162	cmdValidate_Click
36214	Command2_Click
40816	Invalid argument: MsgWaitForMultipleObjectsEx, argument 5, FLAGS: 0x00000004; 0x4 are not valid flags bits. Valid flags t
42940	Invalid argument: GetSystemMetrics, argument 1, DWORD: 0x00002000; Argument must be between 0x00000000 and 0x0
43075	API failure: OpenDesktopW returned 0x00000000. LastError: Toegang geweigerd. (5)

Study the code here. Remember that we only see the specific events and errors now.

Call this the overview windows if you want :)

여기 code 를 공부하자. 기억해. 우리는 오직 명확한 events 와 error 를 봐야 한다.

네가 원할 때 Overview window 를 부르자. :)

INFO :

The code from the next screens can also be found on the included html page

다음 screens 로부터 포함된 html page 에서 code 를 찾았다.

CConvert.exe - Program Results

```

13516 + ⚡ Timer1_Timer
13612 + ⚡ Timer1_Timer
13770 + ⚡ Timer1_Timer
13990 + ⚡ Timer1_Timer
14189 + ⚡ Timer1_Timer
14367 + ⚡ Timer1_Timer
14586 + ⚡ Timer1_Timer
14682 + ⚡ Timer1_Timer
14778 + ⚡ Timer1_Timer
15042 + ⚡ Timer1_Timer
15138 + ⚡ Timer1_Timer
15280 + ⚡ cmdRegister_Click
29305 - ⚡ cmdValidate_Click
29308   ↴ txtUserName.Text
29319     ↴ Len(String:"lena151") returns LONG:7
29330   ↴ txtUserName.Text
29345     ↴ Mid(VARIANT:String:"lena151", long:1, VARIANT:Integer:5)
29351     ↴ Asc(String:"lena1") returns Integer:108
29354     ↴ Str(VARIANT:Integer:108)
29372   ↴ txtRegCode.Text
29383     ↴ Len(String:"47806") returns LONG:5
29394   ↴ txtRegCode.Text
29441     ↴ MsgBox(VARIANT:String:"Sorry, B...", Integer:0, VARIANT:Missing, VARIANT:Missing, VARIANT:Missing) returns Integer
32162   ⚡ cmdValidate_Click
36214 + ⚡ Command2_Click
40816 ✗ Invalid argument: MsgWaitForMultipleObjectsEx, argument 5, FLAGS: 0x00000004; 0x4 are not valid flags bits. Valid flags b
42940 ✗ Invalid argument: GetSystemMetrics, argument 1, DWORD: 0x00002000; Argument must be between 0x00000000 and 0x0
43075 ✗ API failure: OpenDesktopW returned 0x00000000. LastError: Toegang geweigerd. (5)

```

In general :
Len(String:"lena151") returns LONG:7

Explanation:
Get length of String "lena151" which is 7

**Our name is
 7 chars long**

In general :

개념 :

Len(String:"lena151") returns LONG:7

Return 값이 LONG:7

Explanation :

설명 :

Get length of String "lena151" which is 7

"lena151"의 String length 는 7 이다.

Our name is 7 chars long

우리의 name 은 7 글자다.

CConvert.exe - Program Results

```

13516 + ⚡ Timer1_Timer
13612 + ⚡ Timer1_Timer
13770 + ⚡ Timer1_Timer
13990 + ⚡ Timer1_Timer
14189 + ⚡ Timer1_Timer
14367 + ⚡ Timer1_Timer
14586 + ⚡ Timer1_Timer
14682 + ⚡ Timer1_Timer
14778 + ⚡ Timer1_Timer
15042 + ⚡ Timer1_Timer
15138 + ⚡ Timer1_Timer
15280 + ⚡ cmdRegister_Click
29305 - ⚡ cmdValidate_Click
29308   ⚡ txtUserName.Text
        ⚡ Len(String:"lena151") returns LONG:7
29319   ⚡ txtUserName.Text
        ⚡ Mid(VARIANT:String:"lena151", long:1, VARIANT:Integer:5)
29330   ⚡ Asc(String:"lena1") returns Integer:108
29345   ⚡ Str(VARIANT:Integer:108)
29354   ⚡ txtRegCode.Text
29372   ⚡ Len(String:"47806") returns LONG:5
29383   ⚡ txtRegCode.Text
29394   ⚡ MsgBox(VARIANT:String:"Sorry, B...", Integer:0, VARIANT:Missing, VARIANT:Missing, VARIANT:Missing) returns Integer
32162 + ⚡ cmdValidate_Click
36214 + ⚡ Command2_Click
40816 ✗ Invalid argument: MsgWaitForMultipleObjectsEx, argument 5, FLAGS: 0x00000004; 0x4 are not valid flags bits. Valid flags t
42940 ✗ Invalid argument: GetSystemMetrics, argument 1, DWORD: 0x00002000; Argument must be between 0x00000000 and 0x0
43075 ✗ API failure: OpenDesktopW returned 0x00000000. LastError: Toegang geweigerd. (5)

```

In general :
`Mid(VARIANT: String:"abcdefg", long:1, VARIANT:Integer:1)`

Explanation:
Get the 1st character in the string "abcdefg" starting from location 1.

But this means to pick 5 chars starting from position 1

In general :

개념

`Mid(VARIANT: String:"abcdefg", long:1, VARIANT:Integer:1)`

Explanation:

설명 :

Get the 1st character in the string "abcdefg" starting from location 1.

첫번째 1 글자가 "abcdefg"에서 시작하는 주소.

But this means to pick 5 chars starting from position 1

그러나 여기에서는 5 번째 글자부터 1로 시작한다.

Dec	Hex	ASCII		Dec	Hex	ASCII		Dec	Hex	ASCII		Dec	Hex	ASCII
0	00	NUL		32	20	SP		64	40	@		96	60	'
1	01	SOH		33	21	!		65	41	A		97	61	a
2	02	STX		34	22	"		66	42	B		98	62	b
3	03	ETX		35	23	#		67	43	C		99	63	c
4	04	EOT		36	24	\$		68	44	D		100	64	d
5	05	ENQ		37	25	%		69	45	E		101	65	e
6	06	ACK		38	26	&		70	46	F		102	66	f
7	07	BEL		39	27	'		71	47	G		103	67	g
8	08	BS		40	28	(72	48	H		104	68	h
9	09	HT		41	29)		73	49	I		105	69	i
10	0A	LF		42	2A	*		74	4A	J		106	6A	j
11	0B	VT		43	2B	+		75	4B	K		107	6B	k
12	0C	FF		44	2C	,		76	4C	L	108	6C	l	
13	0D	CR		45	2D	-		77	4D	M	109	6D	m	
14	0E	SO		46	2E	.		78	4E	N	110	6E	n	
15	0F	SI		47	2F	/		79	4F	O	111	6F	o	
16	10	DLE		48	30	0		80	50	P	112	70	p	
17	11	DC1		49	31	1		81	51	Q	113	71	q	
18	12	DC2		50	32	2		82	52	R	114	72	r	
19	13	DC3		51	33	3		83	53	S	115	73	s	
20	14	DC4		52	34	4	84	54	T	116	74	t		
21	15	NAK		53	35	5	85	55	U	117	75	u		
22	16	SYN		54	36	6	86	56	V	118	76	v		
23	17	ETB		55	37	7	87	57	W	119	77	w		
24	18	CAN		56	38	8	88	58	X	120	78	x		
25	19	EM		57	39	9	89	59	Y	121	79	y		
26	1A	SUB		58	3A	:	90	5A	Z	122	7A	z		
27	1B	ESC		59	3B	;	91	5B	[123	7B	{		
28	1C	FS		60	3C	<	92	5C	\	124	7C			
29	1D	GS		61	3D	=	93	5D]	125	7D	}		
30	1E	RS		62	3E	>	94	5E	^	126	7E	~		
31	1F	US		63	3F	?	95	5F	_	127	7F	DEL		

CConvert.exe - Program Results

```

13516 + ⚡ Timer1_Timer
13612 + ⚡ Timer1_Timer
13770 + ⚡ Timer1_Timer
13990 + ⚡ Timer1_Timer
14189 + ⚡ Timer1_Timer
14367 + ⚡ Timer1_Timer
14586 + ⚡ Timer1_Timer
14682 + ⚡ Timer1_Timer
14778 + ⚡ Timer1_Timer
15042 + ⚡ Timer1_Timer
15138 + ⚡ Timer1_Timer
15280 + ⚡ cmdRegister_Click
29305 - ⚡ cmdValidate_Click
29308 - ⚡ txtUserName.Text

Ascii for letter "I" is 108 !!!
  
```

29345 Mid(VARIANT:String;"lena1", long:1, VARIANT:Integer:5)
 29351 Asc(String:"lena1") returns Integer:108
 29354 Str(VARIANT:String;long:108)
 29372 txtRegCode.Text
 29383 Len(String:"47806") returns LONG:6
 29394 txtRegCode.Text
 29441 MsgBox(VARIANT:String;"Sorry")
 32162 cmdValidate_Click
 36214 + ⚡ Command2_Click
 40816 ✗ Invalid argument: MsgWaitForMultipleObjects(0, ..., dwMilliseconds:DWORD, ..., Valid flags to be set: 0x00000000)
 42940 ✗ Invalid argument: GetSystemMetrics, argument 1, DWORD: 0x00002000; Argument must be between 0x00000000 and 0x00000000
 43075 ✗ API failure: OpenDesktopW returned 0x00000000. LastError: Toegang geweigerd. (5)

In general :
 Asc(String:"T") returns Integer:84

Explanation:
 Get the decimal value of T which is 84

In general :

Asc(String:"T") returns Integer:84

Explanation:

Get the decimal value of T which is 84

ASCII 의 T decimal 값인 84 를 얻어온다.

Ascii for letter "I" is 108 !!!

즉, lena1 의 I 값의 ascii 코드값이 108 이다.

CConvert.exe - Program Results

```

13516 + ⚡ Timer1_Timer
13612 + ⚡ Timer1_Timer
13770 + ⚡ Timer1_Timer
13990 + ⚡ Timer1_Timer
14189 + ⚡ Timer1_Timer
14367 + ⚡ Timer1_Timer
14586 + ⚡ Timer1_Timer
14682 + ⚡ Timer1_Timer
14778 + ⚡ Timer1_Timer
15042 + ⚡ Timer1_Timer
15138 + ⚡ Timer1_Timer
15280 + ⚡ cmdRegister_Click
29305 - ⚡ cmdValidate_Click
  ↳ txtUserName.Text
  ↳ Len(String:"lena151") returns LONG:5
  ↳ Mid(VARIANT:String:"lena151",
        Asc(String:"lena1") returns Integer:108
        Str(VARIANT:Integer:108)
  ↳ txtRegCode.Text
  ↳ Len(String:"47806") returns LONG:5
  ↳ txtRegCode.Text
  ↳ MsgBox(VARIANT:String:"Sorry, B...", Integer:0, VARIANT:Missing, VARIANT:Missing, VARIANT:Missing) returns Integer:0
32162 - ⚡ cmdValidate_Click
36214 + ⚡ Command2_Click
40816 ✘ Invalid argument: MsgWaitForMultipleObjectsEx, argument 5, FLAGS: 0x00000004; 0x4 are not valid flags bits. Valid flags bits are 0x00000001 to 0x00000004
42940 ✘ Invalid argument: GetSystemMetrics, argument 1, DWord: 0x00002000; Argument must be between 0x00000000 and 0x00000004
43075 ✘ API failure: OpenDesktopW returned 0x00000000. LastError: Toegang geweigerd. (5)

```

Len(String:"47806") returns LONG:5

Explanation:
Get length of String "47806" which is 5

The serial is 5 chars long

Len(String:"47806") returns LONG:5

Explanation:

Get length of String "47806" which is 5

There serial is 5 chars long

"47806"의 length 인 5 를 얻어온다.

Serial 은 5 글자다.

CConvert.exe - Program Results

```

13516 + ⚡ Timer1_Timer
13612 + ⚡ Timer1_Timer
13770 + ⚡ Timer1_Timer
13990 + ⚡ Timer1_Timer
14189 + ⚡ Timer1_Timer
14367 + ⚡ Timer1_Timer
14586 + ⚡ Timer1_Timer
14682 + ⚡ Timer1_Timer
14778 + ⚡ Timer1_Timer
15042 + ⚡ Timer1_Timer
15138 + ⚡ Timer1_Timer
15280 + ⚡ cmdRegister_Click
29305 - ⚡ cmdValidate_Click
  ↳ txtUserName.Text
  ↳ Len(String:"lena151") returns LONG:7
  ↳ txtUserName.Text
  ↳ Mid(VARIANT:String:"lena151", long:1, VARIANT:Integer:5)
  ↳ Asc(String:"lena1") returns Integer:108
  ↳ Str(VARIANT:Integer:108)
  ↳ txtRegCode.Text
  ↳ Len(String:"47806") returns LONG:5
  ↳ txtRegCode.Text
  ↳ MsgBox(VARIANT:String:"Sorry, B...", Integer:0, VARIANT:Missing, VARIANT:Missing, VARIANT:Missing) returns Integer:0
32162 - ⚡ cmdValidate_Click
36214 + ⚡ Command2_Click
40816 ✘ Invalid argument: MsgWaitForMultipleObjectsEx, argument 5, FLAGS: 0x00000004; 0x4 are not valid flags bits. Valid flags bits are 0x00000001 to 0x00000004
42940 ✘ Invalid argument: GetSystemMetrics, argument 1, DWord: 0x00002000; Argument must be between 0x00000000 and 0x00000004
43075 ✘ API failure: OpenDesktopW returned 0x00000000. LastError: Toegang geweigerd. (5)

```

And here is the Badboy already !

CCONVERT.EXE!00000A13D (no debug info)

- prompt (variant)
 - String .bstrVal = 00157484
 - = "Sorry, Better Luck Next Time!"
 - Long .buttons = 0x00000000
- title (variant)
 - Long .scode = -2147352572 0x80020004
- helpfile (variant)
 - Long .scode = -2147352572 0x80020004
- context (variant)
 - Long .scode = -2147352572 0x80020004

And here is the Badboy already!

여기 이미 Badboy 가 있다!

13516 + Timer1_Timer
13612 + Timer1_Timer
13770 + Timer1_Timer
13990 + Timer1_Timer
14189 + Timer1_Timer
14367 + Timer1_Timer
14586 + Timer1_Timer
14682 + Timer1_Timer
14778 + Timer1_Timer
15042 + Timer1_Timer
15138 + Timer1_Timer
15280 + cmdRegister_Click
29305 - cmdValidate_Click
29308 + txUserName.Text
29319 + Len(String: "lena151") returns LONG: 7
29330 + txUserName.Text
29345 + Mid(VARIANT: String: "lena151", long, VARIANT: Integer: 5)
29351 + Asc(String: "len", 1) returns Integer: 108
29354 + Str(VARIANT: Integer: 108)
29372 + txRegCode.Text
29383 + Len(String: "47806") returns LONG: 5
29394 + txRegCode.Text
29441 + MsgBox(VARIANT: String: "Sorry, B...", Integer: 0, VARIANT: Missing, VARIANT: Missing, VARIANT: Missing) returns Integer
32162 + cmdValidate_Click
36214 + Command2_Click
40816 X Invalid argument: MsgWaitForMultipleObjectsEx, argument 5, FLAGS: 0x00000004; 0x4 are not valid flags bits. Valid flags t
42940 X Invalid argument: GetSystemMetrics, argument 1, DWORD: 0x00002000; Argument must be between 0x00000000 and 0x0
43075 X API failure: OpenDesktopW returned 0x00000000. LastError: Toegang geweigerd. (5)

And here, the program is only verifying the length of the serial.

여기, program 은 serial 의 length 만 검사한다.

Now, do you understand that between these two, there must be a compare serial <-----> name ???

이제, 2 가지 값에서, serial 과 name 은 반드시 비교되는 것을 이해했어?

13516 + Timer1_Timer
13612 + Timer1_Timer
13770 + Timer1_Timer
13990 + Timer1_Timer
14189 + Timer1_Timer
14367 + Timer1_Timer
14586 + Timer1_Timer
14682 + Timer1_Timer
14778 + Timer1_Timer
15042 + Timer1_Timer
29330 + txUserName.Text
29345 + Mid(VARIANT: String: "lena151", long, VARIANT: Integer: 5)
29351 + Asc(String: "len", 1) returns Integer: 108
29354 + Str(VARIANT: Integer: 108)
29372 + txRegCode.Text
29383 + Len(String: "47806") returns LONG: 5
29394 + txRegCode.Text
29441 + MsgBox(VARIANT: String: "Sorry, B...", Integer: 0, VARIANT: Missing, VARIANT: Missing, VARIANT: Missing) returns Integer
32162 + cmdValidate_Click
36214 + Command2_Click
40816 X Invalid argument: MsgWaitForMultipleObjectsEx, argument 5, FLAGS: 0x00000004; 0x4 are not valid flags bits. Valid flags t
42940 X Invalid argument: GetSystemMetrics, argument 1, DWORD: 0x00002000; Argument must be between 0x00000000 and 0x0
43075 X API failure: OpenDesktopW returned 0x00000000. LastError: Toegang geweigerd. (5)

This means, that with this line highlighted

이 뜻은, 이 line 은 highlight 됐다.

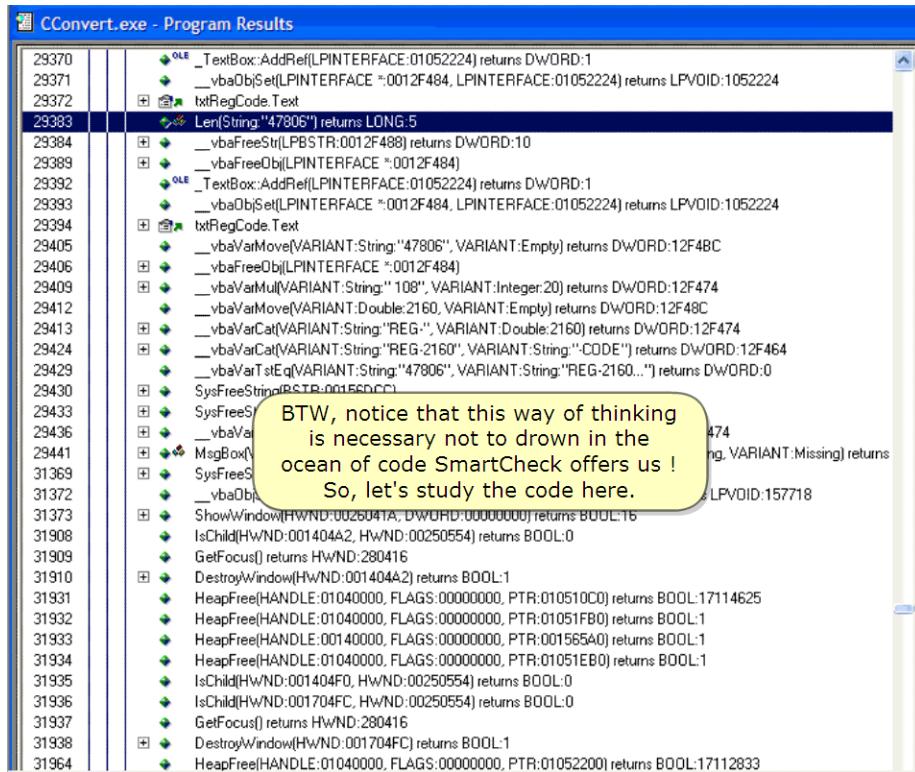
We can click "Show all events"

우리는 "Show all events"를 click 할 수 있다.

... we will find the compare after the highlighted line and before the badboy !!!

Make sure to understand this !!!

... 우리는 badboy 가 나타난 후에 highlight 된 line 에서 비교하는 것을 찾을 수 있다.
이것을 이해해야 한다.



The screenshot shows the SmartCheck results for the CConvert.exe program. It displays a list of API calls with their parameters and return values. A yellow callout box highlights the line: '_vbaVarMove(VARIANT:String:'47806', VARIANT:Empty) returns DWORD:12F4BC'. This line is associated with the variable 'txtRegCode.Text'. The callout box contains the text: "BTW, notice that this way of thinking is necessary not to drown in the ocean of code SmartCheck offers us ! So, let's study the code here."

