

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]



Software	Fotosizer v3.5.2.558 ( <a href="http://fotosizer.com/Default">http://fotosizer.com/Default</a> )
Descarga	<a href="http://fotosizer.com/Download.aspx">http://fotosizer.com/Download.aspx</a> <a href="#">Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x) (Crack+Patch)</a>
Protección	<b>Serial - Crypto Obfuscator for .NET v5.x</b>
Herramientas	Windows 10.0 - Build 15063 x64 Bits (SO donde trabajamos)  dnSpy v3.0.2 (.NET assembly editor, decompiler, and debugger <a href="https://github.com/0xd4d/dnSpy/">https://github.com/0xd4d/dnSpy/</a> )  de4dot v3.1.41592 (.NET deobfuscator and unpacker. <a href="https://github.com/0xd4d/de4dot">https://github.com/0xd4d/de4dot</a> )  Protection iD v0.6.8.5 RDG Packer Detector v0.7.4 dUP2 Diablo's Universal Patcher v2.26

## INTRODUCCIÓN

Un saludo a la comunidad CracksLatinoS y en especial a ti que estás leyendo este tuto, que es mi segundo tutorial, y es una obligación en hacerlo porque pienso que es lo mínimo que uno puede hacer para retribuir en algo por todo lo que la comunidad encabezada por el abanderado y maestro de la mayoría de nosotros, y ya ustedes saben quién es, me refiero a **Ricardo NarvaJa** que nos han dado tanto.

Un poco más de "carreta". Este es el primer programa en .NET que abordo y me tocó desde cero porque ni idea cómo trabajar con un .NET pero ahí se demuestra lo grande que es la lista porque tú puedes encontrar mucho material útil para seguir aprendiendo esto del Cracking (cosa que nunca terminas de aprender) y aquí es donde debo agradecer de forma personal a **sequeyo** que gracias a sus tutes de .NET

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

me mostraron el camino, además me fue muy grato en recibir su respuesta cuando pedí un crackme que resolvió en un concurso ([DOTNET](#)).

Para terminar te aconsejo que si te estás adentrando en los .NET te leas el tutorial de **sequeyo** que te explica de manera sencilla y muy clara lo básico de los .NET y que te darán una base sólida para empezar, es la teoría numera [\[437\]](#) [CRACKEO VB .NET por sequeyo](#).

## ANÁLISIS INICAL

En programa te pide un serial para quedar registrado y completo. Empecé con la convicción de hallar un serial válido y también crear un Keygen suponiendo que fuera de ese tipo o solo el serial si fuera un Hardcoded, pero la vida me llevó por otro camino y terminé cambiando el programa para que aceptara cualquier serial.



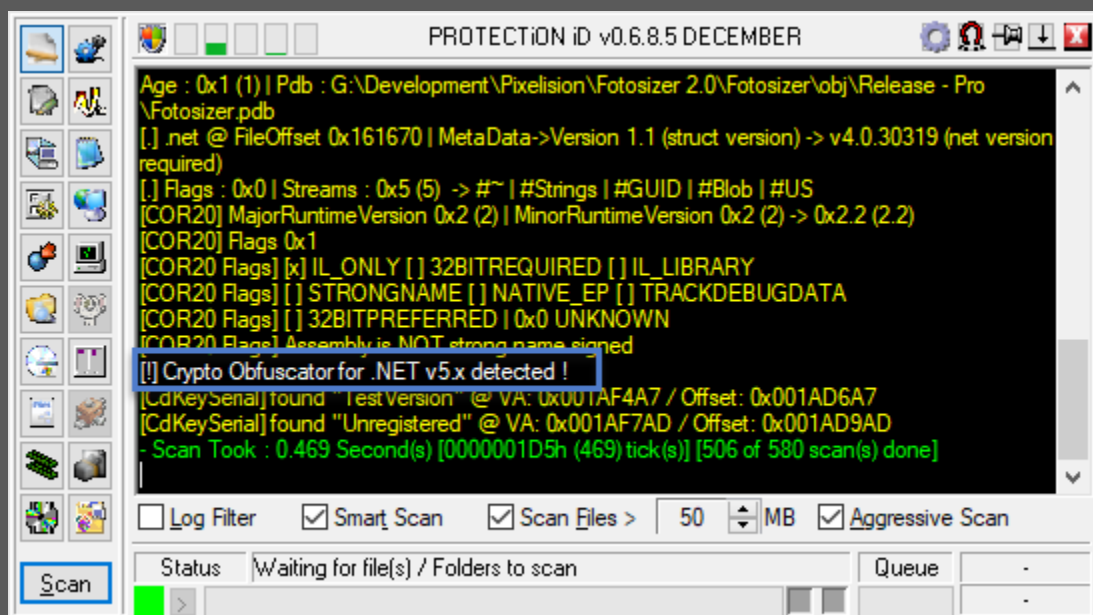
Nos muestra una <NAG Trial> donde nos da la opción para poder ingresar el serial y poder registrar el programa.

Ahora lo analizamos con el <RDG Packer Detector v0.7.4> para ver si está empacado u ofuscado y nos avisa que efectivamente está ofuscado.

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]



También utilicemos el <Protection id v0.6.8.5 December 2016> y que confirma lo dicho por el <RDG Packer Detector>.

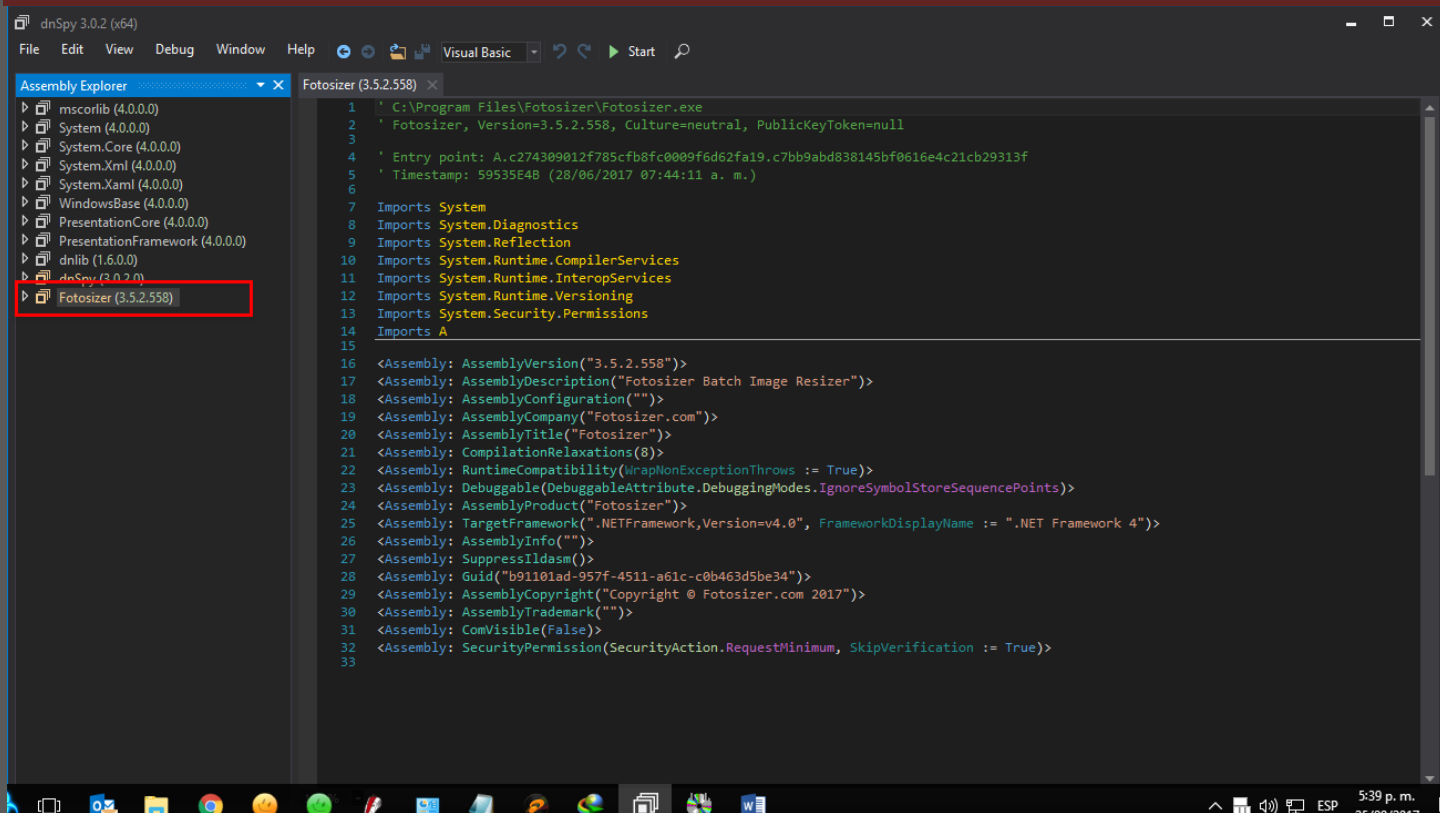


Y como siempre hay que decir cuando los programas no estén protegidos: "*Qué bueno que no tenga protección, así se facilitan las cosas y sea fácil la tarea...jeje*". Por fortuna todavía podemos decir lo anteriormente citado ya que esa ofuscación se supera fácilmente utilizando la herramienta <de4dot>.

## AL ATAQUE

Lo primero que haremos será abrir el programa <Fotosizer.exe> con el <dnSpy v3.0.2> (de aquí en adelante será dnSpy), <File->Open> o <Ctrl+O>, ahí ya lo tenemos cargado.

# [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]



Antes de continuar les recomiendo estas lecturas de las teorías numeradas donde se analizan otros .NET con otras herramientas incluyendo el **dnSpy** y que nos sirve para profundizar mucho más en la teoría y poder entender mejor el manejo de los .NET:

[\[302\] CRACKEANDO APLICACIONES .NET PARTE I por Emadicius](#)

[\[1052\] tutorial principiante .NET+tool RSA](#)

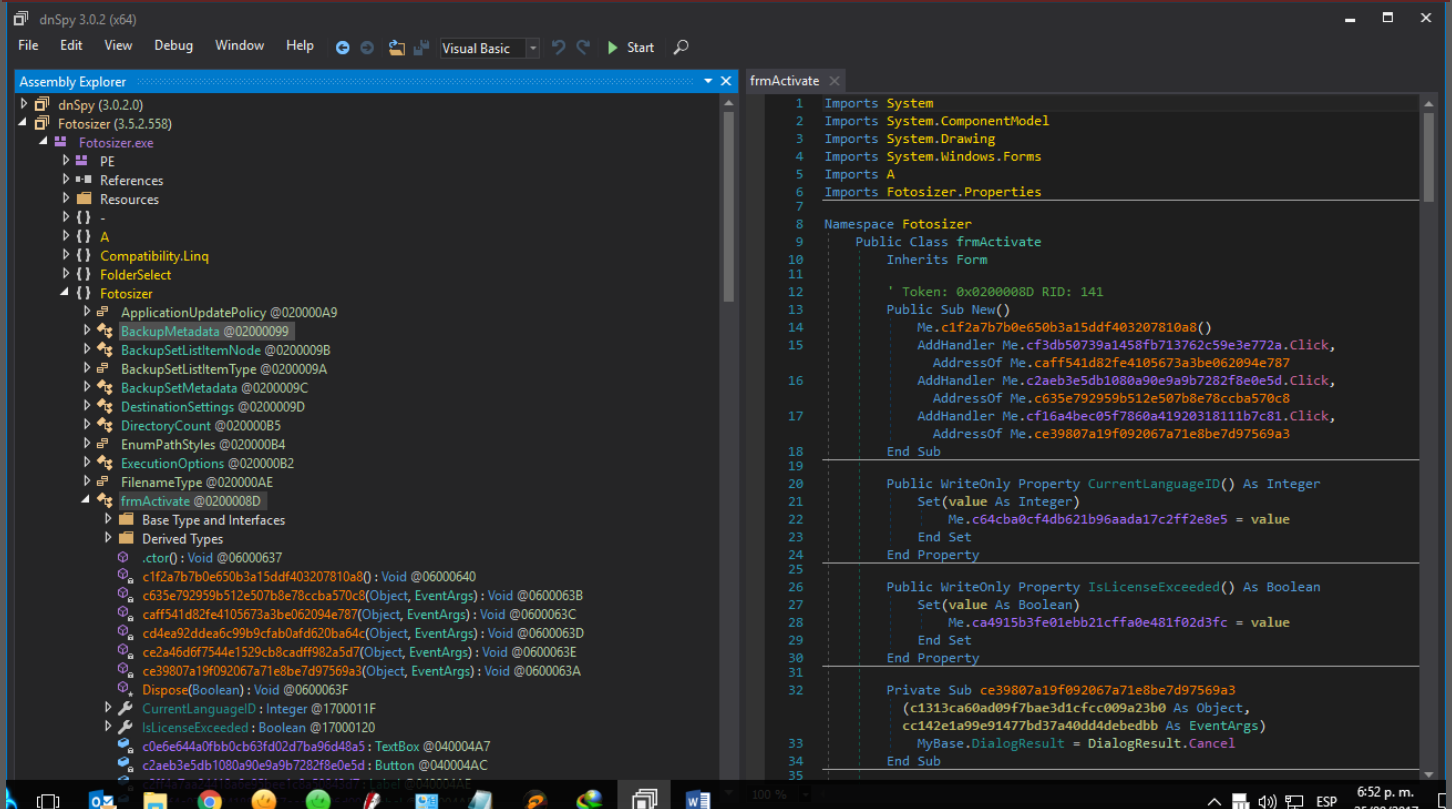
[\[1202\] NovaMind 4 Platinum un .NET por sequeyo](#)

[\[1575\] Jugando con un.NET en dnSpy por SNAT](#)

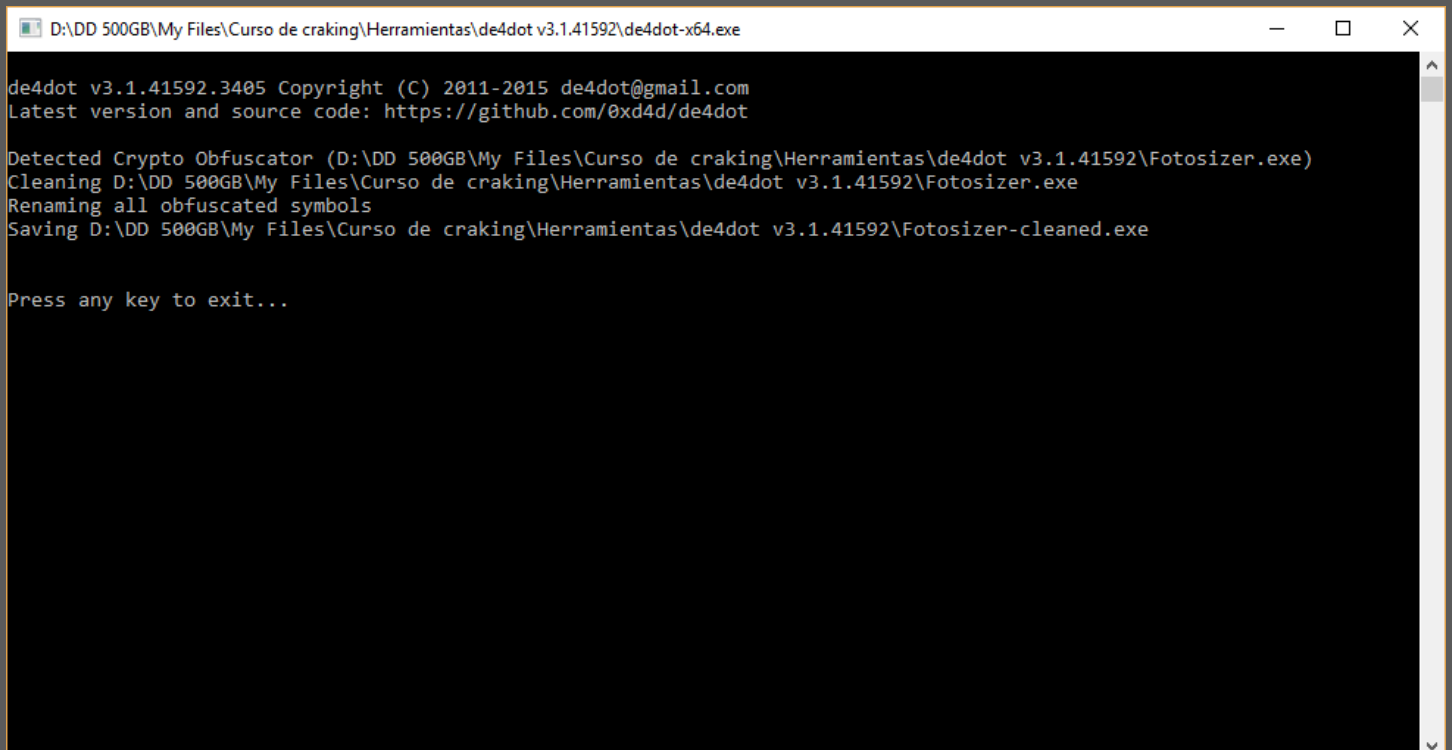
[\[1582\] sequeyo .NET NovaMind 605 pass crackslatinos](#)

He colocado una captura del **dnSpy** con el programa ofuscado y pueden ver claramente que los nombres de los eventos, objetos...etc, no son claros y entonces se complica seguir la ruta correcta.

