Flexnet Publisher V11.16及其以下版本加密种子点恢复方法

By: YangMyron

Flexnet Publisher V11.16已经做了改进，隐藏了原来恢复加密种子点的标志3D4DA1D6，无法实现快速定位。现提供两种方法：通用法和标志法。针对V11.16版，未使用lmgr_trl库链接的守护神可以通过标志0F42400h定位直接得到加密种子点。

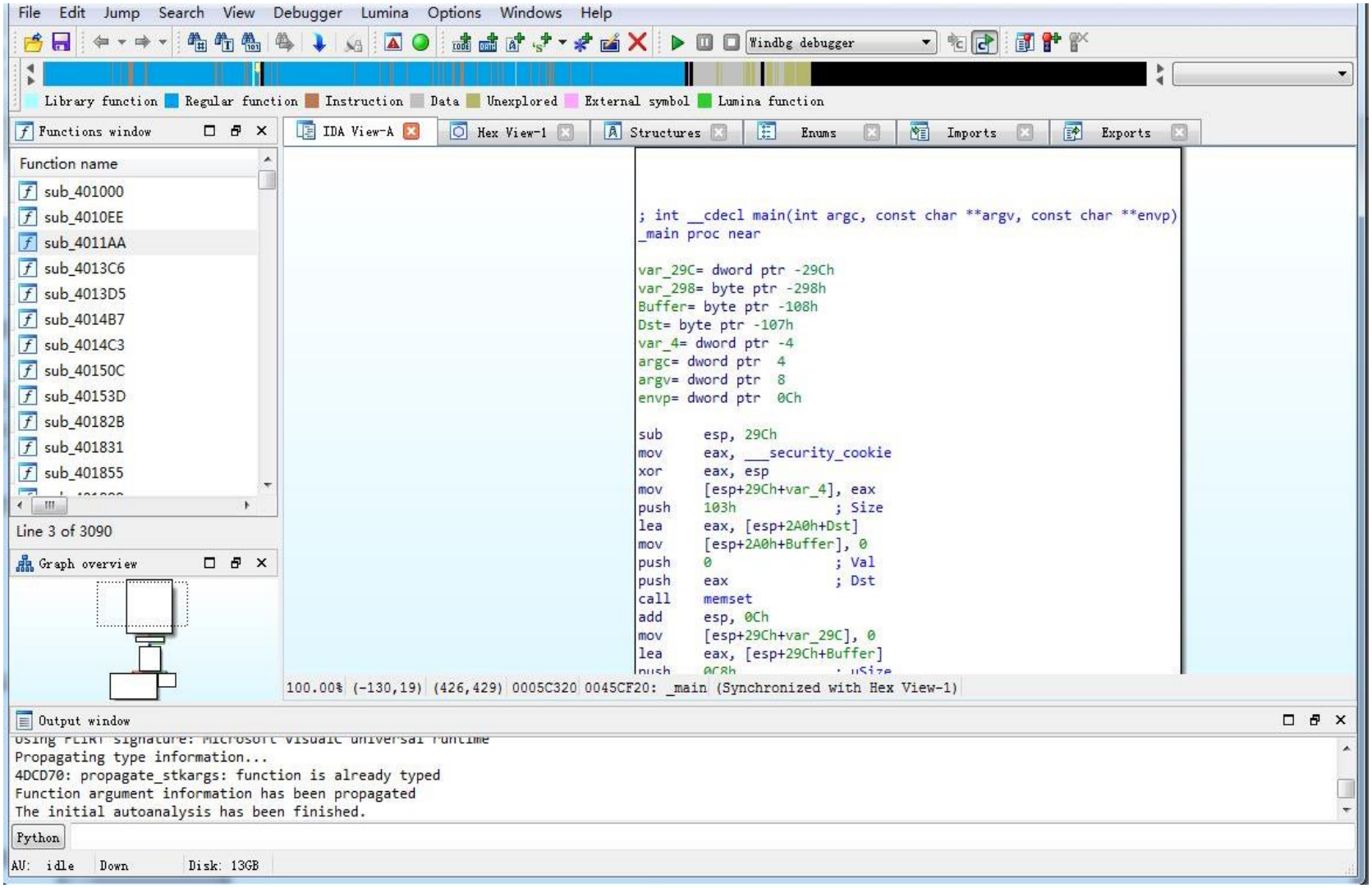