Line Number	API Call	Parameters	Return Value
29370	_TextBox::AddRef(LPINTERFACE:01052224)	returns DWORD:1	
29371	_vbaObjSet(LPINTERFACE :0012F484, LPINTERFACE:01052224)	returns LPVOID:1052224	
29372	[+] txtRegCode.Text		
29383	[+] Len(String:"47806") returns LONG:5		
29384	[+] _vbaFreeStr(LPBSTR:0012F488)	returns DWORD:10	
29389	[+] _vbaFreeObj(LPINTERFACE :0012F484)		
29392	[+] OLE _TextBox::AddRef(LPINTERFACE:01052224)	returns DWORD:1	
29393	[+] _vbaObjSet(LPINTERFACE :0012F484, LPINTERFACE:01052224)	returns LPVOID:1052224	
29394	[+] txtRegCode.Text		
29405	[+] _vbaVarMove(VARIANT:String:'47806', VARIANT:Empty)	returns DWORD:12F4BC	
29406	[+] _vbaFreeObj(LPINTERFACE :0012F484)		
29409	[+] _vbaVarMul(VARIANT:String:'108', VARIANT:Integer:20)	returns DWORD:12F474	
29412	[+] _vbaVarMove(VARIANT:Double:2160, VARIANT:Empty)	returns DWORD:12F48C	
29413	[+] _vbaVarCat(VARIANT:String:'REG...', VARIANT:Double:2160)	returns DWORD:12F474	
29424	[+] _vbaVarCat(VARIANT:String:'REG-2160', VARIANT:String:'CODE')	returns DWORD:12F464	
29429	[+] _vbaVarIstEq(VARIANT:String:'47806', VARIANT:String:'REG-2160...')	returns DWORD:0	
29430	[+] SysFreeString(LPCSTR:0015CDCC)		
29433	[+] SysFreeS		
29436	[+] _vbaVa		
29441	[+] MsgBoxV		
31369	[+] SysFreeS		
31372	[+] _vbaOb		
31373	[+] ShowWindow(HWND:0026041A, DWORD:00000000)	returns BOOL:1	
31908	[+] IsChild(HWND:001404A2, HWND:00250554)	returns BOOL:0	
31909	[+] GetFocus()	returns HWND:280416	
31910	[+] DestroyWindow(HWND:001404A2)	returns BOOL:1	
31931	[+] HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:010510C0)	returns BOOL:17114625	
31932	[+] HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01051FB0)	returns BOOL:1	
31933	[+] HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:001565A0)	returns BOOL:1	
31934	[+] HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01051EB0)	returns BOOL:1	
31935	[+] IsChild(HWND:001404F0, HWND:00250554)	returns BOOL:0	
31936	[+] IsChild(HWND:001704FC, HWND:00250554)	returns BOOL:0	
31937	[+] GetFocus()	returns HWND:280416	
31938	[+] DestroyWindow(HWND:001704FC)	returns BOOL:1	
31964	[+] HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01052200)	returns BOOL:17112833	

BTW, notice that this way of thinking is necessary not to drown in the ocean of code SmartCheck offers us!

By the way, 하나의 방법은 필요하다. 많은 양의 code 에서 익사하지 않게 SmartCheck 는 우리에게 제공한다.

So, let's study the code here.

여기 code 를 공부하자.

CConvert.exe - Program Results

```

29370 |     [OLE] _TextBox:AddRef(LPINTERFACE:01052224) returns DWORD:1
29371 |     [OLE] __vbaObjSet(LPINTERFACE:0012F484, LPINTERFACE:01052224) returns LPVOID:1052224
29372 |     [+] txRegCode.Text
29383 |     [+] Len(String: "47806") returns LONG:5
29384 |     [+] [+] __vbaFreeStr(LPBSTR:0012F488) returns DWORD:10
29389 |     [+] [+] __vbaFreeObj(LPINTERFACE:0012F484)
29392 |     [+] [+] [OLE] _TextBox:AddRef(LPINTERFACE:01052224) returns DWORD:1
29393 |     [+] [+] __vbaObjSet(LPINTERFACE:0012F484, LPINTERFACE:01052224) returns LPVOID:1052224
29394 |     [+] txRegCode.Text
29405 |     [+] [+] __vbaVarMove(VARIANT:String:"47806", VARIANT:Empty) returns DWORD:12F48C
29406 |     [+] [+] __vbaFreeObj(LPINTERFACE:0012F484)
29409 |     [+] [+] __vbaVarMul(VARIANT:String:"108", VARIANT:Integer:20) returns DWORD:12F474
29412 |     [+] [+] __vbaVarMove(VARIANT:Double:2160, VARIANT:Empty) returns DWORD:12F48C
29413 |     [+] [+] __vbaVarCat(VARIANT:String:"REG-", VARIANT:Double:2160) returns DWORD:12F474
29424 |     [+] [+] __vbaVarCat(VARIANT:String:"REG-2160", VARIANT:Double:2160) returns DWORD:12F48A
29429 |     [+] [+] __vbaVarTstEq(VARIANT:String:"47806", VARIANT:Empty)
29430 |     [+] SysFreeString(BSTR:00156DCC)
29433 |     [+] SysFreeString(BSTR:00156DF4)
29436 |     [+] __vbaVarDup(VARIANT:String:"Sorry, B...", VARIANT:Empty)
29441 |     [+] [+] MsgBox(VARIANT:String:"Sorry, B...", Integer:0, VA
31369 |     [+] SysFreeString(BSTR:00157484)
31372 |     [+] [+] __vbaObjSetAddref(LPINTERFACE:0012F484, LPINTERFACE:00157718) returns LPVOID:157718
31373 |     [+] ShowWindow(HWND:0026041A, DWORD:00000000) returns BOOL:1
31908 |     [+] IsChild(HWND:001404A2, HWND:00250554) returns BOOL:0
31909 |     [+] GetFocus() returns HWND:280416
31910 |     [+] DestroyWindow(HWND:001404A2) returns BOOL:1
31931 |     [+] HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:010510C0) returns BOOL:17114625
31932 |     [+] HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01051FB0) returns BOOL:1
31933 |     [+] HeapFree(HANDLE:00140400, FLAGS:00000000, PTR:001565A0) returns BOOL:1
31934 |     [+] HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01051EB0) returns BOOL:1
31935 |     [+] IsChild(HWND:001404F0, HWND:00250554) returns BOOL:0
31936 |     [+] IsChild(HWND:001704FC, HWND:00250554) returns BOOL:0
31937 |     [+] GetFocus() returns HWND:280416
31938 |     [+] DestroyWindow(HWND:001704FC) returns BOOL:1
31964 |     [+] HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01052200) returns BOOL:17112833

```

In fact, it is not difficult at all. We almost immediately see that the first char from our name (value in ascii is 108, see before) is multiplied by 20

In fact, it is not difficult at all. We almost immediately see that the first char from our name(value in ascii is 108, see before) is multiplied by 20

사실, 이것은 어렵지 않다. 우리는 바로 우리의 이름에서 첫번째 문자에 20 곱한 것을 참조 하십시오.

In general :

`_vbaVarMul(VARIANT:String:"1", VARIANT:String:"2") returns ...`

Explanation : Multiply 1 by 2

설명 : 1 에 2 를 곱한다.

CConvert.exe - Program Results

```

29370   + OLE _TextBox:AddRef(LPINTERFACE:01052224) returns DWORD:1
29371   + OLE __vbaObjSet(LPINTERFACE :0012F484, LPINTERFACE:01052224) returns LPVOID:1052224
29372   + txRegCode.Text
29383   + LenString("47806") returns LONG:5
29384   + __vbaFreeStr(LPCTSTR:0012F488) returns DWORD:10
29389   + __vbaFreeObj(LPINTERFACE :0012F484)
29392   + OLE _TextBox:AddRef(LPINTERFACE:01052224) returns DWORD:1
29393   + OLE __vbaObjSet(LPINTERFACE :0012F484, LPINTERFACE:01052224) returns LPVOID:1052224
29394   + txRegCode.Text
29405   + The result is 2160
29406   + __vbaVarMul(VARIANT:String:"108", VARIANT:Integer:20) returns DWORD:12F474
29409   + __vbaVarMove(VARIANT:Double:2160, VARIANT:Empty) returns DWORD:12F48C
29412   + __vbaVarCat(VARIANT:String:"REG-", VARIANT:Double:2160) returns DWORD:12F474
29413   + __vbaVarCat(VARIANT:String:"REG-2160", VARIANT:String:"CODE") returns DWORD:12F464
29424   + __vbaVarCat(VARIANT:String:"47806", VARIANT:String:"REG-2160...") returns DWORD:0
29429   + SysFreeString(BSTR:00156DCC)
29430   + SysFreeString(BSTR:00156DF4)
29433   + __vbaVarUp(VARIANT:String:"Sorry, B...", VARIANT:Empty) returns DWORD:12F474
29441   + MsgBox(VARIANT:String:"Sorry, B...", Integer:0, VARIANT:Missing, VARIANT:Missing, VARIANT:Missing) returns
29444   + SysFreeString(BSTR:00157484)
31369   + __vbaObjSetAddref(LPINTERFACE :0012F484, LPINTERFACE:00157718) returns LPVOID:157718
31372   + ShowWindow(HWND:0026041A, DWORD:00000000) returns BOOL:16
31373   + IsChild(HWND:001404A2, HWND:00250554) returns BOOL:0
31908   + GetFocus() returns HWND:280416
31909   + DestroyWindow(HWND:001404A2) returns BOOL:1
31910   + HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:010510C0) returns BOOL:17114625
31931   + HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01051FB0) returns BOOL:1
31932   + HeapFree(HANDLE:00140000, FLAGS:00000000, PTR:001565A0) returns BOOL:1
31933   + HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01051EB0) returns BOOL:1
31934   + IsChild(HWND:001404F0, HWND:00250554) returns BOOL:0
31935   + IsChild(HWND:001704FC, HWND:00250554) returns BOOL:0
31936   + GetFocus() returns HWND:280416
31937   + DestroyWindow(HWND:001704FC) returns BOOL:1
31938   + HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01052200) returns BOOL:17112833
31964

```

The result is 2160

결과는 2160 이다.

CConvert.exe - Program Results

```

29370   + OLE _TextBox:AddRef(LPINTERFACE:01052224) returns DWORD:1
29371   + OLE __vbaObjSet(LPINTERFACE :0012F484, LPINTERFACE:01052224) returns LPVOID:1052224
29372   + txRegCode.Text
29383   + LenString("47806") returns LONG:5
29384   + __vbaFreeStr(LPCTSTR:0012F488) returns DWORD:10
29389   + __vbaFreeObj(LPINTERFACE :0012F484)
29392   + OLE _TextBox:AddRef(LPINTERFACE:01052224) returns DWORD:1
29393   + OLE __vbaObjSet(LPINTERFACE :0012F484, LPINTERFACE:01052224) returns LPVOID:1052224
29394   + txRegCode.Text
29405   + In general :
29406   + __vbaVarCat(VARIANT:String:"REG-", VARIANT:String:"2160") returns ....
29409   + __vbaVarMove(VARIANT:Double:2160, VARIANT:Empty) returns DWORD:12F48C
29412   + __vbaVarCat(VARIANT:String:"REG-", VARIANT:Double:2160) returns DWORD:12F474
29413   + __vbaVarCat(VARIANT:String:"REG-2160", VARIANT:String:"CODE") returns DWORD:12F464
29424   + __vbaVarCat(VARIANT:String:"47806", VARIANT:String:"REG-2160...") returns DWORD:0
29429   + SysFreeString(BSTR:00156DCC)
29430   + SysFreeString(BSTR:00156DF4)
29433   + __vbaVarUp(VARIANT:String:"Sorry, B...", VARIANT:Empty) returns DWORD:12F474
29441   + MsgBox(VARIANT:String:"Sorry, B...", Integer:0, VARIANT:Missing, VARIANT:Missing, VARIANT:Missing) returns
29444   + SysFreeString(BSTR:00157484)
31369   + __vbaObjSetAddref(LPINTERFACE :0012F484, LPINTERFACE:00157718) returns LPVOID:157718
31372   + ShowWindow(HWND:0026041A, DWORD:00000000) returns BOOL:16
31373   + IsChild(HWND:001404A2, HWND:00250554) returns BOOL:0
31908   + GetFocus() returns HWND:280416
31909   + DestroyWindow(HWND:001404A2) returns BOOL:1
31910   + HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:010510C0) returns BOOL:17114625
31931   + HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01051FB0) returns BOOL:1
31932   + HeapFree(HANDLE:00140000, FLAGS:00000000, PTR:001565A0) returns BOOL:1
31933   + HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01051EB0) returns BOOL:1
31934   + IsChild(HWND:001404F0, HWND:00250554) returns BOOL:0
31935   + IsChild(HWND:001704FC, HWND:00250554) returns BOOL:0
31936   + GetFocus() returns HWND:280416
31937   + DestroyWindow(HWND:001704FC) returns BOOL:1
31938   + HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01052200) returns BOOL:17112833
31964

```

In general :

`_vbaVarCat(VARIANT: String:"REG-", VARIANT:String:"2160") returns`

Explanation : join "2160" to "REG-" to form "REG-2160"

"2160"이 "REG-"에 가서 붙는다. "REG-2160"

The screenshot shows the assembly code for the variable `txtRegCode.Text`. A yellow callout box highlights the instruction `Len(String:"47806") returns LONG:5`, which corresponds to the string "REG-2160". Another yellow callout box highlights the instruction `vbaVarCat(VARIANT:String:"REG-", VARIANT:String:"CODE") returns DWORD:12F464`.

```
29370     _TextBox::AddRef(LPINTERFACE:01052224) returns DWORD:1
29371     _vbaObjSet(LPINTERFACE :0012F484, LPINTERFACE:01052224) returns LPVOID:1052224
29372     [+] txtRegCode.Text
29383     [+] OLE _Len(String:"47806") returns LONG:5
29384     [+] _vbaFreeStr(LPBSTR:0012F488) returns DWORD:10
29389     [+] _vbaFreeObj(LPINTERFACE :0012F484)
29392     [+] OLE _TextBox::AddRef(LPINTERFACE:01052224) returns DWORD:1
29393     [+] _vbaObjSet(LPINTERFACE :0012F484, LPINTERFACE:01052224) returns LPVOID:1052224
29394     [+] [+] txtRegCode.Text
29405     [+] _vbaVarMove(VARIANT:String:"REG-", VARIANT:Empty) returns DWORD:12F4BC
29406     [+] _vbaFreeObj(LPINTERFACE:01052224)
29409     [+] _vbaVarMul(VARIANT:String:"108", VARIANT:Integer:20) returns DWORD:12F474
29412     [+] _vbaVarMove(VARIANT:Double:2160, VARIANT:Empty) returns DWORD:12F48C
29413     [+] _vbaVarCat(VARIANT:String:"REG-", VARIANT:Double:2160) returns DWORD:12F474
29424     [+] _vbaVarCat(VARIANT:String:"REG-2160", VARIANT:String:"CODE") returns DWORD:12F464
29429     [+] _vbaVarTstEq(VARIANT:String:"47806", VARIANT:String:"REG-2160...") returns DWORD:0
29430     [+] SysFreeString(BSTR:00156DCC)
29433     [+] SysFreeString(BSTR:00156DF4)
29436     [+] _vbaVarDup(VARIANT:String:"Sorry, B...") returns DWORD:12F474
29441     [+] MsgBox(VARIANT:String:"Sorry, B...", VARIANT:Integer:0, VARIANT:Missing, VARIANT:Missing) returns
29444     [+] SysFreeString(BSTR:00157484)
31369     [+] _vbaObjSetAddref(LPINTERFACE :0012F484, LPINTERFACE:00157718) returns LPVOID:157718
31372     [+] ShowWindow(HWND:0026041A, DWORD:00000000) returns BOOL:16
31373     [+] IsChild(HWND:001404A2, HWND:00250554) returns BOOL:0
31908     [+] GetFocus() returns HWND:280416
31909     [+] DestroyWindow(HWND:001404A2) returns BOOL:1
31910     [+] HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:010510C0) returns BOOL:17114625
31932     [+] HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01051FB0) returns BOOL:1
31933     [+] HeapFree(HANDLE:00140000, FLAGS:00000000, PTR:001565A0) returns BOOL:1
31934     [+] HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01051EB0) returns BOOL:1
31935     [+] IsChild(HWND:001404F0, HWND:00250554) returns BOOL:0
31936     [+] IsChild(HWND:001704FC, HWND:00250554) returns BOOL:0
31937     [+] GetFocus() returns HWND:280416
31938     [+] DestroyWindow(HWND:001704FC) returns BOOL:1
31964     [+] HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01052200) returns BOOL:17112833
```

And so we form :
REG-2160

And here we form
REG-2160-CODE

And so we form : REG-2160

우리의 form : REG-2160

And here we form REG-2160-CODE

우리의 form REG-2160-CODE

The screenshot shows the assembly code for the variable `txtRegCode.Text`. A yellow callout box highlights the instruction `vbaVarCat(VARIANT:String:"REG-2160", VARIANT:String:"CODE")`. Another yellow callout box highlights the instruction `... which is then compared with our serial`.

```
29370     _TextBox::AddRef(LPINTERFACE:01052224) returns DWORD:1
29371     _vbaObjSet(LPINTERFACE :0012F484, LPINTERFACE:01052224) returns LPVOID:1052224
29372     [+] txtRegCode.Text
29383     [+] OLE _Len(String:"47806") returns LONG:5
29384     [+] _vbaFreeStr(LPBSTR:0012F488) returns DWORD:10
29389     [+] _vbaFreeObj(LPINTERFACE :0012F484)
29392     [+] OLE _TextBox::AddRef(LPINTERFACE:01052224) returns DWORD:1
29393     [+] _vbaObjSet(LPINTERFACE :0012F484, LPINTERFACE:01052224) returns LPVOID:1052224
29394     [+] [+] txtRegCode.Text
29405     [+] _vbaVarMove(VARIANT:String:"REG-", VARIANT:Empty) returns DWORD:12F4BC
29406     [+] _vbaFreeObj(LPINTERFACE:01052224)
29409     [+] _vbaVarMul(VARIANT:String:"108", VARIANT:Integer:20) returns DWORD:12F474
29412     [+] _vbaVarMove(VARIANT:Double:2160, VARIANT:Empty) returns DWORD:12F48C
29413     [+] _vbaVarCat(VARIANT:String:"REG-", VARIANT:Double:2160) returns DWORD:12F474
29424     [+] _vbaVarCat(VARIANT:String:"REG-2160", VARIANT:String:"CODE")
29425     [+] SysAllocStringByteLen(Char :00000000, DWORD:0000001A) returns LPVOID:156DF4
29428     [+] _vbaVarCat returns DWORD:12F464
29429     [+] _vbaVarTstEq(VARIANT:String:"47806", VARIANT:String:"REG-2160...") returns DWORD:0
29430     [+] SysFreeString(BSTR:00156DCC)
29433     [+] SysFreeString(BSTR:00156DF4)
29436     [+] ... which is then
29441     [+] M...y) returns DWORD:12F474
29444     [+] Missing, VARIANT:Missing, VARIANT:Missing) returns
31369     [+] Sys...
31372     [+] _vbaObjSetAddref(LPINTERFACE :0012F484, LPINTERFACE:00157718) returns LPVOID:157718
31373     [+] ShowWindow(HWND:0026041A, DWORD:00000000) returns BOOL:16
31374     [+] IsChild(HWND:001404A2, HWND:00250554) returns BOOL:0
31375     [+] GetFocus() returns HWND:280416
31376     [+] DestroyWindow(HWND:001404A2) returns BOOL:1
31377     [+] HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:010510C0) returns BOOL:17114625
31378     [+] HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01051FB0) returns BOOL:1
31379     [+] HeapFree(HANDLE:00140000, FLAGS:00000000, PTR:001565A0) returns BOOL:1
31380     [+] HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01051EB0) returns BOOL:1
31381     [+] IsChild(HWND:001404F0, HWND:00250554) returns BOOL:0
31382     [+] IsChild(HWND:001704FC, HWND:00250554) returns BOOL:0
31383     [+] GetFocus() returns HWND:280416
```

... which is then compared with our serial

이 다음에 우리의 serial 과 비교한다.

The screenshot shows the assembly dump of CConvert.exe. A yellow callout box highlights the instruction `_vbaVarTstEq(VARIANT:****, VARIANT:****) returns DWORD:0`. Below it, an explanation states: `_vbaVarTstEq is used to compare variants. If they are not the same, DWORD=0`. It also notes that if they are the same, DWORD will be FFFFFFFF (so eax=FFFFFF). Similar to `_vbaVarCmpEq`.

```
29370     .OLE _TextBox>AddRef(LPINTERFACE:01052224) returns DWORD:1
29371     .vbaObjSet(LPINTERFACE:0012F484, LPINTERFACE:01052224) returns LPVOID:1052224
29372     .txtRegCode.Text
29383     .Len(String:"47806")
29384     .vbaFreeStr(LPBS)
29389     .vbaFreeObj(LPIN)
29392     .OLE _TextBox>AddRef(LPINT)
29393     .vbaObjSet(LPINT)
29394     .txtRegCode.Text
29405     .vbaVarMove(VAR)
29406     .vbaFreeObj(LPIN)
29409     .vbaVarMul(VARIAN)
29412     .vbaVarMove(VAR)
29413     .vbaVarCat(VARIAN)
29424     .vbaVarCat(VARIAN)
29425     .SysAllocStringByteLen(char:00000000, DWORD:0000001A) returns LPVOID:156DF4
29428     .vbaVarCat returns DWORD:12F464
29429     .vbaVarTstEq(VARIANT:String:"47806", VARIANT:String:"REG-2160...") returns DWORD:0
29430     .SysFreeString(BSTR:00156DCC)
29433     .SysFreeString(BSTR:00156DF4)
29436     .vbaVarDup(VARIANT:String:"Sorry, B...", VARIANT:Empty) returns DWORD:12F474
29441     .MsgBox(VARIANT:String:"Sorry, B...", Integer:0, VARIANT:Missing, VARIANT:Missing, VARIANT:Missing) returns
31369     .SysFreeString(BSTR:00157484)
31372     .vbaObjSetAddref(LPINTERFACE:0012F484, LPINTERFACE:00157718) returns LPVOID:157718
31373     .ShowWindow(HWND:0026041A, DWORD:00000000) returns BOOL:16
31908     .IsChild(HWND:00140442, HWND:00250554) returns BOOL:0
31909     .GetFocus() returns HWND:280416
31910     .DestroyWindow(HWND:00140442) returns BOOL:1
31931     .HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:010510C0) returns BOOL:17114625
31932     .HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01051FB0) returns BOOL:1
31933     .HeapFree(HANDLE:00140000, FLAGS:00000000, PTR:001565A0) returns BOOL:1
31934     .HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01051EB0) returns BOOL:1
31935     .IsChild(HWND:001404F0, HWND:00250554) returns BOOL:0
31936     .IsChild(HWND:001704FC, HWND:00250554) returns BOOL:0
31937     .GetFocus() returns HWND:280416
```

In general :

`_vbaVarTstEq(VARIANT:****, VARIANT:****) returns DWORD:0`

Explanation:

`_vbaVarTstEq` is used to compare variants.