# [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]



Para arreglar eso utilizaremos el <de4dot v3.1.41592>; yo lo haré de la forma sencilla y es arrastrar el programa sobre el ejecutable del <de4dot> y él se encarga de todo dejándome el archivo <Fotosizer-cleaned.exe>.



## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

Ahora si abriremos nuestro archivo "desofuscado" (por llamarlo de una forma, sin protección) ya podemos ver que ahora tiene todos los nombres correctos y así podemos revisar el código sin perdernos.

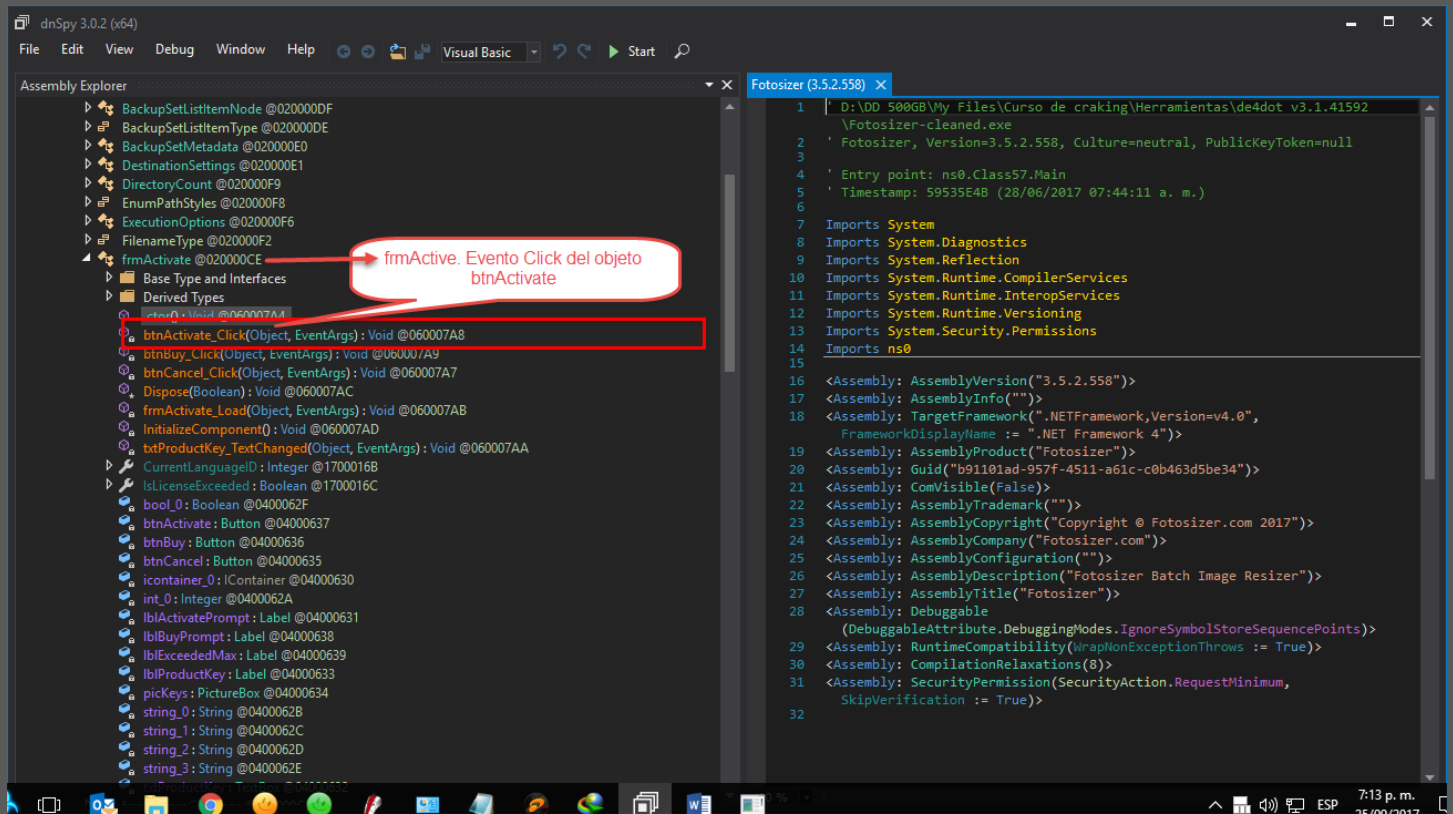
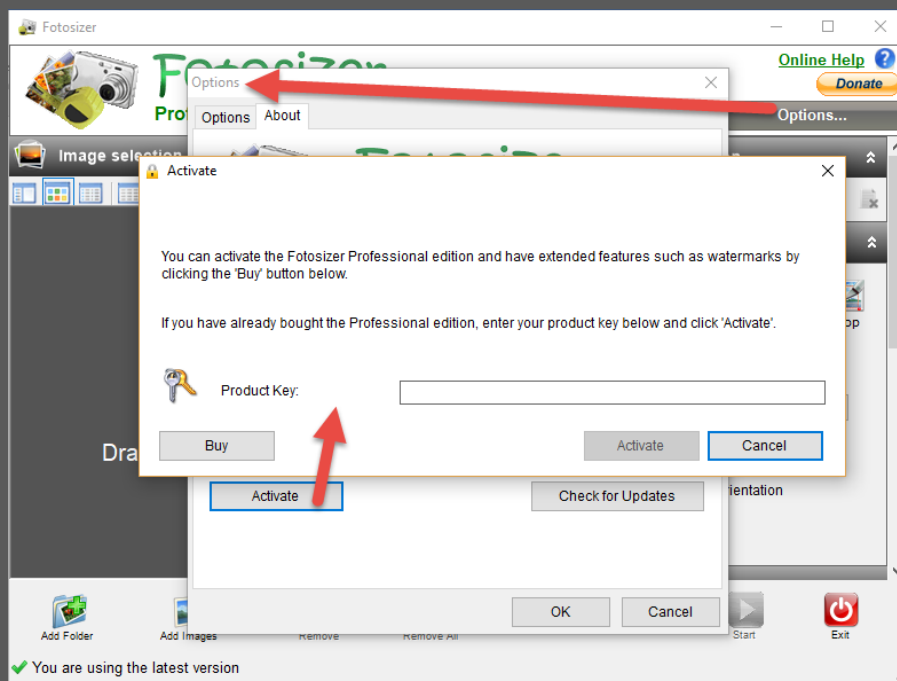


Imagen 1

El **<frmActivate>** es la ventana que nos sale cuando deseamos activar el programa desde el **<About>** del programa, cosa que se hace después aceptar la **<NAG Trial>**.





## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

Revisando más <Form> podemos encontrar el <frmSplash> que es la <NAG Trial>. Lo que debemos hacer siempre es buscar con paciencia y revisar si tienen nombres de interés, como <Activate> en nuestro caso; yo llegué a los <Form> de interés de esa manera mientras entendía como el dnSpy muestra la información.

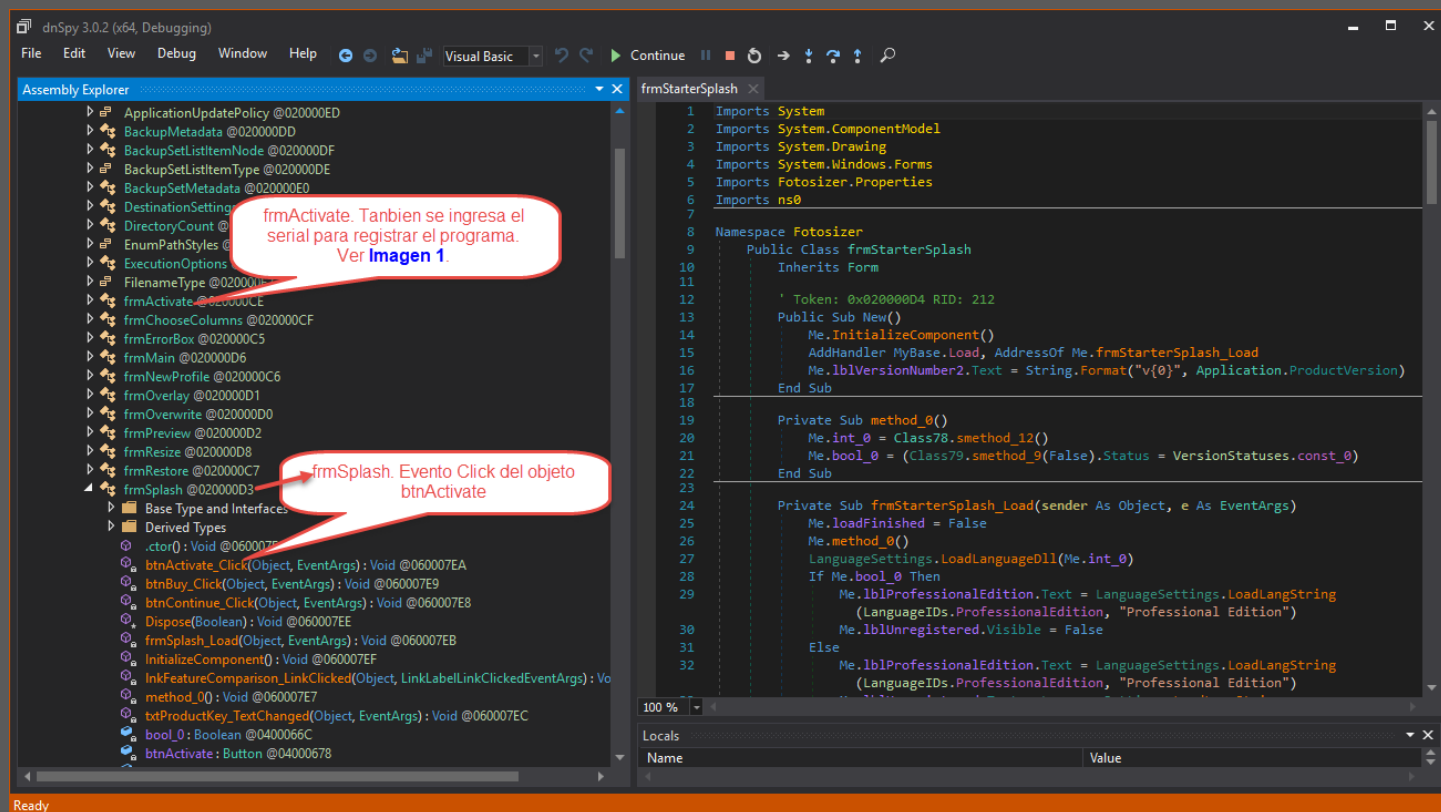


Imagen 2

Como comenté en la introducción terminé cambiando el programa para que acepte cualquier serial pero antes de seguir por ese lado quiero comentarles la lógica del programa para mostrar si está <REGISTRADO> o <NO REGISTRADO> y la explico como yo la entiendo y comienzo diciendo lo que dije en mi primer tuto [\[1630\]](#) [Tuto001 - Exif-Tag Remover v5.1 Serial.Hardcoded.VisualBasicv6.0](#), "la regla general es que los programas siempre guardan los datos de registro y que comprueben estos al iniciar", aclarando, el programa revisa y como no tiene datos de su registro exitoso nos mostrará en primer lugar la <NAG Trial>, en este caso el <Form> <frmSplash> (ver Imagen 2), entonces eso quiere decir que debe haber una comprobación preliminar que siempre es utilizada para comprobar los datos de nuestro registro, ya sea que tengamos registrada la aplicación o cuando ingresemos el serial por primera vez y así funciona este programa y la mayoría.

Solo nos resta encontrar ese procedimiento y buscando con paciencia, este procedimiento es llamado en el evento Load de nuestro <Form> <frmStarterSplash>.