Flexnet Publisher V11.16及其以下版本加密种子点恢复方法

通用法

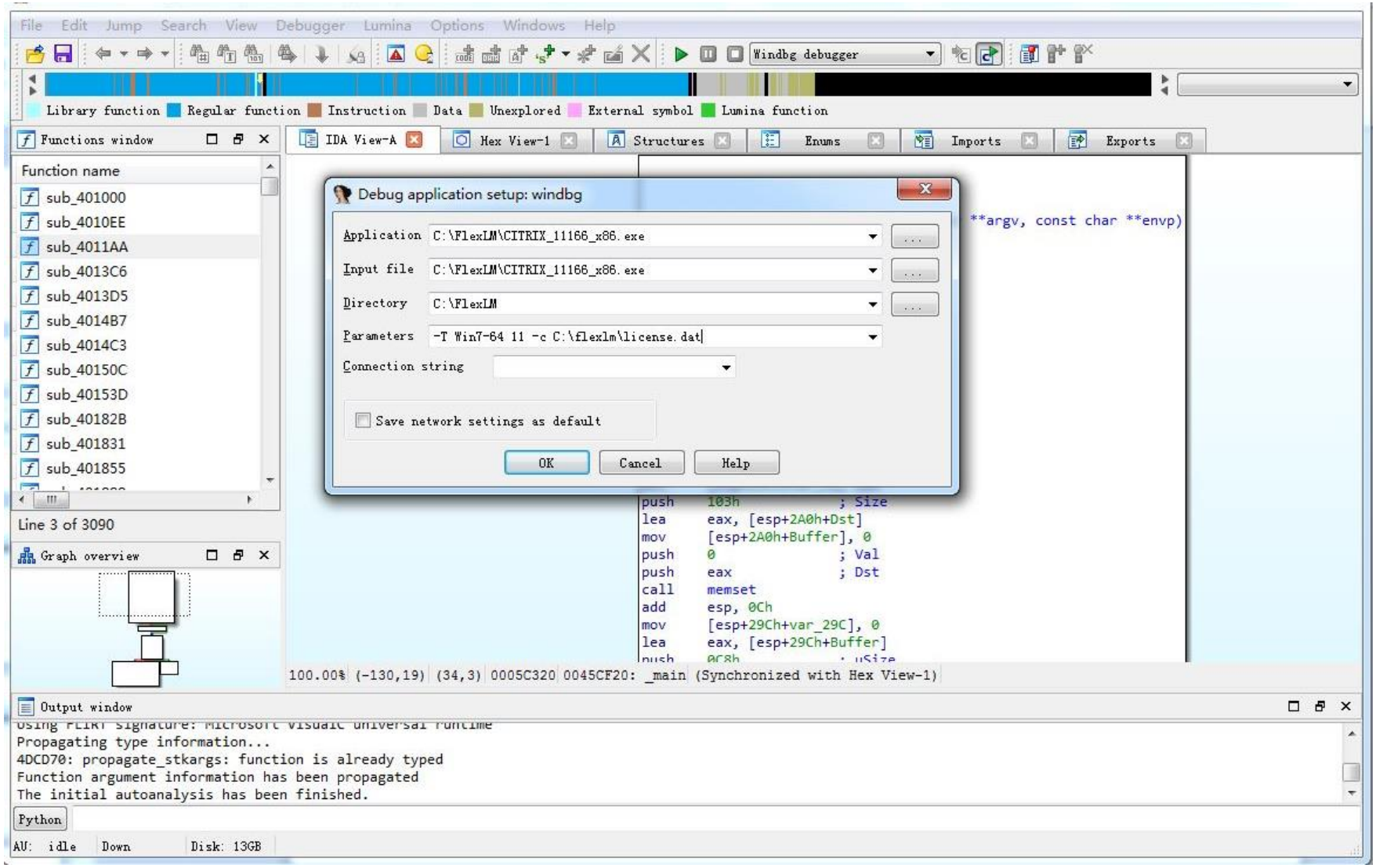
1、准备好格式正确的license.dat文件，一般放在c:\flexlm目录下。

SERVER WIN7-64 [redacted]
DAEMON [redacted] c:\flexlm\[redacted]_11166_x86.exe