`_vbaVarTstEq` 는 비교하는데 사용되는 변종이다.

If they are not the same, DWORD=0

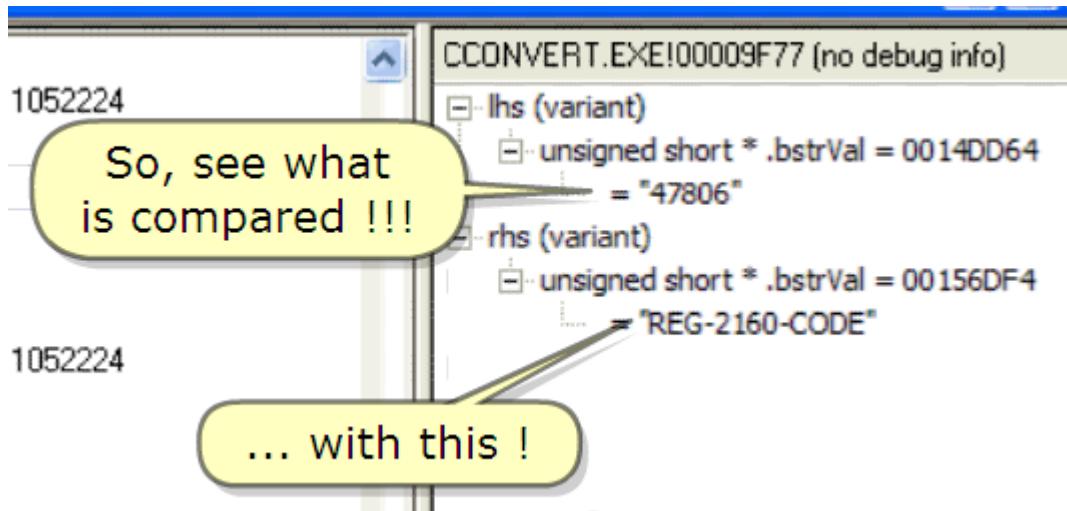
그들이 같지 않다면, DWORD=0

If they are the same, DWORD will be FFFFFFFF (so EAX=FFFFFF)

그들이 같다면, DWORD FFFFFFFF (EAX=FFFFFF) 이 될 것이다.

Similar to `_vbaVarCmpEq`

비슷한 것으로는 `_vbaVarCmpEq` 다.



So, see what is compared !!!

그래서, 무엇을 비교했는지 봐 !!!

... with this !

... 이것과 비교한다 !

CConvert.exe - Program Results	
29370	OLE _TextBox::AddRef(LPINTERFACE::01052224) returns DWORD:1
29371	◆ _vbaObjSet(LPINTERFACE ::0012F484, LPINTERFACE::01052224) returns LPVOID:1052224
29372	[+] txtRegCode.Text
29383	◆ Len(String:"47806") returns LONG:5
29384	[+] ◆ _vbaFreeStr(LPBSTR::0012F488) returns DWORD:10
29389	[+] ◆ _vbaFreeObj(LPINTERFACE ::0012F484)
29392	OLE _TextBox::AddRef(LPINTERFACE::01052224) returns DWORD:1
29393	◆ _vbaObjSet(LPINTERFACE ::0012F484, LPINTERFACE::01052224) returns LPVOID:1052224
29394	[+] txtRegCode.Text
29405	◆ _vbaVarMove(VARIANT:String:"47806", VARIANT:Empty) returns DWORD:12F4BC
29406	◆ _vbaFreeObj(LPINTERFACE ::0012F484)
29409	◆ _vbaVarMul(VARIANT:Empty)
29412	◆ _vbaVarMove(VARIANT:Empty)
29413	◆ _vbaVarCat(VARIANT:Empty)
29424	◆ _vbaVarCat(VARIANT:String REG-2160..., VARIANT:String:"REG-2160...") returns DWORD:12F464
29429	◆ _vbaVarTstEq(VARIANT:String:"47806", VARIANT:String:"REG-2160...") returns DWORD:0
29430	◆ SysFreeString(BSTR::00156DCC)
29433	◆ SysFreeString(BSTR::00156DF4)
29436	◆ _vbaVarDup(VARIANT:String:"Sorry, B...", VARIANT:Empty) returns DWORD:12F474
29441	◆ MsgBox(VARIANT:String:"Sorry, B...", Integer:0, VARIANT:Missing, VARIANT:Missing, VARIANT:Missing) returns
31369	◆ SysFreeString(BSTR::00157484)
31372	◆ _vbaObjSetAddref(LPINTERFACE ::0012F484, LPINTERFACE::00157718) returns LPVOID:157718
31373	◆ ShowWindow(HWND::0026041A, DWORD:00000000) returns BOOL:16
31908	◆ IsChild(HWND::001404A2, HWND:00250554) returns BOOL:0
31909	◆ GetFocus() returns HWND:280416
31910	◆ DestroyWindow(HWND::001404A2) returns BOOL:1
31931	◆ HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:010510C0) returns BOOL:17114625
31932	◆ HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01051FB0) returns BOOL:1
31933	◆ HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:001565A0) returns BOOL:1
31934	◆ HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01051EB0) returns BOOL:1
31935	◆ IsChild(HWND::001404F0, HWND:00250554) returns BOOL:0
31936	◆ IsChild(HWND::001704FC, HWND:00250554) returns BOOL:0
31937	◆ GetFocus() returns HWND:280416
31938	◆ DestroyWindow(HWND::001704FC) returns BOOL:1
31964	◆ HeapFree(HANDLE:01040000, FLAGS:00000000, PTR:01052200) returns BOOL:17112833

In our case, they are not equal ...

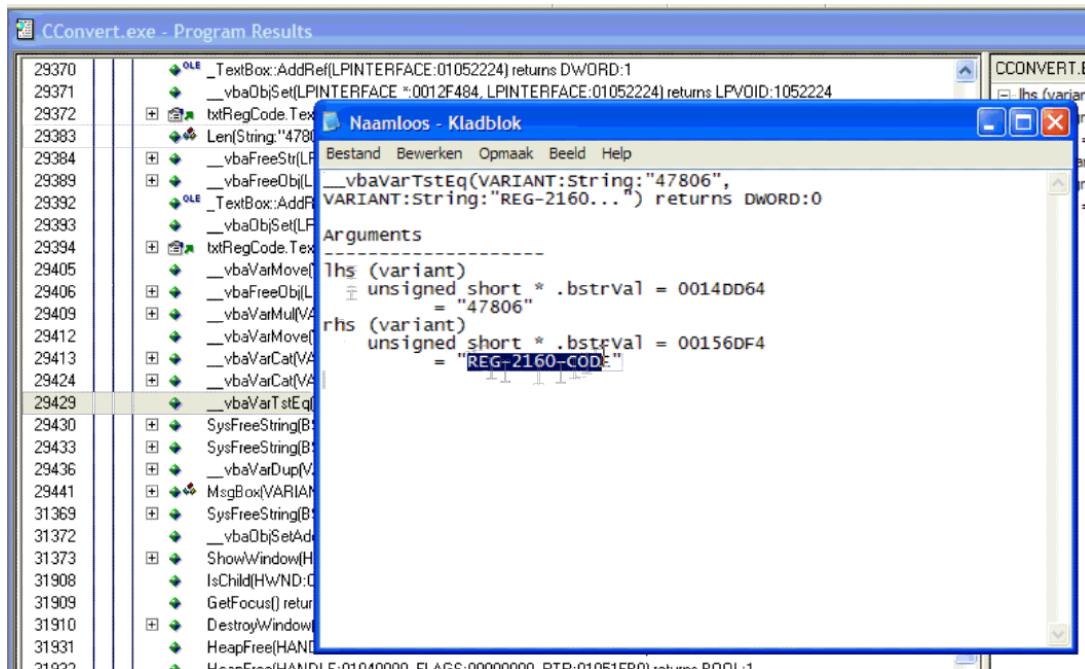
There comes the Badboy !!!

이 경우, 그들은 같지 않다 ...

Badboy 가 온다 !!!

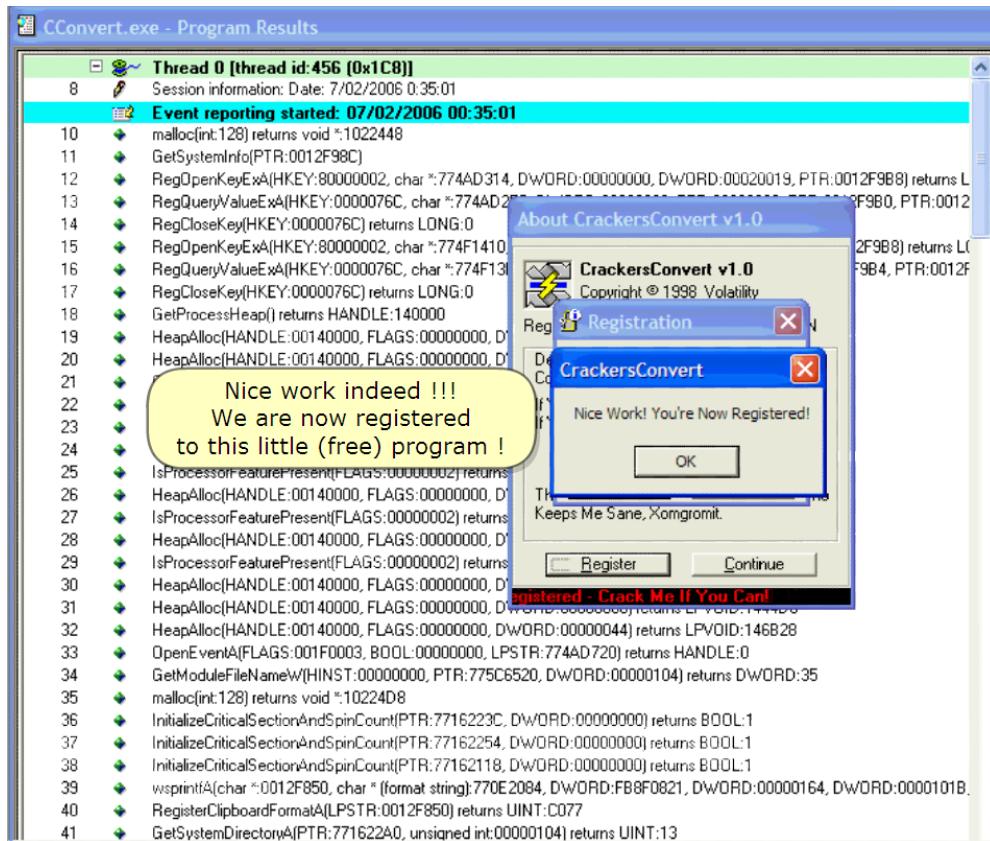
So, press Ctrl-C to copy the data from the highlighted line in notepad and then try it out in the program

Notepad 의 Highlight 된 line 에서 Data 를 copy 하기 위해 Ctrl-C 를 눌러. 그리고 program 의 밖으로 나와.



And press Ctrl-V to paste it into notepad

Notepad 에 붙여넣기 하기 위해 Ctrl-V 눌러.



Nice work indeed !!!

We are now registered to this little (free) program!

정말 좋은 일이다 !!!

우리는 이제 program 을 등록했다.

INFO :

For completeness of this tutorial : the registered program writes the registration data in cconv.\$\$\$ and cconv.ccc and also verifies registration data from these files at each startup

Tutorial 의 완벽함 : 등록된 program 은 registration data 를 cconv.\$\$\$와 cconv.ccc 에 저장했다.

그리고 또한 각 startup 에서 그들의 file 로부터 registration data 를 검사한다.

All right.

This went well, let's try our luck in another ReverseMe

좋아.

이것은 좋다, 다른 ReverseMe 에서 우리의 운을 도전해 보자.

6. ReverseMe2

And so, we land here in Olly where I have opened ReverseMe2 already.

우리는 Olly 에 도착했다. 이미 ReverseMe2 를 열었다.

"In Olly ?" you probably ask.

"Olly 안이야?" 너는 아마 물을 것이다.

Well yes, I first opened this ReverseMe in SmartCheck but SmartCheck went berserk ;)

좋아, 나는 첫번째로 이 SmartCheck에서 ReverseMe를 열었다. 그러나 SmartCheck는 미쳤다. ;)

So, I want to see in Olly what's going on.

그래서, 나는 Olly에서 어떻게 되는지 보기 원한다.

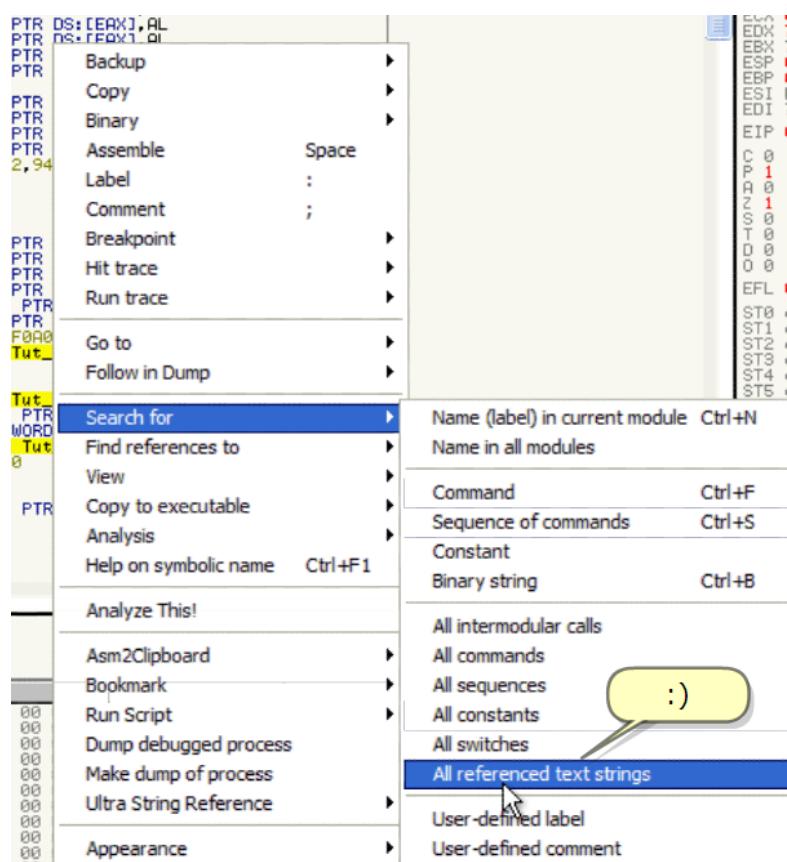
I have a good idea however : this is an anti-SmartCheck trick from the ReverseMe2.

나에게는 좋은 idea가 있다. 그러나 : 이것은 ReverseMe2의 anti-SmartCheck trick이다.

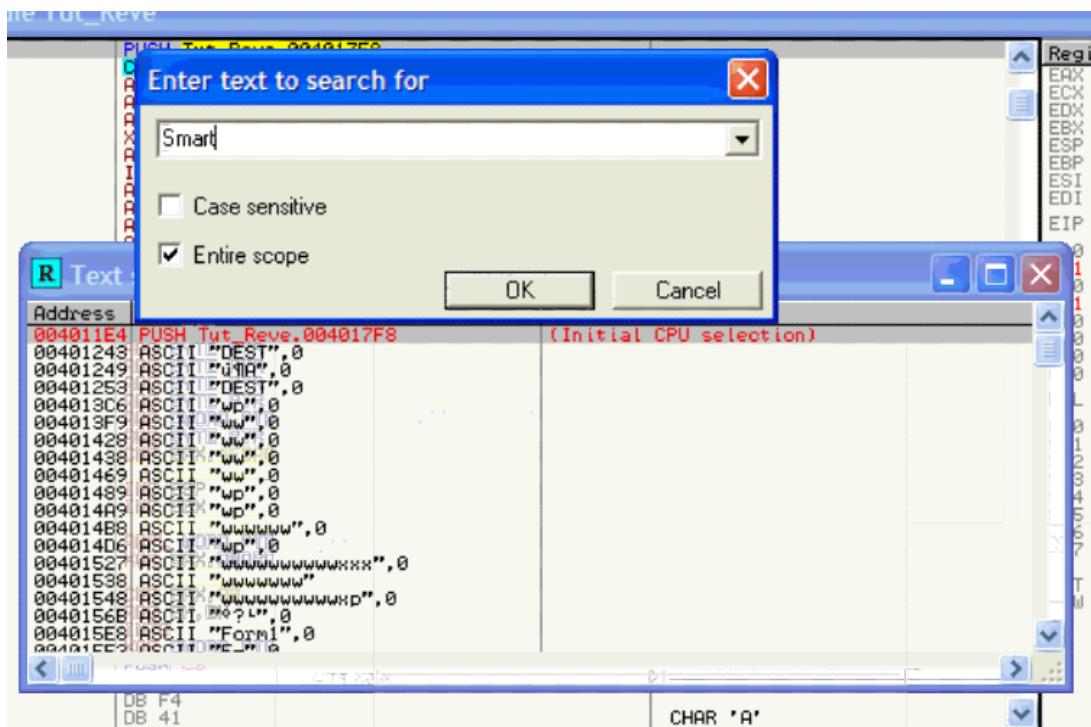
As soon as the ReverseMe notices SmartCheck, it tries to close SmartCheck. Look how to solve it.

즉시 ReverseMe는 SmartCheck에게 알려줬다. SmartCheck를 닫기 위해 도전했다. 어떻게

해결하는지 봐.

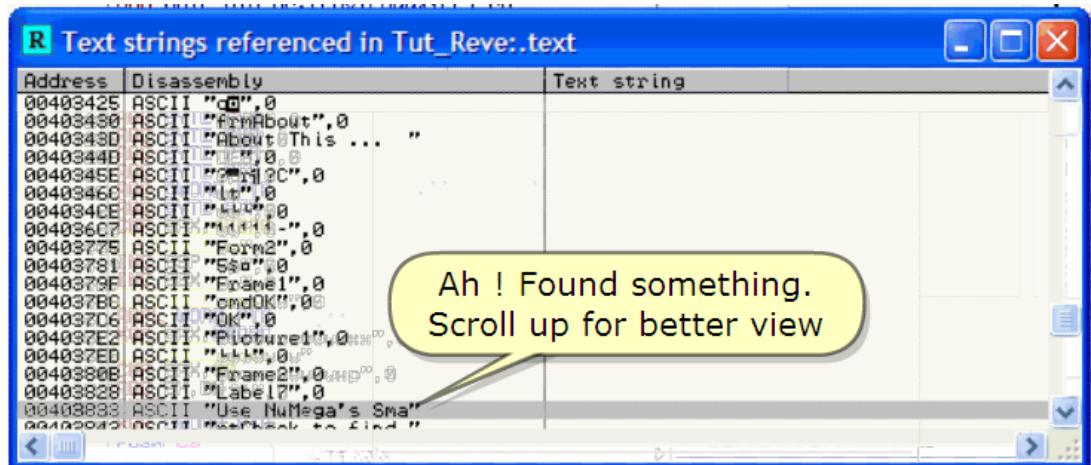


;))



Let's see if this ReverseMe is looking for SmartCheck :)

이 ReverseMe 가 SmartCheck 를 찾고 있는지 봐.



AH ! Found something. Scroll up for better view

아! 무언가를 찾았다. 좀 더 좋은 것을 보기 위해 Scroll 올려.

No! This is piece from the About box.

We've seen when studying the ReverseMe.

Press Ctrl-L to look further

아니다! 이것은 About box 에서 온 조각이다.

ReverseMe 를 공부할 때 우리는 봤다.

멀리 보기 위해 Ctrl-L 를 눌러.

번역 주)Ctrl-L 은 다음 찾기

The screenshot shows the OllyDbg debugger interface. The assembly window at the top displays several ADD instructions with immediate values like 0x41, 0x42, etc., followed by comments such as "Label1", "ReverseMe Tutori", and "a Part10". A yellow callout bubble points to the first ADD instruction with the text "Aha ! Let's see it in the code.". Below the assembly window is a text strings window titled "Text strings referenced in Tut_Reve:.text". It lists memory addresses, disassembly, and text strings. The strings include ASCII and UNICODE representations of various messages and file names like "Label1", "ReverseMe Tutori", "a Part10", "MS Sans Serif", "DB", "NuMega SmartCheck", and "UNREGISTERED" messages.

Address	Disassembly	Text string
00403E40	ASCII "Label1", 0	
00403E4B	ASCII "ReverseMe Tutori"	
00403E5B	ASCII "a Part10"	
00403E6B	ASCII " ", 0	
00403E80	ASCII "DB"	
00403E85	ASCII "MS Sans Serif"	
004040D0	PUSH Tut_Reve.00401E18	UNICODE "reginfo.key"
00404279	PUSH Tut_Reve.00401E8C	UNICODE "REGISTERED? Why did it take so l
0040439E	PUSH Tut_Reve.00401E88	ASCII "3+A"
00404525	MOV DWORD PTR SS:[EBP-104],Tut_Rev	UNICODE "NuMega SmartCheck"
0040458F	PUSH Tut_Reve.00401F78	UNICODE "%(F4)"
004045B9	PUSH Tut_Reve.00401F88	UNICODE "%V"
004045CD	MOV EDI,Tut_Reve.00401E18	UNICODE "reginfo.key"
004046FC	PUSH Tut_Reve.00401F9C	UNICODE "UNREGISTERED: Keyfile found : ali
00404F18	PUSH Tut_Reve.00402020	"Registered? Why did it take so l
00404F5C	PUSH Tut_Reve.00401F9C	UNICODE "UNREGISTERED: Keyfile found : ali
00404F8C	PUSH Tut_Reve.0040206C	UNICODE "UNREGISTERED"
004058B0	ASCII "XXXX", 0	UNICODE "UNREGISTERED"

Aha! Let's see it in the code

아하! Code 에서 보자.