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

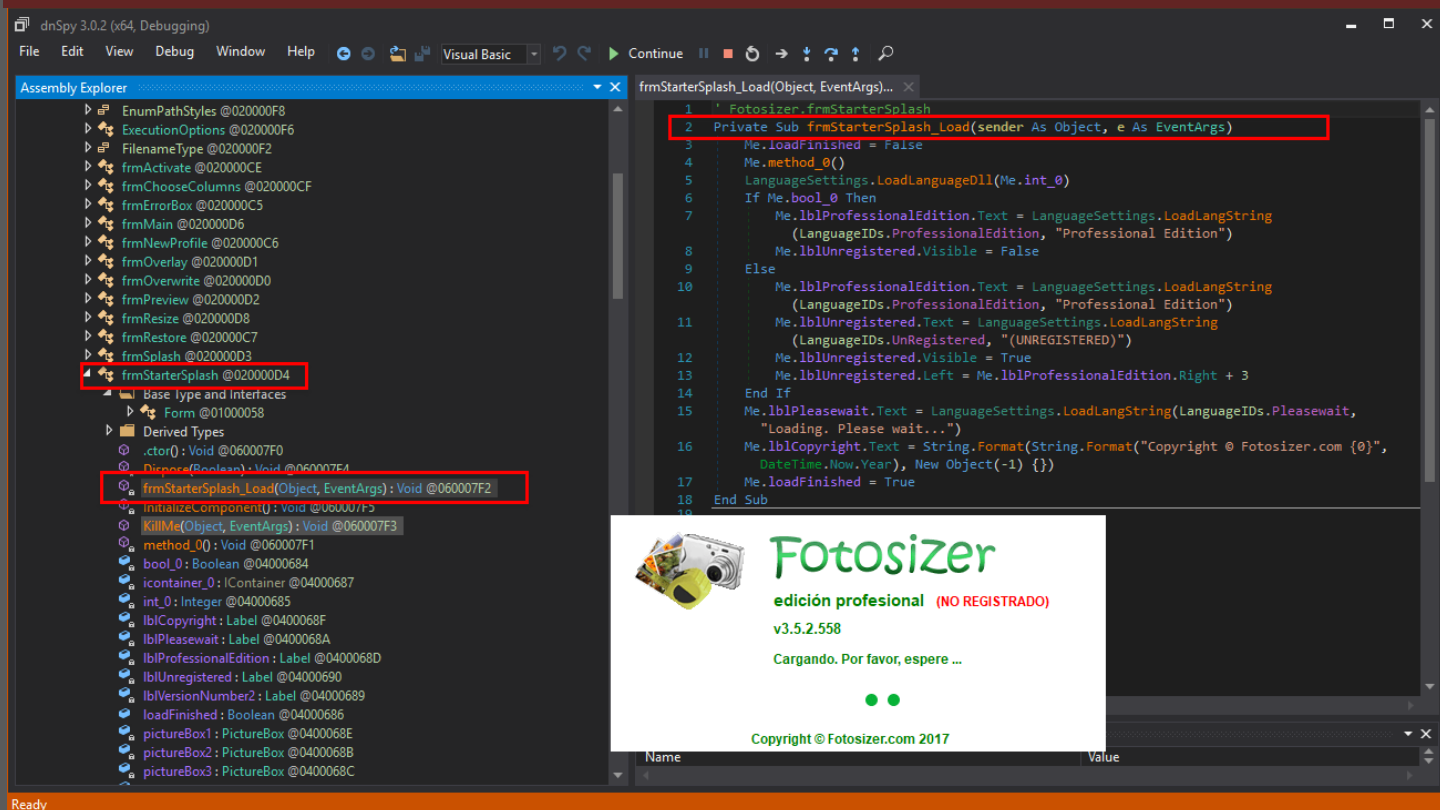


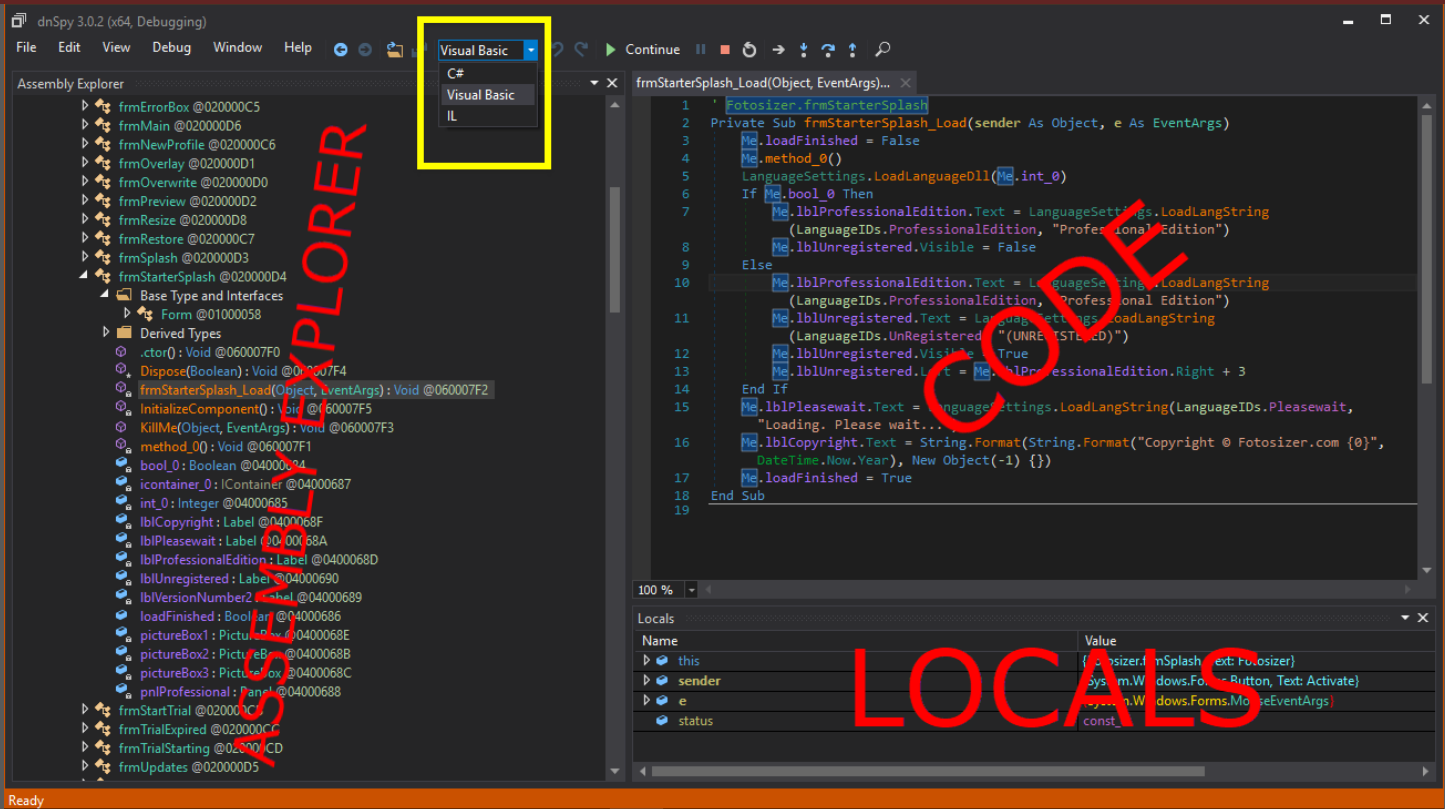
Imagen 3

Ahí lo tenemos claro, en la parte superior derecha tenemos el procedimiento **Private Sub frmStarterSplash\_Load(sender As Object, e As EventArgs)** que tiene un **<If>** en la línea 6 (**If Me.bool\_0 Then**), y de acuerdo al valor tomado por la variable **bool\_0** (Boolean) saltará al **<CHICO BUENO>** o **<CHICO MALO>** y ese valor de **bool\_0** es dado por el procedimiento **Me.method\_0()**. Como ya conozco las rutinas de comprobación no sirve de nada cambiar el salto aquí porque solo quitaríamos el cartelito rojo de **<NO REGISTRADO>** pero nada más porque cuando entras a **Me.method\_0()** es donde comprueba todo, es ahí donde carga todas las funciones Pro. El procedimiento **Me.method\_0()** te llevará a la maraña de comprobaciones y es ahí donde debemos cambiar los saltos para nuestro serial falso. Todo este cuento es para entender un poco lo debemos hacer.

Ahora un poco de lo que podemos ver en el **dnSpy** y lo que podemos hacer con él. El **dnSpy** ofrece tres vistas de código C#, Visual Basic, IL que vemos en la ventana **<CODE>**. Yo utilizaré Visual Basic que es la que más entiendo. Tienes a tu disposición la ventana **<LOCALS>** que te muestra las variables y los valores que tienen cuando ejecutas código, además puedes editar esos valores para que la ejecución de código siga la ruta que tú quieras. Bueno eso a grandes rasgos.



# [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]



Bueno como quiero que acepte mi serial falso, selecciono el **<Form> <frmSplash>** desde el **<ASSEMBLY EXPLORER>** para debugear desde ahí y revisar qué sucede con mi serial falso y los cambios que debo hacer para que funcione. Coloco un **<BrakPoint>** en la línea 4 de mi ventana **<CODE>**, selecciono dando clic en la línea, con **<F9>**, **<Debug->Toggle Breakpoint>** o dándole clic en la barra izquierda el lado de la línea de código.

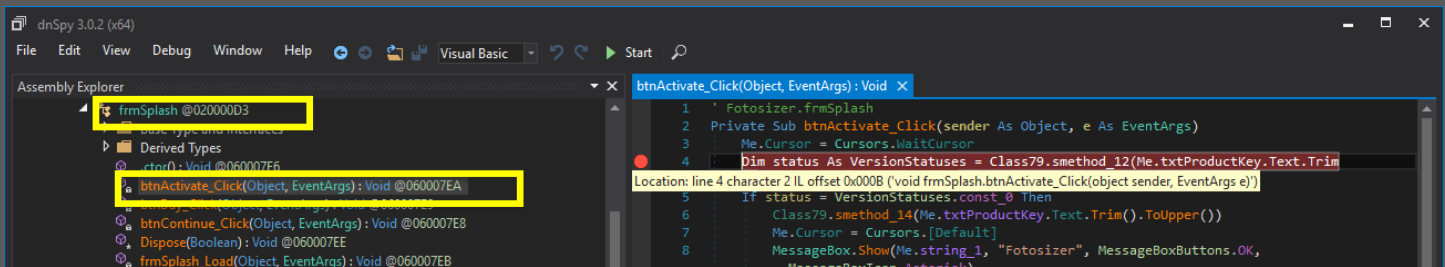


Imagen 3

Analicemos la línea 4 de Código:

**Dim status As VersionStatuses = Class79.smethord\_12(Me.txtProductKey.Text.Trim().ToUpper(), True).Status**

1. Declara la variable **status** como **VersionStatuses**. Revisemos qué significa **VersionStatuses**, para eso nos posicionamos encima de lo que queremos información. En este caso vemos que **VersionStatuses** es una enumeración donde deben estar declarados los valores constantes que será retornado por **Class79.smethord\_12(Me.txtProductKey.Text.Trim().ToUpper(), True).Status** y dependerá de nuestro serial que se almacena en **Me.txtProductKey.Text**. Para ver las declaraciones con

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

<Ctrl+Clic> sobre la misma y así se nos abre una nueva pestaña y no perdemos la que tenemos abierta.

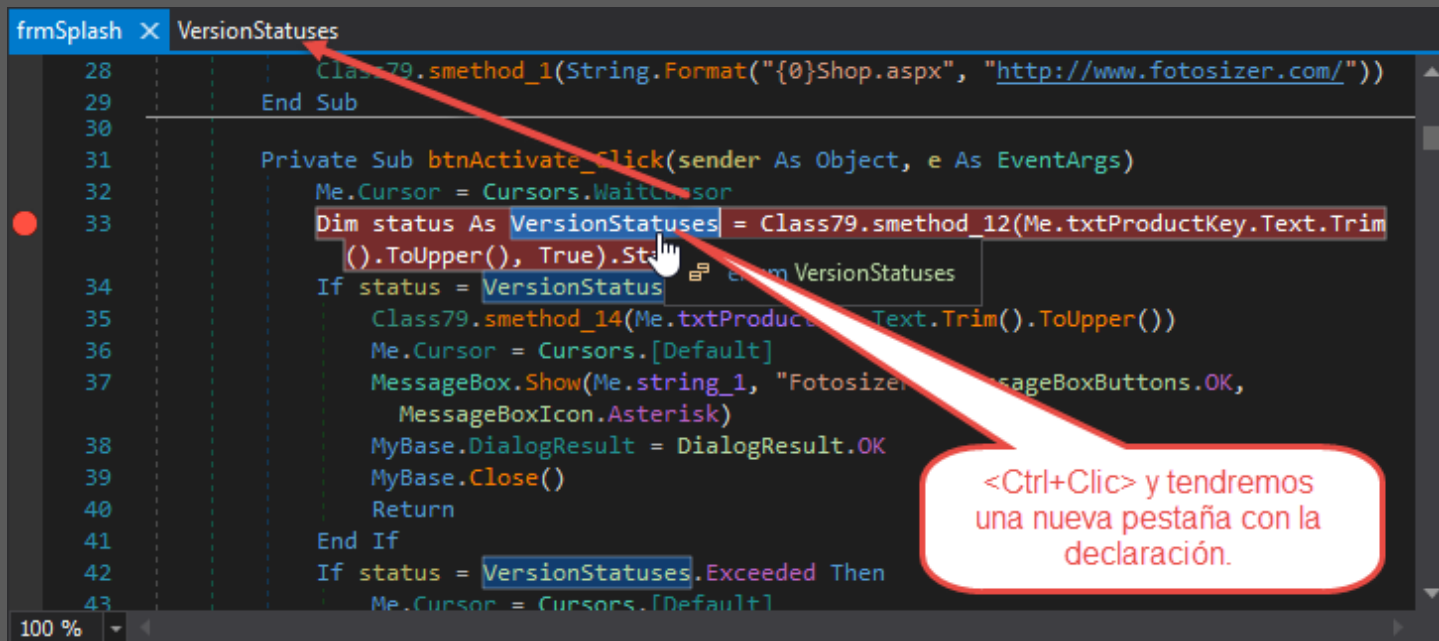


Imagen 4

2. Recordar que el código se ejecuta en un orden determinado, desde el interior al exterior. El orden sería `Me.txtProductKey.Text.Trim()` que sería eliminar los espacios del inicio o final del serial, si este los tuviera, luego seguiría `ToUpper()`, que es para pasar a mayúsculas el serial, eso indica que el serial verdadero, el original, debe ser puras letras o por lo menos letras y números.

Un poco de ejecución de código con **dnSpy**. Para iniciar el debugger en la barra de herramientas **Start** o <F5>. Tendrás la ventana <Debug Assembly> y lo más relevante para mí es que te puedes escoger cuando detener el debugger.

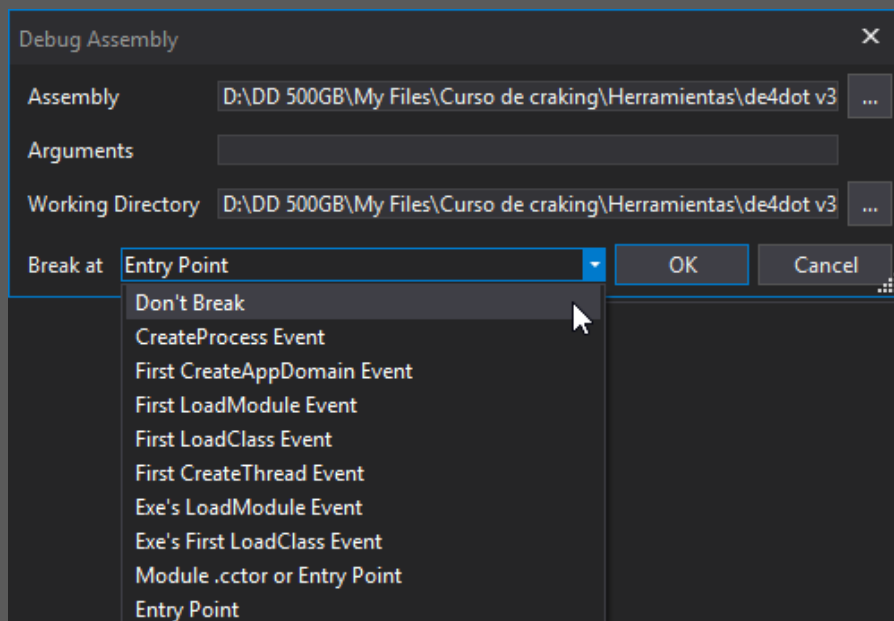



Imagen 5

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

Iniciado el Debugger, podemos tracear paso a paso, o si queremos pasar por encima de las instrucciones o no. , <Step Into> o <F11> para tracear entrando en todo el código; <Step Over> o <F10> para tracear por instrucciones.

Bueno, ahora si a lo que venimos, volvamos al <Breakpoint> de la [Imagen 4](#), lo colocamos e iniciamos el debugger <F5> y escogemos que no pare (Don't Break) en la ventana <Debug Assembly> ([Imagen 5](#)) e iniciamos el Debugger, y el programa se ejecuta mostrándonos la <NAG Trial>. Ahí colocamos nuestro serial "passfalso"



Y empieza lo bueno, clic en <Activate> y el programa para en nuestro <Breakpoint>. Este código lo explicamos ya ([Imagen 4](#)), y nos sirvió para explicar un poco del dnSpy y cómo funciona el código.

Debemos seguir [Class79.smethod\\_12](#) abriéndolo en otra pestaña, lo recuerdo una última vez, <Ctrl+Clic> sobre la rutina de interés es este caso es [smethod\\_12](#). Así podemos avanzar y ver el código sin tracear y poder colocar <Breakpoints> en lugares de interés. Coloco un <Breakpoint> en [Class79.smethod\\_12](#) y <F5> para continuar la ejecución, se detendrá en el <Breakpoint> [Class79.smethod\\_12](#).