INCREMENT session [redacted] 8.8 31-dec-2037 9998 D4D693AE978F HOSTID=ANY \
user_info=<aalbuquerque@agillity.com.br> ISSUER="GraphOn \
Corporation" ISSUED=21-Nov-14 NOTICE="Copyright (C) GraphOn \
Corporation. All Rights Reserved" ck=152 \
SN=FXQ3BJ_PRODUCTCODE=1410011625249 34M53E622qXwAVT TS_OK \
SIGN=81BE7DE89164
INCREMENT any_app [redacted] 8.8 31-dec-2137 9999 2A56248BCDD5 HOSTID=ANY \
user_info=<aalbuquerque@agillity.com.br> ISSUER="GraphOn \
Corporation" ISSUED=21-Nov-14 NOTICE="Copyright (C) GraphOn \
Corporation. All Rights Reserved" ck=179 \
SN=FXQ3BJ_PRODUCTCODE=1410011625249 34M53E622qXwAVT TS_OK \
SIGN=C676EAB876E6

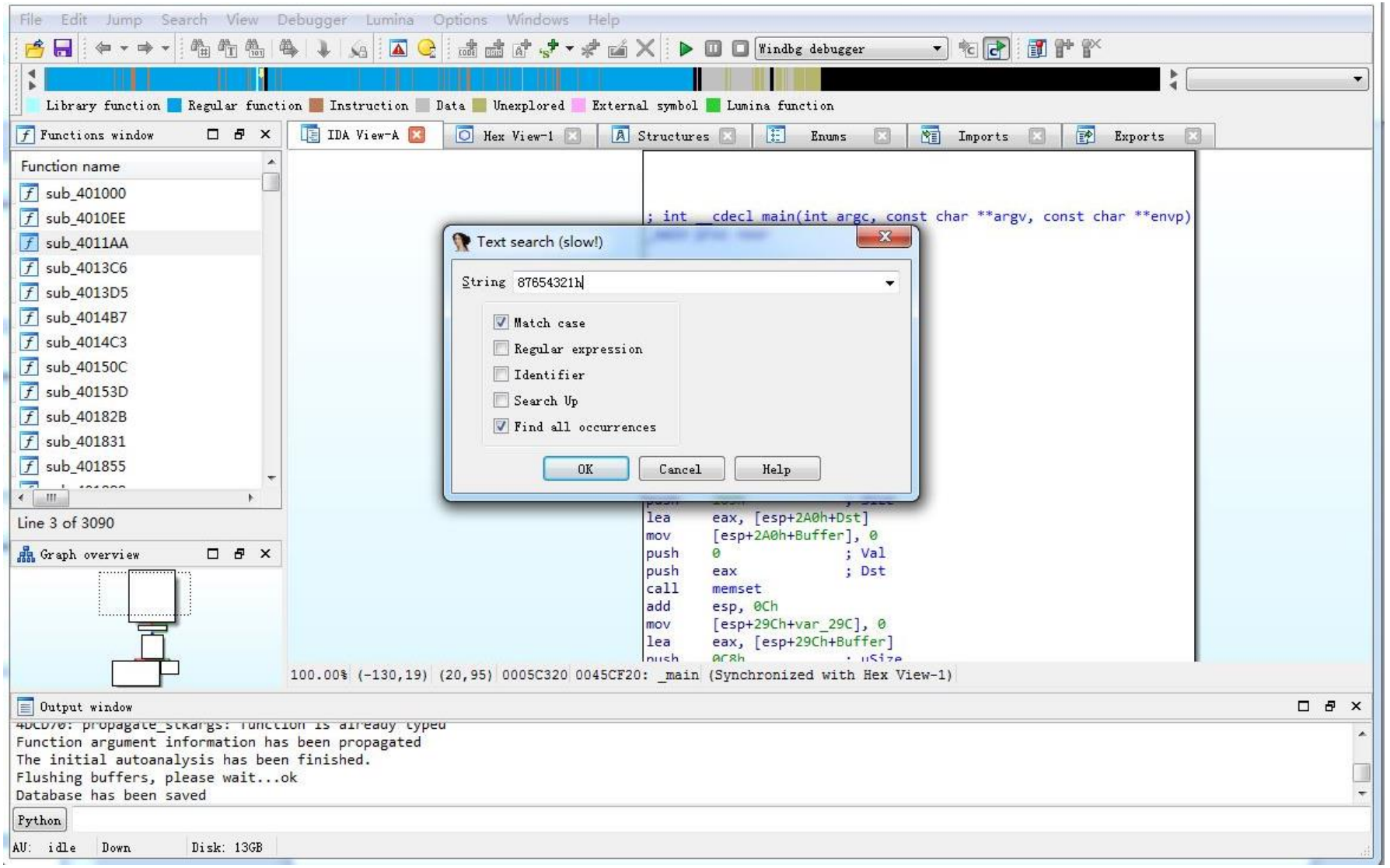