Right! We found the guilty! Now, how to circumvent this?

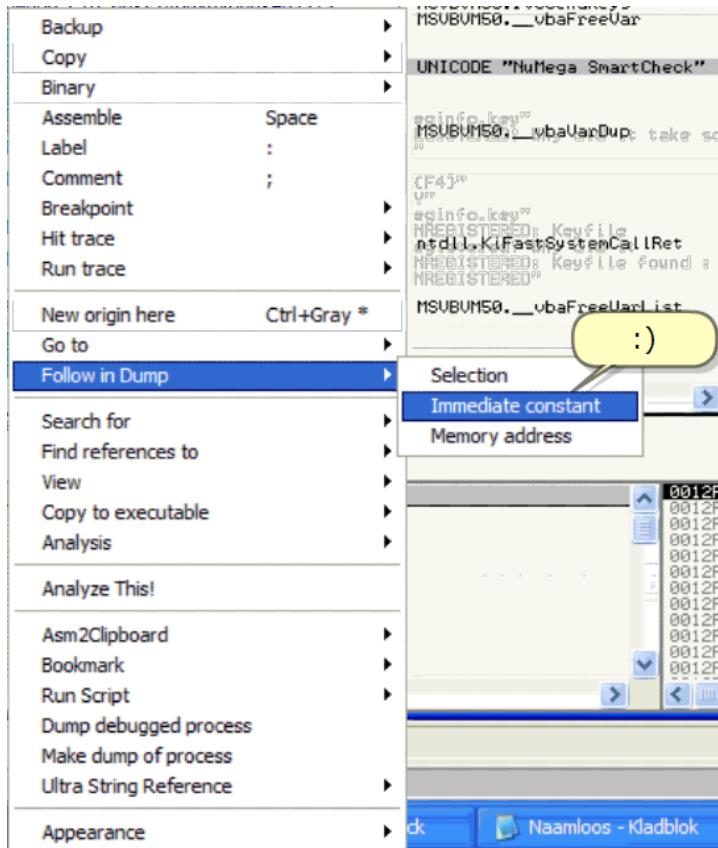
좋아! 우리는 의심스러운 것을 찾았다! 이제, 이것을 어떻게 회피해요?

Well, just make the ReverseMe search for some other name or.... Change all occurrences "NuMega SmartCheck" in SmartCheck with Olly !

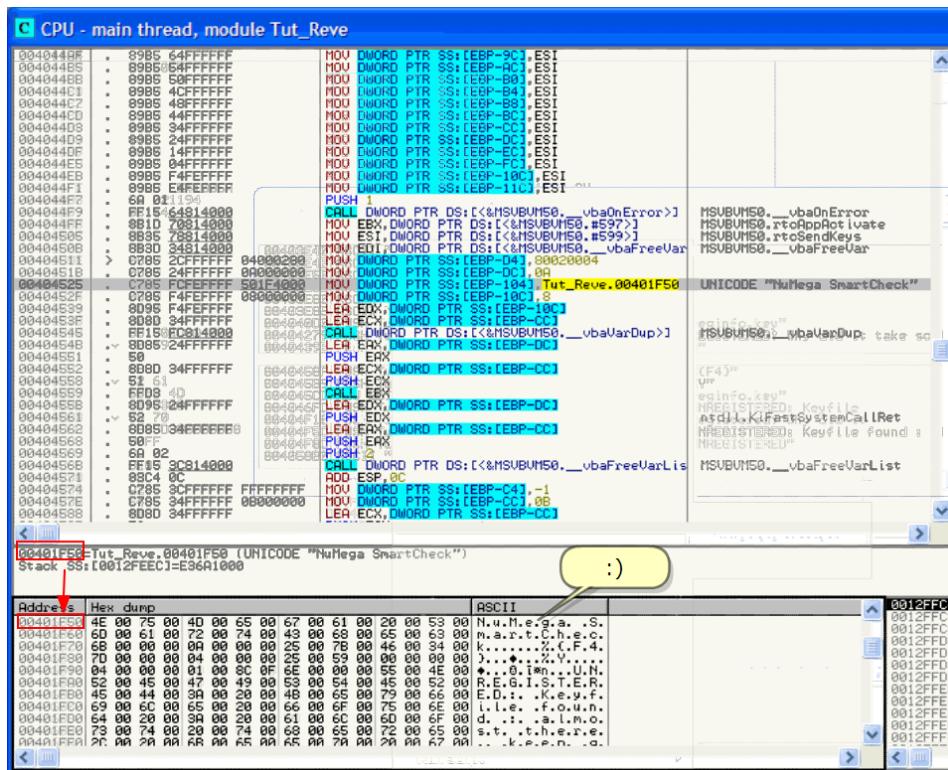
좋아, 바로 ReverseMe 가 약간의 다른 name 을 찾도록 하거나 ... SmartCheck 에서 Olly 와 함께 발생하는 모든 것을 "NuMega SmartCheck" 바꿔.

I will show you the easiest here which is make the ReverseMe search for a bogus name.

너에게 쉽다는 것을 보여줄 것이다. ReverseMe 를 bogus name 을 찾게 시켰다.



1



1

To make changes in the dump, highlight a byte and just press any number key on the key pad
Dump 에 있는 내용을 바꾸자. Highligh 된 byte 와 keypad 에 있는 모든 number key 를 눌러라.

And now press another key

그리고 이제 다른 key 를 눌러.

So, we will change 4D

우리는 4D 를 바꿀 수 있다.

Into 55. Press enter

55 로 바꾼다. Enter 를 눌러.

Dec	Hex	ASCII		Dec	Hex	ASCII		Dec	Hex	ASCII		Dec	Hex	ASCII
0	00	NUL		32	20	SP		64	40	@		96	60	'
1	01	SOH		33	21	!		65	41	A		97	61	a
2	02	STX		34	22	"		66	42	B		98	62	b
3	03	ETX		35	23	#		67	43	C		99	63	c
4	04	EOT		36	24	\$		68	44	D		100	64	d
5	05	ENQ		37	25	%		69	45	E		101	65	e
6	06	ACK		38	26	&		70	46	F		102	66	f
7	07	BEL		39	27	'		71	47	G		103	67	g
8	08	BS		40	28	(72	48	H		104	68	h
9	09	HT		41	29)		73	49	I		105	69	i
10	0A	LF		42	2A	*		74	4A	J		106	6A	j
11	0B	VT		43	2B	+		75	4B	K		107	6B	k
12	0C	FF		44	2C	,		76	4C	L		108	6C	l
13	0D	CR		45	2D	-		77	4D	M		109	6D	m
14	0E	SO		46	2E	.		78	4E	N		110	6E	n
15	0F	SI		47	2F	/		79	4F	O		111	6F	o
16	10	DLE		48	30	0		80	50	P		112	70	p
17	11	DC1		49	31	1		81	51	Q		113	71	q
18	12	DC2		50	32	2		82	52	R		114	72	r
19	13	DC3		51	33	3		83	53	S		115	73	s
20	14	DC4		52	34	4		84	54	T		116	74	t
21	15	NAK		53	35	5		85	55	U		117	75	u
22	16	SYN		54	36	6		86	56	V		118	76	v
23	17	ETB		55	37	7		87	57	W		119	77	w
24	18	CAN		56	38	8		88	58	X		120	78	x
25	19	EM		57	39	9		89	59	Y		121	79	y
26	1A	SUB		58	3A	:		90	5A	Z		122	7A	z
27	1B	ESC		59	3B	;		91	5B	[123	7B	{
28	1C	FS		60	3C	<		92	5C	\		124	7C	
29	1D	GS		61	3D	=		93	5D]		125	7D	}
30	1E	RS		62	3E	>		94	5E	^		126	7E	~
31	1F	US		63	3F	?		95	5F	_		127	7F	DEL

Which is 55 (ascii == U).

55 는 (ascii == U)

00401F50 | : C785 34FFFFFF 0B000000 MOU DWORD PTR SS:[EBP-CC],0B
00401F58 | : 8D8D 34FFFFFF LEA ECX,DWORD PTR SS:[EBP-CC]

00401F50=Tut_Reve.00401F50 (UNICODE "NuUega SmartCheck")
Stack SS:[0012FE0C]=E36A1000

Address	Hex dump	ASCII
00401F50	4E 00 75 00 55 00 65 00 67 00 61 00 20 00 53 00	N.u.U.e.g.a..S.
00401F58	60 00 61 00 72 00 74 00 43 00 68 00 65 00 63 00	m.a.r.t.C.h.e.c.
00401F70	68 00 00 00 00 00 00 25 00 78 00 46 00 34 00	k.....%(.F.4.
00401F80	70 00 00 00 00 04 00 00 00 25 00 59 00 00 00 00	J...♦...%V...
00401F90	04 00 00 00 01 00 8C 0F 6E 00 00 00 55 00 4E 00	♦.I.e.n..U.N.
00401FA0	52 00 45 00 47 00 49 00 53 00 54 00 45 00 52 00	R.E.G.I.S.T.E.R.
00401FB0	45 00 44 00 39 00 28 00 48 00 65 00 79 00 66 00	E.D.: .K.e.y.f.
00401FC0	69 00 6C 00 65 00 28 00 66 00 6F 00 75 00 6E 00	i.l.e..f.o.u.n.
00401FD0	64 00 20 00 94 00 28 00 61 00 6C 00 60 00 6F 00	d.::..a.l.m.o.
00401FE0	73 00 24 00 28 00 74 00 60 00 65 00 72 00 65 00	s.t..t.h.e.r.e.
00401FF0	2F 00 20 00 45 00 45 00 7A 00 20 00 67 00k.e.p.n..o.

:)

00404569 | : 6A 02 00 00 00 00 00 00 00 00 00 00 00 00 00 00 PUSH EBP
0040456B | : FF15 3C814000 CALL DWORD PTR DS:[<&MSUBUM50_ vbaFreeVarList MSUBUM50_ vbaFreeVarList

00404571 | : 88C4 0C
00404574 | : C785 3CFFFFFF FFFFFFFF
0040457E | : C785 34FFFFFF 0B000000
00404588 | : 8D8D 34FFFFFF

00401F50=Tut_Reve.00401F50 (UNICODE "NuUega SmartCheck")
Stack SS:[0012FE0C]=E36A1000

Address	Hex dump	ASCII
00401F50	4E 00 75 00 55 00 65 00 67 00 61 00 20 00 53 00	N.u.U.e.g.a..S.
00401F58	60 00 61 00 72 00 74 00 43 00 44 00 55 00 63 00	m.a.r.t.C.♦.e.c.
00401F70	68 00 00 00 00 00 00 25 00 78 00 46 00 34 00	k.....%(.F.4.
00401F80	70 00 00 00 00 04 00 00 00 25 00 59 00 00 00 00	J...♦...%V...
00401F90	04 00 00 00 01 00 8C 0F 6E 00 00 00 55 00 4E 00	♦.I.e.n..U.N.
00401FA0	52 00 45 00 47 00 49 00 53 00 54 00 45 00 52 00	R.E.G.I.S.T.E.R.
00401FB0	45 00 44 00 39 00 28 00 48 00 65 00 79 00 66 00	E.D.: .K.e.y.f.
00401FC0	69 00 6C 00 65 00 28 00 66 00 6F 00 75 00 6E 00	i.l.e..f.o.u.n.
00401FD0	64 00 20 00 94 00 28 00 61 00 6C 00 60 00 6F 00	d.::..a.l.m.o.
00401FE0	73 00 24 00 28 00 74 00 60 00 65 00 72 00 65 00	s.t..t.h.e.r.e.
00401FF0	2F 00 20 00 45 00 45 00 7A 00 20 00 67 00k.e.p.n..o.

If I save these changes in the ReverseMe, it will search for "NuUega SmartCDeck", I doubt it will find something :)

ReverseMe에서 바뀐 것들을 저장한다면, 그것은 "NuUega SmartCDeck"를 찾을 것이다. 그것이 무언가를 찾을 수 있을지 의심한다.

Save the changes. Load the saved ReverseMe in SmartCheck and

... we land here (removed to reduce movie size)

바뀐 것들을 저장해. SmartCheck에서 저장된 ReverseMe를 불러 와. 그리고 ...

우리는 여기에 도착했다(movie size를 줄이기 위해 삭제했다)

Aha! No problems any more with this ReverseMe2 in SmartCheck !!!

I suppose you understand that I tricked the ReverseMe into searching for a wrong name in SmartCheck?

아하! SmartCheck에서 이 ReverseMe2와 같이 더 이상 문제없다 !!!

네가 ReverseMe를 SmartCheck에서 잘못된 name으로 검색하게 trick 했던 것을 이해 했을 것이라 생각한다.

INFO :

This basic anti-technique can also be used for other tools like Olly, LordPE, ImpRec, etc etc.

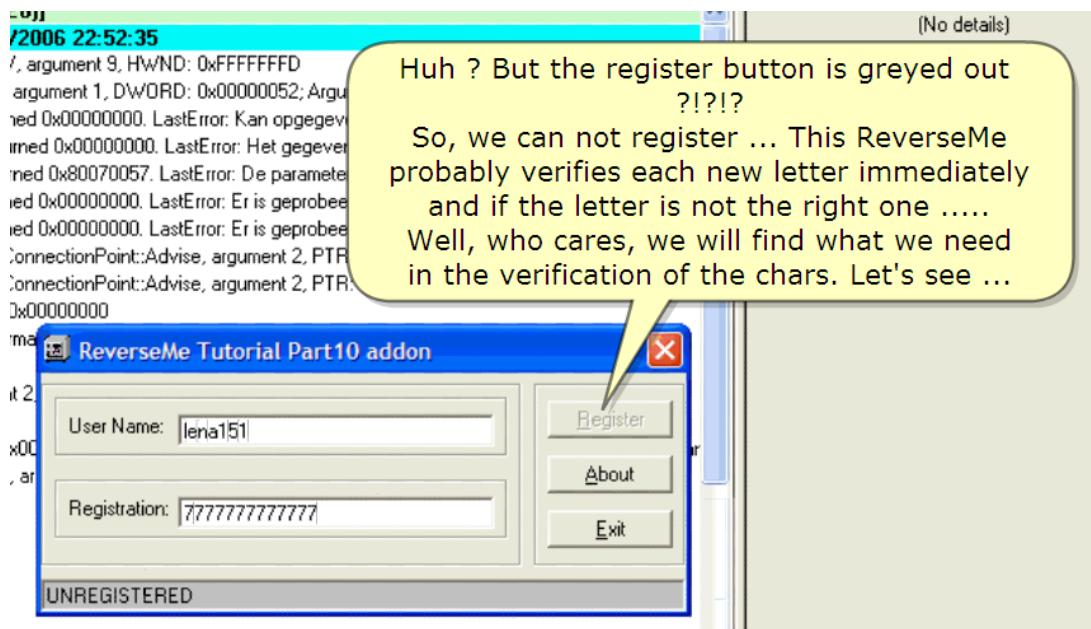
기본 anti-technique에는 Olly, LordPE, ImpRec, etc 등 같은 다른 tools들이 사용됐다.

But there exist also tools to do this and much more and better too ;)

그러나 그것은 또한 하기 위해 존재한다. 그리고 좀 더 좋아진다. ;)

Let's register this ReverseMe !

이제 ReverseMe 를 등록하자!



Huh? But the register button is greyed out ?!?!?

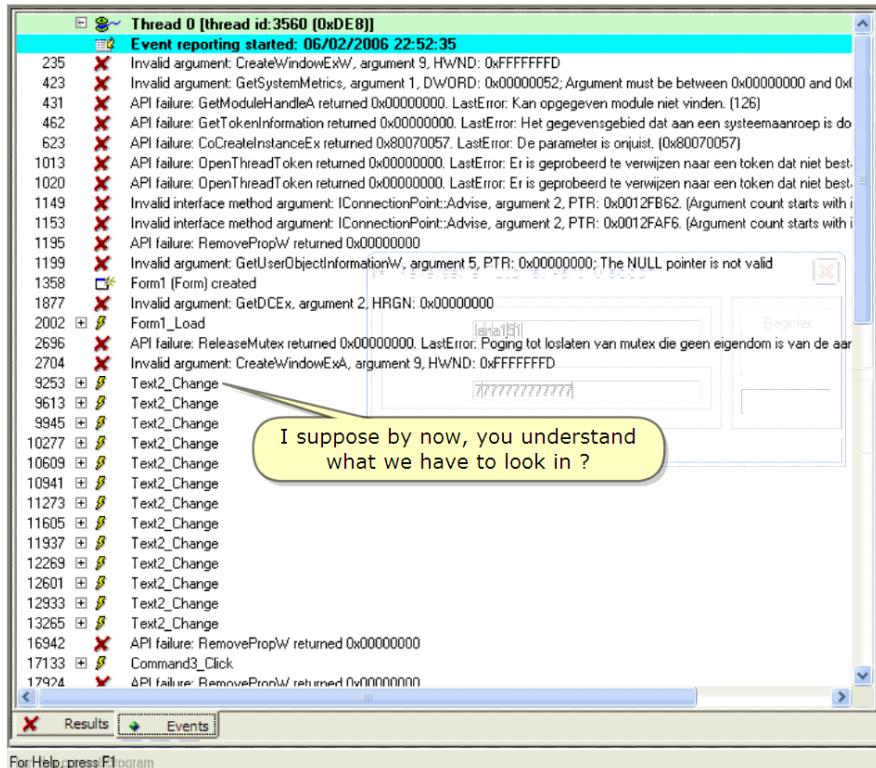
잉? Register button 이 회색이네 ?!?!?

So, we can not register ... This ReverseMe probably verifies each new letter immediately and if the letter is not the right one

그래서, 우리는 등록할 수 없다. 이 ReverseMe 는 아마 각각 새로운 문자를 즉시 검증한다. 만약에 문자가 올바르지 않다면

Well, who cares, we will find what we need in the verification of the chars. Let's see ...

좋아, 걱정하지마, 우리는 검증하는데 필요한 문자를 찾을 수 있다. 보자...



I suppose by now, you understand what we have to look in?

이제 우리가 무엇을 봐야 하는지 이해 하겠어?

Scroll down

Scroll 내려

Study the code

Code 공부해.

Here, we see the calculation of some chars.

여기, 우리는 약간 문자를 계산하는 것을 보자.

```

1149 ✗ Invalid interface method argument: IConnectionPoint::Advise, argument 2, PTR: 0x0012FB62 [Argument count starts with i
1153 ✗ Invalid interface method argument: IConnectionPoint::Advise, argument 2, PTR: 0x0012FAF6. [Argument count starts with i
1195 ✗ API failure: RemovePropW returned 0x00000000
1199 ✗ Invalid argument: GetUserObjectInformationW, argument 5, PTR: 0x00000000; The NULL pointer is not valid
1358 Form1 [Form] created
1877 ✗ Invalid argument: GetDCEx, argument 2, HRGN: 0x00000000
2002 + ⚡ Form1_Load
2696 ✗ API failure: ReleaseMutex returned 0x00000000. LastError: Poging tot loslaten van mutex die geen eigendom is van de aar
2704 ✗ Invalid argument: CreateWindowExA, argument 9, HWND: 0xFFFFFFF
9253 - ⚡ Text2_Change
9256   Text1.Text
9267     Left(VARIANT:String:"lena151", long:1)
9271     Asc(String:"I") returns Integer:108
9287   Text1.Text
9298     Mid(VARIANT:String:"lena151", long:2, VARIANT:Integer:1)
9302     Asc(String:"e") returns Integer:101
9318   Text1.Text
9329     Mid(VARIANT:String:"lena151", long:3, VARIANT:Integer:1)
9333     Asc(String:"n") returns Integer:110
9349   Text1.Text
9360     Mid(VARIANT:String:"lena151", long:4, VARIANT:Integer:1)
9364     Asc(String:"a") returns Integer:97
9380   Text1.Text
9391     Mid(VARIANT:String:"lena151", long:5, VARIANT:Integer:1)
9395     Asc(String:"I") returns Integer:49
9467     Mid(VARIANT:String:"10810111...", long:2, VARIANT:Integer:10)
9478   Text2.Text
9536     Text2_Change
9613 + ⚡ Text2_Change
9945 + ⚡ Text2_Change
10277 + ⚡ Text2_Change
10609 + ⚡ Text2_Change
10941 + ⚡ Text2_Change
11273 + ⚡ Text2_Change

```

But when digging in the Text2.Text ... I couldn't find anything relevant for the case. (Removed from movie)

But when digging in the Text2.Text ... I couldn't find anything relevant for the case.(Removed from movie)

그러나 Text2.Text 를 파고들 때, 나는 이번 경우와 관계 있는 어떤 것도 찾을 수 없었다.(movie 에서 삭제했다)

```

1149 ✗ Invalid interface method argument: IConnectionPoint::Advise, argument 2, PTR: 0x0012FB62 [Argument count starts with i
1153 ✗ Invalid interface method argument: IConnectionPoint::Advise, argument 2, PTR: 0x0012FAF6. [Argument count starts with i
1195 ✗ API failure: RemovePropW returned 0x00000000
1199 ✗ Invalid argument: GetUserObjectInformationW, argument 5, PTR: 0x00000000; The NULL pointer is not valid
1358 Form1 [Form] created
1877 ✗ Invalid argument: GetDCEx, argument 2, HRGN: 0x00000000
2002 + ⚡ Form1_Load
2696 ✗ API failure: ReleaseMutex returned 0x00000000. LastError: Poging tot loslaten van mutex die geen eigendom is van de aar
2704 ✗ Invalid argument: CreateWindowExA, argument 9, HWND: 0xFFFFFFF
9253 - ⚡ Text2_Change
9256   Text1.Text
9267     Left(VARIANT:String:"lena151", long:1)
9271     Asc(String:"I") returns Integer:108
9287   Text1.Text
9298     Mid(VARIANT:String:"lena151", long:2, VARIANT:Integer:1)
9302     Asc(String:"e") returns Integer:101
9318   Text1.Text
9329     Mid(VARIANT:String:"lena151", long:3, VARIANT:Integer:1)
9333     Asc(String:"n") returns Integer:110
9349   Text1.Text
9360     Mid(VARIANT:String:"lena151", long:4, VARIANT:Integer:1)
9364     Asc(String:"a") returns Integer:97
9380   Text1.Text
9391     Mid(VARIANT:String:"lena151", long:5, VARIANT:Integer:1)
9395     Asc(String:"I") returns Integer:49
9467     Mid(VARIANT:String:"10810111...", long:2, VARIANT:Integer:10)
9478   Text2.Text
9536     Text2_Change
9613 + ⚡ Text2_Change
9945 + ⚡ Text2_Change
10277 + ⚡ Text2_Change
10609 + ⚡ Text2_Change
10941 + ⚡ Text2_Change
11273 + ⚡ Text2_Change

```

That made me think....
Mmmm, perhaps the ReverseMe knows at startup (when loading the form) that it is not registered (indeed, the main window states "UNREGISTERED"). Mmmm, let's see there.

