第十五章：调用栈的使用

一、简介

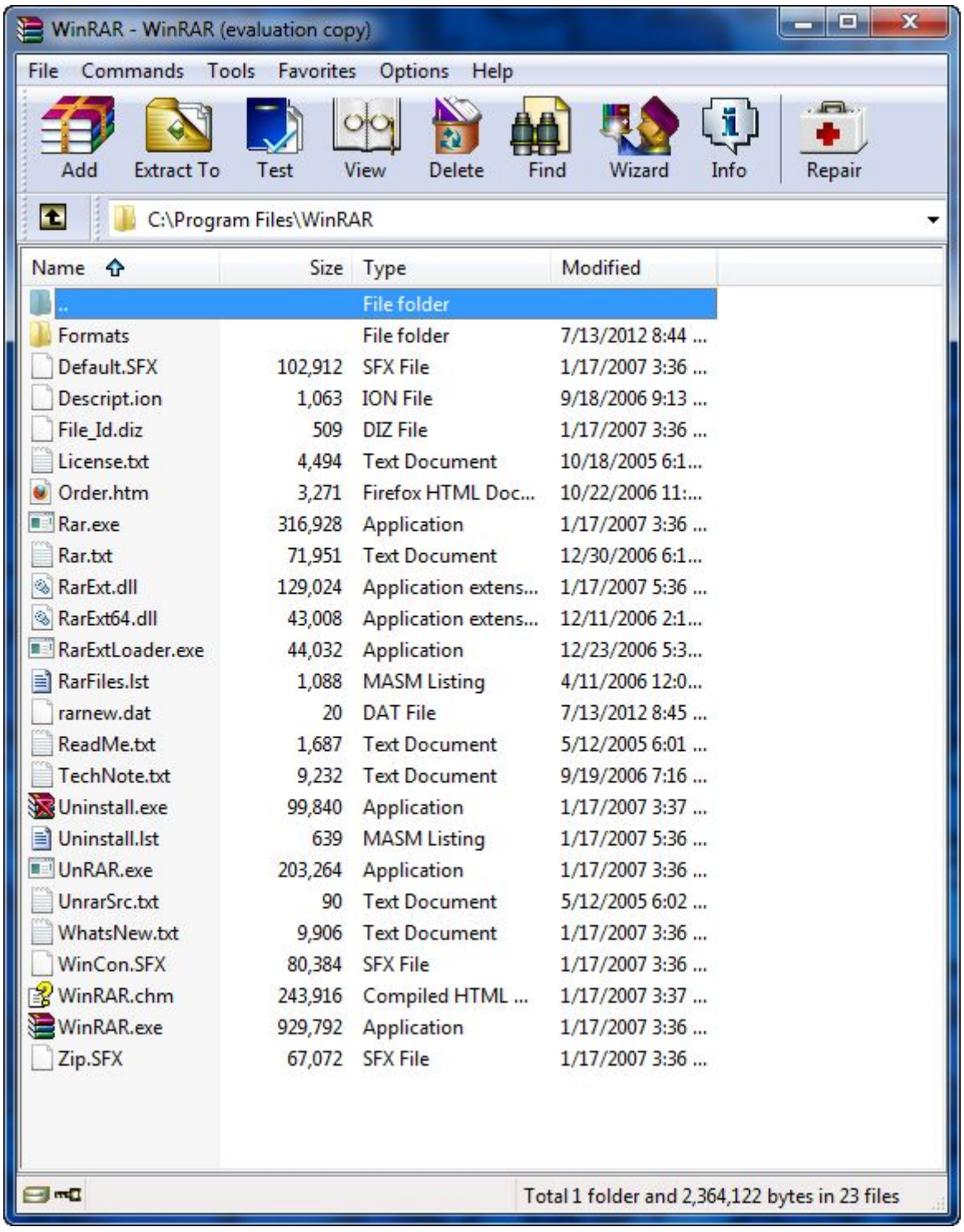
本章我们会删除一个“真正”的程序的 nag 窗口。为了试图帮助作者，因为他们花了大量的时间来创造这些应用，我试着挑出一个能将伤害降到最低的应用。这次，我用 Google 搜索了下“Cracked Software”，这个程序有着最高的点击率，包括教程、序列号、keygen，应有尽有。因为获得该软件的破解版是如此的简单，我想不管怎么获得它都不太可能有麻烦。不过我们请求你，如果你确实喜欢它，那就买它。

我也会添加一些技巧到咱们的逆向兵器库中。有一点需要注意，如果你是在 64 位 Windows7 下学习本章（像我一样），Olly1.1 版甚至我的版本，调用栈这招就不好用了。我的建议是，做我所做的：用 Olly2.0 版来学习新的技巧（取得正确的地址），然后再转到我所用的 Olly 来做其他的。或者就用 Olly2.0 版，它有很多很好的特性，它可以在 64 位操作系统下工作。（译者注：既然有了 Olly2.0，那我们为什么在平时用的时候还是以 1.1 的版本居多呢？因为 1.1 版的 Olly 拥有大量的插件，这给破解带来了很大的便利，而 2.0 版的插件则弱了很多。）

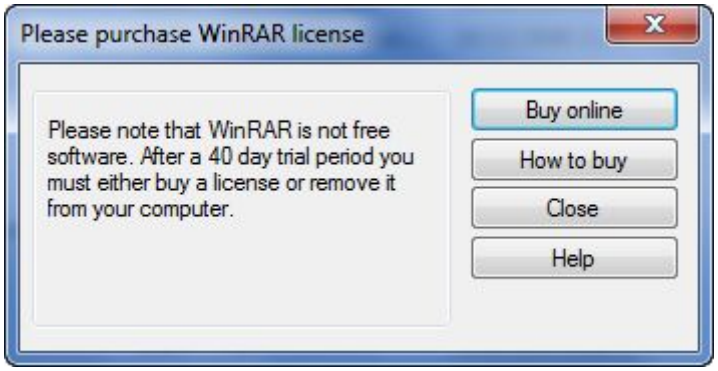
你可以在教程页下载本章的相关文件。

二、探究该应用

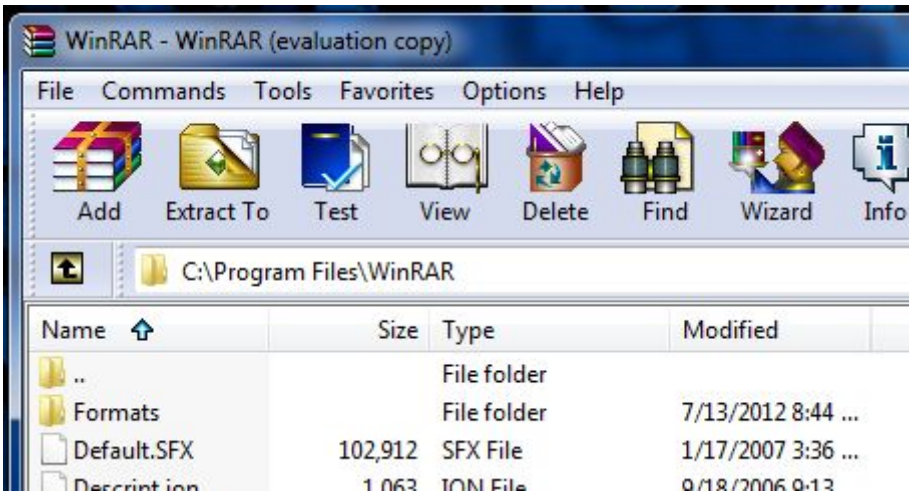
该程序有一个 40 天（可能和圣经有关？）的使用限制，40 天后会弹出一个 nag 窗口。相信我，根据以往的经验，它肯定很啰嗦。不幸的是，因为 nag 窗口 40 天（40 个晚上？-对不起。）内都不会出现，所以你有两个选择：你可以安装该应用，然后等上 40 天再阅读本章；或者你可以将系统时间设置成今天加上 41 天后练习本章，然后再将日期设置回今天。你要确保在练习本章前做了其中的一个，否则它不匹配。