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

The screenshot displays a Visual Studio code editor with a VB.NET function named `smethod_12`. The function takes two parameters: `string_7` (String) and `bool_0` (Boolean), and returns a `Class80` object. The function's logic is as follows:

- Line 218: `Dim [class] As Class80 = New Class80() With { .Key = String.Empty, .Status = VersionStatuses.Invalid, .Trial = False, .DaysRemaining = 0 }`
- Line 219: `If string_7.Length = 22 Then`
- Line 220: `Try`
- Line 221: `[class] = Class79.smethod_13(string_7.ToUpper())`
- Line 222: `If [class] IsNot Nothing AndAlso [class].Status = VersionStatuses.const_0 AndAlso bool_0 Then`
- Line 223: `Dim status As VersionStatuses = Class82.smethod_3(string_7)`
- Line 224: `[class].Status = status`
- Line 225: `End If`
- Line 226: `Catch ex As Exception`
- Line 227: `Console.WriteLine("Error validating license: " + ex.Message)`
- Line 228: `[class].Status = VersionStatuses.Invalid`
- Line 229: `End Try`
- Line 230: `End If`
- Line 231: `Return [class]`
- Line 232: `End Function`

The `Locals` window at the bottom shows the current state of the variables:

Name	Value
<code>string_7</code>	"PASSFALSO"
<code>bool_0</code>	true
<code>class</code>	null
<code>status</code>	const_0
<code>ex</code>	null

Two red arrows point from the `string_7` variable in the `Locals` window to the `string_7` parameter in the function signature and the `string_7.Length` property access in the `If` statement.

Analizando un poco podemos ver que la variable `string_7` tiene nuestro serial y lo ha puesto en mayúsculas, en la línea 219 `If string_7.Length = 22` se comprueba la longitud de mi serial que debe ser igual a 22, ahora mi serial "PASSFALSO" tiene longitud de 9, así que nos botará al <CHICO MALO>. Tracemos con <F10> hasta llegar a `Private Sub btnActivate_Click(sender As Object, e As EventArgs)`.

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

The screenshot shows the Visual Studio IDE with the `frmSplash` form selected. The code in the `Private Sub btnActivate_Click` method is visible. A red arrow points from the `status` variable in the `Locals` window to its assignment in the code.

```
29 End Sub
30
31 Private Sub btnActivate_Click(sender As Object, e As EventArgs)
32     Me.Cursor = Cursors.WaitCursor
33     Dim status As VersionStatuses = Class79.smethod_12(Me.txtProductKey.Text.Trim
34         ().ToUpper(), True).Status
35     If status = VersionStatuses.const_0 Then
36         Class79.smethod_14(Me.txtProductKey.Text.Trim().ToUpper())
37         Me.Cursor = Cursors.[Default]
38         MessageBox.Show(Me.string_1, "Fotosizer", MessageBoxButtons.OK,
39             MessageBoxIcon.Asterisk)
40         MyBase.DialogResult = DialogResult.OK
41         MyBase.Close()
42         Return
43     End If
44     If status = VersionStatuses.Exceeded Then
45         Me.Cursor = Cursors.[Default]
46         MessageBox.Show(String.Format("{0}{1}{1}{2}", Me.string_2,
47             Environment.NewLine, Me.string_3), "Fotosizer", MessageBoxButtons.OK,
48             MessageBoxIcon.Exclamation)
49         Return
50     End If
51     Me.Cursor = Cursors.[Default]
52 End Sub
```

**Locals**

Name	Value
<code>this</code>	{Fotosizer.frmSplash, Text: Fotosizer}
<code>sender</code>	{System.Windows.Forms.Button, Text: Activate}
<code>e</code>	{System.Windows.Forms.MouseEventArgs}
<code>status</code>	Invalid

Podemos ver que la variable `status` es `<Invalid>` y debería ser `<const_0>` para sacar al **<CHICO BUENO>**.

Sigamos con **<F5>** hasta sacar al **<CHICO MALO>** y colocamos un serial de longitud 22, colocaré este `"LUISFECAB-5D4F-BA72F16"` y volvemos a tratar hasta llegar de nuevo a `Public Function smethod_12(string_7 As String, bool_0 As Boolean) As Class80` donde tenemos un **<Breakpoint>**.

The screenshot shows the `Public Function smethod_12` function. A green box highlights the assignment of `[class]` to `Class79.smethod_13`.

```
217 Public Function smethod_12(string_7 As String, bool_0 As Boolean) As Class80
218     Dim [class] As Class80 = New Class80() With { .Key = String.Empty, .Status =
219         VersionStatuses.Invalid, .Trial = False, .DaysRemaining = 0 }
220     If string_7.Length = 22 Then
221         Try
222             [class] = Class79.smethod_13(string_7.ToUpper())
223             If [class].IsNot Nothing AndAlso [class].Status =
224                 VersionStatuses.const_0 AndAlso bool_0 Then
225                 Dim status As VersionStatuses = Class82.smethod_3(string_7)
226                 [class].Status = status
227             End If
228         Catch ex As Exception
229             Console.WriteLine("Error validating license: " + ex.Message)
230             [class].Status = VersionStatuses.Invalid
231         End Try
232     End If
233     Return [class]
234 End Function
```

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

Ahora si pasé esa comprobación de longitud de serial=22 y entramos la línea 221 `Class79.smethod_13` que va a realizar más comprobaciones de mi serial.

```
234      Friend Function smethod_13(string_7 As String) As Class80
235          Dim [class] As Class80 = Class79.smethod_15(string_7, False)
236          Dim class2 As Class80 = Class79.smethod_15(string_7, True)
237          If [class].Status <> VersionStatuses.const_0 Then
238              Dim arg_1E_0 As VersionStatuses = class2.Status
239          End If
240          If [class].Status <> VersionStatuses.const_0 Then
241              Return class2
242          End If
243          Return [class]
244      End Function
```

Tenemos dos comprobaciones más con nuestro serial. Entremos a la primera que es la línea 235 `Class79.smethod_15(string_7, False)`

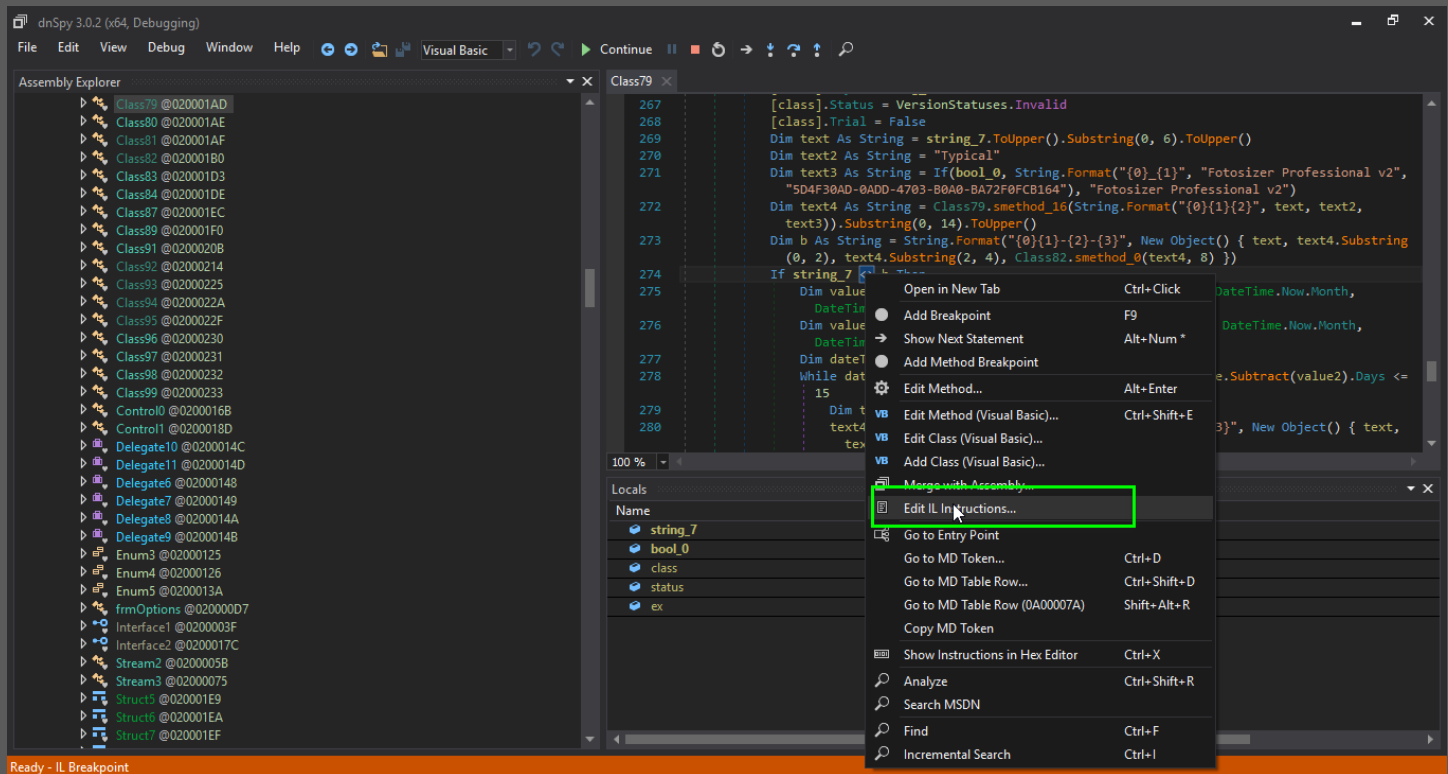
```
264      Private Function smethod_15(string_7 As String, bool_0 As Boolean) As Class80
265          Dim [class] As Class80 = New Class80()
266          [class].Key = string_7
267          [class].Status = VersionStatuses.Invalid
268          [class].Trial = False
269          Dim text As String = string_7.ToUpper().Substring(0, 6).ToUpper()
270          Dim text2 As String = "Typical"
271          Dim text3 As String = If(bool_0, String.Format("{0}_{1}", "Fotosizer Professional v2",
272              "5D4F30AD-0ADD-4703-B0A0-BA72F0FCB164"), "Fotosizer Professional v2")
273          Dim text4 As String = Class79.smethod_16(String.Format("{0}_{1}{2}", text, text2,
274              text3)).Substring(0, 14).ToUpper()
275          Dim b As String = String.Format("{0}_{1}-{2}-{3}", New Object() { text, text4.Substring(0, 2),
276              text4.Substring(2, 4), Class82.smethod_0(text4, 8) })
277          If string_7 <> b Then
278              Dim value As DateTime = New DateTime(DateTime.Now.Year, DateTime.Now.Month, DateTime.Now.Day)
279              Dim value2 As DateTime = New DateTime(DateTime.Now.Year, DateTime.Now.Month,
280                  DateTime.Now.Day)
281              Dim dateTime As DateTime = value2.AddDays(15.0)
282              While dateTime.Subtract(value2).Days > 0 AndAlso dateTime.Subtract(value2).Days <= 15
283                  Dim text5 As String = value2.ToString("yyyy-MM-dd")
284                  text4 = Class79.smethod_16(String.Format("{0}_{1}{2}{3}", New Object() { text, text2,
285                      text3, text5 })).Substring(0, 14).ToUpper()
286                  b = String.Format("{0}_{1}-{2}-{3}", New Object() { text, text4.Substring(0, 2),
287                      text4.Substring(2, 4), Class82.smethod_0(text4, 8) })
288                  If string_7 = b Then
289                      [class].Status = VersionStatuses.const_0
290                      [class].Trial = True
291                      [class].DaysRemaining = value2.Subtract(value).Days
292                      Exit While
293                  End If
294                  value2 = value2.AddDays(1.0)
295              End While
296          Else
297              [class].Trial = False
298              [class].Status = VersionStatuses.const_0
299          End If
300          Return [class]
301      End Function
```

Aquí es donde debemos hacer los primeros cambios para el serial sea tomado como verdadero, así que debemos cambiar los saltos para que pase como comprobación correcta. En la línea 274 `If string_7 <> b Then` las variables `string_7` y `b` deben ser iguales

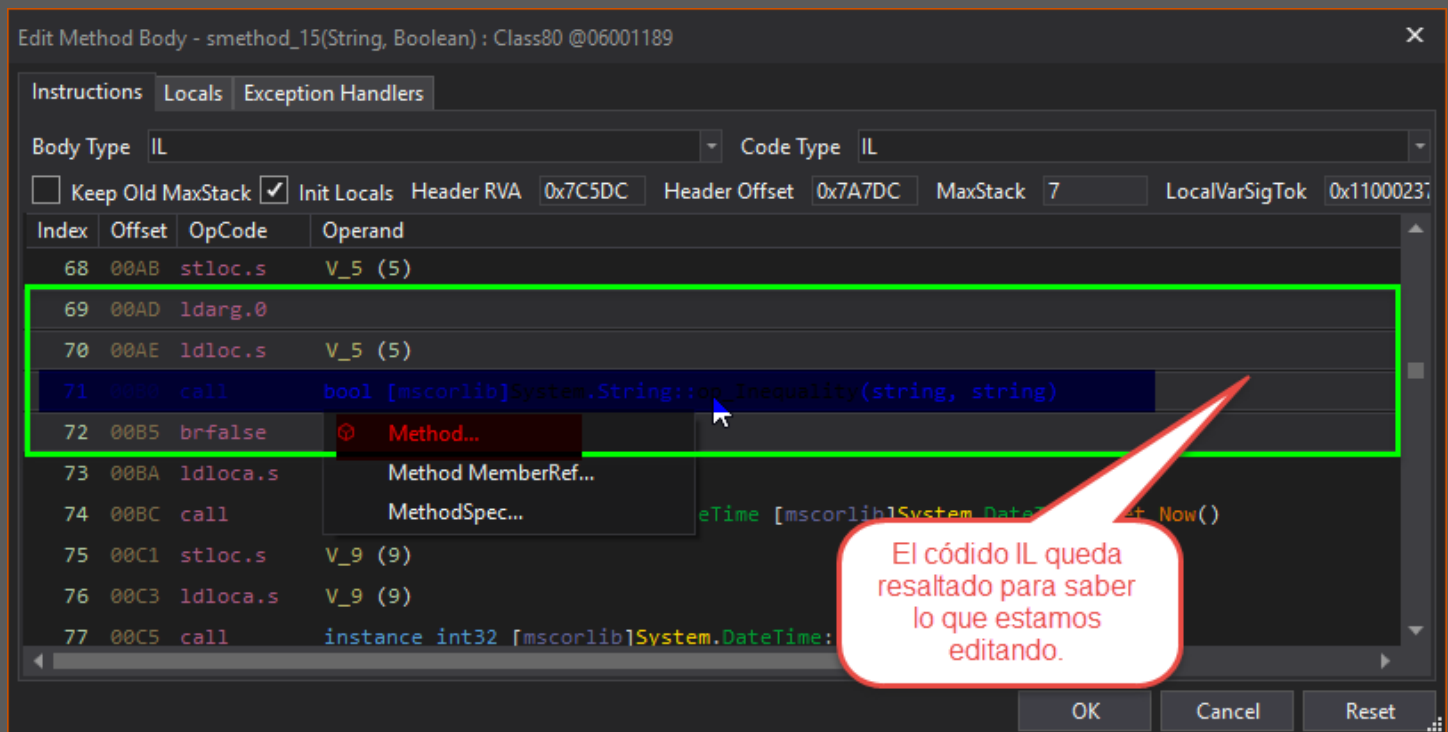


## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

para que se tome el salto y en nuestro caso `string_7` y `b` no son iguales, así que debemos hacer la comparación con igualdad para que salte `If string_7 = b Then`.