That made me think....

그것을 나를 생각하게 만들었다.

Mmmm, perhaps the ReverseMe knows at startup (when loading the form) that it is not registered (indeed, the main window states "UNREGISTERED").

음, 아마 ReverseMe 는 startup 할 때 (form 을 loading 할 때) 그것은 등록되지 않았다.(정말, main window 상태는 "UNREGISTERED")

Mmmm, let's see there.

음, 그곳을 보자.

The screenshot shows a debugger interface with assembly code. A yellow callout bubble points from the text "Mmmmm. Yes indeed! Let's see this more in detail ..." to the assembly instruction at address 2221. The assembly code at 2221 is:

```
2221  mov     eax,dword ptr [Text3.Text] ; Text3.Text <- "UNREGISTERED" (String)
```

The tooltip contains the text: "Mmmmm. Yes indeed! Let's see this more in detail ...".

Below the assembly code, the debugger shows several other assembly instructions and their corresponding opcodes and arguments, such as:

- 2270 mov eax,dword ptr [Form1_Load]
- 2696 xor eax,eax
- 2704 mov eax,dword ptr [Text2_Change]
- 9253 mov eax,dword ptr [Text1_Text]
- 9256 mov eax,dword ptr [Text1_Text]
- 9267 mov eax,dword ptr [Text1_Text]
- 9271 mov eax,dword ptr [Text1_Text]
- 9274 mov eax,dword ptr [Text1_Text]
- 9278 mov eax,dword ptr [Text1_Text]
- 9287 mov eax,dword ptr [Text1_Text]
- 9298 mov eax,dword ptr [Text1_Text]
- 9302 mov eax,dword ptr [Text1_Text]
- 9318 mov eax,dword ptr [Text1_Text]
- 9329 mov eax,dword ptr [Text1_Text]
- 9333 mov eax,dword ptr [Text1_Text]
- 9349 mov eax,dword ptr [Text1_Text]
- 9360 mov eax,dword ptr [Text1_Text]
- 9364 mov eax,dword ptr [Text1_Text]
- 9380 mov eax,dword ptr [Text1_Text]
- 9391 mov eax,dword ptr [Text1_Text]
- 9395 mov eax,dword ptr [Text1_Text]
- 9467 mov eax,dword ptr [Text2_Text]
- 9478 mov eax,dword ptr [Text2_Text]
- 9536 mov eax,dword ptr [Text2_Change]
- 9613 mov eax,dword ptr [Text2_Change]

Mmmmm. Yes indeed! Let's see this more in detail ...

음. 정말! 좀 더 자세히 보자...

The screenshot shows the ReverseMe debugger interface with assembly code and annotations. Annotations include:

- INFO :** See what our ReverseMe verifies now :))
- ... but was not found ;)**

```

1149  Invalid interface method argument: IConnectionPoint::Advise, argument 2, PTR: 0x0012FB62. (Argument count starts with i
1153  Invalid interface method argument: IConnectionPoint::Advise, argument 2, PTR: 0x0012FAF6. (Argument count starts with i
1195  API failure: RemovePropW returned 0x00000000
1199  Invalid argument: GetUs...
1358  Form1 [Form] created
1877  Invalid argument: GetDCEx...
2002  Form1_Load
2003      OnError(long:1)
2013  AppActivate(VARIANT:String:"NuJega S...", VARIANT:Missing) fails
2197  Dir(VARIANT:String:"reginfo...", FLAGS:00000000)
2221  Text3.Text <- "UNREGISTERED" (String)

2270  Form1_Load
2696  API failure: ReleaseMutex returned 0x00000000. LastError: Poging tot loslaten van mutex die geen eigendom is van de aanroepende applicatie
2704  Invalid argument: CreateWindowExA, argument 9, HWND: 0xFFFFFFFF
9253  Text2_Change
9256      Text1.Text
9267          Left(VARIANT:String:"lena151", long:1)
9271              Asc(String:"I") returns Integer:108
9287          Text1.Text
9298              Mid(VARIANT:String:"lena151", long:2, VARIANT:Integer:1)
9302                  Asc(String:"e") returns Integer:101
9318          Text1.Text
9329              Mid(VARIANT:String:"lena151", long:3, VARIANT:Integer:1)
9333                  Asc(String:"n") returns Integer:110
9349          Text1.Text
9360              Mid(VARIANT:String:"lena151", long:4, VARIANT:Integer:1)
9364                  Asc(String:"a") returns Integer:97
9380          Text1.Text
9391              Mid(VARIANT:String:"lena151", long:5, VARIANT:Integer:1)
9395                  Asc(String:"I") returns Integer:49
9467          Text2.Text
9536      Text2_Change
9613  Text2_Change

```

INFO :

See what our ReverseMe verifies now :))

...but was not found ;)

ReverseMe 가 검증하고 있는 것을 봐. :))

그러나 찾지 못했다. ;)

Oh yes, that looks promising too !!!

오 좋아, 그것은 유망하다 !!!

```

2002  Form1_Load
2003      OnError(long l)
2008      __vbaVarDup(VARIANT::String:"NuUega S...", VARIANT::Empty) returns DWORD:12FA34
2013      AppActivate(VARIANT::String:"NuUega S...", VARIANT::Missing) fails
2186      RtlUnwind(DWORD:0012FAE4, DWORD:74132085, DWORD:00000000, DWORD:00000000) returns DWORD:0
2187      __vbaFreeStrList() returns DWORD:0
2188      __vbaFreeObjList(unsigned int:00000002)
2189      SysFreeString(BSTR:0014EED4)
2192      __vbaVarDup(VARIANT::String:"reginfo...", VARIANT::Empty) returns DWORD:12FA34
2197      Dir(VARIANT::String:"reginfo...", FLAGS:00000000)

Aha ! A test !!!
2213      __vbaFreeVar(VARIANT::String:"") returns DWORD:20
2218      __vbaVarIstEq(VARIANT::String:"", VARIANT::Const String "") returns DWORD:0
2219          OLE _TextBox::AddRef(LPINTERFACE:0103CAC4) returns DWORD:1
2220          __vbaObjSet(LPINTERFACE :0012FA48, LPINTERFACE:0103CAC4) returns LPVOID:103CAC4
2221      Text3.Text <- "UNREGISTERED" (String)
2245          __vbaFreeObj(LPINTERFACE :0012FA48)
2248          __vbaExitProc() returns DWORD:0
2257          __vbaFreeVar(VARIANT::Empty) returns DWORD:10
2258          I'm sure you understand we have to search
2259          before the UNREGISTERED string ???
2260          __vbaFreeVar(VARIANT::String:"") returns DWORD:10
2265          __vbaFreeVar(VARIANT::Empty) returns DWORD:10
2266          __vbaFreeVar(VARIANT::Empty) returns DWORD:10
2267          __vbaFreeVar(VARIANT::Empty) returns DWORD:10
2268          __vbaFreeVar(VARIANT::Empty) returns DWORD:10
2269          __vbaFreeVar(VARIANT::Empty) returns DWORD:10
2270      Form1_Load
2271          OLE _Form::Release(LPINTERFACE:0103C0C4) returns DWORD:0
2272          GetDesktopWindow() returns HWND:10014
2273          GetWindowRect(HWND:00010014, PTR:0012FB9C) returns BOOL:1
2274          SetWindowPos(HWND:00070510, Hwnd:00000000, int:288, int:274, int:0, int:0, SWP_FLAGS:00000015:SWP_NOSI
2445          ShowWindow(HWND:00070510, DWORD:00000001) returns BOOL:0
3119          PostThreadMessage(HWND:00070510, MSG:110691, WPARAM:0nnnnnnnn, LPARAM:0nnnnnnnn) returns BOOL:1

```

I'm sure you understand we have to search before the UNREGISTERED string ??

우리는 미등록 된 문자열을 찾은 후에 검색하는가??

Aha! A test!!!

아하! Test!!!

```

2002  Form1_Load
2003      OnError(long l)
2008      __vbaVarDup(VARIANT::String:"NuUega S...", VARIANT::Empty) returns DWORD:12FA34
2013      AppActivate(VARIANT::String:"NuUega S...", VARIANT::Missing) fails
2186      RtlUnwind(DWORD:0012FAE4, DWORD:74132085, DWORD:00000000, DWORD:00000000) returns DWORD:0
2187      __vbaFreeStrList() returns DWORD:0
2188      __vbaFreeObjList(unsigned int:00000002)
2189      SysFreeString(BSTR:0014EED4)
2192      __vbaVarDup(VARIANT::String:"reginfo...", VARIANT::Empty) returns DWORD:12FA34
2197      Dir(VARIANT::String:"reginfo...", FLAGS:00000000)
2212      __vbaVarMove(VARIANT::String:"", VARIANT::Empty) returns DWORD:12FAA4
2213      __vbaFreeVar(VARIANT::String:"reginfo...") returns DWORD:20

2218      __vbaVarIstEq(VARIANT::String:"", VARIANT::Const String "") returns DWORD:0
2219          OLE _TextBox::AddRef(LPINTERFACE:0103CAC4) returns DWORD:1
2220          __vbaObjSet(LPINTERFACE :0012FA48, LPINTERFACE:0103CAC4) returns LPVOID:103CAC4
2221      Text3.Text <- "UNREGISTERED" (String)
2248          __vbaFreeObj(LPINTERFACE :0012FA48)
2249          __vbaExitProc() returns DWORD:0
2257          __vbaFreeVar(VARIANT::Empty) returns DWORD:0
2258          __vbaFreeVar(VARIANT::Empty) returns DWORD:0
2259          __vbaFreeVar(VARIANT::Empty) returns DWORD:0
2260          __vbaFreeVar(VARIANT::String:"") returns DWORD:10
2265          __vbaFreeVar(VARIANT::Empty) returns DWORD:10
2266          __vbaFreeVar(VARIANT::Empty) returns DWORD:10
2267          __vbaFreeVar(VARIANT::Empty) returns DWORD:10
2268          __vbaFreeVar(VARIANT::Empty) returns DWORD:10
2269          __vbaFreeVar(VARIANT::Empty) returns DWORD:10
2270      Form1_Load
2271          OLE _Form::Release(LPINTERFACE:0103C0C4) returns DWORD:0
2272          GetDesktopWindow() returns HWND:10014
2273          GetWindowRect(HWND:00010014, PTR:0012FB9C) returns BOOL:1
2274          SetWindowPos(HWND:00070510, Hwnd:00000000, int:288, int:274, int:0, int:0, SWP_FLAGS:00000015:SWP_NOSI
2445          ShowWindow(HWND:00070510, DWORD:00000001) returns BOOL:0
3119          PostThreadMessage(HWND:00070510, MSG:110691, WPARAM:0nnnnnnnn, LPARAM:0nnnnnnnn) returns BOOL:1

```

No clear info !
Let's dig deeper

No clear info !

Let's dig deeper

정확하지 않은 정보다!

깊게 파고들자.

Immunity Debugger Screenshot:

- Assembly pane:
 - Line 2197: `Dir(VARIANT String: "reginfo...", FLAGS: 00000000)`
 - Line 2214: `VariantString("reginfo...", 001504E4)`
 - Line 2218: `vbaVarTstEq(VARIANT String: "", VARIANT Const String: "")`
 - Line 2221: `Text3.Text <- "UNREGISTERED"`
- Registers pane: Shows CPU, Registers, Stack, and Flags.
- Memory dump pane (top right):
 - Pathname (variant): String .bstrVal = 001504E4 = "reginfo.key"
 - Integer attributes = 0x0000
- Bottom status bar: Results and Events buttons.

That's it! The ReverseMe is looking for a key file !

좋아! ReverseMe 는 key file 을 찾고 있다.

And if the key file is not found

그리고 만약에 key file 을 찾지 못했다면

... it puts UNREGISTERED in the main window but doesn't really bother what we type in the registration box!!!

UNREGISTERED 를 main window 에 넣는다. 정말 registration box 의 type 을 신경 쓰지 않는다.

```

2002   Form1_Load
2003     OnError(long 1)
2008     _vbaVarDup(VARIANT:String:"NuUega S...", VARIANT:Empty) returns DWORD:12FA34
2013     AppActivate(VARIANT:String:"NuUega S...", VARIANT:Missing) fails
2186     RtlUnwind(DWORD:0012FAE4, DWORD:74132085, DWORD:00000000, DWORD:00000000) returns DWORD:0
2187     _vbaFreeStrList() returns DWORD:0
2188     _vbaFreeObjList(unsigned int:00000000)
2189     SysFreeString(BSTR:0014EED4)
2192     _vbaVarDup(VARIANT:String:"reginfo...", VARIANT:Empty) returns DWORD:12FA34
2197   Dir(VARIANT:String:"reginfo...", FLAGS:00000000)
2198     WideCharToMultiByte(unsigned int:00000000, FLAGS:00000000, LPWSTR:001504E4, int:-1, PTR:0012F3C8, DV
2199     GetFullPathNameA(LPSTR:0012F3C8, DWORD:00000010, PTR:0012F60C, PTR:0012F384) returns DWORD:3
2200     WideCharToMultiByte(un
2201     FindFirstFileA(LPSTR:001504E4, PTR:0012F858, DV
2202     GetLastError() returns D
2203     GetLastError() returns DWORD:2
2204     SetLastError(DWORD:00000002)
2205     GetLastError() returns DWORD:2
2206     SetLastError(DWORD:00000002)
2207     SysAllocString(wchar_t *:7407152C) returns LPVOID:14DCF4
2211   Dir
2212     _vbaVarMove(VARIANT:String:"", VARIANT:Empty) returns DWORD:12FAA4
2213     _vbaFreeVar(VARIANT:String:"reginfo...")
2214     SysFreeString(BSTR:001504E4)
2217     _vbaFreeVar returns DWORD:20
2218     _vbaVarListEq(VARIANT:String:"", VARIANT:Const String:"") returns DWORD:0
2219     OLE_TextBox: AddRef(LPINTERFACE:0103CAC4) returns DWORD:1
2220     _vbaObjSet(LPINTERFACE *:0012FA48, LPINTERFACE:0103CAC4) returns LPVOID:103CAC4
2221     Text3.Text <= "UNREGISTERED" (String)
2245     _vbaFreeObj(LPINTERFACE *:0012FA48)
2248     _vbaExitProc() returns DWORD:0
2257     _vbaFreeVar(VARIANT:Empty) returns DWORD:0
2258     _vbaFreeVar(VARIANT:Empty) returns DWORD:0
2259     vbaFreeVar(VARIANT:Empty) returns DWORD:0

```

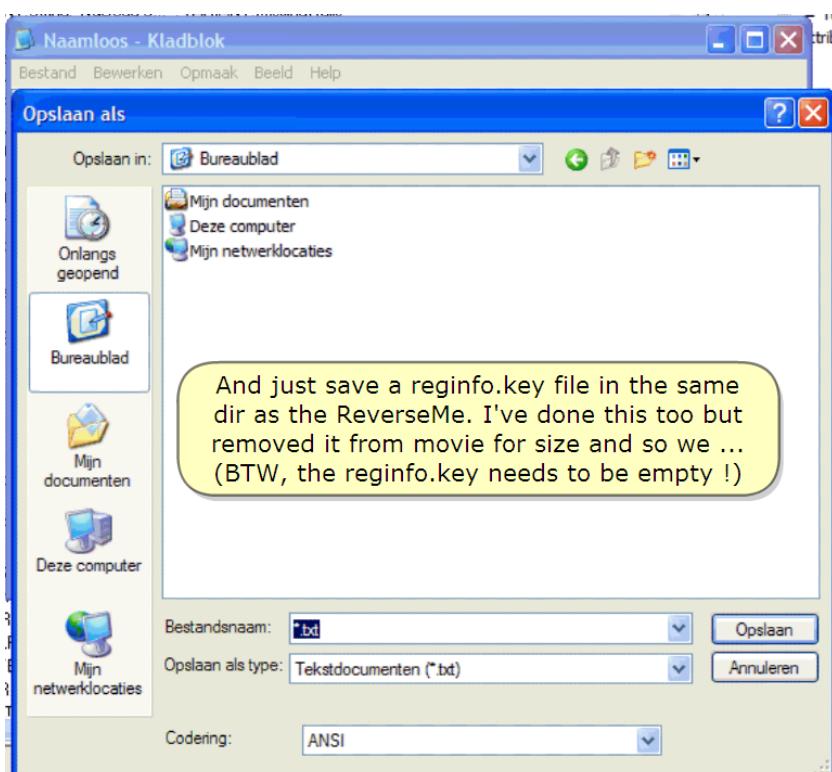
That means we need a reginfo.key, well, let's make one !!!

That means we need a reginfo.key, well, let's make one !!!

우리에게 reginfo.key 이 필요하다는 것을 뜻한다. 좋아. 하나 만들자 !!!

Meanwhile, I have opened Notepad (removed from movie for size)

동안, Notepad 를 열어놨다.(size 를 위해 movie 에서 삭제했다)



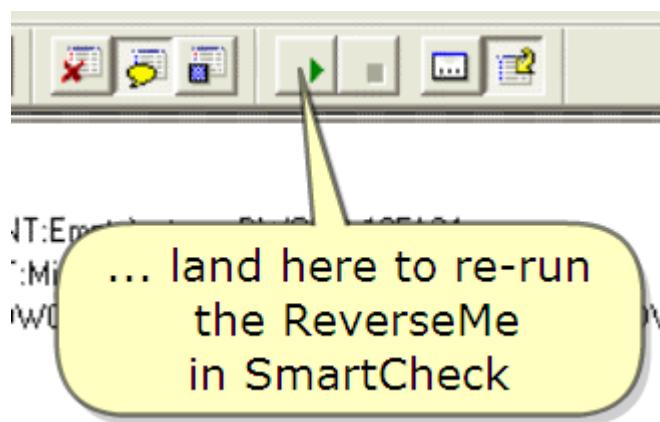
And just save a reginfo.key file in the same dir as the ReverseMe.

ReverseMe 일 때 Reginfo.key file 을 같은 dir 에 저장해라.

I've done this too but removed it from movie for size and so we...

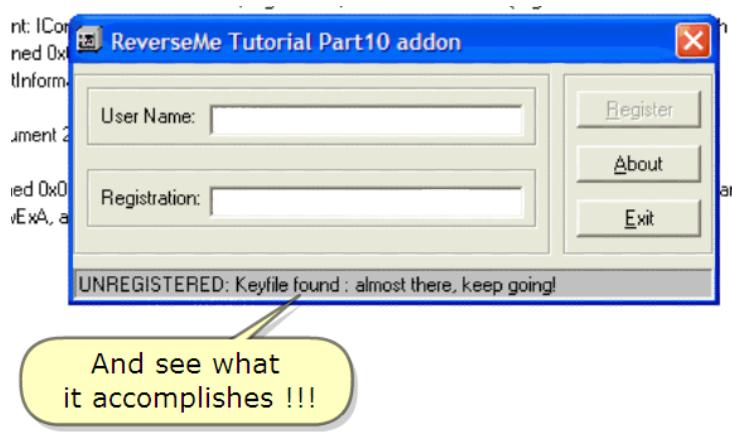
(BTW, the reginfo.key needs to be empty !)

나는 이것을 끝냈다. 그러나 movie size 를 줄이기 위해 삭제했다. 그리고 우리는
(By the way, reginfo.key 는 비어있어야 한다. !)

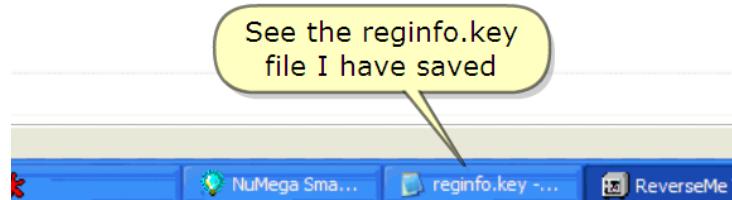


... land here to re-run the ReverseMe in SmartCheck

ReverseMe 를 재시작 하기 위해 도착했다.



And see what it accomplishes !!!