WinRAR - WinRAR (evaluation copy)
File Commands Tools Favorites Options Help
Add Extract To Test View Delete Find Wizard Info Repair
C:\Program Files\WinRAR
Name Size Type Modified
.. File folder
Formats File folder 7/13/2012 8:44 ...
Default.SFX 102,912 SFX File 1/17/2007 3:36 ...
Descript.ion 1,063 ION File 9/18/2006 9:13 ...
File_Id.diz 509 DIZ File 1/17/2007 3:36 ...
License.txt 4,494 Text Document 10/18/2005 6:1...
Order.htm 3,271 Firefox HTML Doc... 10/22/2006 11:...
Rar.exe 316,928 Application 1/17/2007 3:36 ...
Rar.txt 71,951 Text Document 12/30/2006 6:1...
RarExt.dll 129,024 Application extens... 1/17/2007 5:36 ...
RarExt64.dll 43,008 Application extens... 12/11/2006 2:1...
RarExtLoader.exe 44,032 Application 12/23/2006 5:3...
RarFiles.lst 1,088 MASM Listing 4/11/2006 12:0...
rarnew.dat 20 DAT File 7/13/2012 8:45 ...
ReadMe.txt 1,687 Text Document 5/12/2005 6:01 ...
TechNote.txt 9,232 Text Document 9/19/2006 7:16 ...
Uninstall.exe 99,840 Application 1/17/2007 3:37 ...
Uninstall.lst 639 MASM Listing 1/17/2007 5:36 ...
UnRAR.exe 203,264 Application 1/17/2007 3:36 ...
UnrarSrc.txt 90 Text Document 5/12/2005 6:02 ...
WhatsNew.txt 9,906 Text Document 1/17/2007 3:36 ...
WinCon.SFX 80,384 SFX File 1/17/2007 3:36 ...
WinRAR.chm 243,916 Compiled HTML ... 1/17/2007 3:37 ...
WinRAR.exe 929,792 Application 1/17/2007 3:36 ...
Zip.SFX 67,072 SFX File 1/17/2007 3:36 ...
Total 1 folder and 2,364,122 bytes in 23 files

过了一会...。


Please purchase WinRAR license
Please note that WinRAR is not free software. After a 40 day trial period you must either buy a license or remove it from your computer.
Buy online
How to buy
Close
Help

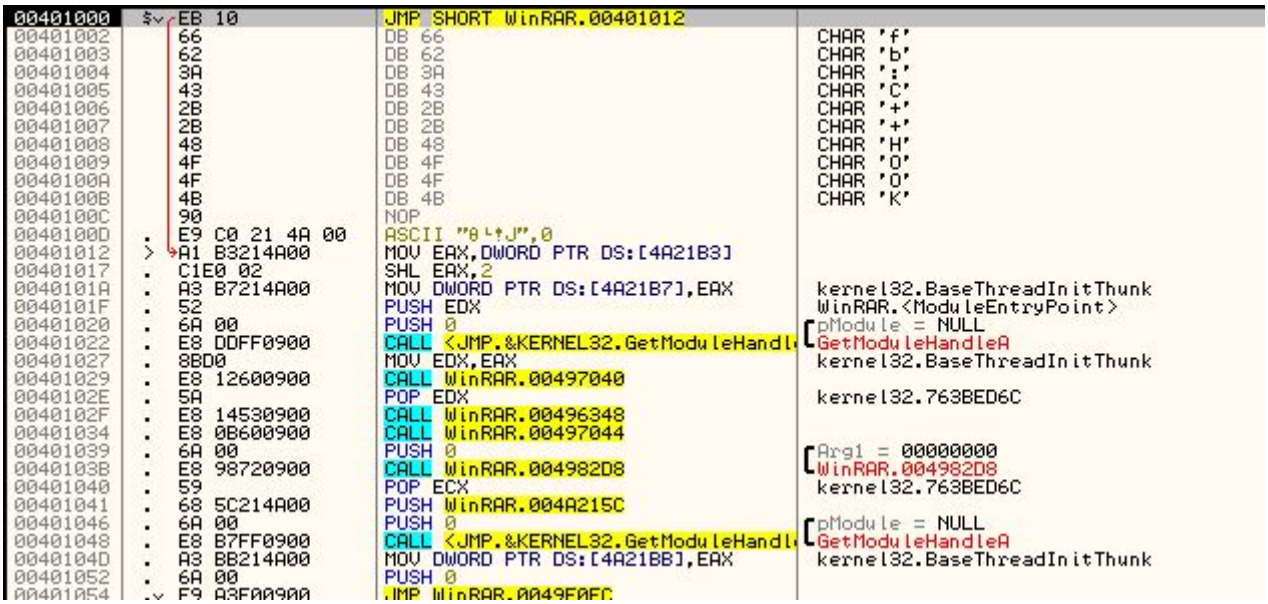
nag 窗口弹出来了。在使用的时候它出现了很多次。这真是很烦人。我们在顶部也可以看到 “evaluation copy”:



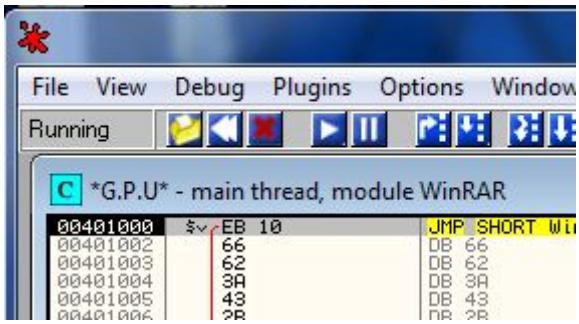
我打算介绍两种方法来获取相关注册码。

三、第一种方法

Olly 载入并启动程序:



启动应用后，等 nag 出现。它一出现（在关闭它以前），切换到 Olly，点击暂停按钮（靠近 play 的那个）:

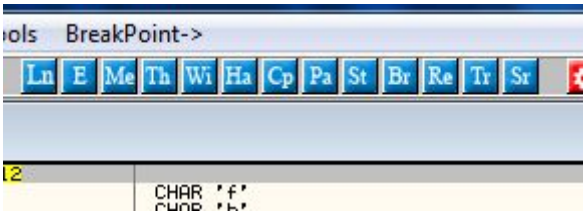


现在，我们要找出那个 nag 窗口是从哪来的，最后找出是谁让它显示的。我们当然也可以搜索字符串或模块间调用，不过我向你保证，这些技巧对于外面的大部分应用都不管用。所以咱们学习另外一个技巧...。

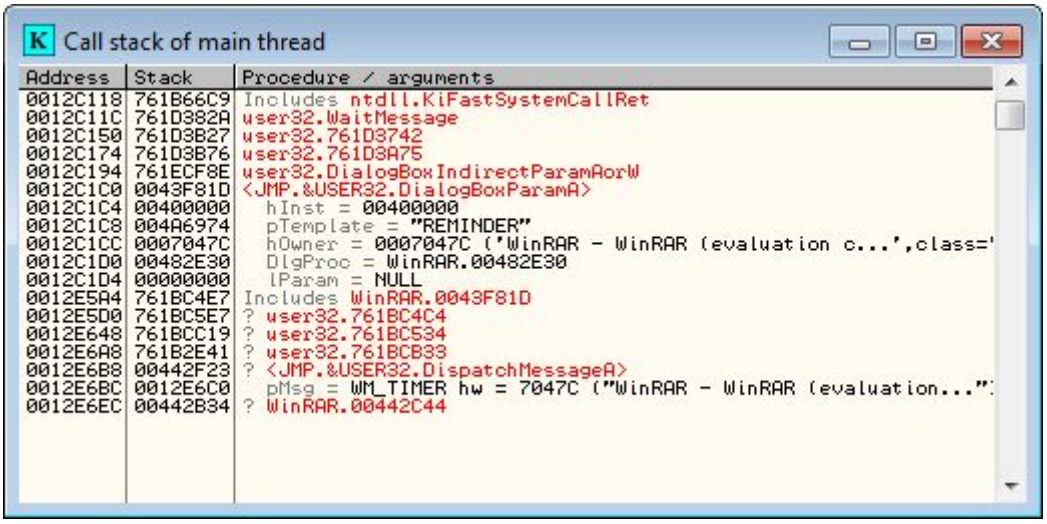
四、调用栈

调用栈是 Olly 尝试跟踪让我们到达某处位置的代码，从而试着找出哪个函数被调用。它也尝试向你显示被传递给函数的参数。所有的这些都可以通过右下角的“普通”的堆栈来完成，不过通过调用栈用来查看这些数据要更好用。要记住 Olly 在这方面不是很完美，你不能把这个窗口的所有东西都当做福音（糟糕，我又犯了同样的错误。译者注：我也不知道这句话啥意思。）。要做很多猜测。当然，有很多次，这个窗口是空的。通常是因为 Olly 完全糊涂了，或者是在逆向一个 VB 程序（VB 程序在调用函数的方式上和真正的程序不太一样）。

要查看调用栈，如果你用的是我的版本的话，点击“St”按钮:



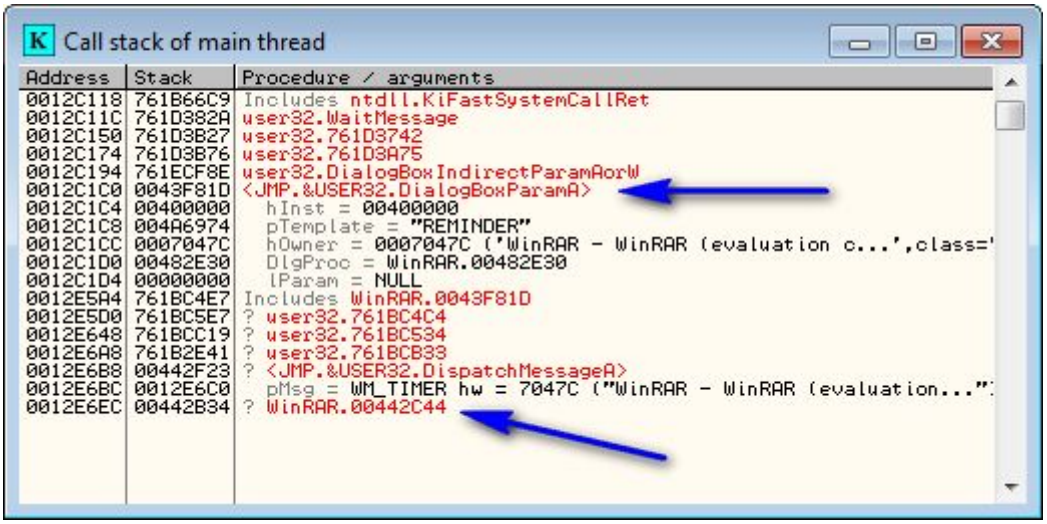
如果用的是原版的 Olly，点击工具栏中的“K”按钮。似乎“Call”这个词在作者的母语中是以“K”打头的:



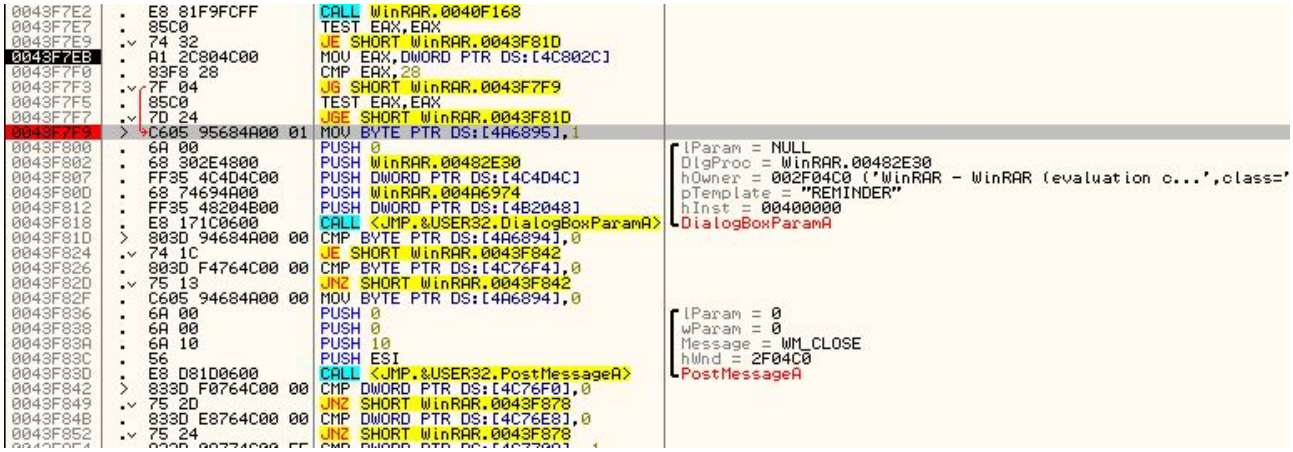
有几件事需要点出来...。最近的调用是在顶部，和堆栈类似。“Includes”意思是该指令在那个 CALL 中有涉及到，不过 Olly 不能够准确的知道是怎么回事。问号的意思是 Olly 对该行没有把握，所以你得为自己带盐（译者注：原话是“take it with a grain of salt”，意思是需要进行分析，要斟酌斟酌，不能全信。）。

在咱们的例子中，可以看到一个 ntdll 函数、几个 user32 函数、对 DialogBoxParamA 的带参数调用、又是几个对 user32 的调用，底部是对程序 WinRAR 本身函数的调用。下面是对这些内容的思考：WinRAR 在地址 442C44

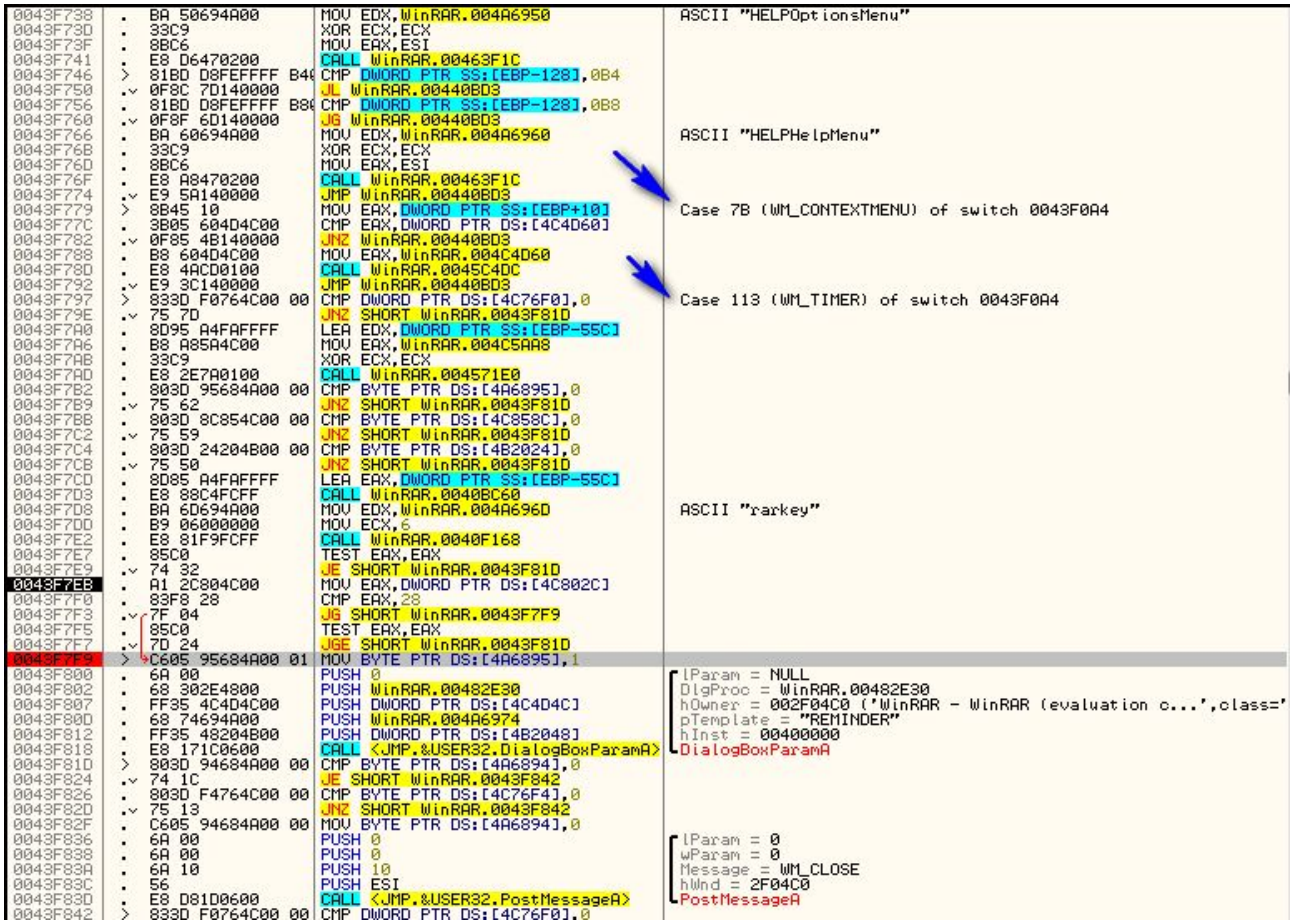
处调用了 DispatchMessageA，这里用一个消息来显示对话框。然后 User32 调用了 DialogBoxParamA 函数来显示对话框，标题是 “evaluation copy”，还有其他几个参数。然后 User32 显示对话框并等待我们的输入，它使用 WaitMessage 来做这个。