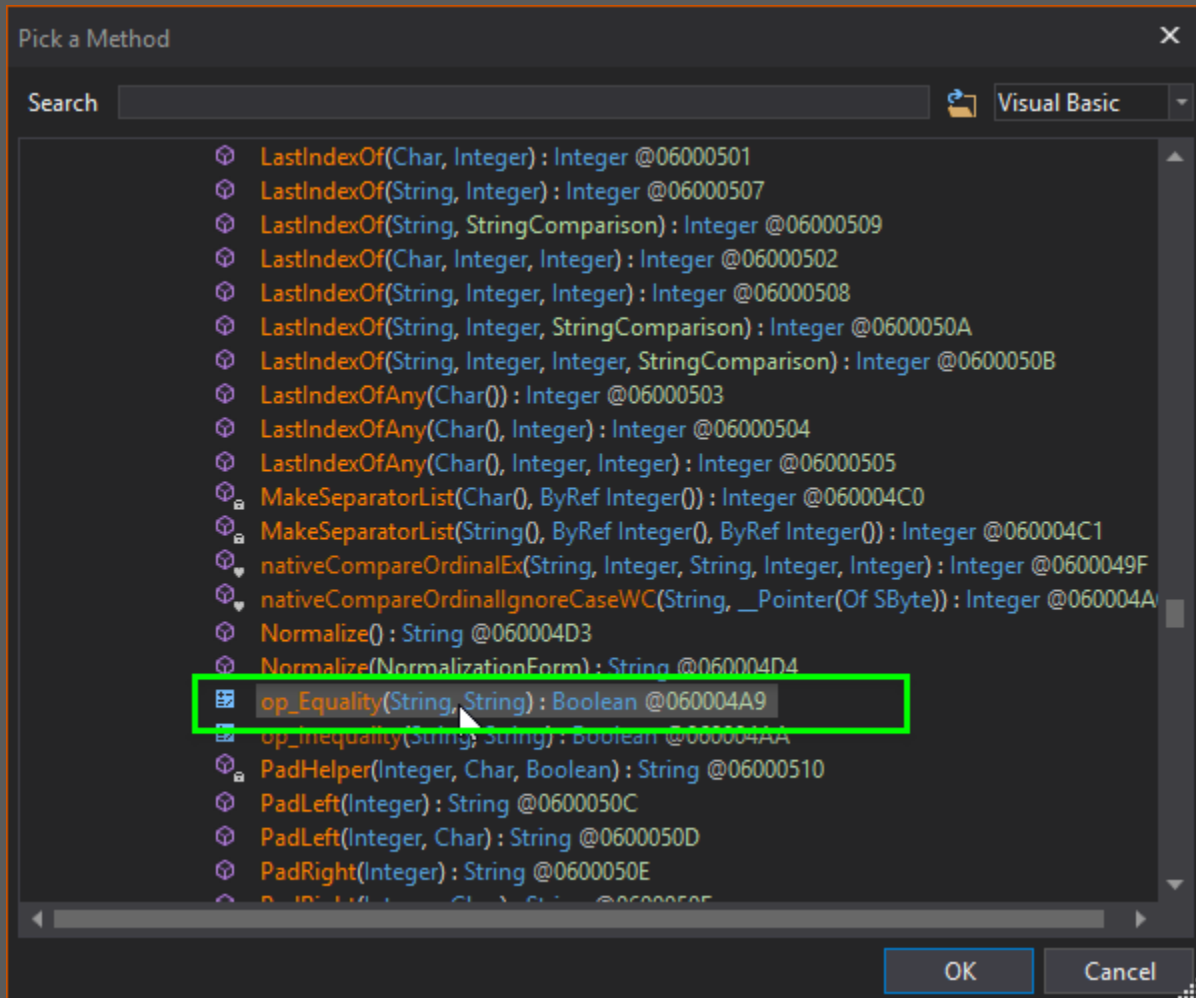


Nos posicionamos sobre el código que deseamos cambiar y clic derecho (secundario) y escogemos la opción **<Edit IL Instructions...>**.

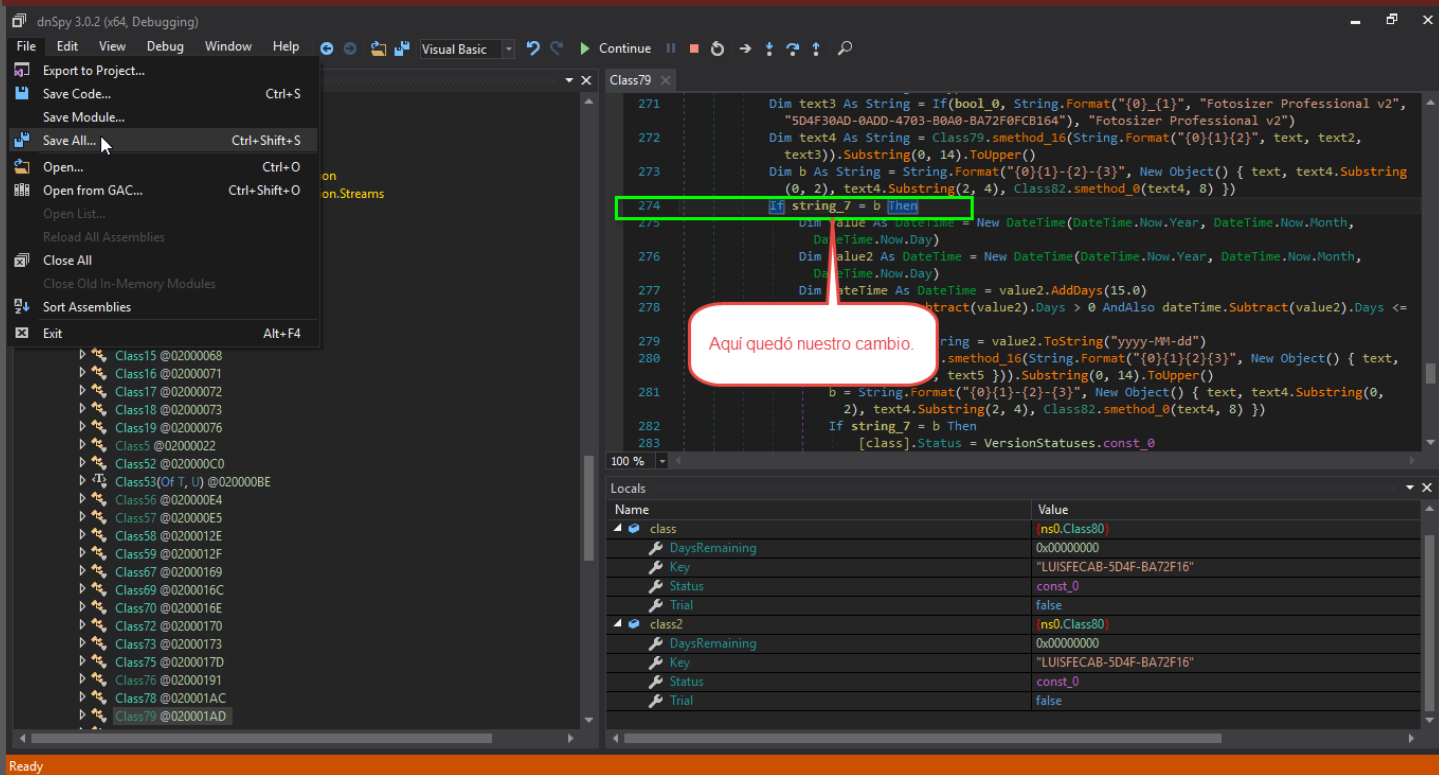


## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

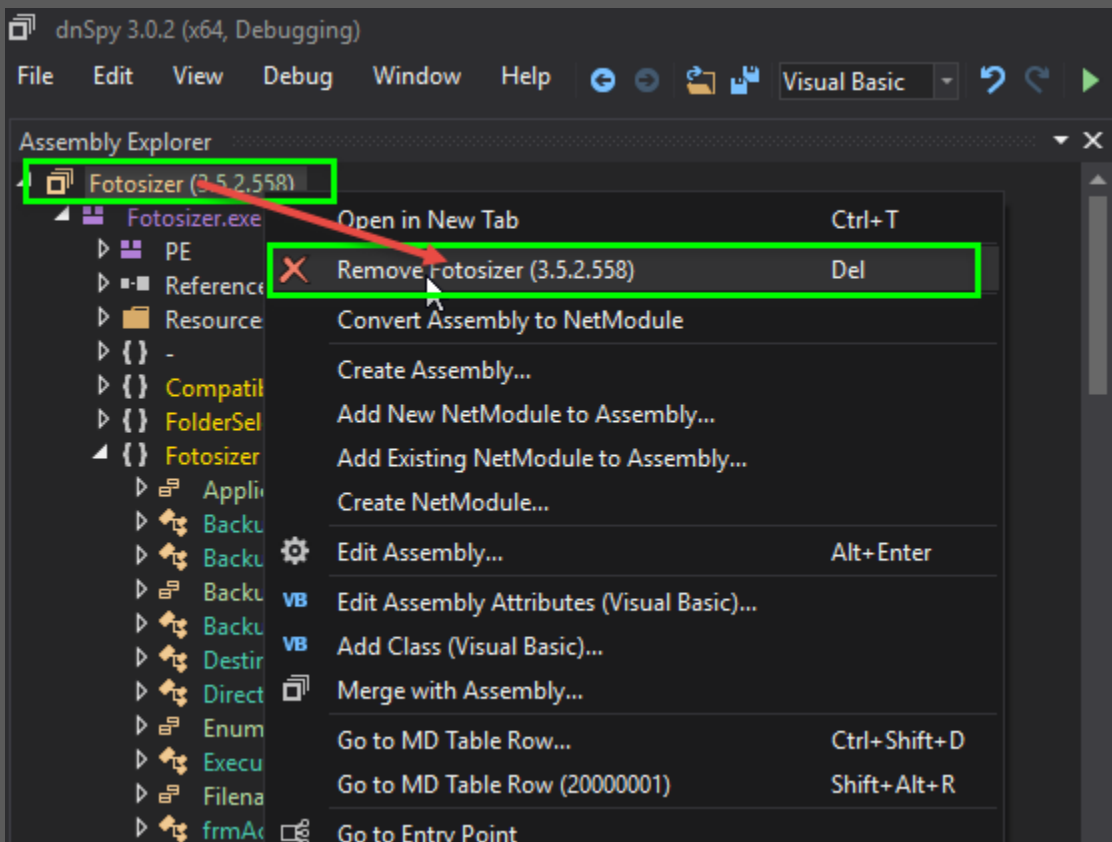
Se nos abre la ventana con el código IL. Damos clic al código a cambiar, en este caso es el index 71 (resaltado azul) y escogemos Method (resaltado rojo) y tendremos la ventana para cambiar la comparación a igualdad.



Seleccionamos la igualdad, OK, y vemos que ya cambió la desigualdad `If string_7 <> b Then` por nuestra igualdad `If string_7 = b Then`. Solo queda guardar los cambios en un nuevo archivo <File->Save All...> y abrirlo.



Es bueno que limpies del **dnSpy** el programa que abrimos y depugeamos primero para no confundirnos. Lo seleccionas del **<Assembly Explorer>**, clic derecho y lo remueves. Como veras en la imagen para próximas ocasiones con la tecla **<Del>**.



## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

Yo guardé el archivo como <Fotosizer-cleaned1.exe>. Lo abrimos, vayamos donde hicimos el cambio y ponemos un <Breakpoint>; ejecutamos el debugger <F5>, colocamos nuestro serial falso y paramos en nuestro <Breakpoint>.

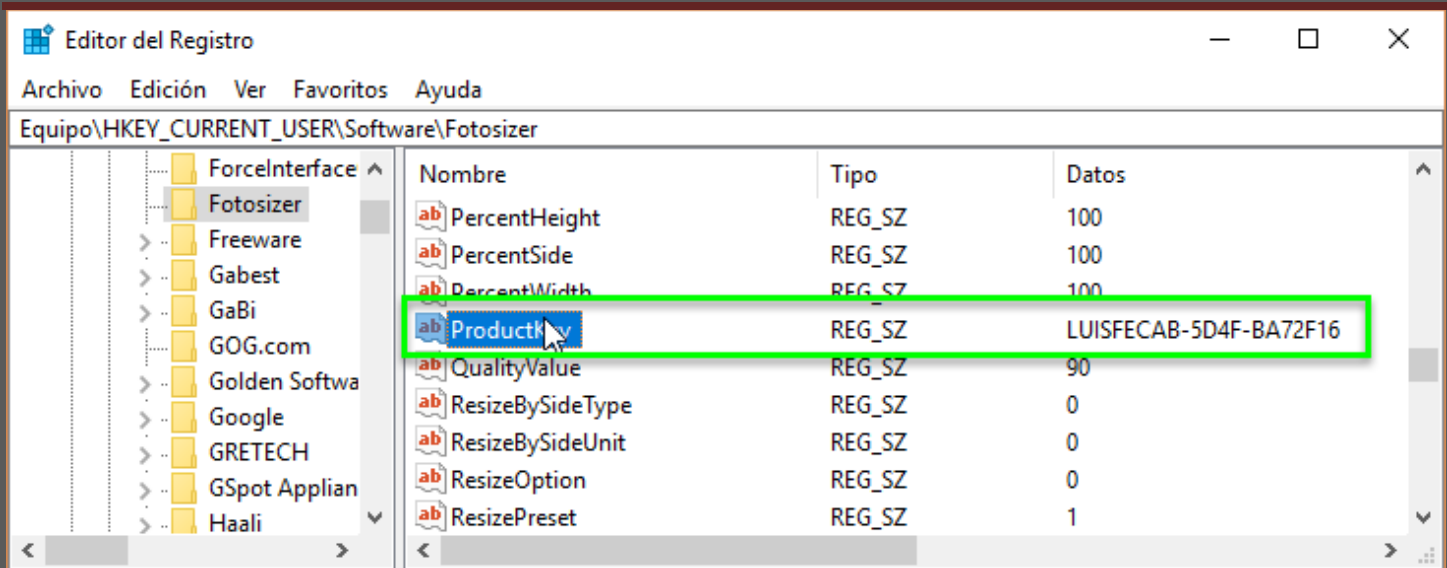
```
264 Private Function smethod_15(string_7 As String, bool_0 As Boolean) As Class80
265     Dim [class] As Class80 = New Class80()
266     [class].Key = string_7
267     [class].Status = VersionStatuses.Invalid
268     [class].Trial = False
269     Dim text As String = string_7.ToUpper().Substring(0, 6).ToUpper()
270     Dim text2 As String = "Typical"
271     Dim text3 As String = If(bool_0, String.Format("{0}_{1}", "Fotosizer Professional v2",
272         "5D4F30AD-0ADD-4703-B0A0-BA72F0FCB164"), "Fotosizer Professional v2")
273     Dim text4 As String = Class79.smethod_16(String.Format("{0}{1}{2}", text, text2, text3)).Substring
274         (0, 14).ToUpper()
275     Dim b As String = String.Format("{0}{1}-{2}-{3}", New Object() { text, text4.Substring(0, 2),
276         text4.Substring(2, 4), Class82.smethod_0(text4, 8) })
277     If string_7 = b Then
278         Dim value As DateTime = New DateTime(DateTime.Now.Year, DateTime.Now.Month, DateTime.Now.Day)
279         Dim value2 As DateTime = New DateTime(DateTime.Now.Year, DateTime.Now.Month, DateTime.Now.Day)
280         Dim dateTime As DateTime = value2.AddDays(15.0)
281         While dateTime.Subtract(value2).Days > 0 AndAlso dateTime.Subtract(value2).Days <= 15
282             Dim text5 As String = value2.ToString("yyyy-MM-dd")
283             text4 = Class79.smethod_16(String.Format("{0}{1}{2}{3}", New Object() { text, text2,
284                 text3, text5 })).Substring(0, 14).ToUpper()
285             b = String.Format("{0}{1}-{2}-{3}", New Object() { text, text4.Substring(0, 2),
286                 text4.Substring(2, 4), Class82.smethod_0(text4, 8) })
287             If string_7 = b Then
288                 [class].Status = VersionStatuses.const_0
289                 [class].Trial = True
290                 [class].DaysRemaining = value2.Subtract(value).Days
291                 Exit While
292             End If
293             value2 = value2.AddDays(1.0)
294         End While
295     Else
296         [class].Trial = False
297         [class].Status = VersionStatuses.const_0
298     End If
299     Return [class]
300 End Function
```

Presionas <F10> para seguir debuguando y de una nos salta a nuestro punto de interés donde línea 291 [class].Trial = False (Ya no es Trial) y línea 292 [class].Status = VersionStatuses.const\_0 (Status tiene la constante de versión registrada).