See the reginfo.key file I have saved

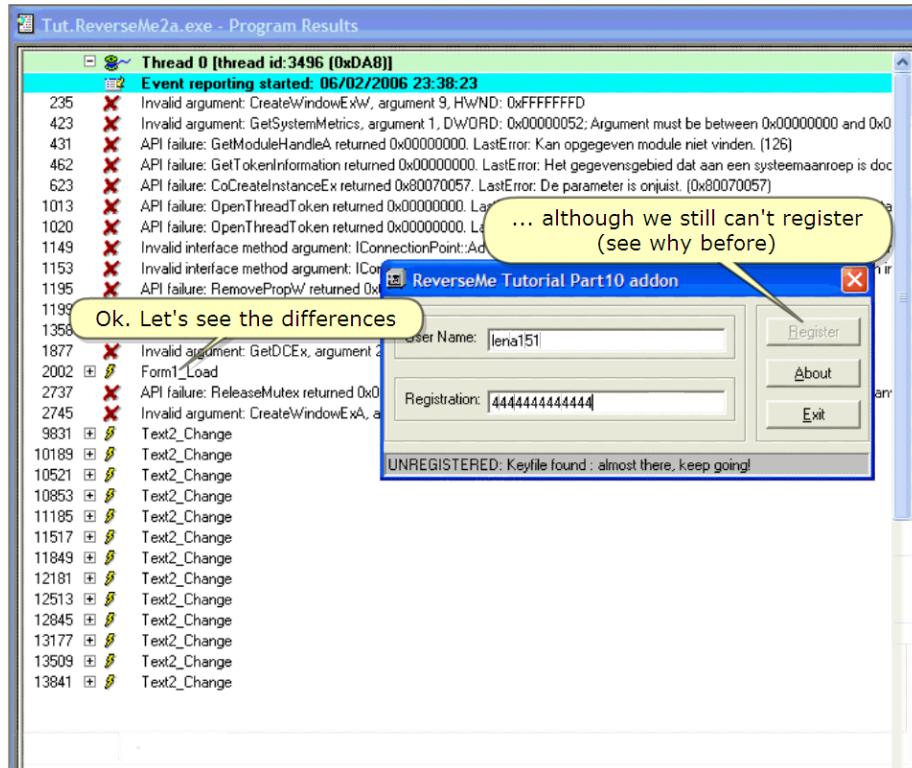
And see what it accomplishes !!!

저장했던 Reginfo.key file 을 봐.

어떤 것을 성취했는지 봐 !!!

Can we register now?

이제 우리는 등록할 수 있을까?



Ok. Let's see the differences

좋아. 다른 것들을 봐.

... although we still can't register (see why before)

우리는 여전히 등록할 수 없지만(왜 그런지는 전의 것을 참조 해)

Tut.ReverseMe2a.exe - Program Results

Thread 0 [thread id: 3496 (0xDA8)]

Event reporting started: 06/02/2006 23:38:23

```

235 ✗ Invalid argument: CreateWindowExW, argument 9, HwND: 0xFFFFFFFF
423 ✗ Invalid argument: GetSystemMetrics, argument 1, DWORD: 0x00000052; Argument must be between 0x00000000 and 0x0
431 ✗ API failure: GetModuleHandleA returned 0x00000000. LastError: Kan opgegeven module niet vinden. (126)
462 ✗ API failure: GetTokenInformation returned 0x00000000. LastError: Het gegevensgebied dat aan een systeemaanroep is doc
623 ✗ API failure: CoCreateInstanceEx returned 0x80070057. LastError: De parameter is onjuist. (0x80070057)
1013 ✗ API failure: OpenThreadToken returned 0x00000000. LastError: Er is geprobeerd te verwijzen naar een token dat niet besta
1020 ✗ API failure: OpenThreadToken returned 0x00000000. LastError: Er is geprobeerd te verwijzen naar een token dat niet besta
1149 ✗ Invalid interface method argument: IConnectionPoint::Advise, argument 2, PTR: 0x0012FB62. (Argument count starts with ir
1153 ✗ Invalid interface method argument: IConnectionPoint::Advise, argument 2, PTR: 0x0012FB62. (Argument count starts with ir
1195 ✗ API failure: RemovePropW returned 0x00000000
1199 ✗ Invalid argument: GetUserObjectInformationW, argument 5, PTR: 0x00000000; The NULL pointer is not valid
1358 ☐ Form1 [Form] created
1877 ✗ Invalid argument: GetDCE, argument 2, HRGN: 0x00000000
2002 ☐ Form1_Load
2003 ☐ OnError(long:1)eMutex returned 0x0
2013 ☐ AppActivate(VARIANT:String:"NuUega S...", VARIANT:Missing) fails
2200 ☐ Dir(VARIANT:String:"reginfo...", FLAGS:00000000)
2231 ☐ Open(String:"reginfo...", Integer:1, long:-1, long:1)
2242 ☐ EOF(Integer:1)
2245 ☐ Close(Integer:1)
2249 ☐ Len(VARIANT:Empty) returns LONG:1243700
2260 ☐ Text3.Text <- "UNREGISTERED: Keyfile found : almost there, keep going!" (String)
2311 ☐ Form1_Load
2327 ✗ API failure: ReleaseMutex returned 0x00000000. LastError: Poging tot loslaten van mutex die geen eigendom is van de aan
2745 ✗ Invalid argument: CreateWindowExA, argument 9, HwND: 0xFFFFFFFF
9831 ☐ Text2_Change
10189 ☐ Text2_Change
10521 ☐ Text2_Change
10853 ☐ Text2_Change
11185 ☐ Text2_Change
11517 ☐ Text2_Change
11849 ☐ Text2_Change
12181 ☐ Text2_Change

```

For Help, press F1 RUN Program Events: 16128

:)

Tut.ReverseMe2a.exe - Program Results

Thread 0 [thread id: 3496 (0xDA8)]

Event reporting started: 06/02/2006 23:38:23

```

235 ✗ Invalid argument: CreateWindowExW, argument 9, HwND: 0xFFFFFFFF
423 ✗ Invalid argument: GetSystemMetrics, argument 1, DWORD: 0x00000052; Argument must be between 0x00000000 and 0x0
431 ✗ API failure: GetModuleHandleA returned 0x00000000. LastError: Kan opgegeven module niet vinden. (126)
462 ✗ API failure: GetTokenInformation returned 0x00000000. LastError: Het gegevensgebied dat aan een systeemaanroep is doc
623 ✗ API failure: CoCreateInstanceEx returned 0x80070057. LastError: De parameter is onjuist. (0x80070057)
1013 ✗ API failure: OpenThreadToken returned 0x00000000. LastError: Er is geprobeerd te verwijzen naar een token dat niet besta
1020 ✗ API failure: OpenThreadToken returned 0x00000000. LastError: Er is geprobeerd te verwijzen naar een token dat niet besta
1149 ✗ Invalid interface method argument: IConnectionPoint::Advise, argument 2, PTR: 0x0012FB62. (Argument count starts with ir
1153 ✗ Invalid interface method argument: IConnectionPoint::Advise, argument 2, PTR: 0x0012FB62. (Argument count starts with ir
1195 ✗ API failure: RemovePropW returned 0x00000000
1199 ✗ Invalid argument: GetUserObjectInformationW, argument 5, PTR: 0x00000000; The NULL pointer is not valid
1358 ☐ Form1 [Form] created
1877 ✗ Invalid argument: GetDCE, argument 2, HRGN: 0x00000000
2002 ☐ Form1_Load
2003 ☐ OnError(long:1)eMutex returned 0x0
2013 ☐ AppActivate(VARIANT:String:"NuUega S...", VARIANT:Missing) fails
2200 ☐ Dir(VARIANT:String:"reginfo...", FLAGS:00000000)
2231 ☐ Open(String:"reginfo...", Integer:1, long:-1, long:1)
2242 ☐ EOF(Integer:1)
2245 ☐ Close(Integer:1)
2249 ☐ Len(VARIANT:Empty) returns LONG:1243700
2260 ☐ Text3.Text <- "UNREGISTERED: Keyfile found : almost there, keep going!" (String)
2311 ☐ Form1_Load
2327 ✗ API failure: ReleaseMutex returned 0x00000000. LastError: Poging tot loslaten van mutex die geen eigendom is van de aan
2745 ✗ Invalid argument: CreateWindowExA, argument 9, HwND: 0xFFFFFFFF
9831 ☐ Text2_Change
10189 ☐ Text2_Change
10521 ☐ Text2_Change
10853 ☐ Text2_Change
11185 ☐ Text2_Change
11517 ☐ Text2_Change
11849 ☐ Text2_Change
12181 ☐ Text2_Change

```

It seems we did well !!

우리는 좋게 해낸걸 봐 !!

Tut.ReverseMe2a.exe - Program Results

Thread 0 [thread id:3496 (0xDA8)]

Event reporting started: 06/02/2006 23:38:23

```

235 ✗ Invalid argument: CreateWindowExW, argument 9, HwND: 0xFFFFFFF
423 ✗ Invalid argument: GetSystemMetrics, argument 1, DWORD: 0x00000052; Argument must be between 0x00000000 and 0x0
431 ✗ API failure: GetModuleHandleA returned 0x00000000. LastError: Kan opgegeven module niet vinden. (126)
462 ✗ API failure: GetTokenInformation returned 0x00000000. LastError: Het gegevensgebied dat aan een systeemaanroep is doc
623 ✗ API failure: CoCreateInstanceEx returned 0x80070057. LastError: De parameter is onjuist. (0x80070057)
1013 ✗ API failure: OpenThreadToken returned 0x00000000. LastError: Er is geprobeerd te verwijzen naar een token dat niet besta
1020 ✗ API failure: OpenThreadToken returned 0x00000000. LastError: Er is geprobeerd te verwijzen naar een token dat niet besta
1149 ✗ Invalid interface method argument: IConnectionPoint::Advise, argument 2, PTR: 0x0012FB62. (Argument count starts with ir
1153 ✗ Invalid interface method argument: IConnectionPoint::Advise, argument 2, PTR: 0x0012FAF6. (Argument count starts with ir
1195 ✗ API failure: RemovePropW returned 0x00000000
1199 ✗ Invalid argument: GetUserObjectInformationW, argument 5, PTR: 0x00000000; The NULL pointer is not valid
1358 Form1 [Form] created
1877 ✗ Invalid argument: GetDCEx, argument 2, HRGN: 0x00000000
2002 Form1_Load
2003 ✗ DnError(long:1)@Mutex returned 0x0
2013 ✗ AppActivate(VARIANT:String:"NuUega S...", VARIANT:Missing) fails
2200 ✗ D:\Dir(VARIANT:String:"reginfo...", FLAGS:00000000)
2231 ✗ Open(String:"reginfo...", Integer:1, long:-1, long:1)
2242 ✗ EOF(Integer:1)
2245 ✗ Close(Integer:1)
2249 ✗ Len(VARIANT:Empty) returns LONG:1243700
2260 Text3.Text <- "UNREGISTERED: Keyfile found : almost there, keep going!" (String)

2311 Form1_Load@0
12737 ✗ API failure: ReleaseMutex, argument 1, PTR: 0x00000000; Failing tot loslaten van mutex die geen eigendom is van de aan
12745 ✗ Invalid argument: CreateFileA, argument 1, PTR: 0x00000000; FFFFFFFD
19831 Text2_Change
10188 Text2_Change
10521 Text2_Change
10853 Text2_Change
11185 Text2_Change
11517 Text2_Change
11849 Text2_Change
12181 Text2_Change

```

Let's now see
in the verification
of the serial

Let's now see in the verification of the serial

Serial 을 검증하는 것을 봐.

Scroll down

Scroll 내려.

See the included "The usage of SmartCheck.html" file for more info on the next texts

첨부된 "The usage of SmartCheck.html" file 을 다음 text에서 많은 정보를 얻기 위해서 봐.

Tut.ReverseMe2a.exe - Program Results

```

2200 Dir(VARIANT:String:"reginfo...", FLAGS:00000000)
2231 Open(String:"reginfo...", Integer:1, long:-1, long:1)
2242 EOF(Integer:1)
2245 Close(Integer:1)
2249 Len(VARIANT:Empty) returns LONG:1243700
2260 Text3.Text <- "UNREGISTERED: Keyfile found : almost there, keep going!!" (String)

2311 Form1_Load
2737 API failure: ReleaseMutex()
2745 Invalid argument: CreateFile()
9831 Text2_Change
9834 Text1.Text
9845 Left(VARIANT:String:"lena151", long:1)
9849 Asc(String:"l") returns Integer:108
9865 Text1.Text
9876 Mid(VARIANT:String:"lena151", long:2, VARIANT:Integer:1)
9880 Asc(String:"e") returns Integer:101
9896 Text1.Text
9907 Mid(VARIANT:String:"lena151", long:3, VARIANT:Integer:1)
9911 Asc(String:"n") returns Integer:110
9927 Text1.Text
9938 Mid(VARIANT:String:"lena151", long:4, VARIANT:Integer:1)
9942 Asc(String:"a") returns Integer:97
9958 Text1.Text
9969 Mid(VARIANT:String:"lena151", long:5, VARIANT:Integer:1)
9973 Asc(String:"i") returns Integer:49
10043 Mid(VARIANT:String:"10810111...", long:2, VARIANT:Integer:10)
10054 Text2.Text
10112 Text2_Change
10189 Text2_Change
10521 Text2_Change
10853 Text2_Change
11185 Text2_Change
11517 Text2_Change
11849 Text2_Change
12181 Text2_Change

```

The diagram illustrates the process of extracting characters from the string 'lena151'. It shows four main steps:

- Grab the first char from the name**: Extracts the first character 'l' at index 1.
- and convert its value in ascii**: Converts the ASCII value of 'l' (108) to its decimal representation (108).
- Grab the 2nd char from the name**: Extracts the second character 'e' at index 2.
- and convert its value in ascii**: Converts the ASCII value of 'e' (101) to its decimal representation (101).

Below these, a note states: **Mmmmm, but here the chars are joined together already. So, we are missing a couple of steps. Let's see it in detail !!!**

Grab the first char from the name

Name에서 첫번째 문자를 잡는다.

And convert its value in ascii

그리고 ascii에서 dec 값으로 변환한다.

Grab the 2nd char from the name

Name에서 2 번째 문자를 잡는다.

And convert its value in ascii

그리고 ascii에서 dec 값으로 변환한다.

And do the same for the 3rd, 4th and the fifth until

그리고 3 번째, 4 번째, 5 번째 마지막까지 같은 방법을 사용한다.

Mmmmm, but here the chars are joined together already. So, we are missing a couple of steps.

Let's see it in detail !!!

음, 그러나 여기 문자는 이미 같이 합병됐다. 그래서, 몇 가지가 누락됐다. 자세하게 보자.

Tut.ReverseMe2a.exe - Program Results

```

2200     Dir(VARIANT:String:"reginfo...", FLAGS:00000000)
2231     Open(String:"reginfo...", Integer:1, long:-1, long:1)
2242     EOF(Integer:1)
2249     Close(Integer:1)
2260     Len(VARIANT:Empty) returns LONG:1243700
2311     Text3.Text <- "UNREGISTERED: Keyfile found : almost there, keep going!" (String)
2737     Form1_Load
2745     API failure: ReleaseMutex returned 0x00000000. LastError: Poging tot loslaten van mutex die geen eigendom is van de aan
2745     Invalid argument: CreateWindowExA, argument 9, Hwnd: 0xFFFFFFFF
9831     Text2_Change
9834     Text1.Text
9845     Left(VARIANT:String:"lena151", long:1)
9849     Asc(String:"I") returns Integer:108
9865     Text1.Text
9876     Mid(VARIANT:String:"lena151", long:2, VARIANT:Integer:1)
9880     Asc(String:"e") returns Integer:101
9896     Text1.Text
9907     Mid(VARIANT:String:"lena151", long:3, VARIANT:Integer:1)
9911     Asc(String:"n") returns Integer:110
9927     Text1.Text
9938     Mid(VARIANT:String:"lena151", long:4, VARIANT:Integer:1)
9942     Asc(String:"a") returns Integer:97
9958     Text1.Text
9969     Mid(VARIANT:String:"lena151", long:5, VARIANT:Integer:1)
9973     Asc(String:"I") returns Integer:49
10043     Mid(VARIANT:String:"10810111...", long:2, VARIANT:Integer:10)
10054     Text2.Text
10112     Text2_Change
10188     Text2_Change
10521     Text2_Change
10853     Text2_Change
11185     Text2_Change
11517     Text2_Change
11849     Text2_Change
12181     Text2_Change

```

Remember that we will need to look before and after the highlighted line

Remember that we will need to look before and after the highlighted line

기억해. 우리는 highlight 된 line 을 보기 전에 필요하다.

Tut.ReverseMe2a.exe - Program Results

```

9973     Asc(String:"I") returns Integer:49
9976     _vbaVarMove(VARIANT:Integer:49, VARIANT:Empty) returns DWORD:12F43C
9977     _vbaFreeStr(LPCTSTR:0012F434) returns DWORD:0
9978     _vbaFreeObj(LPINTERFACE:<0012F430)
9981     SysFreeString(BSTR:001505DC)
9984     SysFreeString(BSTR:0014DD44)
9987     _vbaVarCall(VARIANT:Integer:108, VARIANT:Integer:101) returns DWORD:12F420
10002     _vbaVarCall(VARIANT:String:"108101", VARIANT:Integer:110) returns DWORD:12F410
10011     _vbaVarCall(VARIANT:String:"10810111...", VARIANT:Integer:97) returns DWORD:12F400
10022     _vbaVarCall(VARIANT:String:"10810111...", VARIANT:Integer:49) returns DWORD:12F3F0
10033     _vbaVarMove(VARIANT:String:"10810111...", VARIANT:Empty) returns DWORD:12F48C
10034     SysFreeString(BSTR:0015067C)
10037     SysFreeString(BSTR:001505DC)
10040     SysFreeString(BSTR:00154304)
10043     Mid(VARIANT:String:"10810111...", long:2, VARIANT:Integer:10)
10046     _vbaVarMove(VARIANT:String:"08101110...", VARIANT:String:"10810111...") returns DWORD:12F48C
10051     _vbaFreeVar(VARIANT:Integer:10) returns DWORD:12F48C
10052     _OLE_TextBox:AllocRef(LPINTERFACE:0012F430, LPINTERFACE:0103CE2C) returns DWORD:2
10053     _vbaObjSet(LPINTERFACE:<0012F430, LPINTERFACE:0103CE2C) returns LPVOID:103CE2C
10054     Text2.Text
10067     _vbaVarSub(VARIANT:String:"4", VARIANT:String:"08101110...") returns DWORD:12F410
10093     _vbaVarSetEq(VARIANT:Double:-8.10111e+008, VARIANT:Const Integer:0) returns DWORD:0
10094     _vbaFreeObj(LPINTERFACE:<0012F430)
10097     _vbaFreeVar(VARIANT:String:"4") returns DWORD:20
10102     _vbaFreeVar(VARIANT:String:"08101110...") returns DWORD:20
10107     _vbaFreeVar(VARIANT:Integer:108) returns DWORD:20
10108     _vbaFreeVar(VARIANT:Integer:101) returns DWORD:20
10109     _vbaFreeVar(VARIANT:Integer:110) returns DWORD:20
10110     _vbaFreeVar(VARIANT:Integer:97) returns DWORD:20
10111     _vbaFreeVar(VARIANT:Integer:49) returns DWORD:20
10112     Text2_Change
10113     _OLE_TextBox:Release(LPINTERFACE:0103CE2C) returns DWORD:0
10114     GetFocus() returns Hwnd:1E040E
10115     SelectObject(HGDIOBJ:DA010EA9, HGDIOBJ:2C0A0BA4) returns HGDIOBJ:18A0021
10116     GetTextMetricsA(HGDIOBJ:DA010EA9, PTR:0012FC84) returns BOOL:L1

```

Study the code ...

Study the code ...

Code 를 공부해 ...

We saw the last char converted to ascii
...and indeed, here the ascii values are joined

```

9973 | [+] Asc(String:"1") returns Integer:49
9976 |   [+] _vbaVarMove(VARIANT:Integer:49, VARIANT:Empty) returns DWORD:12F43C
9977 |   [+] _vbaFreeStr(LPCTSTR:0012F434) returns DWORD:0
9978 |   [+] _vbaFreeObj(LPINTERFACE :0012F430)
9981 |   [+] SysFreeString(BSTR:001505DC)
9984 |   [+] SysFreeString(BSTR:0014DD44)
9987 |   [+] _vbaVarCat(VARIANT:Integer:108, VARIANT:Integer:101) returns DWORD:12F420
10002 |   [+] _vbaVarCat(VARIANT:String:"108101", VARIANT:Integer:110) returns DWORD:12F410
10011 |   [+] _vbaVarCat(VARIANT:String:"10810111...", VARIANT:Integer:97) returns DWORD:12F400
10022 |   [+] _vbaVarCat(VARIANT:String:"10810111...", VARIANT:Integer:49) returns DWORD:12F3F0
10033 |   [+] _vbaVarMove(VARIANT:String:"10810111...", VARIANT:Empty) returns DWORD:12F48C
10034 |   [+] SysFreeString(BSTR:0015067C)
10037 |   [+] SysFreeString(BSTR:001505DC)
10040 |   [+] SysFreeString(BSTR:00154304)
10043 |   [+] Mid(VARIANT:String:"10810111...", long:2, VARIANT:Integer:10)
10046 |   [+] _vbaVarMove(VARIANT:String:"08101110...", VARIANT:String:"10810111...") returns DWORD:12F48C
10051 |   [+] _vbaFreeVar(VARIANT:Integer:10) returns DWORD:12F48C
10052 |   [+] OLE _TextBox::AddRef(LPINTERFACE:0103CE2C) returns DWORD:2
10053 |   [+] _vbaObjSet(LPINTERFACE :0012F430, LPINTERFACE:0103CE2C) returns LPVOID:103CE2C
10054 | [+] Text2.Text
10067 |   [+] _vbaVarSub(VARIANT:String:"4", VARIANT:String:"08101110...") returns DWORD:12F410
10093 |   [+] _vbaVarTstEq(VARIANT:Double:-8.10111e+008, VARIANT:Const Integer:0) returns DWORD:0
10094 |   [+] _vbaFreeObj(LPINTERFACE :0012F430)
10097 |   [+] _vbaFreeVar(VARIANT:String:"4") returns DWORD:20
10102 |   [+] _vbaFreeVar(VARIANT:String:"08101110...") returns DWORD:20
10107 |   [+] _vbaFreeVar(VARIANT:Integer:108) returns DWORD:20
10108 |   [+] _vbaFreeVar(VARIANT:Integer:101) returns DWORD:20
10109 |   [+] _vbaFreeVar(VARIANT:Integer:110) returns DWORD:20
10110 |   [+] _vbaFreeVar(VARIANT:Integer:97) returns DWORD:20
10111 |   [+] _vbaFreeVar(VARIANT:Integer:49) returns DWORD:20
10112 |   [+] Text2_Change
10113 |   [+] OLE _TextBox::Release(LPINTERFACE:0103CE2C) returns DWORD:0
10114 |   [+] GetFocus() returns HWND:1E040E
10115 |   [+] SelectObject(HGDIOBJ:DA010EA9, HGDIOBJ:2C0A0BA4) returns HGDIOBJ:18A0021
10116 |   [+] GetTextMetricsA(HGDIOBJ:DA010EA9, PTR:0012FC84) returns BOOL:1

```

We saw the last char converted to ascii

...and indeed, here the ascii values are joined

우리는 ascii 로 마지막 바뀐 마지막 문자를 봤다.