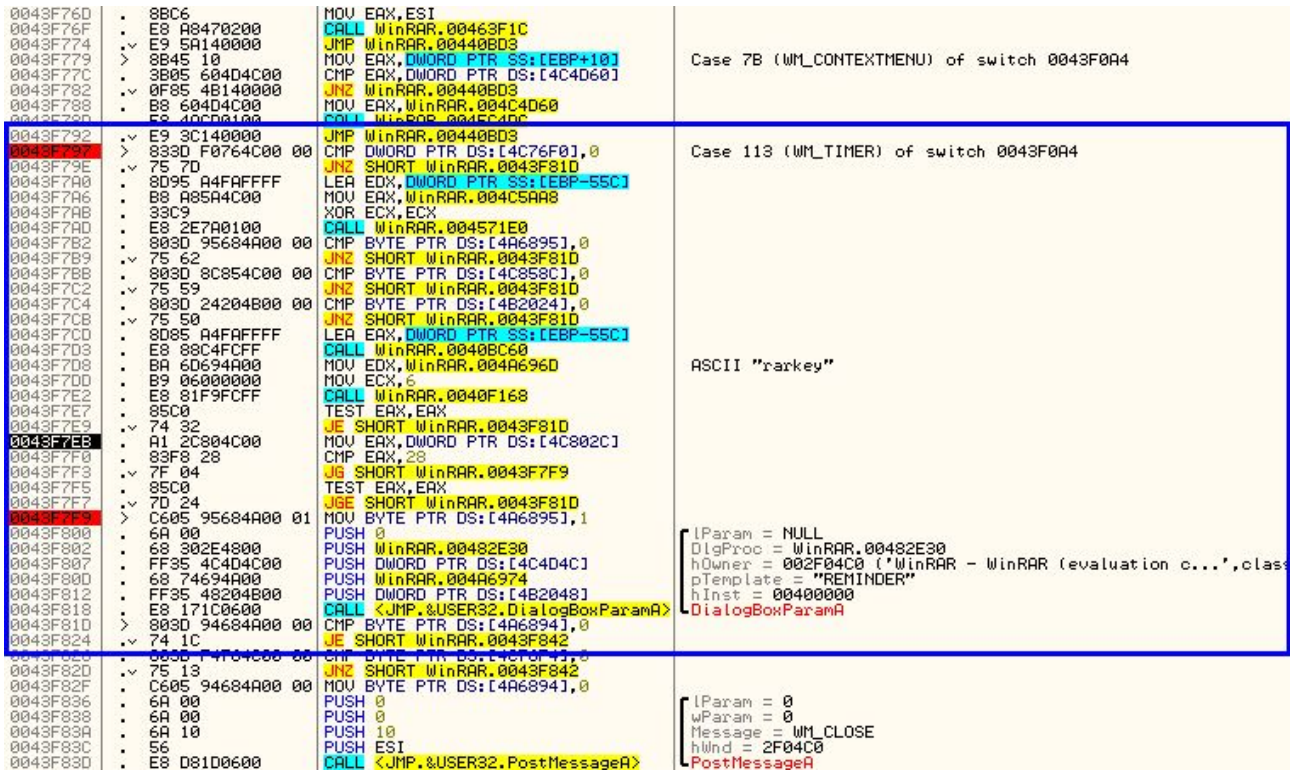
这个窗口里重要的是调用显示对话框的那个 CALL 以及应用自己的 CALL。通常使用调用栈时，从顶部开始，找到你感兴趣的可以用来找到代码区块的第一项。如果这个不好用，继续向下找，检查每一个函数调用，直到你“回到”代码足够的远，以找出决定该函数是否被调用的那个 比较/跳转 指令组合。通过双击那行，咱们来试试检查下对 DialogBoxParamA 调用的那个 CALL:



我们来到了调用 DialogBoxParamA 的地方。我在执行设置以及调用显示对话框这些指令的开始处设置了一个 BP。在它的上面，有几个条件跳转映入眼帘。如果你再向上滚动，你会发现有几个写着 “Case XX (WM_Something) of switch 0043F0A4” 的注释，这里的 XX 是一个十六进制数:

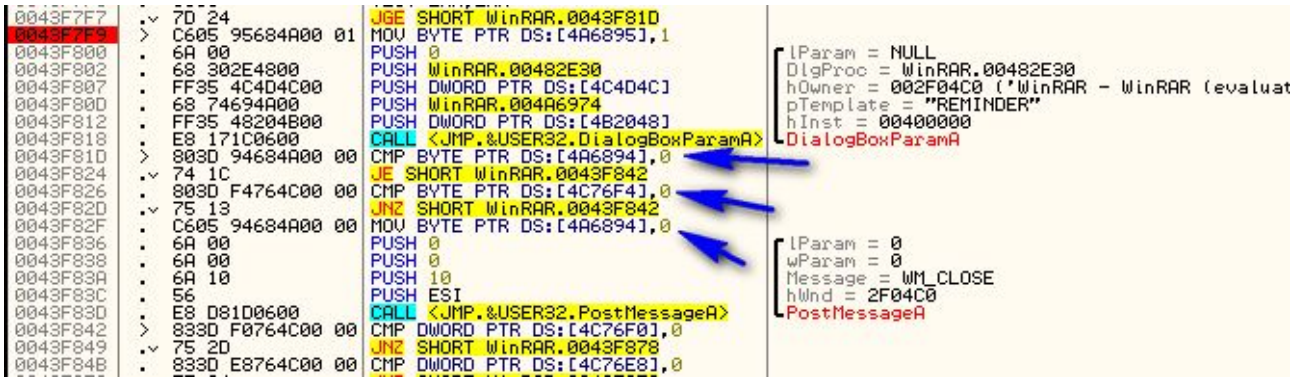


这是 Olly 显示 switch 语句的方式。如果你往上滚动，你会发现这真是一个相当大的 switch 语句。如果你有 Windows 编程经验，你可以认出“WM_SOMETHING”句子是 Windows 消息，你也可以认出这一整块代码是作为 Windows 消息的消息处理过程。如果你对这些一无所知也没关系，在后面的章节中我们会非常细致的讲解 windows 消息处理过程。目前来说，我们只对涉及到对话框调用的部分感兴趣。下面，我们可以看看这整个分支（case）:



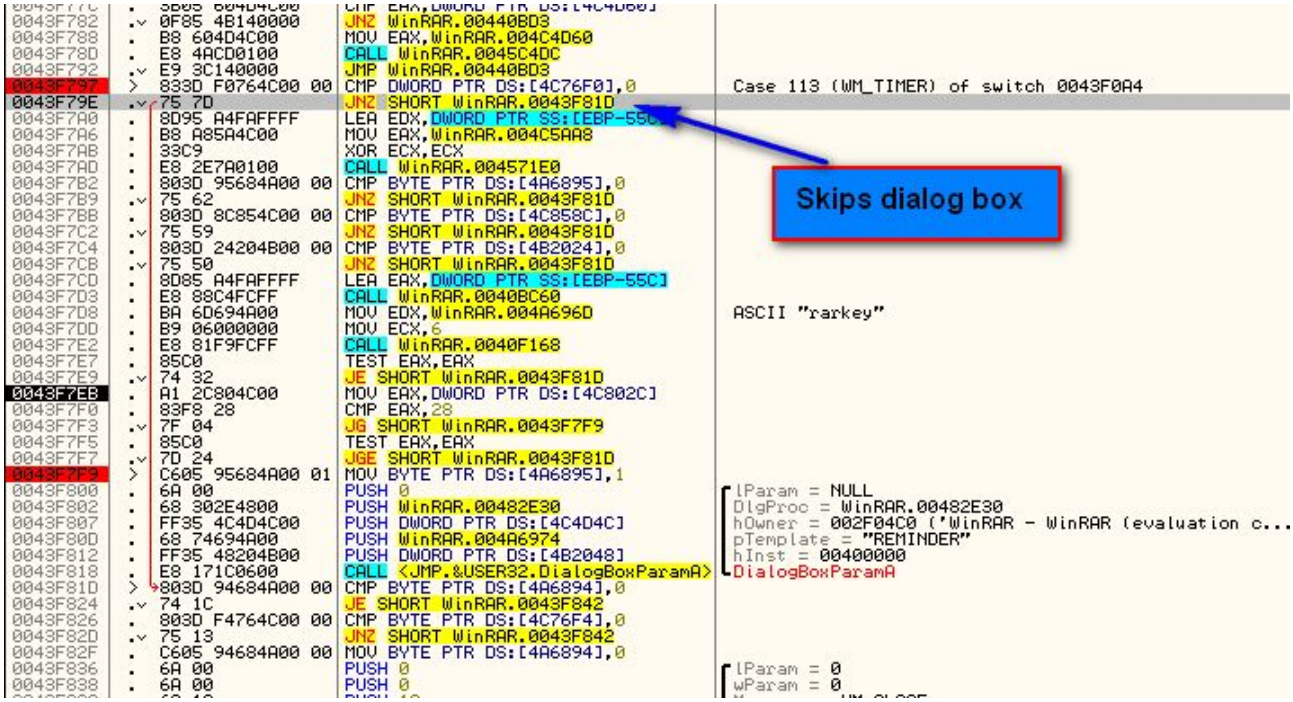
你会发现它是在处理 WM_TIMER 消息的区块。这很能说明一些问题。为什么我们的对话框是在一个对计时器超时的消息处理中？我们马上就会看到....