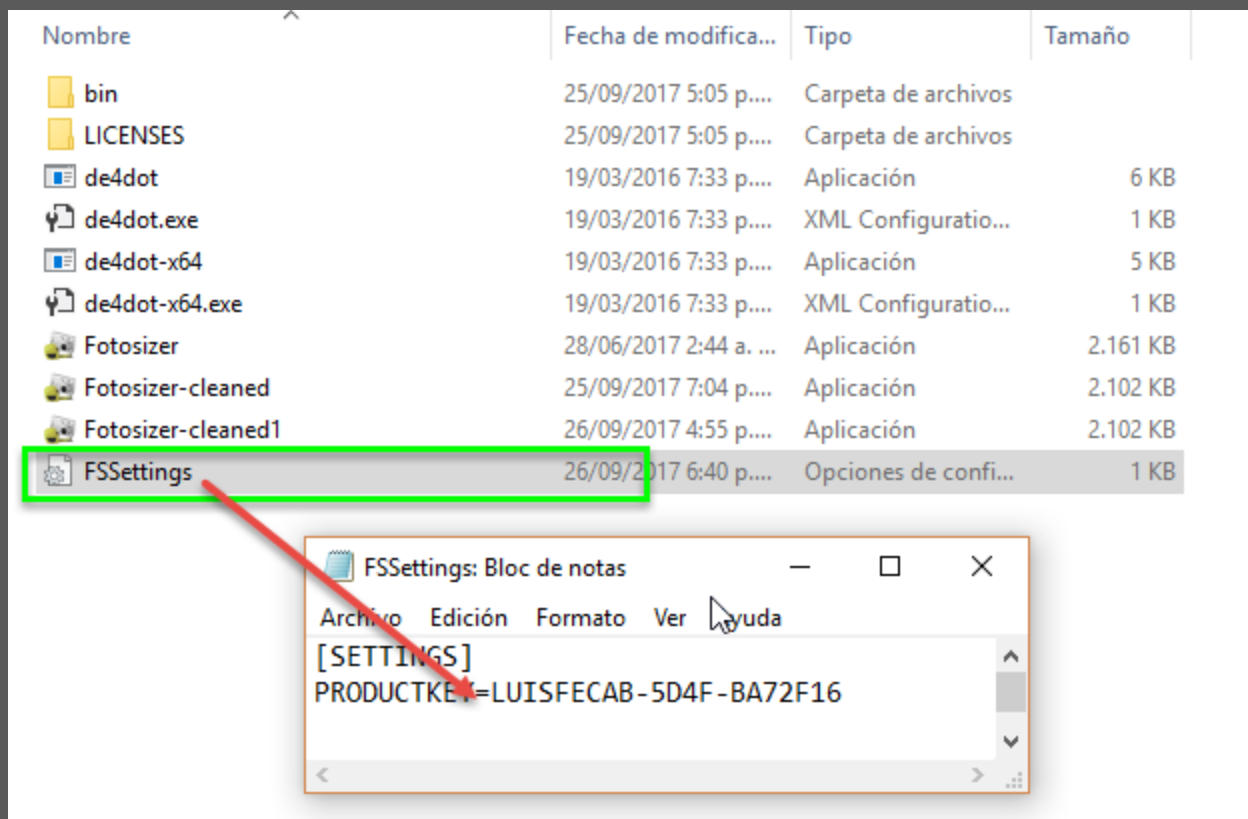
Presiona <F5> y pararas de nuevo en tu <Breakpoint> pero no habrá problema siempre saltarás al lugar correcto y con eso nuestro serial falso será tomado como verdadero, siempre y cuando no estés conectado a internet porque tiene una comprobación del serial por Internet.

Si lo registras desconectado de Internet el serial es tomado correctamente y guarda tu serial en el registro.

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]



Y lo guarda también en un archivo que crea, FSSettings.ini



Podemos solucionar eso siguiendo el siguiente procedimiento cuando instalemos y registremos:

1. Desconectar el Internet.
2. Instalar el programa.
3. Reemplazar el programa con nuestro crack.
4. Iniciar el programa y registrar con cualquier serial que tenga longitud de 22 (LUISFECAB-5D4F-BA72F16)
5. Bloquear el programa desde el Firewall.

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

Bueno, pensaba dejarlo ahí, pero decidí seguir adelante y burlar la comprobación del serial a través de Internet y me encontré con la grata sorpresa que era sencillo, igual como hicimos con el salto del serial y aquí hacemos lo mismo.

```
217 Public Function smethod_12(string_7 As String, bool_0 As Boolean) As Class80
218     Dim [class] As Class80 = New Class80() With { .Key = String.Empty, .Status =
        VersionStatuses.Invalid, .Trial = False, .Proxy = True }
219     If string_7.Length = 22 Then
220         Try
221             [class] = Class79.smethod_13(string_7.ToUpper())
222             If [class] IsNot Nothing AndAlso [class].Status = VersionStatuses.const_0
                AndAlso bool_0 Then
223                 Dim status As VersionStatuses = Class82.smethod_3(string_7)
224                 [class].Status = status
225             End If
226         Catch ex As Exception
227             Console.WriteLine("Error validating license: " + ex.Message)
228             [class].Status = VersionStatuses.Invalid
229         End Try
230     End If
231     Return [class]
232 End Function
```

Estamos en la parte de la longitud de serial

Class82.smethod\_3(string\_7) - Función para comprobar el Serial via Internet.

La función `Class82.smethod_3(string_7)`, que verifica el serial por Internet está dentro de la función `Class79.smethod_12`, que es la donde verifica la longitud del serial. Abramos la función `Class82.smethod_3(string_7)` en otra pestaña para ver el cambio que debemos hacer.

```
42 Public Function smethod_3(string_0 As String) As VersionStatuses
43     If string_0 IsNot Nothing AndAlso string_0.Trim().Length > 0 Then
44         Dim result As VersionStatuses
45         Try
46             Dim text As String = Class82.smethod_6()
47             Dim text2 As String = Class82.smethod_7()
48             Dim args As String() = New String() { string_0, text, text2 }
49             Dim versionStatus As String = New VersionInfo() With { .Proxy =
                WebRequest.DefaultWebProxy, .Url = String.Format("{0}VersionInfo.aspx",
                "http://www.fotosizer.com/") }.GetVersionStatus(args)
50             Dim expr_68 As XmlDocument = New XmlDocument()
51             expr_68.LoadXml(versionStatus)
52             Dim innerText As String = expr_68.DocumentElement.SelectSingleNode("//
                VersionStatusInfo/Status").InnerText
53             If Not(innerText = "0") Then
54                 If Not(innerText = "2") Then
55                     result = VersionStatuses.const_0
56                 Else
57                     result = VersionStatuses.Exceeded
58                 End If
59             Else
60                 result = VersionStatuses.Invalid
61             End If
62         Catch ex_B0 As Exception
63             result = VersionStatuses.const_0
64         End Try
65         Return result
66     End If
67     Return VersionStatuses.const_0
68 End Function
```

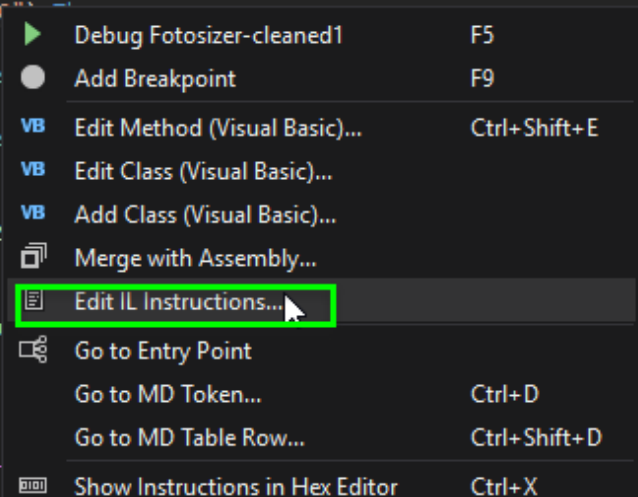
EXCEPCIÓN



## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

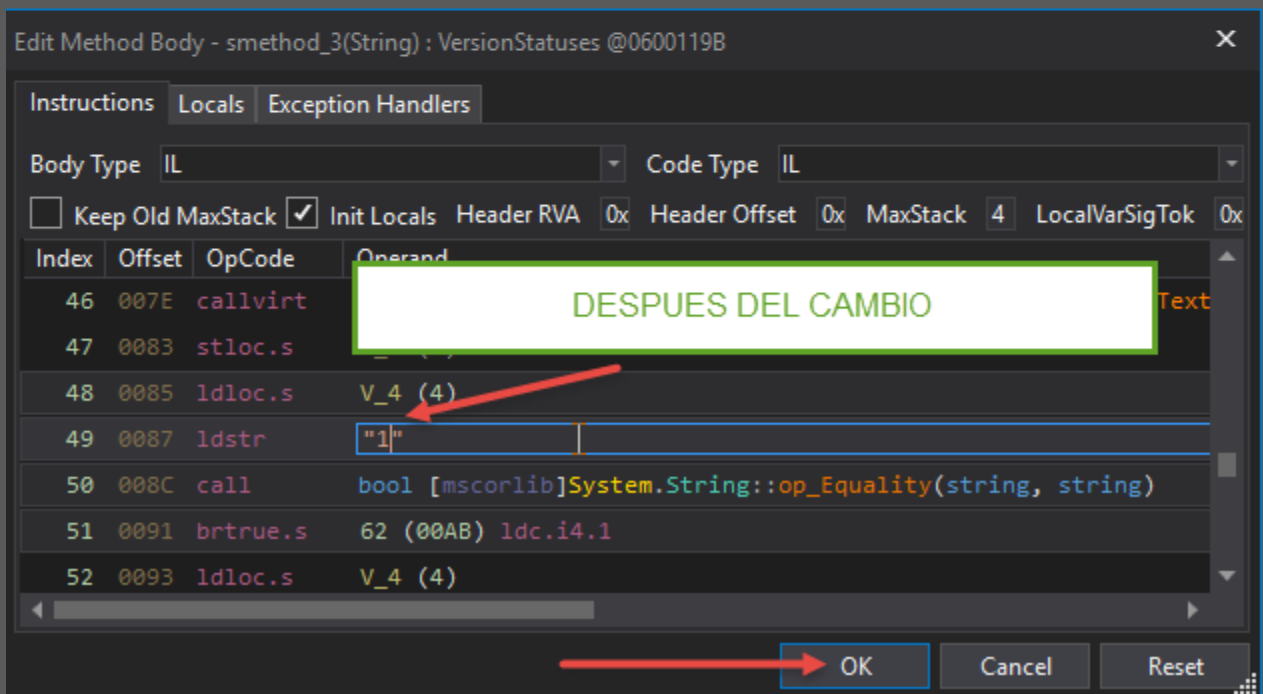
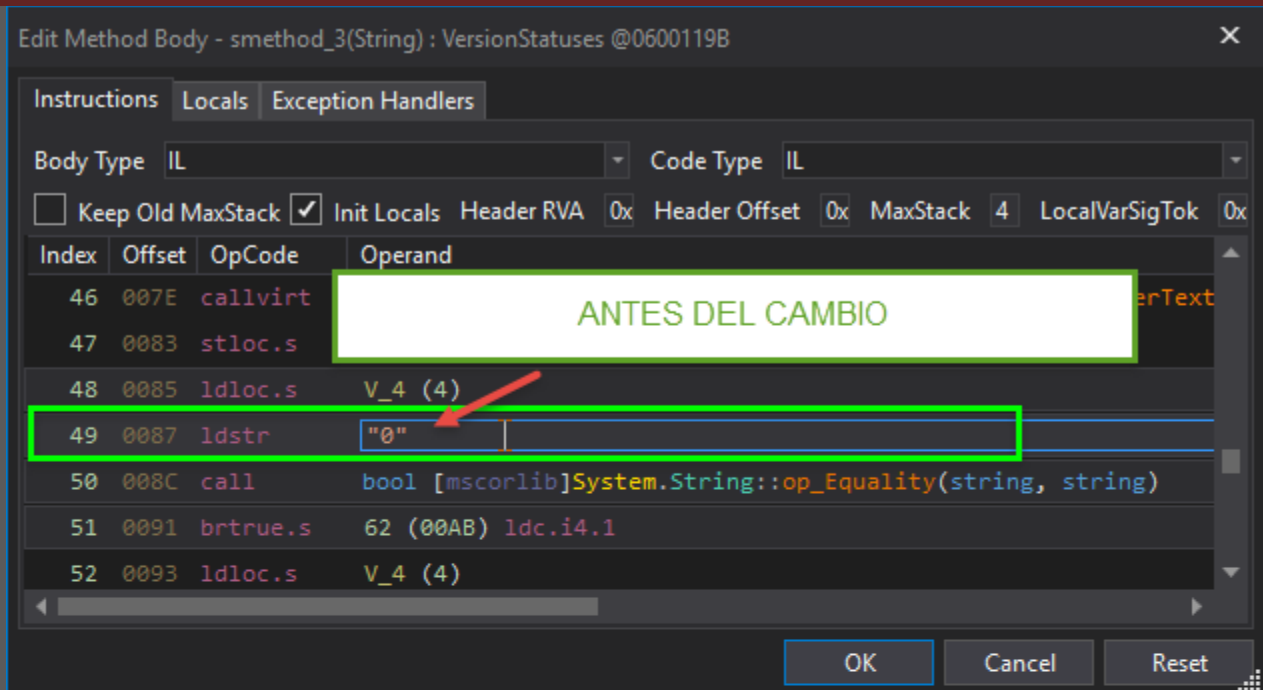
Lo primero que podemos analizar es que la función empiece con un manejador de excepciones, línea 45 Try, y si ocurre cualquier excepción, como cuando no tenemos Internet conectado saltara a la línea 63 `result = VersionStatuses.const_0` que nos deja como si tuviéramos el programa activado. Por eso nuestro serial funciona cuando estamos desconectados de Internet o hemos bloqueado el programa desde el Firewall de Windows. Ahora si estamos conectados es validado y el salto ocurre comparando el valor de la variable `innerText`, así que si `innerText="0"` salta a nuestro <CHICO MALO>. Podemos cambiar la comparación, a que compare a si es diferente, editando de la misma forma que con el serial, quedando `innerText <> "0"` o cambiamos el valor de la constante a hacer comparado y es lo que haré, cambiaré `innerText="0"` por `innerText="1"`. Hagamos el cambio. Nos posicinamos sobre la línea que queremos editar, <Clic Derecho>, escogemos <Edit IL Instructions...>

```
42 Public Function smethod_3(string_0 As String) As VersionStatuses
43     If string_0 IsNot Nothing AndAlso string_0.Trim().Length > 0 Then
44         Dim result As VersionStatuses
45         Try
46             Dim text As String = Class82.smethod_6()
47             Dim text2 As String = Class82.smethod_7()
48             Dim args As String() = New String() { string_0, text, text2 }
49             Dim versionStatus As String = New VersionInfo() With { .Proxy =
                    WebRequest.DefaultWebProxy, .Url = String.Format("{0}VersionInfo.aspx",
                    "http://www.fotosizer.com/") }.GetVersionStatus(args)
50             Dim expr_68 As XmlDocument = New XmlDocument()
51             expr_68.LoadXml(versionStatus)
52             Dim innerText As String = expr_68.DocumentElement.SelectSingleNode("//
                    VersionStatusInfo/Status").InnerText
53             If Not(innerText = "0") Then
54                 If Not(innerText
55                     result = Ver
56                 Else
57                     result = Ver
58                 End If
59             Else
60                 result = Version
61             End If
62             Catch ex_B0 As Exception
63                 result = VersionStat
64             End Try
65             Return result
66         End If
67         Return VersionStatuses.const_
68     End Function
```



Y cambiamos el valor constante 0 por 1.