정말, ascii 값은 합병됐다.

Oh, but this looks important : we grab the digits from #2 to #10

```

9973 | [+] Asc(String:"1") returns Integer:49
9976 |   [+] _vbaVarMove(VARIANT:Integer:49, VARIANT:Empty) returns DWORD:12F43C
9977 |   [+] _vbaFreeStr(LPCTSTR:0012F434) returns DWORD:0
9978 |   [+] _vbaFreeObj(LPINTERFACE :0012F430)
9981 |   [+] SysFreeString(BSTR:001505DC)
9984 |   [+] SysFreeString(BSTR:0014DD44)
9987 |   [+] _vbaVarCat(VARIANT:Integer:108, VARIANT:Integer:101) returns DWORD:12F420
10002 |   [+] _vbaVarCat(VARIANT:String:"108101", VARIANT:Integer:110) returns DWORD:12F410
10011 |   [+] _vbaVarCat(VARIANT:String:"10810111...", VARIANT:Integer:97) returns DWORD:12F400
10022 |   [+] _vbaVarCat(VARIANT:String:"10810111...", VARIANT:Integer:49) returns DWORD:12F3F0
10033 |   [+] _vbaVarMove(VARIANT:String:"10810111...", VARIANT:Empty) returns DWORD:12F48C
10034 |   [+] SysFreeString(BSTR:0015067C)
10037 |   [+] SysFreeString(BSTR:00154304)
10040 |   [+] Mid(VARIANT:String:"10810111...", long:2, VARIANT:Integer:10)
10043 |   [+] _vbaVarMove(VARIANT:String:"08101110...", VARIANT:String:"10810111...") returns DWORD:12F48C
10046 |   [+] _vbaFreeVar(VARIANT:Integer:10) returns DWORD:12F48C
10051 |   [+] OLE _TextBox::AddRef(LPINTERFACE:0103CE2C) returns DWORD:2
10052 |   [+] _vbaObjSet(LPINTERFACE :0012F430, LPINTERFACE:0103CE2C) returns LPVOID:103CE2C
10054 | [+] Text2.Text
10067 |   [+] _vbaVarSub(VARIANT:String:"4", VARIANT:String:"08101110...") returns DWORD:12F410
10093 |   [+] _vbaVarTstEq(VARIANT:Double:-8.10111e+008, VARIANT:Const Integer:0) returns DWORD:0
10094 |   [+] _vbaFreeObj(LPINTERFACE :0012F430)
10097 |   [+] _vbaFreeVar(VARIANT:String:"4") returns DWORD:20
10102 |   [+] _vbaFreeVar(VARIANT:String:"08101110...") returns DWORD:20
10107 |   [+] _vbaFreeVar(VARIANT:Integer:108) returns DWORD:20
10108 |   [+] _vbaFreeVar(VARIANT:Integer:101) returns DWORD:20
10109 |   [+] _vbaFreeVar(VARIANT:Integer:110) returns DWORD:20
10110 |   [+] _vbaFreeVar(VARIANT:Integer:97) returns DWORD:20
10111 |   [+] _vbaFreeVar(VARIANT:Integer:49) returns DWORD:20
10112 |   [+] Text2_Change
10113 |   [+] OLE _TextBox::Release(LPINTERFACE:0103CE2C) returns DWORD:0
10114 |   [+] GetFocus() returns HWND:1E040E
10115 |   [+] SelectObject(HGDIOBJ:DA010EA9, HGDIOBJ:2C0A0BA4) returns HGDIOBJ:18A0021
10116 |   [+] GetTextMetricsA(HGDIOBJ:DA010EA9, PTR:0012FC84) returns BOOL:1

```

Oh, but this looks important : we grab the digits from #2 to #10

이것은 중요하다 : 우리는 2에서 10까지 숫자를 잡았다.

The screenshot shows the assembly code for the program. Several lines of code are annotated with yellow callouts:

- Line 10043: `Mid(VARIANT String:"10810111...", Integer:10)` has a callout saying "but then ... some dark subtraction".
- Line 10046: `some unimportant stuff` is annotated.
- Line 10051: `some unimportant stuff` is annotated.
- Line 10052: `some unimportant stuff` is annotated.
- Line 10053: `some unimportant stuff` is annotated.
- Line 10054: `Text2.Text` is annotated.
- Line 10067: `Mmmm, we need to see that more in detail. Expand the tree !!!` is annotated.

Some unimportant stuff

중요하지 않은 재료다.

But then ... some "dark" subtraction

약간의 "dark" 빼기다.

After which the test is done

여기에서 Test 를 끝냈다.

Mmmm, we need to see that more in detail. Expand the tree !!!

음, 우리는 자세하게 보기 위해 tree 를 확장하는 것이 필요하다. !!!

All right, let's see this.

Aha, but the subtraction is not used for the compare (Test)

```

10046  _vbaVarMove(VARIANT:String:"08101110...", VARIANT:String:"10810111...") returns DWORD:12F48C
10051  _vbaFreeVar(VARIANT:Integer:10) returns DWORD:12F48C
10052  _OLE_TextBox>AddRef(LPINTERFACE:0103CE2C) returns DWORD:2
10053  _vbaObjSet(LPINTERFACE :0012F430, LPINTERFACE:0103CE2C) returns LPVOID:103CE2C
10054  Text2.Text
10055  _vbaVarSub(VARIANT:String:"4", VARIANT:String:"08101110...") 
    VariantChangeTypeEx(VARIANT:Empty, VARIANT:String:"4", LCID:00000400, FLAGS:00000002, VARTYPE:1
    LoadLibraryW(LPTR:0012F04C) returns HINSTANCE:7C800000
    GetProcAddress(HINST:7C800000, LPSTR:770F2584) returns FARPROC:7C839230
    GetProcessHeap() returns HANDLE:140000
    HeapAlloc(HANDLE:00140000, FLAGS:00000008, DWORD:00000064) returns LPVOID:151F18
    GetLocaleInfoW(LCID:00000813, DWORD:00000014, PTR:00151F30, DWORD:00000001, PTR:0012F138, C
    LCMMapStringW(LCID:00000813, FLAGS:00400100, PTR:00151F30, DWORD:00000001, PTR:0012F124, C
    LCMMapStringW(LCID:00000813, FLAGS:00800200, PTR:00151F30, DWORD:00000001, PTR:0012F124, C
    GetLocaleInfoW(LCID:00000813, DWORD:0000000E, PTR:0012F14C, DWORD:00000004) returns INT:2
    GetLocaleInfoW(LCID:00000813, DWORD:00000016, PTR:0012F14C, DWORD:00000004) returns INT:2
    GetLocaleInfoW(LCID:00000813, DWORD:00000051, PTR:0012F14C, DWORD:00000004) returns INT:2
    GetLocaleInfoW(LCID:00000813, DWORD:00000017, PTR:0012F14C, DWORD:00000004) returns INT:2
    GetLocaleInfoW(LCID:00000813, DWORD:0000000F, PTR:0012F14C, DWORD:00000004) returns INT:2
    GetStringExW(LCID:00000813, FLAGS:00000001, LPWSTR:0012F0F0, DWORD:FFFFFFF, PTR:001
    GetLocaleInfoA(LCID:00000813, DWORD:00000012, PTR:0012F120, DWORD:00000002) returns INT:2
    GetLocaleInfoA(LCID:00000813, DWORD:00000010, PTR:0012F14C, DWORD:00000004) returns INT:4
    GetLocaleInfoA(LCID:00000813, DWORD:00000018, PTR:0012F14C, DWORD:00000004) returns INT:4
    GetLocaleInfoA(LCID:00000813, DWORD:00000011, PTR:0012F14C, DWORD:00000004) returns INT:2
    GetLocaleInfoA(LCID:00000813, DWORD:00000013, DWORD:00000001, PTR:0012F14C, DWORD:00000004) returns INT:2
    GetLocaleInfoA(LCID:00000813, DWORD:00000015, PTR:0012F14C, DWORD:00000004) returns INT:2
    VariantChangeTypeEx returns HRESULT:0
    VariantChangeTypeEx(VARIANT:Empty, VARIANT:String:"08101110...", LCID:00000400, FLAGS:00000002, V
    _vbaVarSub returns DWORD:12F410
10093  _vbaVarIntEq(VARIANT:Double:-8.10111e+008, VARIANT:Const Integer:0) returns DWORD:0
10094  _vbaFreeObj(LPINTERFACE :0012F430)
10097  _vbaFreeVar(VARIANT:String:"4") returns DWORD:20
10102  _vbaFreeVar(VARIANT:String:"08101110...") returns DWORD:20

```

All right, let's see this.

좋아, 이것을 봐.

Aha, but the subtraction is not used for the compare (Test)

아하, 뺄셈은 compare 를 위해서 사용되지 않는다 (Test)

But here !!! The real serial is only changed
for the compare into the double dblval

```

10046  _vbaVarMove(VARIANT:String:"08101110...", VARIANT:String:"10810111...") returns DWORD:12F48C
10051  _vbaFreeVar(VARIANT:Integer:10) returns DWORD:12F48C
10052  _OLE_TextBox>AddRef(LPINTERFACE:0103CE2C) returns DWORD:2
10053  _vbaObjSet(LPINTERFACE :0012F430, LPINTERFACE:0103CE2C) returns LPVOID:103CE2C
10054  Text2.Text
10055  _vbaVarSub(VARIANT:String:"4", VARIANT:String:"08101110...") 
    VariantChangeTypeEx(VARIANT:Empty, VARIANT:String:"4", LCID:00000400, FLAGS:00000002, VARTYPE:1
    LoadLibraryW(LPTR:0012F04C) returns HINSTANCE:7C800000
    GetProcAddress(HINST:7C800000, LPSTR:770F2584) returns FARPROC:7C839230
    GetProcessHeap() returns HANDLE:140000
    HeapAlloc(HANDLE:00140000, FLAGS:00000008, DWORD:00000064) returns LPVOID:151F18
    GetLocaleInfoW(LCID:00000813, DWORD:00000014, PTR:00151F30, DWORD:00000001, PTR:0012F138, C
    LCMMapStringW(LCID:00000813, FLAGS:00400100, PTR:00151F30, DWORD:00000001, PTR:0012F124, C
    LCMMapStringW(LCID:00000813, FLAGS:00800200, PTR:00151F30, DWORD:00000001, PTR:0012F124, C
    GetLocaleInfoW(LCID:00000813, DWORD:0000000E, PTR:0012F14C, DWORD:00000004) returns INT:2
    GetLocaleInfoW(LCID:00000813, DWORD:00000016, PTR:0012F14C, DWORD:00000004) returns INT:2
    GetLocaleInfoW(LCID:00000813, DWORD:00000051, PTR:0012F14C, DWORD:00000004) returns INT:2
    GetLocaleInfoW(LCID:00000813, DWORD:00000017, PTR:0012F14C, DWORD:00000004) returns INT:2
    GetLocaleInfoW(LCID:00000813, DWORD:0000000F, PTR:0012F14C, DWORD:00000004) returns INT:2
    GetStringExW(LCID:00000813, FLAGS:00000001, LPWSTR:0012F0F0, DWORD:FFFFFFF, PTR:001
    GetLocaleInfoA(LCID:00000813, DWORD:00000012, PTR:0012F120, DWORD:00000002) returns INT:2
    GetLocaleInfoA(LCID:00000813, DWORD:00000010, PTR:0012F14C, DWORD:00000004) returns INT:4
    GetLocaleInfoA(LCID:00000813, DWORD:00000018, PTR:0012F14C, DWORD:00000004) returns INT:4
    GetLocaleInfoA(LCID:00000813, DWORD:00000011, PTR:0012F14C, DWORD:00000004) returns INT:2
    GetLocaleInfoA(LCID:00000813, DWORD:00000013, DWORD:00000001, PTR:0012F14C, DWORD:00000004) returns INT:2
    GetLocaleInfoA(LCID:00000813, DWORD:00000015, PTR:0012F14C, DWORD:00000004) returns INT:2
    VariantChangeTypeEx returns HRESULT:0
    VariantChangeTypeEx(VARIANT:Empty, VARIANT:String:"08101110...", LCID:00000400, FLAGS:00000002, V
    _vbaVarSub returns DWORD:12F410
10093  _vbaVarIntEq(VARIANT:Double:-8.10111e+008, VARIANT:Const Integer:0) returns DWORD:0
10094  _vbaFreeObj(LPINTERFACE :0012F430)
10097  _vbaFreeVar(VARIANT:String:"4") returns DWORD:20
10102  _vbaFreeVar(VARIANT:String:"08101110...") returns DWORD:20

```

Tut.ReverseMe2a.exe - Program Results

```

10046  + vbaVarMove(VARIANT:String:"08101110...", VARIANT:String:"08101110...") returns DWORD:12F48C
10051  + vbaFreeVar(VARIANT:Integer:10) returns DWORD:12F48C
10052  + OLE _TextBox: AddRef(LPINTERFACE:0103CE2C) returns DWORD:2
10053  + vbaObjSet(LPINTERFACE:0012F430, LPINTERFACE:0103CE2C) returns LPVOID:103CE2C
10054  + Text2.Text
10067  + vbaVarSub(VARIANT:String:"4", VARIANT:String:"08101110...") 
    |   VariantChangeTypeEx(VARIANT:Empty, VARIANT:String:"4", LCID:00000400, FLAGS:00000002, VARTYPE:1)
    |       LoadLibraryA(LPSTR:0012F04C) returns HINSTANCE:7C800000
    |       GetProcAddress(HINST:7C800000, LPSTR:770F2584) returns FARPROC:7C839230
    |       GetProcAddress() returns HANDLE:140000
    |       HeapAlloc(HANDLE:00140000, FLAGS:00000008, DWORD:00000064) returns LPVOID:151F18
    |       GetLocaleInfoW(LCID:00000813, DWORD:00000014, PTR:00151F30, DWORD:00000004) returns INT:2
    |       LCMMapStringW(LCID:00000813, FLAGS:00400100, PTR:00151F30, DWORD:00000001, PTR:0012F138, C)
    |       LCMMapStringW(LCID:00000813, FLAGS:00800200, PTR:00151F30, DWORD:00000001, PTR:0012F124, C)
    |       GetLocaleInfoW(LCID:00000813, DWORD:0000000E, PTR:0012F14C, DWORD:00000004) returns INT:2
    |       GetLocaleInfoW(LCID:00000813, DWORD:00000016, PTR:0012F14C, DWORD:00000004) returns INT:2
    |       GetLocaleInfoW(LCID:00000813, DWWORD:00000051, PTR:0012F14C, DWWORD:00000004) returns INT:2
    |       GetLocaleInfoW(LCID:00000813, DWWORD:00000017, PTR:0012F14C, DWWORD:00000004) returns INT:2
    |       GetLocaleInfoW(LCID:00000813, DWWORD:0000000F, PTR:0012F14C, DWWORD:00000004) returns INT:2
    |       GetStringTypeExW(LCID:00000813, FLAGS:00000001, LPWSTR:0012F0F0, DWWORD:FFFFFF, PTR:001
    |       GetLocaleInfoA(LCID:00000813, DWORD:00000012, PTR:0012F120, DWWORD:00000002) returns INT:2
    |       GetLocaleInfoW(LCID:00000813, DWWORD:00000010, PTR:0012F14C, DWWORD:00000004) returns INT:4
    |       GetLocaleInfoW(LCID:00000813, DWWORD:00000018, PTR:0012F14C, DWWORD:00000004) returns INT:4
    |       GetLocaleInfoW(LCID:00000813, DWWORD:00000011, PTR:0012F14C, DWWORD:00000004) returns INT:2
    |       like our serial is changed too just before
    |       But here !!! The real serial is only changed for the compare into the double dblval
    |       GetLocaleInfoW(LCID:00000813, DWWORD:00000013, PTR:0012F14C, DWWORD:00000004) returns INT:2
    |       GetLocaleInfoW(LCID:00000813, DWWORD:0000001B, PTR:0012F14C, DWWORD:00000004) returns INT:2
    |       VariantChangeTypeEx returns HRESULT:0
    |       VariantChangeTypeEx(VARIANT:Empty, VARIANT:String:"08101110...", LCID:00000400, FLAGS:00000002, V
    |       vbaVarSub returns DWORD:12F410
    |       vbaVarTstEq(VARIANT:Double:8.10111e+008, VARIANT:Const Integer:0) returns DWORD:0
10093  + vbaFreeObj(LPINTERFACE:0012F430)
10094  + vbaFreeVar(VARIANT:String:"4") returns DWORD:20
10097  + vbaFreeVar(VARIANT:String:"08101110...") returns DWORD:20
10102  + vbaFreeVar(VARIANT:String:"08101110...") returns DWORD:20

```

For Help, press F1

RUN Program Events: 16128

But here !!! The real serial is only changed for the compare into the double dblval

그러나 여기 !!! Real serial 은 오직 compare 하기 위해 double dblval 로 변화했다.

Like our serial is changed too just before

우리의 serial 은 이전에 막 바뀌었다.

Tut.ReverseMe2a.exe - Program Results

```

10046  + vbaVarMove(VARIANT:String:"08101110...", VARIANT:String:"08101110...") returns DWORD:12F48C
10051  + vbaFreeVar(VARIANT:Integer:10) returns DWORD:12F48C
10052  + OLE _TextBox: AddRef(LPINTERFACE:0103CE2C) returns DWORD:2
10053  + vbaObjSet(LPINTERFACE:0012F430, LPINTERFACE:0103CE2C) returns LPVOID:103CE2C
10054  + Text2.Text
10067  + vbaVarSub(VARIANT:String:"4", VARIANT:String:"08101110...") 
    |   VariantChangeTypeEx(VARIANT:Empty, VARIANT:String:"4", LCID:00000400, FLAGS:00000002, VARTYPE:1)
    |       LoadLibraryA(LPSTR:0012F04C) returns HINSTANCE:7C800000
    |       GetProcAddress(HINST:7C800000, LPSTR:770F2584) returns FARPROC:7C839230
    |       GetProcAddress() returns HANDLE:140000
    |       HeapAlloc(HANDLE:00140000, FLAGS:00000008, DWORD:00000064) returns LPVOID:151F18
    |       GetLocaleInfoW(LCID:00000813, DWORD:00000014, PTR:00151F30, DWORD:00000004) returns INT:2
    |       LCMMapStringW(LCID:00000813, FLAGS:00400100, PTR:00151F30, DWORD:00000001, PTR:0012F138, C)
    |       LCMMapStringW(LCID:00000813, FLAGS:00800200, PTR:00151F30, DWORD:00000001, PTR:0012F124, C)
    |       GetLocaleInfoW(LCID:00000813, DWORD:0000000E, PTR:0012F14C, DWORD:00000004) returns INT:2
    |       GetLocaleInfoW(LCID:00000813, DWORD:00000016, PTR:0012F14C, DWORD:00000004) returns INT:2
    |       GetLocaleInfoW(LCID:00000813, DWWORD:00000051, PTR:0012F14C, DWWORD:00000004) returns INT:2
    |       GetLocaleInfoW(LCID:00000813, DWWORD:00000017, PTR:0012F14C, DWWORD:00000004) returns INT:2
    |       GetLocaleInfoW(LCID:00000813, DWWORD:0000000F, PTR:0012F14C, DWWORD:00000004) returns INT:2
    |       GetStringTypeExW(LCID:00000813, FLAGS:00000001, LPWSTR:0012F0F0, DWWORD:FFFFFF, PTR:001
    |       GetLocaleInfoA(LCID:00000813, DWWORD:00000012, PTR:0012F120, DWWORD:00000002) returns INT:2
    |       GetLocaleInfoW(LCID:00000813, DWWORD:00000010, PTR:0012F14C, DWWORD:00000004) returns INT:4
    |       GetLocaleInfoW(LCID:00000813, DWWORD:00000018, PTR:0012F14C, DWWORD:00000004) returns INT:4
    |       GetLocaleInfoW(LCID:00000813, DWWORD:00000011, PTR:0012F14C, DWWORD:00000004) returns INT:2
    |       GetLocaleInfoW(LCID:00000813, DWWORD:00000013, PTR:0012F14C, DWWORD:00000004) returns INT:2
    |       Aha ! It's only a conversion for the serial 0810111097 to perform the compare.
    |       GetLocaleInfoW(LCID:00000813, DWWORD:0000001B, PTR:0012F14C, DWWORD:00000004) returns INT:2
    |       VariantChangeTypeEx returns HRESULT:0
    |       VariantChangeTypeEx(VARIANT:Empty, VARIANT:String:"08101110...", LCID:00000400, FLAGS:00000002, V
    |       vbaVarSub returns DWORD:12F410
10093  + vbaFreeObj(LPINTERFACE:0012F430)
10094  + vbaFreeVar(VARIANT:String:"4") returns DWORD:20
10097  + vbaFreeVar(VARIANT:String:"08101110...") returns DWORD:20
10102  + vbaFreeVar(VARIANT:String:"08101110...") returns DWORD:20

```

Aha! It's only a conversion for the serial 0810111097 to perform the compare.