2、用IDA Pro打开守护神文件，等待分析完成。



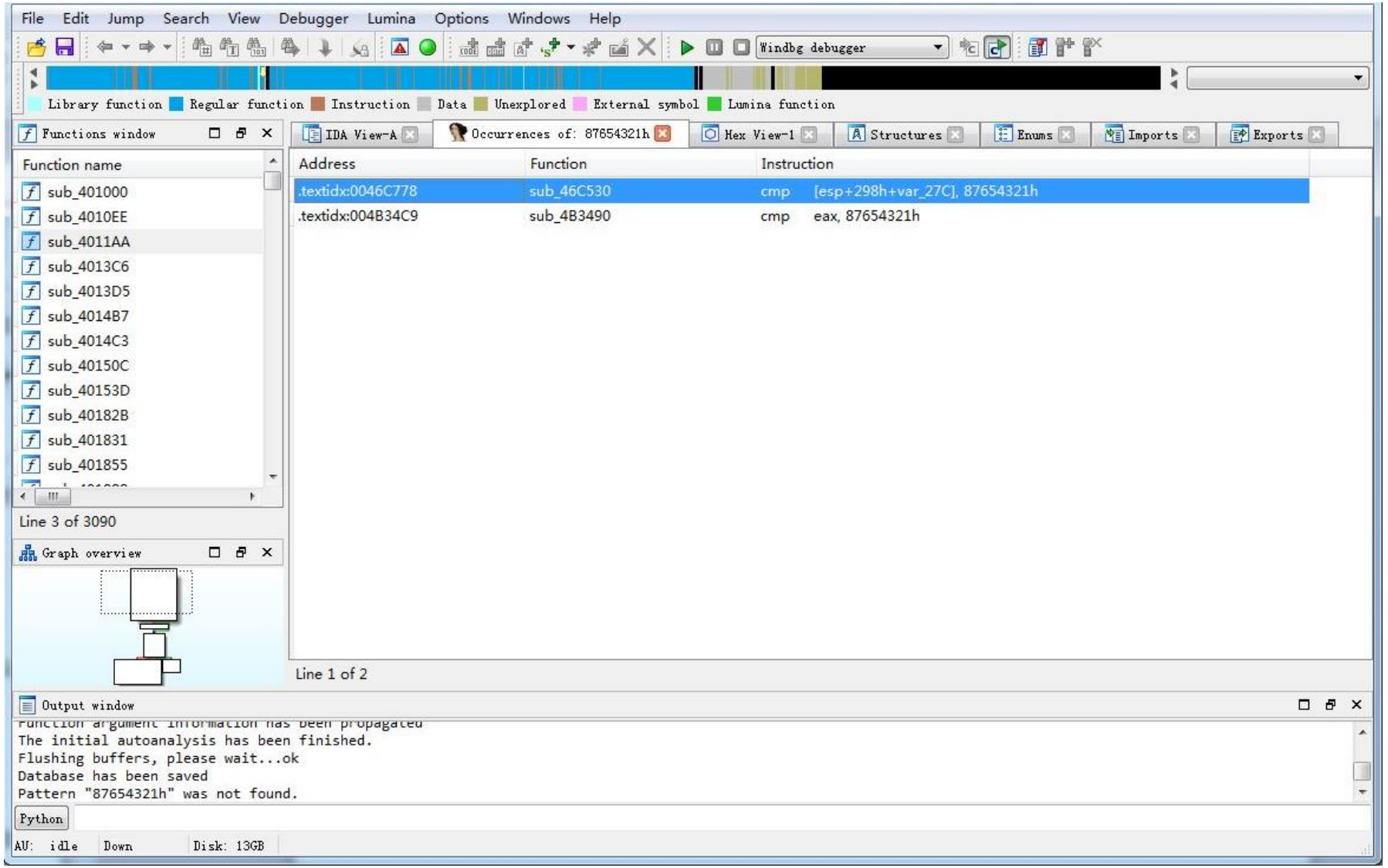
3、设置调试参数：-T Win7-64 11 -c C:\flexlm\license.dat



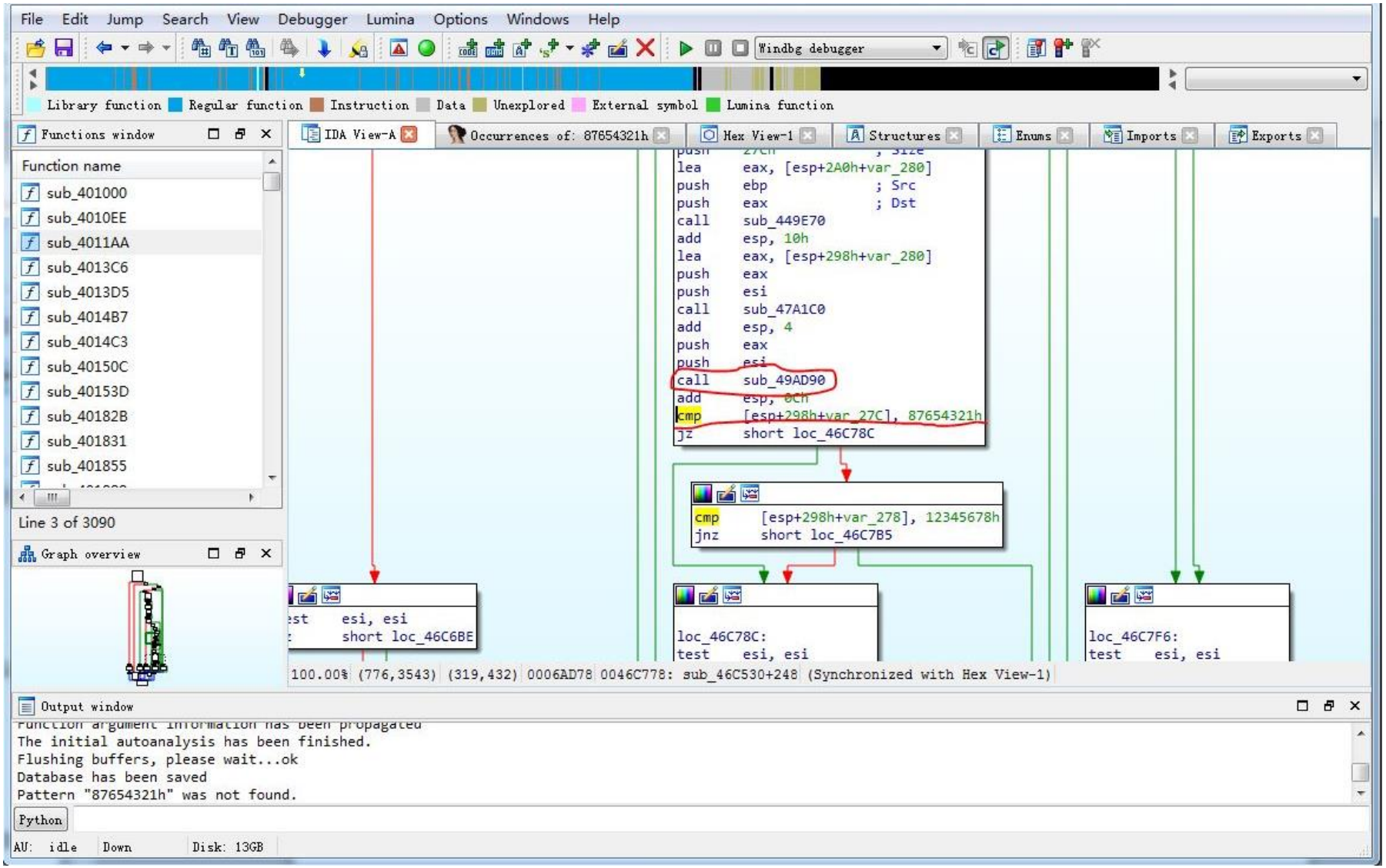
4、在汇编窗口查找“87654321h”



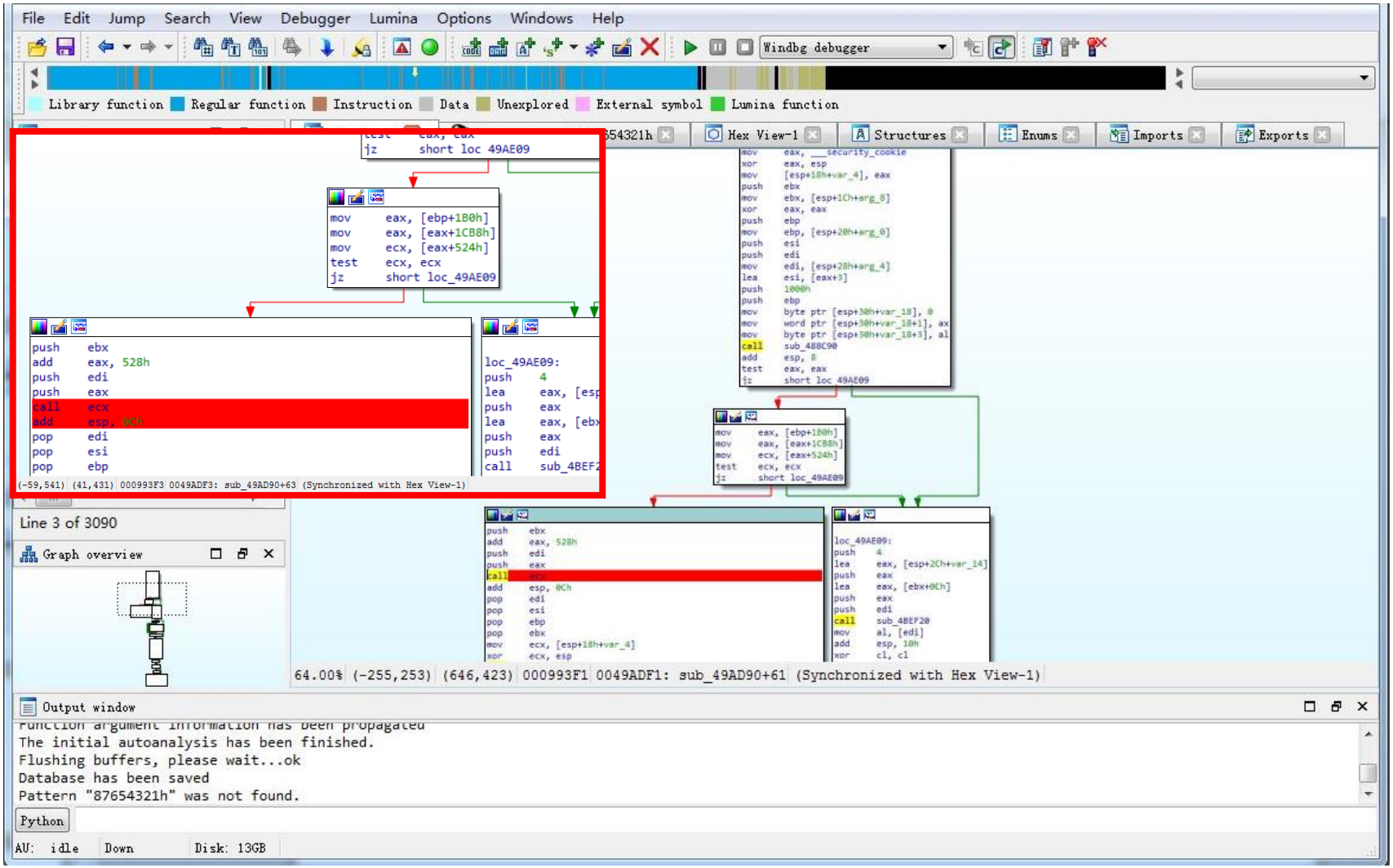