还要注意在对话框被调用后，有几个条件跳转和比较语句:

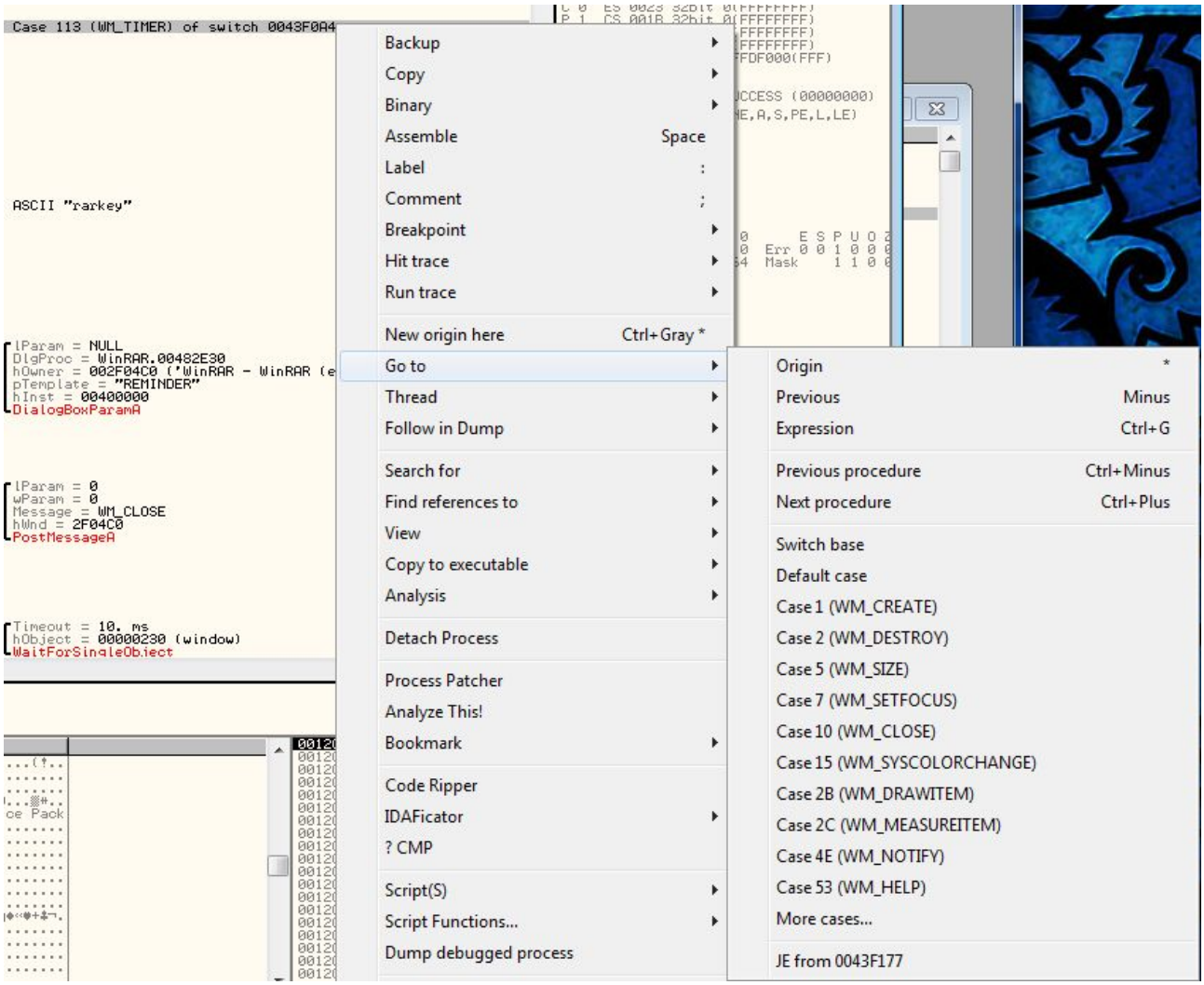


这些跳转执行的代码依赖于我们点了对话框中的什么。如果你点的是”Close“，它会跳到关闭窗口的代码等。

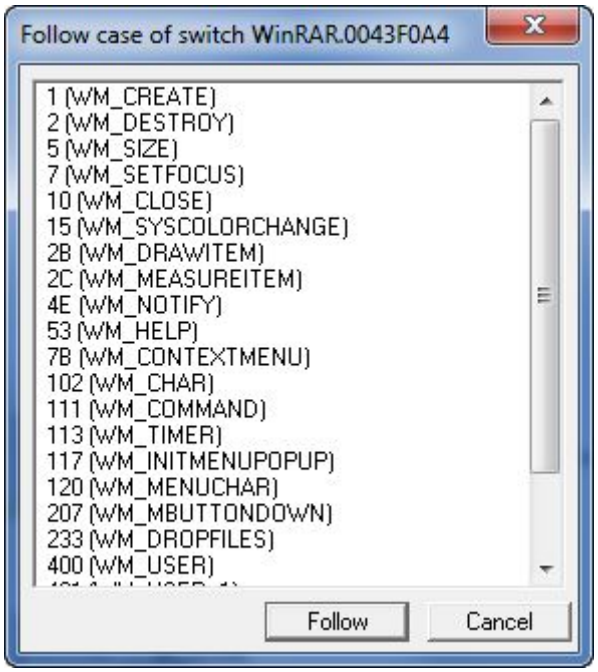
滚动到 switch/case 语句块的起始处，能够发现那里有一个初始的比较和跳转指令:



这个跳转跳过了打开 nag 窗口的那个 CALL(还有其他很多代码也被跳过)。咱们来看看这个初始的 比较/跳转 是啥。在该行上右键，也就是有“Case 113 (WM_TIMER)”的那行，选择“Goto”:



在弹出的下拉菜单中你可以看到，Olly 向我们显示了可以被这个 switch 语句处理的好几个 case。点击 “More cases. . . ”，会弹出一个对话框向我们显示全部的 case:

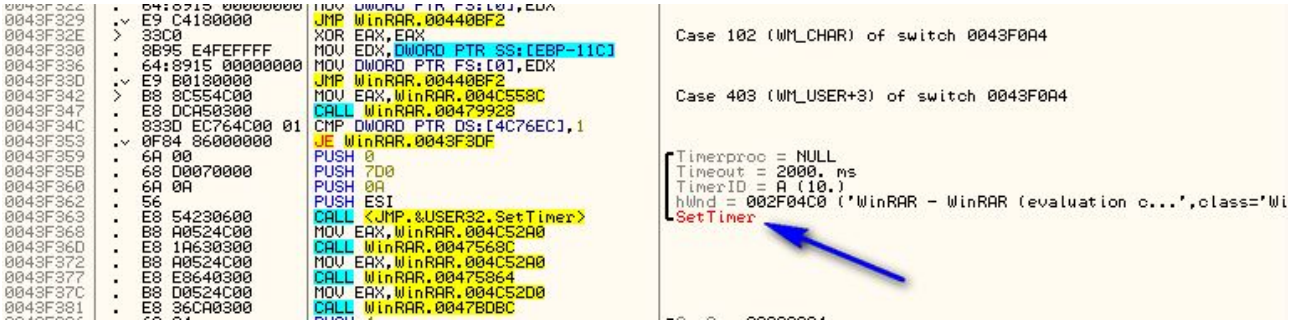


点击其中几个，然后点击 “Follow”，你会跳到处理相应 case 的代码。你会发现，所有这些 case 的开始都是一个 比较/跳转 组合。意思就是汇编语言处理 switch/case 语句，是作为巨大的 if/then 语句来处理的。有点像下面这样（用伪代码来表示）:

```
if (msg != WM_CREATE)
    jump to next if
Do WM_CREATE code
Jump to end
if (msg != WM_DESTROY)
    jump to next if
Do WM_DESTROY code
Jump to end
if (msg != WM_SIZE)
    jump to next if
Do WM_SIZE code
...
```

所以在每一个 case 的起始处，都要检测该 case 是不是用于处理特定的消息，如果不是就跳到下一个比较。如果是，就忽略跳转，直接转到处理消息的代码部分。

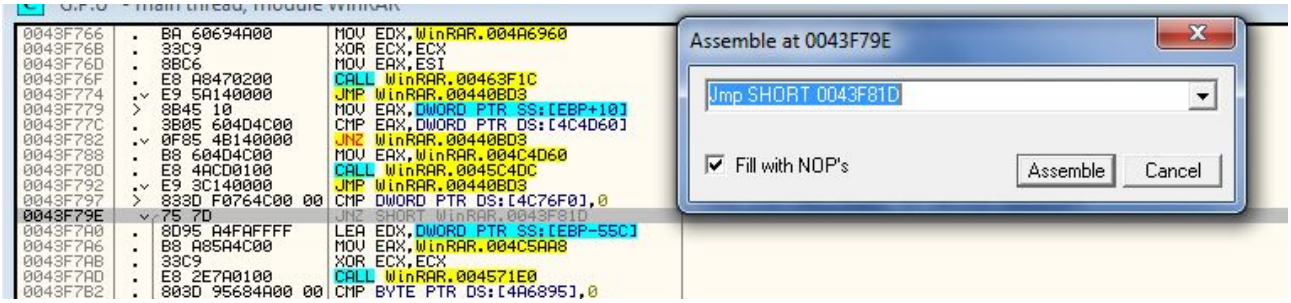
现在，因为我们的 case 是 WM_TIMER 消息，我们可以知道（通过在 Google 搜索 WM_TIMER 消息），这个是用来处理计时器超时时的消息。也就是说，在某个地方计时器必须被启动。向上滚动（多滚一点），我们看到了罪魁祸首：



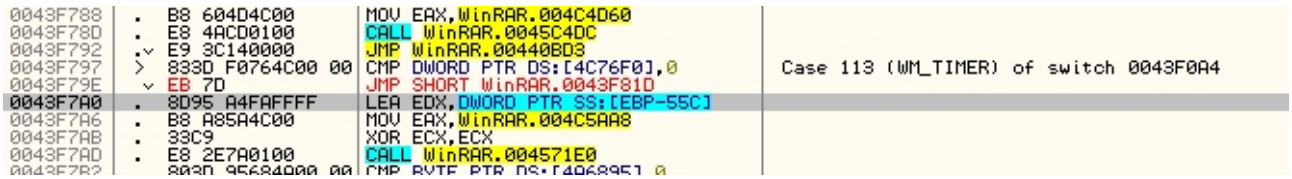
那么现在，我们可以猜测怎么来覆盖这个 nag 窗口了....