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

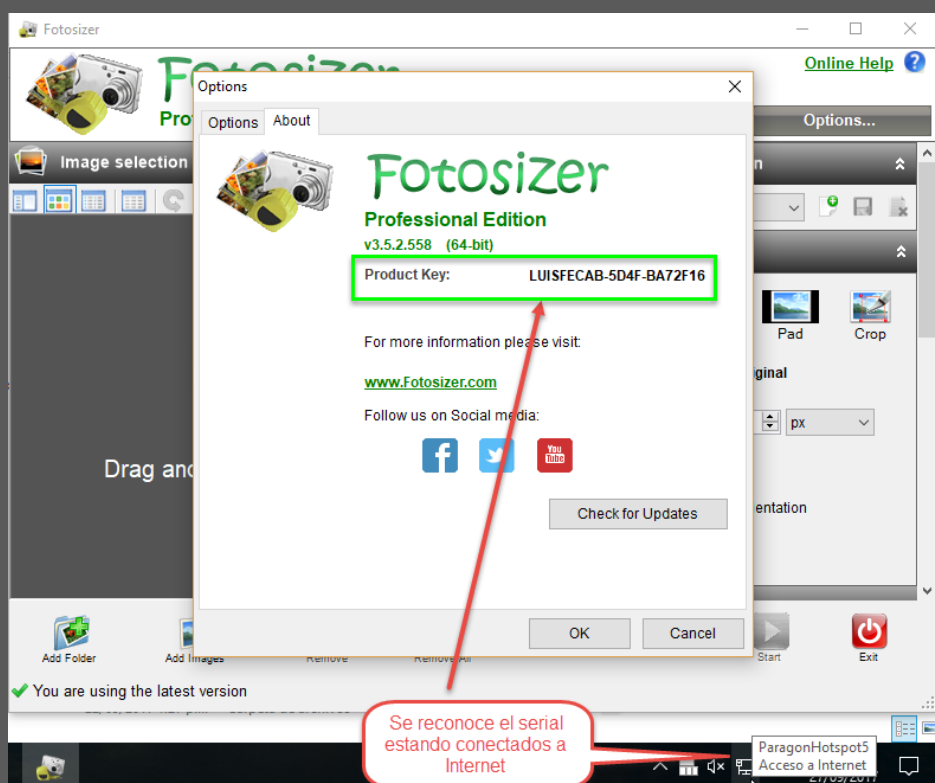


Y nos queda la línea 53 `If Not(innerText = "1") Then.`

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

```
42 Public Function smethod_3(string_0 As String) As VersionStatuses
43     If string_0 IsNot Nothing AndAlso string_0.Trim().Length > 0 Then
44         Dim result As VersionStatuses
45         Try
46             Dim text As String = Class82.smethod_6()
47             Dim text2 As String = Class82.smethod_7()
48             Dim args As String() = New String() { string_0, text, text2 }
49             Dim versionStatus As String = New VersionInfo() With { .Proxy =
                    WebRequest.DefaultWebProxy, .Url = String.Format("{0}VersionInfo.aspx",
                    "http://www.fotosizer.com/") }.GetVersionStatus(args)
50             Dim expr_68 As XmlDocument = New XmlDocument()
51             expr_68.LoadXml(versionStatus)
52             Dim innerText As String = expr_68.DocumentElement.SelectSingleNode("//
                    VersionStatusInfo/Status").InnerText
53             If Not(innerText = "1") Then
54                 If Not(innerText = "2") Then
55                     result = VersionStatuses.const_0
56                 Else
57                     result = VersionStatuses.Exceeded
58                 End If
59             Else
60                 result = VersionStatuses.Invalid
61             End If
62             Catch ex_B0 As Exception
63                 result = VersionStatuses.const_0
64             End Try
65             Return result
66         End If
67         Return VersionStatuses.const_0
68     End Function
```

Guardamos los cambios en un nuevo archivo que sería nuestro Crack 100% funcional. Lo ejecutamos con el Internet conectado y funciona perfecto.



# CREANDO EL PATCH

Me siento animado a hacer algo más y quiero hacer un Patch para el programa. Este será el primer Patch que me animo a hacer y lo haré de la forma sencilla comparando el original con el que yo crackie. Utilizaré el **<dUP2 Diablo's Universal Patcher v2.26>**

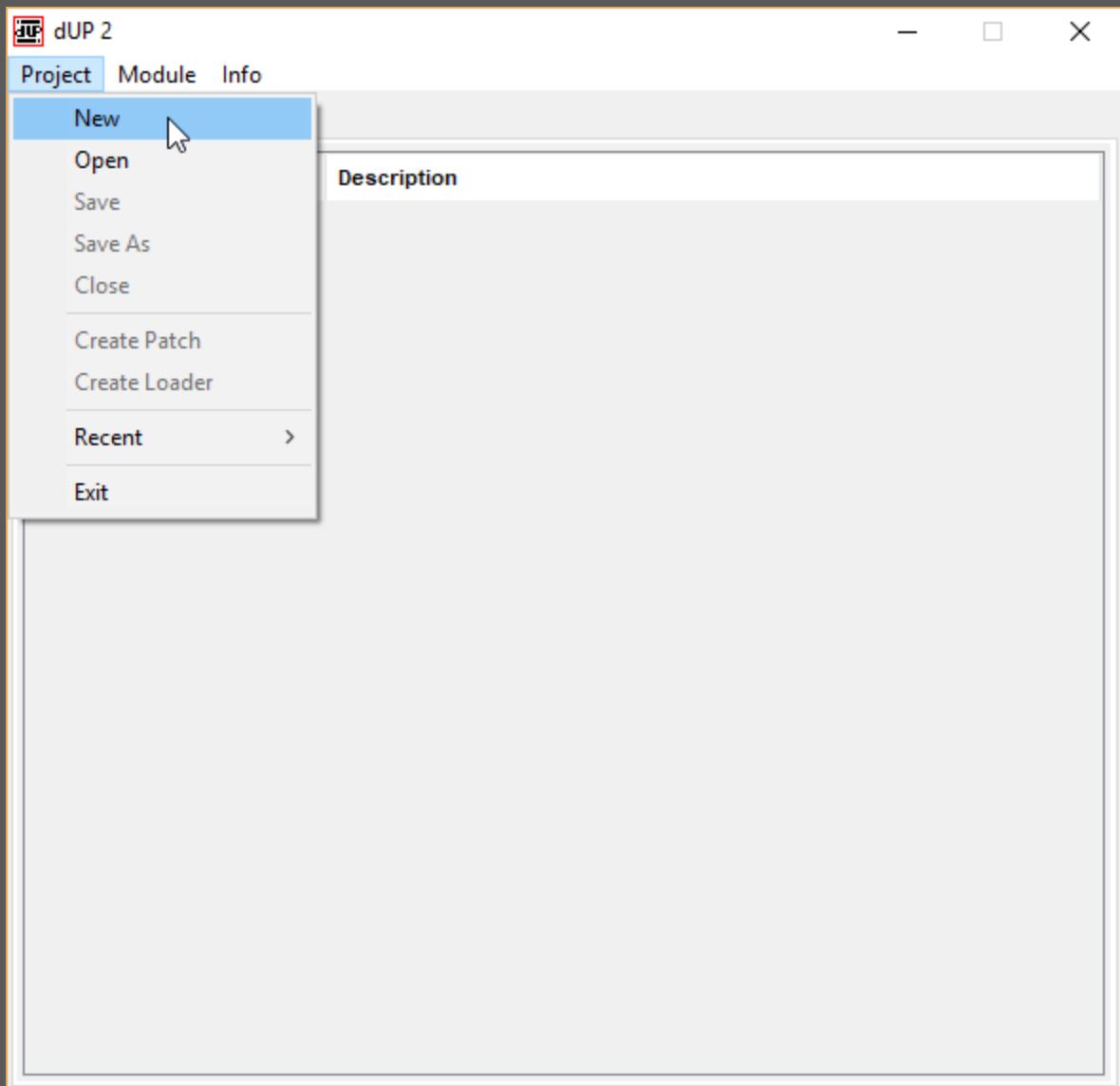
Recomiendo un tutorial de ShaDDy que nos explica el funcionamiento del dUP2.

[\[1009\] dUP2.Diablo's Universal Patcher v2.17 By ShaDDy](#)

También les dejo otro tuto de DavicoRm donde crea un Patch y nos sirve para profundizar.

[Tutorial SGTaller v3.xx Universal Patch by DavicoRm](#)

Ejecuto el dUP2. Vamos a **<Project->New>**



## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

Agrego los datos que quiero y claro está el programa original a ser parcheado.

**Patch Info**

Patcher Caption: LUISFECAB - CracksLatinoS

Application: Fotosizer v3.5.2.558 Patch

Filename (s): Fotosizer.exe

URL: <http://ricardo.crver.net/>

Author: LUISFECAB

Release Date: Septiembre 27, 2017

Release Info: "CrackSLatinoS"  
LUISFECAB  
Colombia  
2017

About Box Message: Fotosizer v3.5.2.558 Patch  
LUISFECAB  
Disfruta de este Patch

Scrolltext:

☒ Show this dialog when create a new project

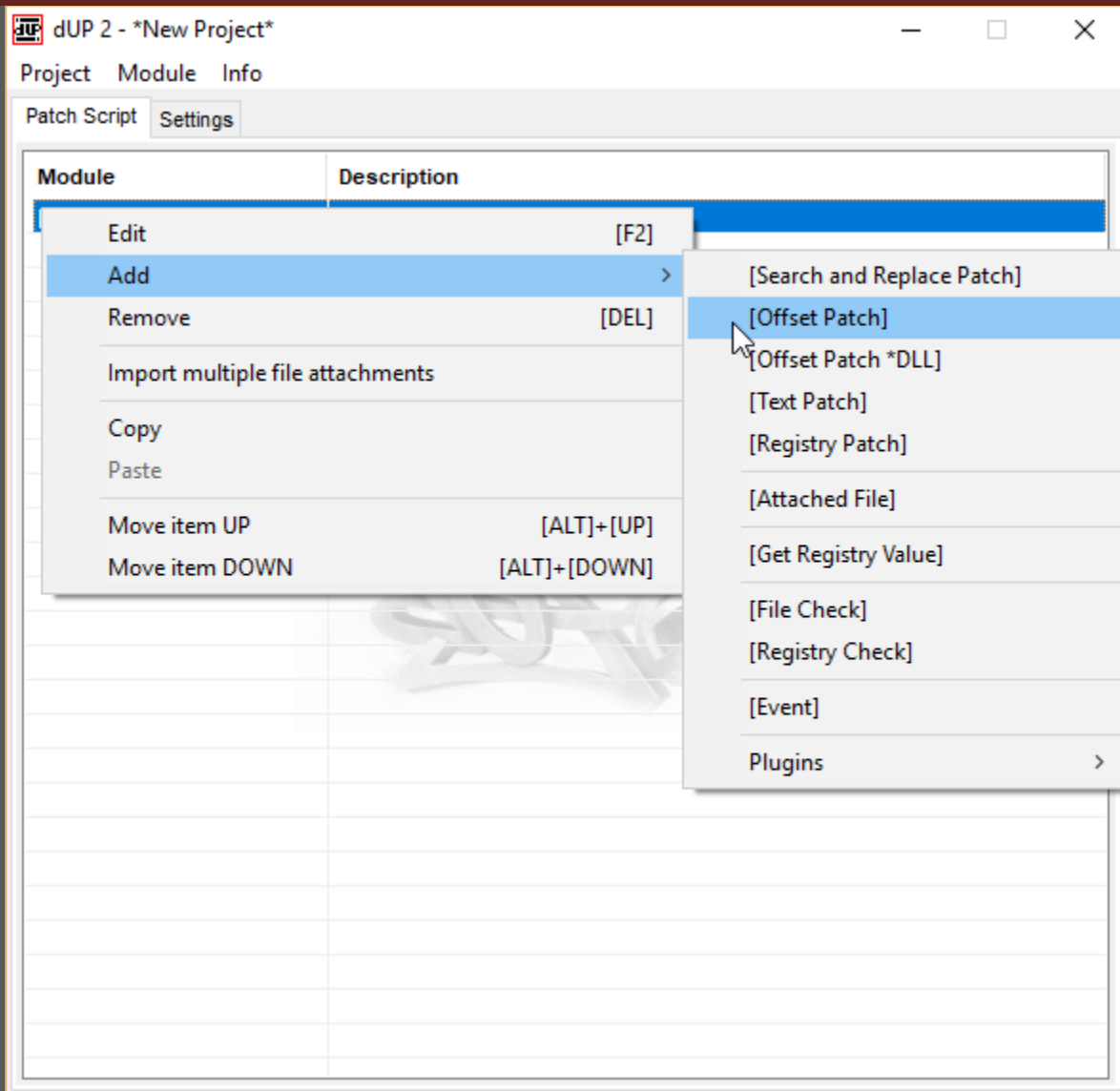
☒ Run patch with administrator rights

☐ No Backup by default

Cancel Save

Una vez salvo el proyecto con **<Save>**, me queda la siguiente ventana, que para tener acceso a las opciones de **<Editar>**, **<Agregar>**, **<Remover>**, **<Mover>**...etc, lo hago con **<Click Derecho>**. De esa forma escojo la forma de crear mi Patch, y escojo **<[Offset Patch]>**.

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]



En el tuto de ShaDDy se explica muy bien este procedimiento. Tenemos el **<[Offset Patch]>** en la ventana principal, **<Doble Clic>** sobre este y se nos abre una ventana, ahí coloco y agrego todo lo necesario.



## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

Offset Patchdata

Target File  
File: %SystemDrive%\Fotosizer\Fotosizer.exe

Old Filesize: 0021C400 New Filesize: 0020D600  
CRC32: F3A524C0 ☒ Enable CRC32 check  
☐ Don't check original (old) bytes ☐ Try patching used file  
☐ Keep original file time ☐ Fix PE CheckSum  
☐ Disable WOW64 FS Redirector

Patchmode  
☒ File Offset  
☐ Virtual Address (VA)  
☐ Relative Virtual Address (RVA)

Compare Files  
Original File: C:\Program Files\Fotosizer\Fotosizer.exe  
Patched File: D:\DD 500GB\My Files\Curso de craking\Herramientas\de4dot v3.1.4159  
Compare

RAW Offset	Old Byte	New Byte
00000097	00	01
0000009D	5A	6C
0000009E	1F	1E
000000A1	C0	68
000000A2	21	02
000000A8	0E	6E
000000A9	78	8A
000000AA	1F	1E
000000B1	80	A0
000000B2	1F	1E
000000D1	20	40
000000D2	22	21
00000100	B4	18
00000101	77	8A

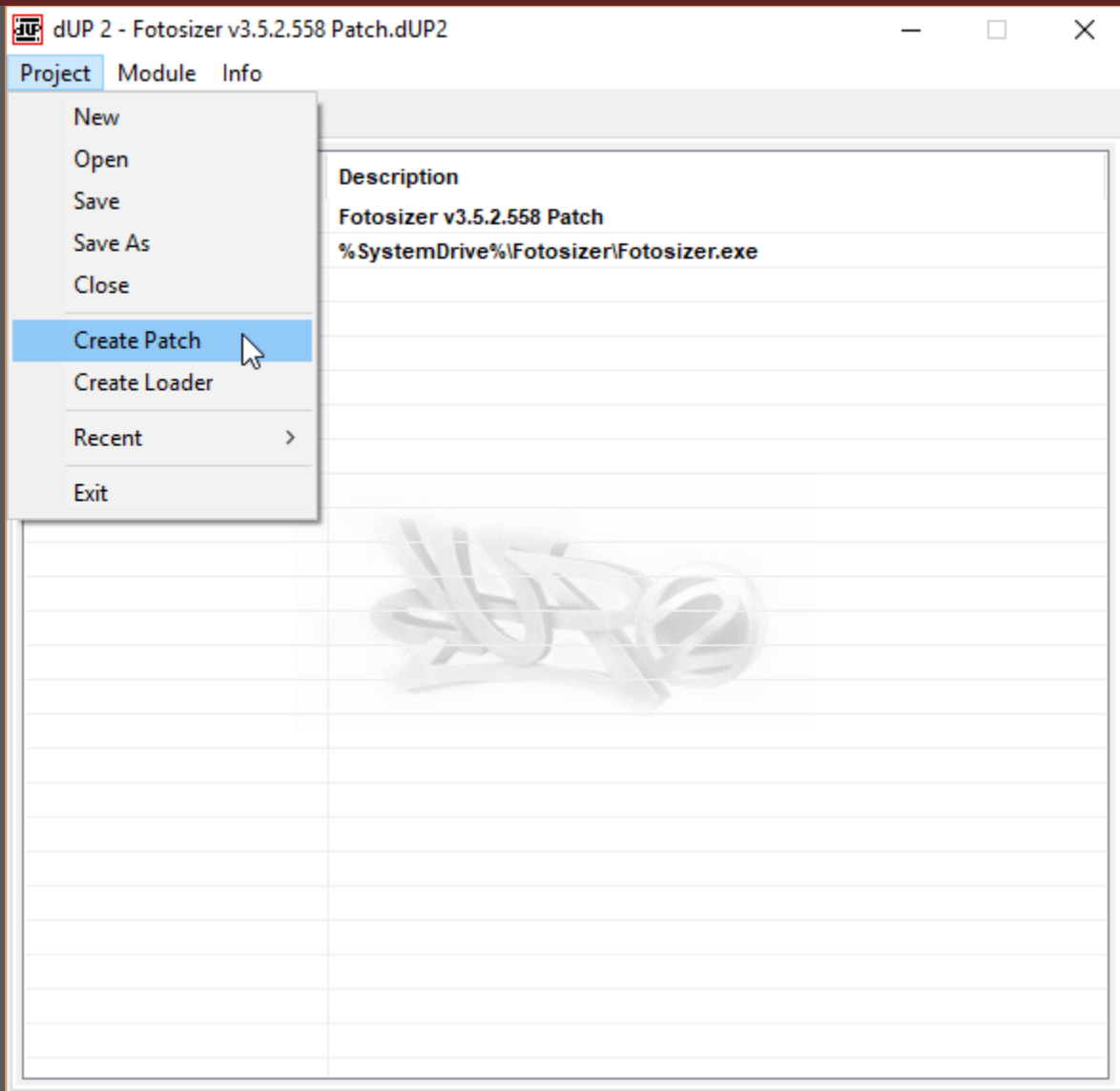
Add and Edit  
Offset:   
Original Byte:   
Patched Byte:   
Edit Add  
Remove Clear List

MemCheck [for Loaders]  
☐ Enable MemCheck ?  
MemoryAddress:   
MemoryValue:   
Up Down  
Items: 2046408

File to load in Ollydbg:  Follow in Ollydbg  
Cancel Save

Se puede observar que son demasiados cambios y eso me llamó mucho la atención en su momento porque yo solo cambié solo dos cositas dentro del programa, más sin embargo lo guardé con <Save> y creé mi patch <Project->Create Patch> con el nombre <fotosizer.v3.5.2.558.patch.exe>.

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]




Muy contento, lo probé y funcionó; solo estaba un detalle que y cuando miré su tamaño era de 18.8 MB. No había caído en cuenta y mi teoría es, y estoy seguro que no me equivoco, que como compara el archivo <original ofuscado> con mi <archivo crackeado desofuscado> entonces son muchos los reemplazos y por eso el tamaño tan grande.

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

Nombre	Fecha de modifica...	Tipo	Tamaño	
bin	25/09/2017 5:05 p....	Carpeta de archivos		
LICENSES	25/09/2017 5:05 p....	Carpeta de archivos		
de4dot	19/03/2016 7:33 p....	Aplicación	6 KB	
de4dot.exe	19/03/2016 7:33 p....	XML Configuratio...	1 KB	
de4dot-x64	19/03/2016 7:33 p....	Aplicación	5 KB	
de4dot-x64.exe	19/03/2016 7:33 p....	XML Configuratio...	1 KB	
Fotosizer v3.5.2.558 Patch	27/09/2017 6:29 p....	Archivo DUP2	11.992 KB	
Fotosizer	28/06/2017 2:44 a....	Aplicación	2.161 KB	
fotosizer.v3.5.2.558.patch	27/09/2017 6:11 p....	Aplicación	19.272 KB	
Fotosizer8	27/09/2017 6:50 p....	Aplicación	2.119 KB	
Fotosizer-cleaned	25/09/2017 7:04 p....	Aplicación	2.102 KB	
Fotosizer-cleaned1	26/09/2017 4:55 p....	Aplicación	2.102 KB	
Fotosizer-cleaned2	26/09/2017 7:29 p....	Aplicación	2.102 KB	
Fotosizer-cleaned3	27/09/2017 3:05 p....	Aplicación	2.102 KB	
FSSettings	26/09/2017 7:31 p....	Opciones de confi...	1 KB	

fotosizer.v3.5.2.558.patch

Aplicación



Fecha de modificación: 27/09/2017 6:11 p. m.  
Tamaño: 18,8 MB  
Fecha de creación: 27/09/2017 6:11 p. m.  
Disponibilidad: Disponible sin conexión

De aquí en adelante escribo el tuto mientras pruebo mi teoría. Entonces lo que debo hacer, es hacer (perdón por la redundancia) los cambios en el ejecutable ofuscado para que cuando haga la comparación solo capte mis dos cambios.

Trabajar con el archivo ofuscado se complica un poco, pero por fortuna ya conocemos el código y podemos ir a las funciones que debemos parchear. Abajo las capturas con los cambios.

```
403 Private Function cc90ad79c2e0cf4d5c005c9275cd53c84(cb12490b51f7d3ef652a07ccd19bf2ddb As
String, cb78ae9bd6fe07f581c3f62b2a42b7d9e As Boolean) As
c0c1bcb0ff3c933e31d33c8adc048c2c6
404 Dim c0c1bcb0ff3c933e31d33c8adc048c2c As c0c1bcb0ff3c933e31d33c8adc048c2c6 = New
c0c1bcb0ff3c933e31d33c8adc048c2c6()
405 c0c1bcb0ff3c933e31d33c8adc048c2c.Key = cb12490b51f7d3ef652a07ccd19bf2ddb
406 c0c1bcb0ff3c933e31d33c8adc048c2c.Status = VersionStatuses.Invalid
407 c0c1bcb0ff3c933e31d33c8adc048c2c.Trial = False
408 Dim text As String = cb12490b51f7d3ef652a07ccd19bf2ddb.ToUpper().Substring(0,
6).ToUpper()
409 Dim text2 As String =
ce645c335b15ba33b58a4e03f19089d69.ce72d84ff6652332c40693c535f893782(88846)
410 Dim text3 As String = If(cb78ae9bd6fe07f581c3f62b2a42b7d9e, String.Format
(ce645c335b15ba33b58a4e03f19089d69.ce72d84ff6652332c40693c535f893782(88912),
ce645c335b15ba33b58a4e03f19089d69.ce72d84ff6652332c40693c535f893782(88861),
ce645c335b15ba33b58a4e03f19089d69.ce72d84ff6652332c40693c535f893782(88927)),
ce645c335b15ba33b58a4e03f19089d69.ce72d84ff6652332c40693c535f893782(88861))
411 Dim text4 As String =
c0ad817653f09e60fcd1dba949bc7235e6.c9677642ae96f2ab05de8e645a7fcb074(String.Format
(ce645c335b15ba33b58a4e03f19089d69.ce72d84ff6652332c40693c535f893782(28206), text,
text2, text3)).Substring(0, 14).ToUpper()
412 Dim b As String = String.Format
(ce645c335b15ba33b58a4e03f19089d69.ce72d84ff6652332c40693c535f893782(89000), New
Object() { text, text4.Substring(0, 2), text4.Substring(2, 4),
c650ff0e2a6daf8f2a1e7d943067492c0.c34782dd016fb3cdd0ddc4bfb44c80dc(text4 - 8) })
413 If cb12490b51f7d3ef652a07ccd19bf2ddb = b Then
414 While True
415 Select Case 6
416 Case 0
417 Continue While
418 End Select
419 Exit While
```

Primer parcheo

ControlFlow

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

En segundo parcheo era cambiar "0" por "1"., que era la comprobación vía Internet y me encuentro que la constante "1" no está cargada directamente, si no que la saca de un **<CALL>**.

```
if innerText = ce645c335b15ba33b58a4e03f19089d69.ce72d84ff6652332c40693c535f893782(27709)
```

Ahí lo vemos clarito es una función que calcula el "<0>" a partir de del número <27709> y que en hexadecimal es <0x6C3D>. Entonces si cambio ese valor obtendré un valor diferente y eso es lo que necesitamos, lo cambiaré por <0x1>. Hagamos los cambios.

Edit Method Body - cbf05d386895e7050b0622b7a4e574ebe(String) : VersionStatuses @060011A7

Instructions Locals Exception Handlers

Body Type IL

☐ Keep Old MaxStack ☒ Init Locals Header RVA 0x9 LocalVarSigTok 0x1100023F

**ANTES DEL CAMBIO**

Index	Offset	OpCode	Operand
74	00B8	ldloc.s	V_4 (4)
75	00BA	ldc.i4	0x6C3D
76	00BF	call	string A.ce645c335b15ba33b58a4e03f19089d69::ce72d84ff6652332c40693c535f893782(int32)
77	00C4	dup	
78	00C5	pop	
79	00C6	call	bool [mscorlib]System.String::op_Equality(string, string)
80	00CB	brtrue.s	91 (00EE) ldc.i4.1

OK Cancel Reset

Edit Method Body - cbf05d386895e7050b0622b7a4e574ebe(String) : VersionStatuses @060011A7

Instructions Locals Exception Handlers

Body Type IL

☐ Keep Old MaxStack ☒ Init Locals Header RVA 0 LocalVarSigTok 0x1100023F

**DESPUES DEL CAMBIO**

Index	Offset	OpCode	Operand
74	00B8	ldloc.s	V_4 (4)
75	00BA	ldc.i4	0x1
76	00BF	call	string A.ce645c335b15ba33b58a4e03f19089d69::ce72d84ff6652332c40693c535f893782(int32)
77	00C4	dup	
78	00C5	pop	
79	00C6	call	bool [mscorlib]System.String::op_Equality(string, string)
80	00CB	brtrue.s	91 (00EE) ldc.i4.1

**GUARDAMOS LOS CAMBIOS** → OK Cancel Reset

Listo, ahí cambiamos el valor y seguro el valor que retornará ese **<CALL>** ya no será "0". Guardemos los cambios del código IL con **<OK>**.

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

```
73      If innerText =  
74          ce645c335b15ba33b58a4e03f19089d69.ce72d84ff6652332c40693c535f893782(1) Then  
75          result = VersionStatuses.Invalid  
76          Return result  
77      End If  
78      If Not(innerText =  
79          ce645c335b15ba33b58a4e03f19089d69.ce72d84ff6652332c40693c535f893782(27712))  
80      Then  
81          While True  
82              Select Case 3  
83                  Case 0  
84                      Continue While  
85              End Select  
86          End While  
87          Exit While  
88      End While  
89      result = VersionStatuses.OK
```

Segundo Parcheo

Listo, ya tenemos nuestro segundo parcheo. Ahora guardamos nuestro archivo ofuscado parcheado, lo guardé como <Fotosizer8.exe>.

Yo no se ustedes, pero tengo curiosidad saber el valor de retorno del <CALL>, para hacer eso ponemos un <Breakpoint> en nuestro segundo parcheo y traceamos con <F11> para entrar a ese <CALL> que fijo hace parte del ofuscamiento porque el archivo desofuscado no lo tiene.

```
72      Dim [string] As String = Encoding.Unicode.GetString  
73      (ce645c335b15ba33b58a4e03f19089d69.c68e49d64562aad9d43bd76bf30fafb79,  
74      c7d00945a1d060d6819835cba2485040b, num)  
75      Return String.Intern([string])  
76      End Function  
77      Friend Shared c68e49d64562aad9d43bd76bf30fafb79 As Byte()  
78      End Class  
79      End Namespace
```

Locals


Name	Value
c7d00945a1d060d6819835cba2485040b	0x00000002
num	0x00000012
string	"{0} ({1})"

Ahí lo podemos ver en la imagen anterior nada parecido a <"0">.

Bueno llegó la hora de hacer de nuevo el patch con el <Ofuscado Original> y el <Ofuscado Parcheado>. No voy a hacer eso en el tuto, ya está más que claro el procedimiento. A cruzar los dedos.

## [Tutorial - Fotosizer v3.5.2.558 (Crypto Obfuscator for .NET v5.x)]

Nombre	Fecha de modifica...	Tipo	Tamaño	fotosizer.v3.5.2.558.patch.ofuscado-patch
bin	25/09/2017 5:05 p....	Carpeta de archivos		Aplicación
LICENSES	25/09/2017 5:05 p....	Carpeta de archivos		
de4dot	19/03/2016 7:33 p....	Aplicación	6 KB	
de4dot.exe	19/03/2016 7:33 p....	XML Configuratio...	1 KB	
de4dot-x64	19/03/2016 7:33 p....	Aplicación	5 KB	
de4dot-x64.exe	19/03/2016 7:33 p....	XML Configuratio...	1 KB	
Fotosizer v3.5.2.558 Patch Ofuscado	28/09/2017 1:40 p....	Archivo DUP2	12.054 KB	
Fotosizer v3.5.2.558 Patch	27/09/2017 6:29 p....	Archivo DUP2	11.992 KB	
Fotosizer	28/06/2017 2:44 a. ...	Aplicación	2.161 KB	
fotosizer.v3.5.2.558.patch	27/09/2017 6:11 p....	Aplicación	19.272 KB	
fotosizer.v3.5.2.558.patch.ofuscado-patch	28/09/2017 1:40 p....	Aplicación	12.115 KB	
Fotosizer8	28/09/2017 12:48 ...	Aplicación	2.118 KB	
Fotosizer-cleaned	25/09/2017 7:04 p....	Aplicación	2.102 KB	
Fotosizer-cleaned1	26/09/2017 4:55 p....	Aplicación	2.102 KB	
Fotosizer-cleaned2	26/09/2017 7:29 p....	Aplicación	2.102 KB	
Fotosizer-cleaned3	27/09/2017 3:05 p....	Aplicación	2.102 KB	
FSSettings	26/09/2017 7:31 p....	Opciones de confi...	1 KB	



Fecha de modificación: 28/09/2017 1:40 p. m.  
Tamaño: 11,8 MB  
Fecha de creación: 28/09/2017 1:40 p. m.  
Disponibilidad: Disponible sin conexión

No nada, sigue extremadamente pesado. Mi teoría no es válida...Ploop. El patch funciona bien, lo que no entiendo es ese tamaño tan excesivo. La verdad es que hasta ahí dejo el Patch, ya funciona; es tremendamente pesado pero que sirve, sirve.

## PARA TERMINAR

Me he entusiasmado con colocar muchas imágenes y no se si fui muy redundante en algunas partes tratando de explicar algo, es que trataba de auto explicarme para mejorar los fundamentos en entender los .NET debugado con el **dnSpy**.

Me queda pendiente lograr un patch que no sea tan pesado y que solo cambie los bytes de mi parcheo y no esa cantidad tan grande.

Espero el tuto sea de ayuda y que pueda aclarar dudas, aportar algo nuevo a la comunidad.

Saludos a la comunidad.