5、在搜索结果窗口双击第一个找到的“87654321h”



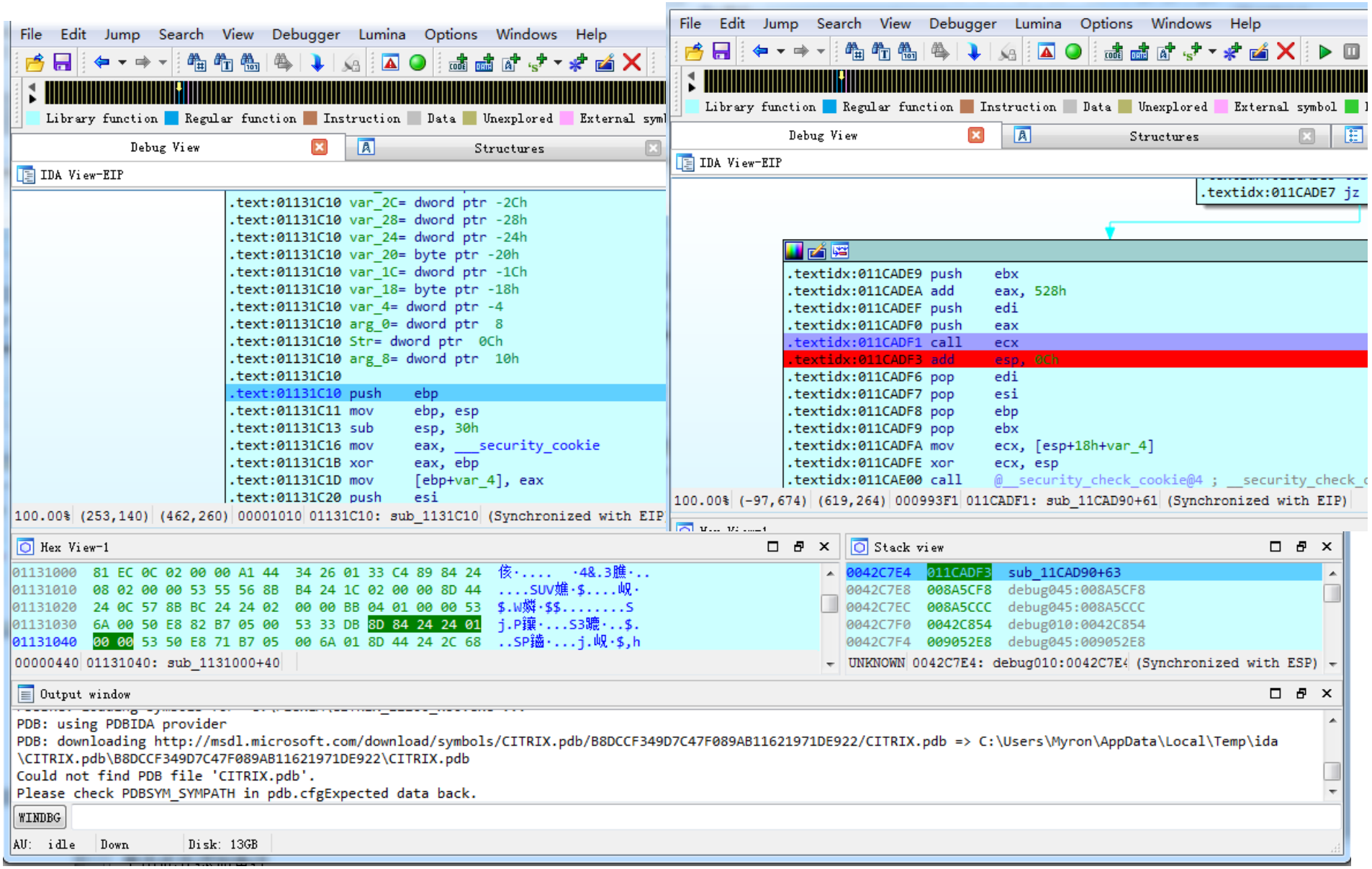
6、在汇编窗口找到“87654321h”上面紧挨着的那个call



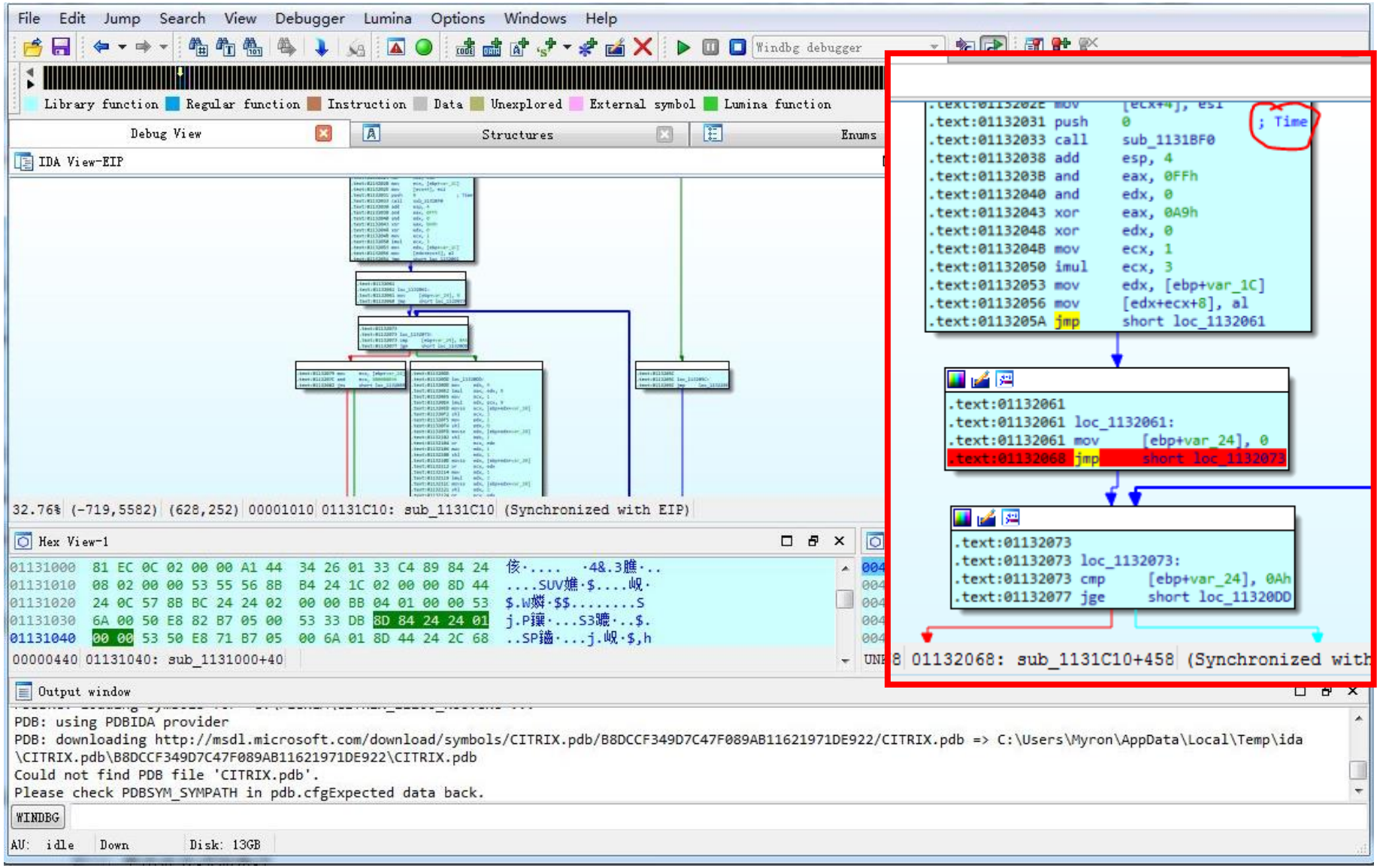
7、双击上步查找的call后面的函数（上图示例为sub_49AD90），在流程图显示第二个“jz”语句下的第一call