五、给程序打补丁

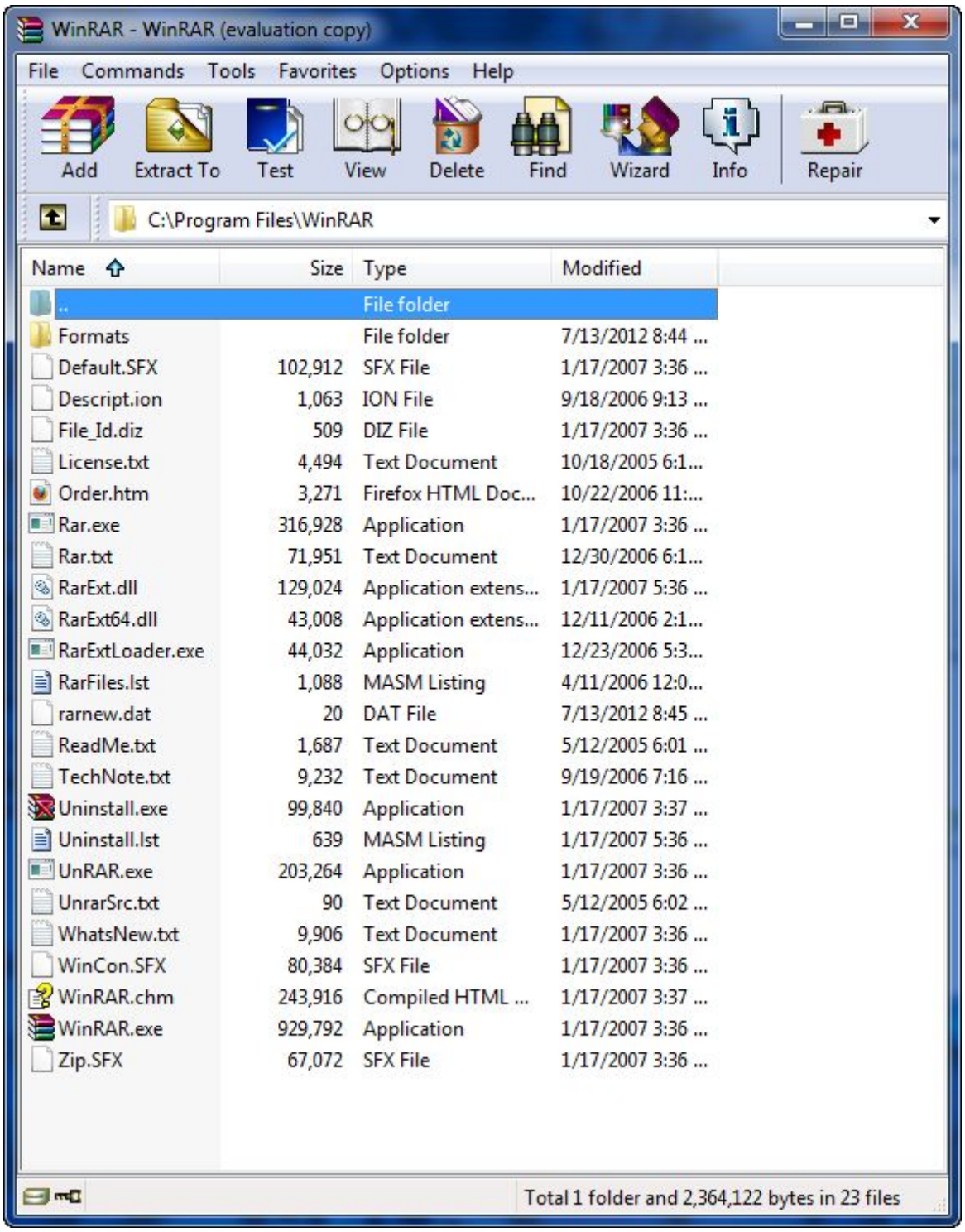
最简单的方法是，当计时器超时时，让消息处理过程什么都不做。做这个的最简单的方法是保证我们每一次都跳过这个 case。那转到该 case（113-WM_TIMER）的起始处，在这里它会检测是不是正确的 case，把它改成总是跳转：



下面就是打过补丁之后的样子：



现在，无论什么时候该消息过程得到了计时器超时的消息，它都会简单的忽略它。运行程序，过一会你会发现那个 nag 窗口再也不会出现了：



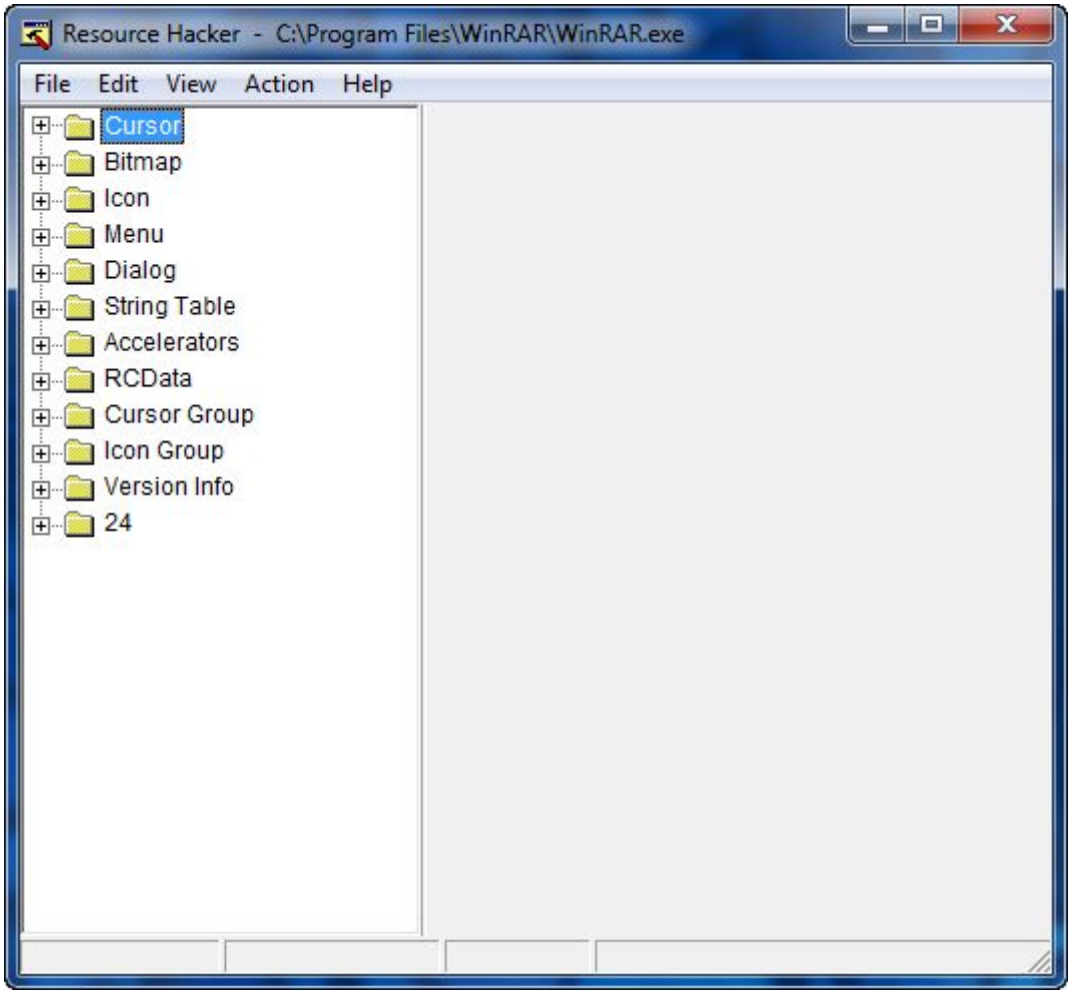
这个程序仍然会在标题栏显示“evaluation”，后面的章节我们会回到这个话题（修改这个有点复杂，你可以试试。这是学习的最好方法！）。不过目前，即使它显示“evaluation”，它也工作的很好并且永远不会过期。好吧，确切的说这不是真的，它仍然会过期，不过它却什么都不会做。确定你保存了补丁（和我们前面几章一样）以保存所做的修改。