아하! 이것은 오직 serial 0810111097 을 compare 실행하기 위한 대화다.

So, our serial in fact is

사실 우리의 serial 은

The program has converted the five first chars from our name to ascii, joined these and then grabbed the second till tenth digit to make a serial.

MSVBVM50.DLL!0001B1F3 (no debug info)

- pvargDest (variant)
 - unsigned short .vt = 0x0000
- pvarSrc (variant)
 - unsigned short * .bstrVal = 0014DD44
= '0810111097'
- unsigned long lid = 1024 0x00000400
- unsigned short wFlags = 2 0x0002
- unsigned short vt = 5 0x0005

Yep !

The program has converted the five first chars from our name to ascii, joined these and then grabbed the second till tenth digit to make a serial.

옙!

Program 은 우리의 name 에서 첫번째 5 글자를 ascii 로 바꿨다. 그리고 그것들을 조합했고 2 번째 숫자부터 10 번째 숫자까지 serial 로 만들기 위해 모았다.

번역 주)

Dec	Ascii
108	L
101	e
110	n
97	a
49	1
53	5
49	1

Lena151 은

Hex 로는 $2 \times 7 = 14$ byte 인데 dec 로는 $3+3+3+2+2+2+2 = 17$ byte 다.

10810111097495349

여기에서 2 번째부터 10 번째 까지만 끄집어 내면 된다.

0810111097 이 된다.

위에서 보면 ascii 로 바꿨다고 하는데 잘못됐다. Decimal로 바꿨다. 이것 때문에 해맸네.._-_-그래도 Lena에게 감사한다. I love Lena.

Let's try it !!!

도전해 봐 !!!

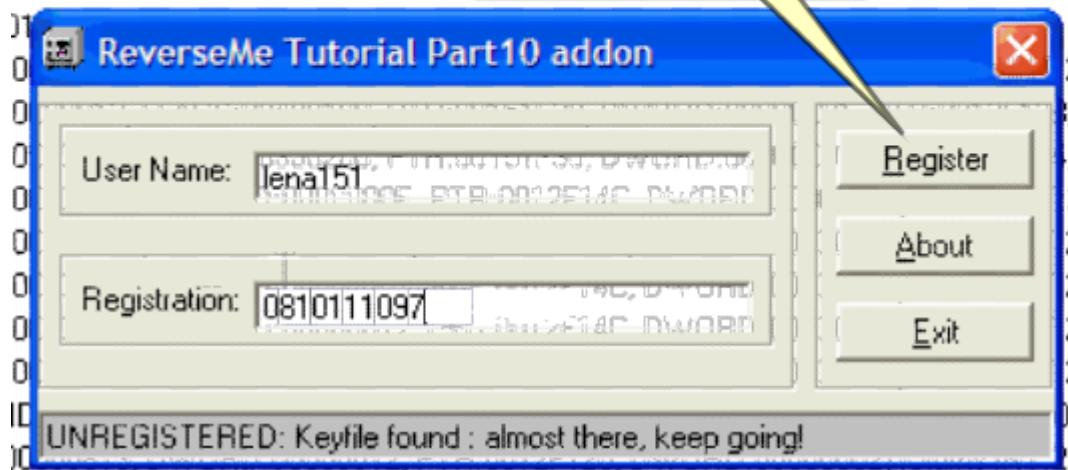


See the register button just before typing the last digit from the serial

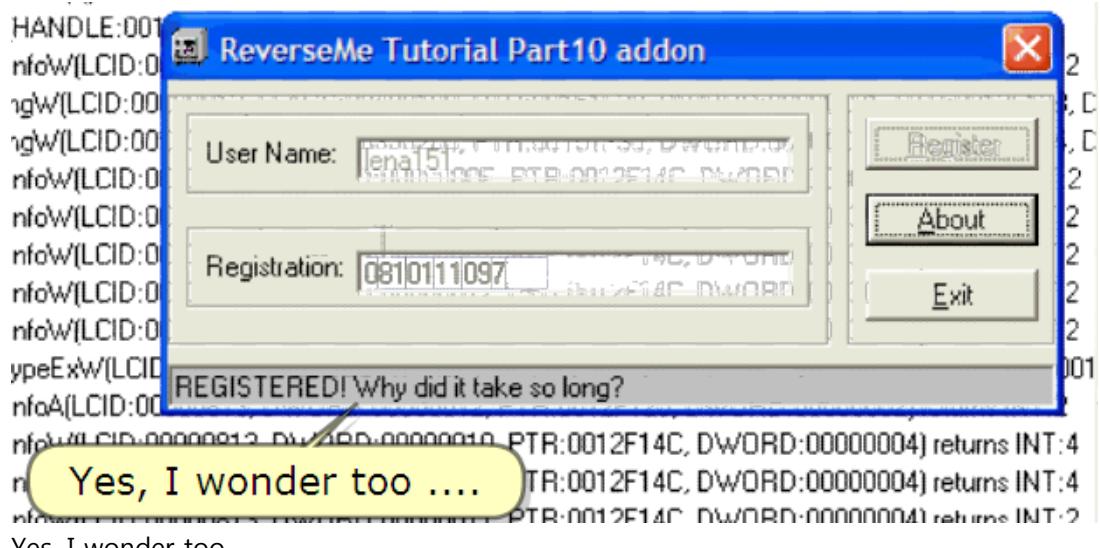
Serial에서 마지막 숫자를 입력하기 전에 Register button 을 봐.

```
ng:"4", VARIANT:String:"08101  
RIANT:Empty, VARIANT:String  
I012F04C) returns HINSTANCE  
T:7C800000, LPSTR:770F2584  
ms HANDLE:140000
```

Our serial must be right ! The registered button became available



Our serial must be right! The registered button became a variable
우리의 serial 은 올바를 것이다! 등록되는 button 은 다양할 것이다.



Yes, I wonder too

예, 나는 궁금하다...

Yep. Registration succeeded. Here, I skip a piece of this movie

예. Registration 성공했다. 여기, movie 조각을 skip 했다.

And then we land here

우리는 여기에 도착했다.

Because I want to show you once again the power of decompiling a program.

왜냐하면 너에게 다시 한 번 program 을 decompiling 힘을 보여주고 싶다.

Meanwhile, I have decompiled the original ReverseMe2 in VB Decompiler Lite 2.2 and let me show you immediately what we could have seen if I had decompiled it to start with

그 동안에, 나는 VB Decompiler Lite 2.2 에서 original ReverseMe2 를 decompile 했다. 그리고 시작하기 위해 decompile 했던 것을 보여주겠다. 우리가 봤던 것이다.

INFO :

I deliberately did not show you this before.

나는 고의로 이전에 보여주지 않았다.

Like I already mentioned : it's better to learn building up from the base ...

나는 이미 언급했다. : 이것은 base 에서 배우기 좋다.

The screenshot shows the VB Decompiler Lite interface with the following details:

- MenuBar:** File, Tools, Plugins, Help
- FileName:** C:\Documents and Settings\Wij\Bureaublad\ReverseMe2.exe
- Buttons:** ... (Browse), Decompile
- Objects Tree:** Project, Code, Form1 (selected), Form1_404450, Text2_405090, Command1_404090, Command2_4042F0, Command3_4043E0, frmAbout, API
- Interface:** Shows assembly code for the selected form:

```
004044C1: mov [ebp-000000B4h], esi
004044C7: mov [ebp-000000B8h], esi
004044CD: mov [ebp-000000BCh], esi
004044D3: mov [ebp-000000CCh], esi
004044D9: mov [ebp-000000DCh], esi
004044DF: mov [ebp-000000ECh], esi
004044E5: mov [ebp-000000FCh], esi
004044EB: mov [ebp-0000010Ch], esi
004044F1: mov [ebp-0000011Ch], esi
004044F7: push 00000001h
004044F9: call [00408164h] ; MSVBVM50.DLL._vbaOnError
004044FF: mov ebx, [00408170h] ; rtcAppActivate
00404505: mov esi, [00408178h] ; rtcSendKeys
0040450B: mov edi, [00408134h] ; MSVBVM50.DLL._vbaFreeVar
00404511: mov [ebp-000000D4h], 80020004h
0040451B: mov [ebp-000000DCh], 0000000Ah
00404525: mov [ebp-00000104h], 00401F50h ; NuMega SmartCheck
0040452F: mov [ebp-0000010Ch], 0000000h
00404539: lea edx, [ebp-0000010Ch]
0040453F: lea ecx, [ebp-000000CCh]
00404545: call [004081FCh] ; MSVBVM50.
0040454B: lea eax, [ebp-000000DCh]
00404551: push eax
00404552: lea ecx, [ebp-000000CCh]
00404558: push ecx
00404559: call ebx
0040455B: lea edx, [ebp-000000DCh]
00404561: push edx
00404562: lea eax, [ebp-000000CCh]
00404568: push eax
00404569: push 00000002h
0040456B: call [0040813Ch] ; MSVBVM50.DLL._vbaFreeVarList
00404571: add esp, 0000000Ch
```
- Annotations:** A callout bubble points to the instruction `call [004081FCh]` with the text "Finding this in a ReverseMe in VB if that isn't a good hint :)"
- Status Bar:** Decompiled OK

Finding this in a ReverseMe in VB

If that isn't a good hint :)

VB 안의 ReverseMe에서 이것을 찾아라.

아주 좋은 hint 가 되지 않는다면 :)

Interface:

```
00404528: lea ecx, [ebp-000000CCh]
0040452E: push ecx
0040452F: push 00401F78h ; %(F4)
00404594: call esi
00404596: lea ecx, [ebp-000000CCh]
0040459C: call edi
0040459E: mov [ebp-000000C4h], FFFFFFFFh
004045A2: mov [ebp-000000CCh], 0000000Bh
004045B2: lea edx, [ebp-000000CCh]
004045B8: push edx
004045B9: push 00401F88h
004045BE: call esi
004045C0: lea ecx, [ebp-000000CCh]
004045C6: call edi
004045C8: jmp 00404511h
004045CD: mov edi, 00401E18h ; reginfo.key
004045D2: mov [ebp-000000104h], edi
004045D8: mov esi, 0000000h
004045DD: mov [ebp-00000010Ch], esi
004045E3: lea edx, [ebp-00000010Ch]
004045E9: lea ecx, [ebp-000000CCh]
004045EF: call [004081FCh] ; MSVBVM50.DLL.__vbaVarDup
004045F5: push 0000000h
004045F7: lea eax, [ebp-000000CCh]
004045FD: push eax
004045FE: call [004081CCh] ; rtcDir
00404604: mov [ebp-000000D4h], eax
0040460A: mov [ebp-000000DCh], esi
00404610: lea edx, [ebp-000000DCh]
00404616: lea ecx, [ebp-5Ch]
00404619: mov esi, [00408130h] ; MSVBVM50.DLL.__vbaVarMove
0040461F: call esi
00404621: lea ecx, [ebp-000000CCh]
```

... and something like this?
What do you think ???

... and something like this? What do you think???

무언가와 같지? 네 생각은 어때???

Interface:

```
004046CC: lea ecx, [ebp-00000010Ch]
004046D2: push ecx
004046D3: call [0040817Ch] ; MSVBVM50.DLL.__vbaVarTstLt
004046D9: test ax, ax
004046DC: jz 404715h
004046DE: mov eax, [ebp+08h]
004046E1: mov edx, [eax]
004046E3: push eax
004046E4: call [edx+000000314h]
004046EA: push eax
004046EB: lea eax, [ebp-000000B8h]
004046F1: push eax
004046F2: call [00408160h] ; MSVBVM50.DLL.__vbaObjSet
004046F8: mov esi, eax
004046FA: mov ecx, [esi]
004046FC: push 00401F9Ch ; UNREGISTERED: Key File found, just do keep on going !!!
00404701: push esi
00404702: call [ecx+000000A4h]
00404708: test eax, eax
0040470A: jnl 00404FAEH
00404710: jmp 00404F9Ch
00404715: push 00000001h
00404717: lea edx, [ebp-2Ch]
0040471A: push edx
0040471B: lea eax, [ebp-000000CCh]
00404721: push eax
00404722: call [00408200h] ; rtcLeftCharVar
00404728: lea ecx, [ebp-000000CCh]
0040472E: push ecx
0040472F: lea edx, [ebp-000000B0h]
00404735: push edx
00404736: mov edi, [004081C4h] ; MSVBVM50.DLL.__vbaStrVarVal
0040473C: call edi
```

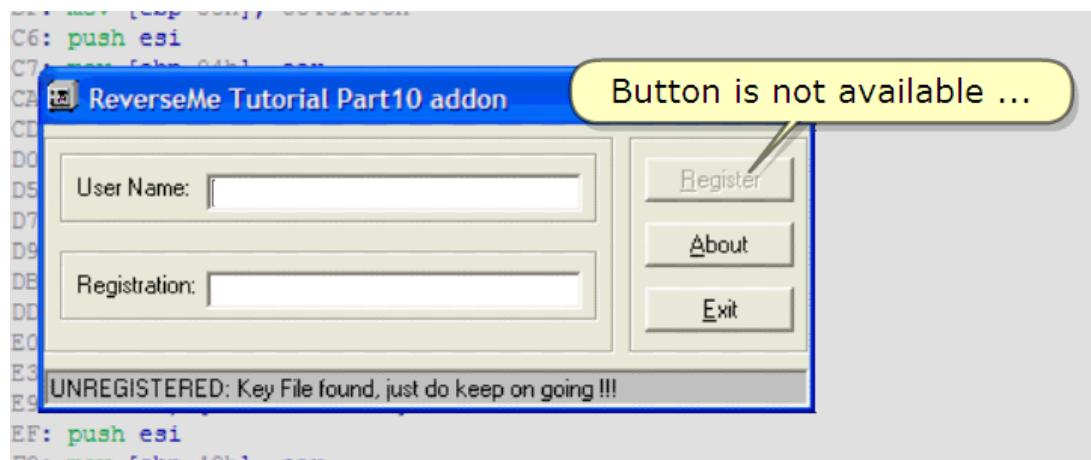
;))

;))

In this context, I also want to mention Veoveo. This tool can enable/disable any button and more.

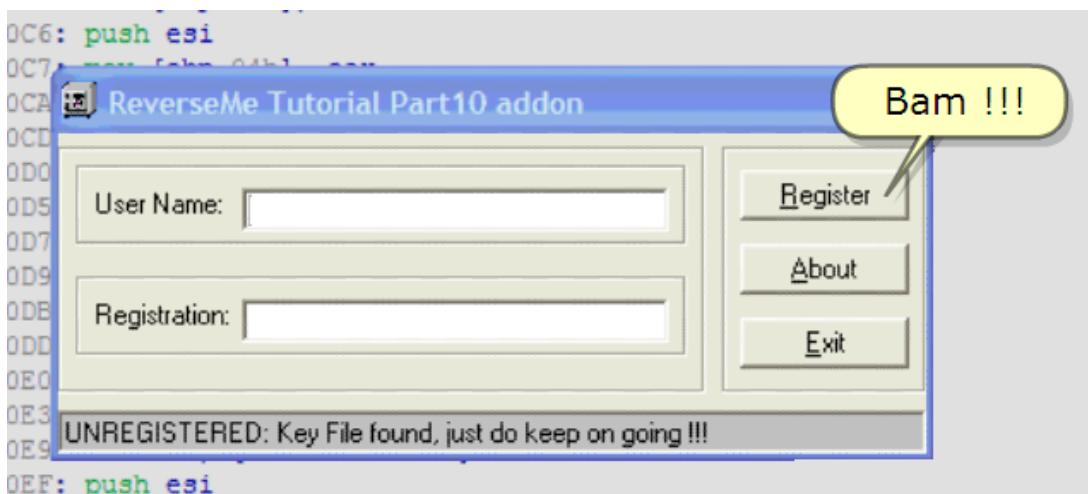
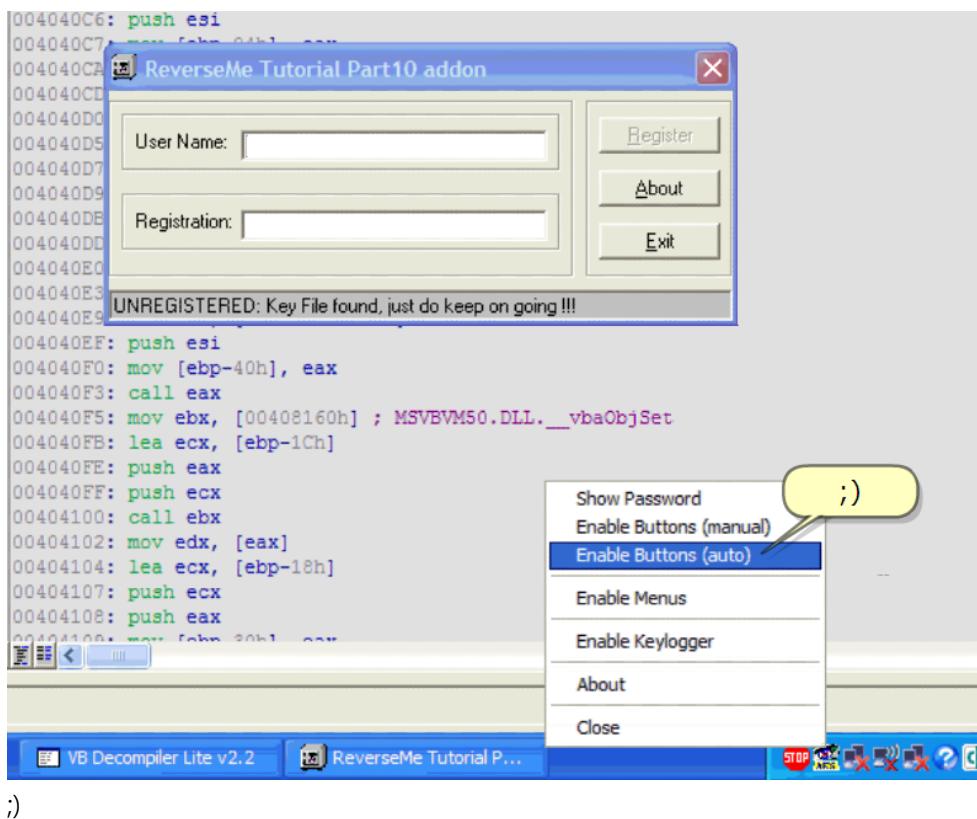
Just see ...

이 문맥에서, 나는 Veoveo 를 언급하기를 원한다. 이 tool 은 어떤 button 이라도 사용할 수도 있고 사용하지 않을 수도 있다. 바로 보자...



Button is not available

... starting Veoyeo



Have you seen how easy this works?

어떻게 이 일을 쉽게 하는지 봤어?

Of course it can't register us because the serial is verified, but it comes certainly in handy from time to time ...

물론 이것은 등록할 수 없다. 왜냐하면 serial 은 검증됐다. 그러나 이것은 명확히 쉽게 time 에서 time 까지 가져온다.

물론 이것은 등록할 수 없다. 왜냐하면 serial 은 검증됐다. 그러나 이것은 때때로 유용하게 확인 할 수 있다.

For your convenience, I have included the english version hereby.

너의 편리함을 위해, 나는 english version 을 여기에 첨부했다.

7. Conclusion

We have learned about programs made in Visual Basic that SmartCheck can be a help in finding serials, especially when used together with other tools like OllyDbg and VB Decompilers.

나는 Visual Basic 에서 만들어진 program 의 행동을 배웠다. SmartCheck 는 serials 을 찾는데 도움을 줄 수 있다. 특별히 OllyDbg 와 VB Decompilers 같은 다른 tools 도 사용했다.

In this part 10 of this reversing series, the primary goal was to study the behaviour of a program compiled in VB.

이번 reversing series 의 Part 10 에서, 제일 중요한 목표는 VB 에서 compile 된 program 의 behaviour 를 배우는 것이다.

We have also reversed the registration scheme using these tools for two ReverseMe's and a freeware program, and studied a very easy anti-SmartCheck technique.

우리는 또한 registration scheme 를 사용하고 있는 ReverseMe's 를 위해 그들의 tools 로 reverse 했다. 그리고 freeware program 이고, 매우 쉽게 anti-SmartCheck technique 를 배울 수 있었다.

I hope you understood everything fine and I also hope someone somewhere learned something from this. See me back in part 11 ;)

네가 모든 것을 좋게 이해했기를 희망한다. 또한 누구든지 어디서든지 이것에서 무언가를 배웠으면 좋겠다. Part 11 에서 보자.

The other parts are available at

다른 parts 는 사용 가능하다.

<http://tinyurl.com/27dzdn> (tuts4you)

<http://tinyurl.com/r89zq> (SnD Filez)

<http://tinyurl.com/l6srv> (fixdown)

Regards to all and especially to you for taking the time to look at this tutorial.

Lena151 (2006, updated 2007)

모두에게 안부를 전하고 특별히 이 tutorial 에 시간을 투자해준 너에게 감사한다.