六、第二种方法

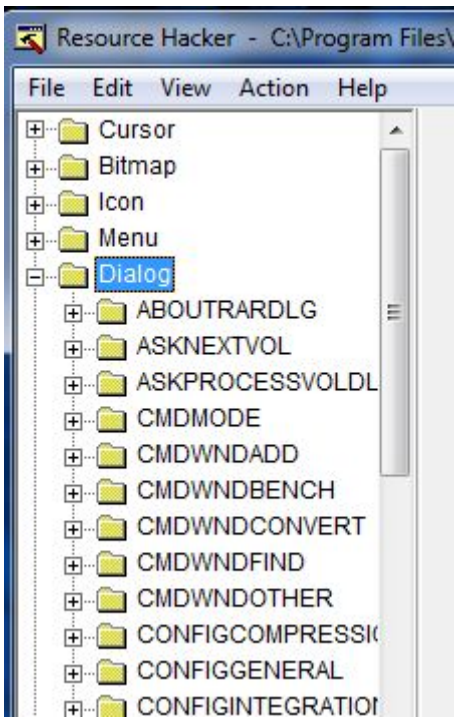
现在，我向你介绍我们能够用来找出对话框区块的另外一个方法（除了调用栈）。如果你还没有的话，先下载一个 Resource Hacker。你可以在工具页获

得它。Resource Hacker 可以让你查看和操纵一个 PE 文件内的资源。当我们讨论 PE 文件的构成时会更深入的讨论资源，不过就目前来说我们只需要知道应用程序所使用的任何资源（包括按钮、对话框、位图、图标、文本字符串）都存储在文件的独立区块，和代码是分开的。真正的，看看我所说的最好的方法是打开 Resource Hacker，载入几个程序看看。

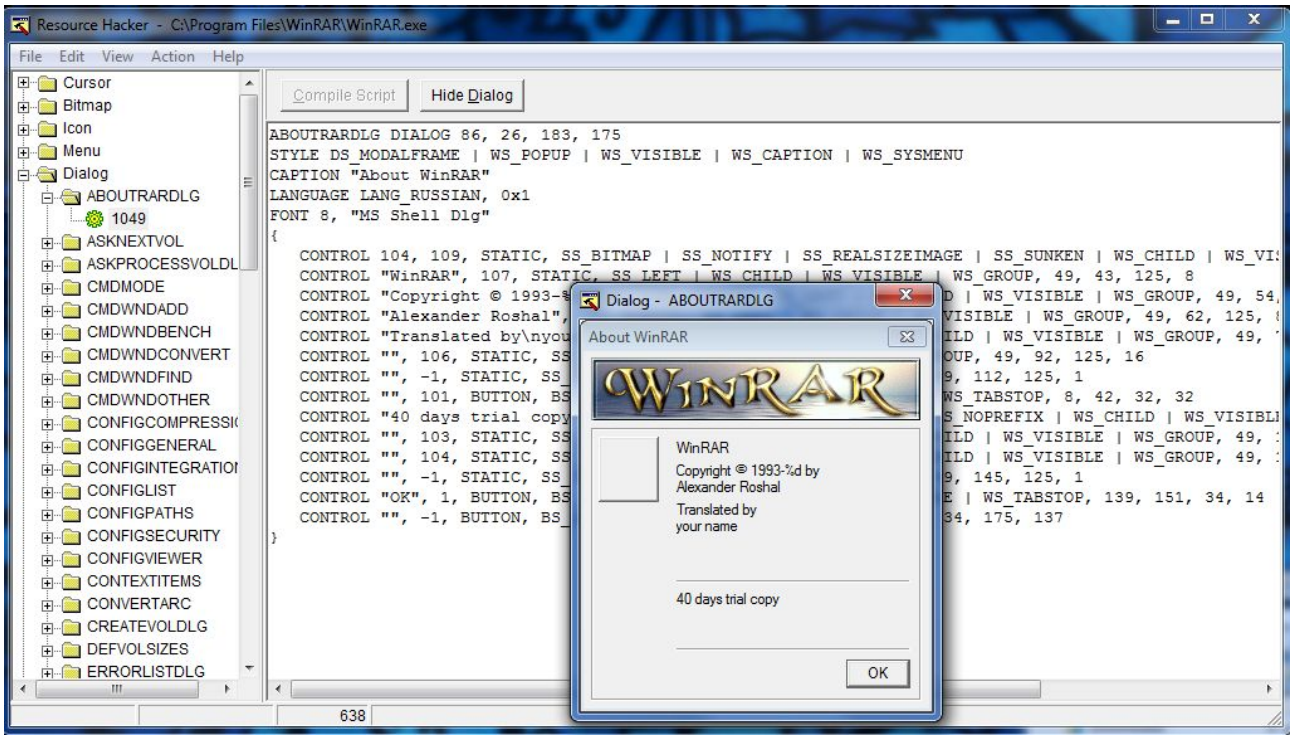
那么咱们就这么干....。打开 Resource Hacker，载入我们的应用：



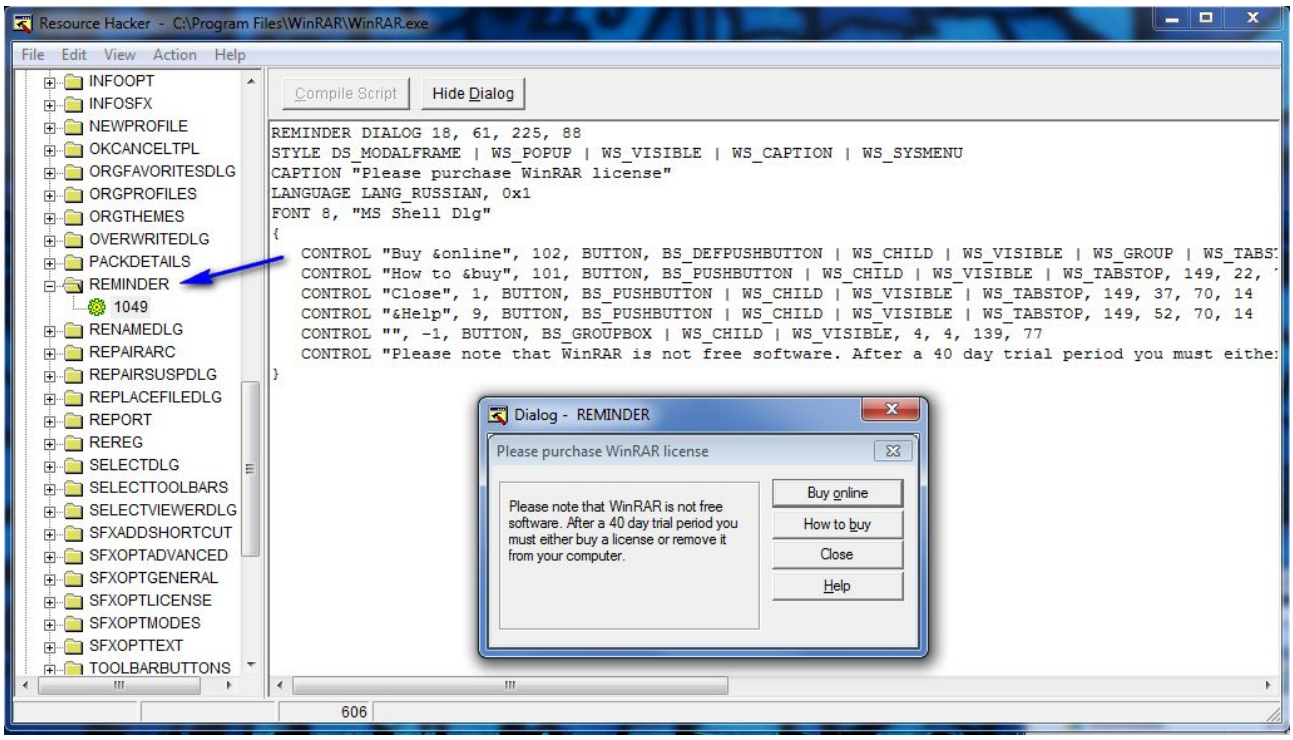
左边的树形列表显示了应用程序的各种资源。可以看到它包含有位图、图标、菜单，以及最重要的对话框：



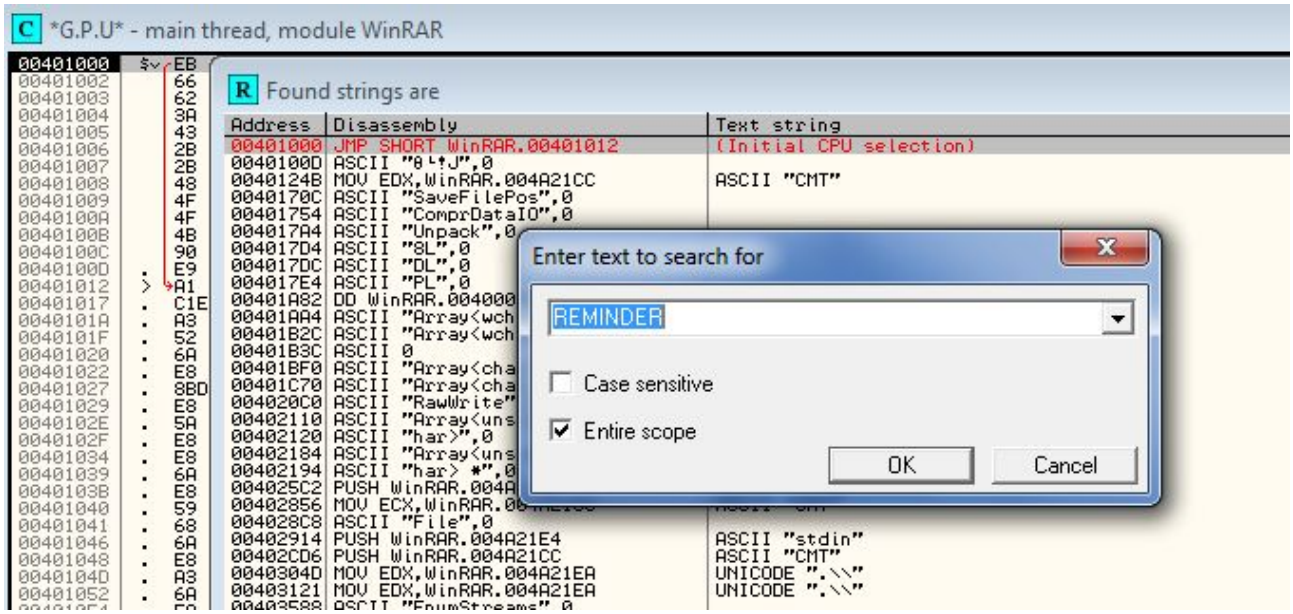
该应用有很多对话框。咱们继续，点击第一个，就是 ABOUTRARDIALOG:



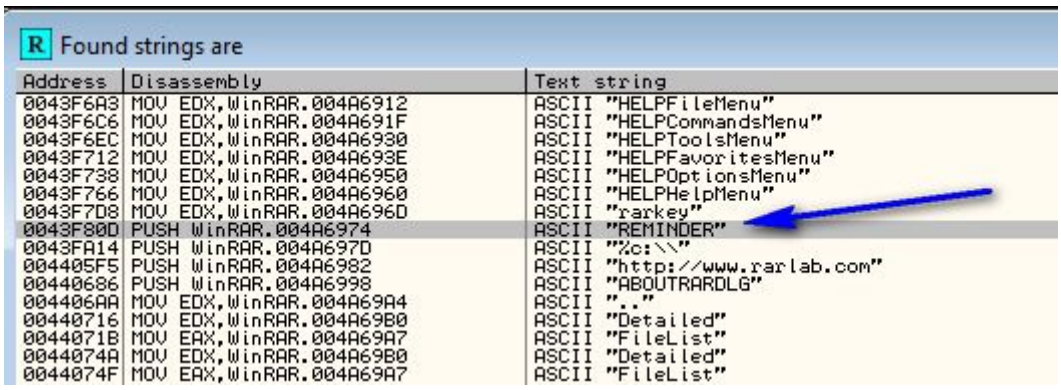
Resource Hacker 向我们显示了该对话框的相关数据，包括标题（显示在窗口标题栏上的），该对话框的相关按钮，以及它的各种设置。它也打开了一个窗口向我们准确的显示了该对话框的样子，这里的是关于对话框。在点了左侧一堆目录以后，就找到了我们想要的：



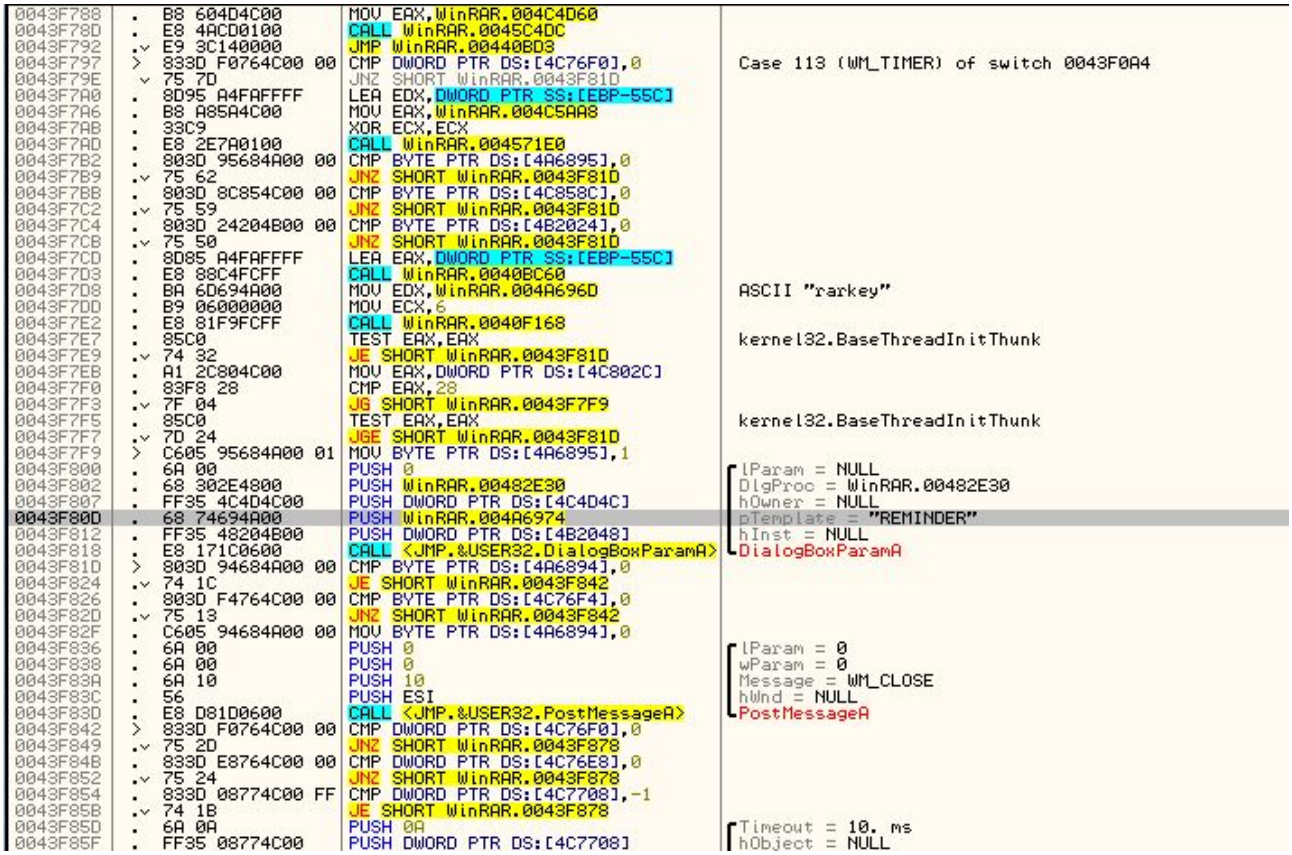
是不是看起来挺面熟的？注意对话框的名字是“REMINDER”。有时候 Windows 用名字来引用一个对话框，有时候用 ID。这里它用名字“REMINDER”。现在我们知道了所有我们需要的，Olly 载入应用，转到“search for strings”。咱们来搜索“REMINDER”：



咱们在列表中看到它了：



双击它，我们来到了与使用调用栈同样的区块:



事实上，你可以看到传递给 DialogBoxParamA 的其中一个参数就是“REMINDER”。如果资源是通过 ID 而不是名字来标示，我们可以通过右键反汇编窗口，选择“Search for” -> “Command”来找到它。然后在弹出的对话框中输入“PUSH xx”，xx 是资源的 ID（十六进制的）。这也会将你带至对话框的调用 CALL。

七、最后一件事

如果你看看众多的破解二进制文件的教程，会发现有一个方法是简单的通过 Resource Hacker 来删除对话框来实现的。本例中这是管用的，你再也不会看到 nag 窗口，但是这个方法不总是好用，因为它完全依赖于程序是如何处理丢失的资源的。之所以介绍这个简单的技术，是因为它总是值得试试的。

ps: 别忘了将你的日期改回去。