（一般为寄存器或地址表达式形式），按F2下断点，紧挨其后的语句也按F2下断点。



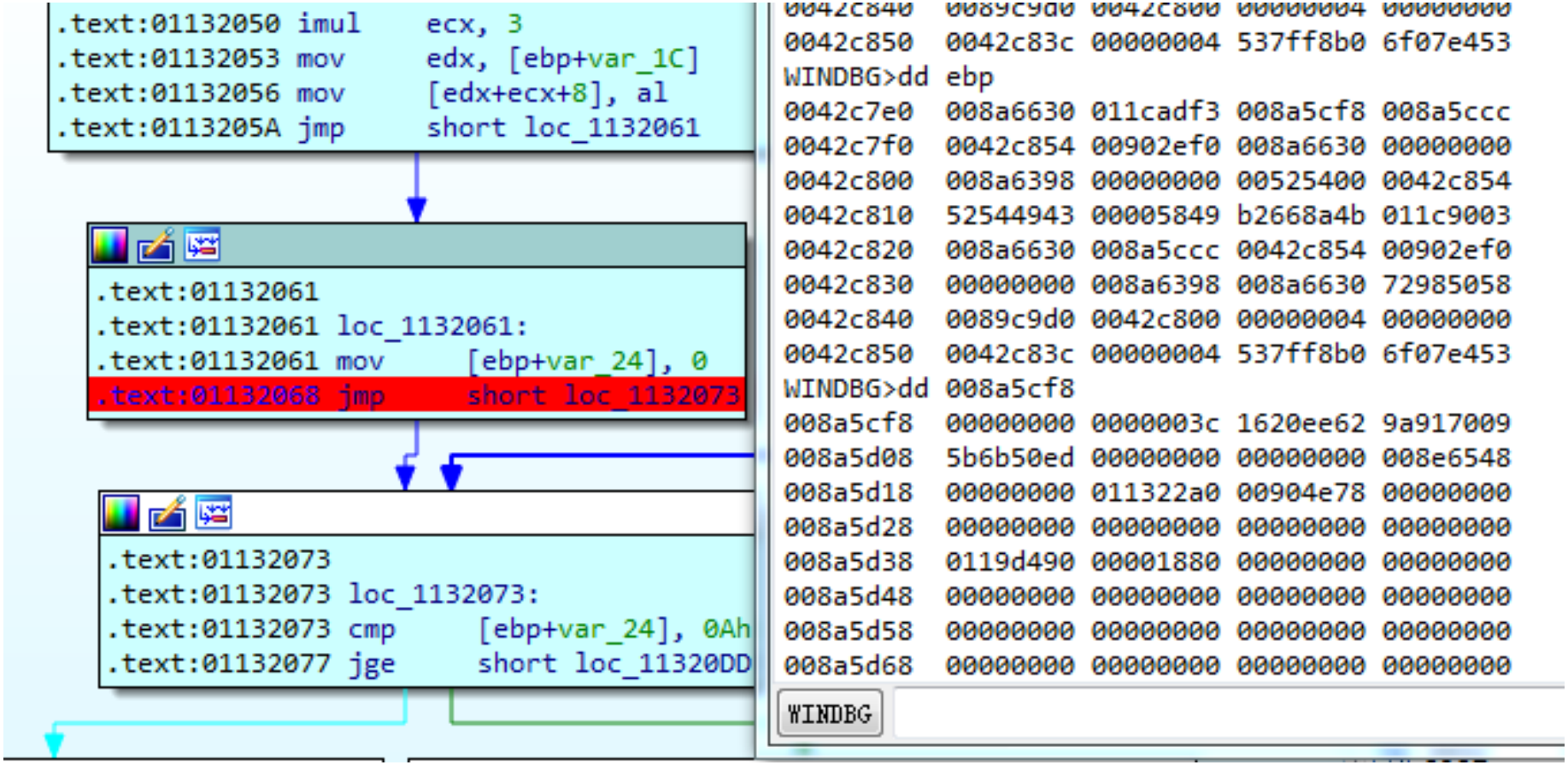
8、按F9开始调试守护神程序，直到在第一个call断点处停下，按F7步进入子函数



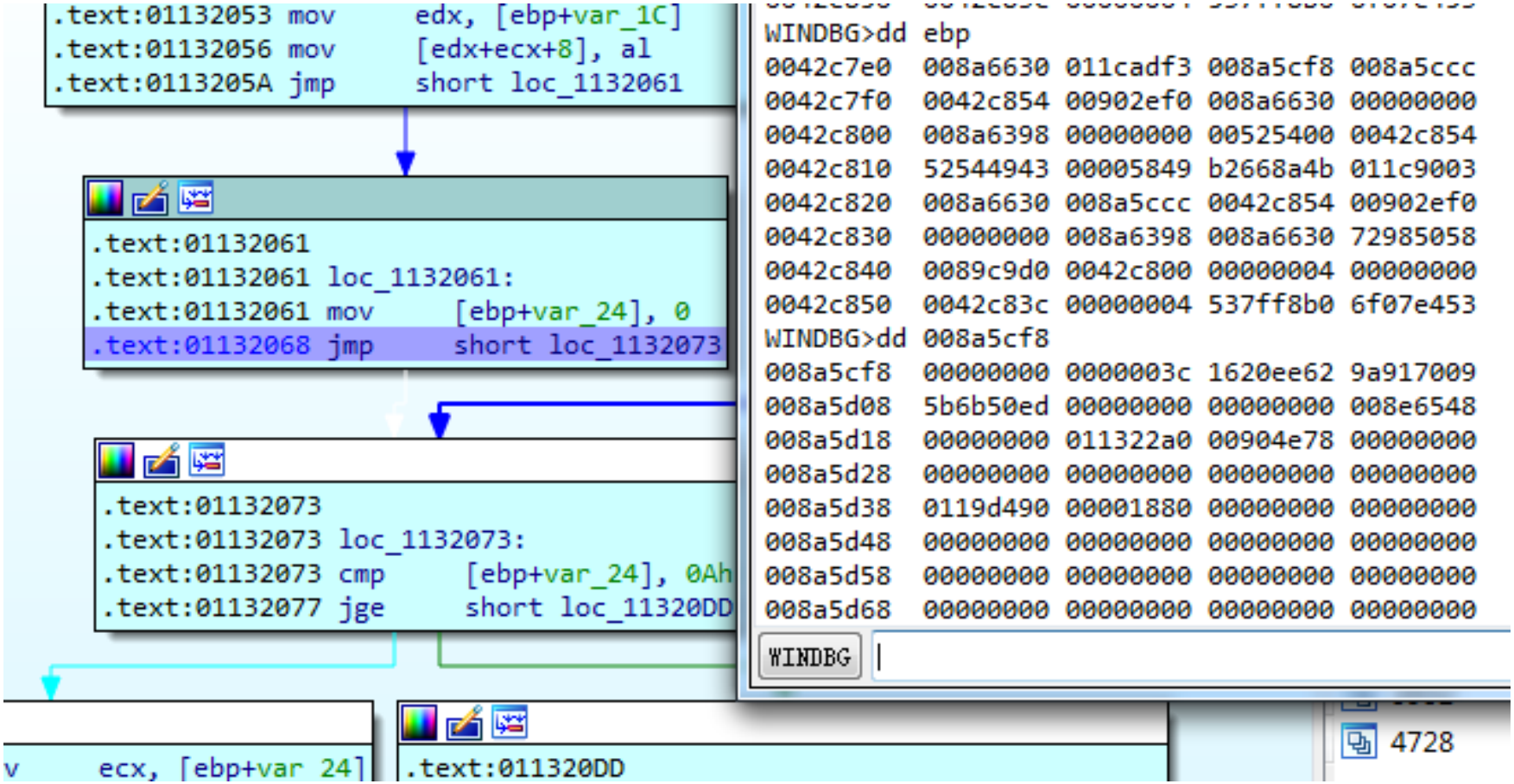
9、用Ctrl+鼠标滚轮缩放流程图，找到一长串有”time”注释后面的第二个JMP语句，F2下断点



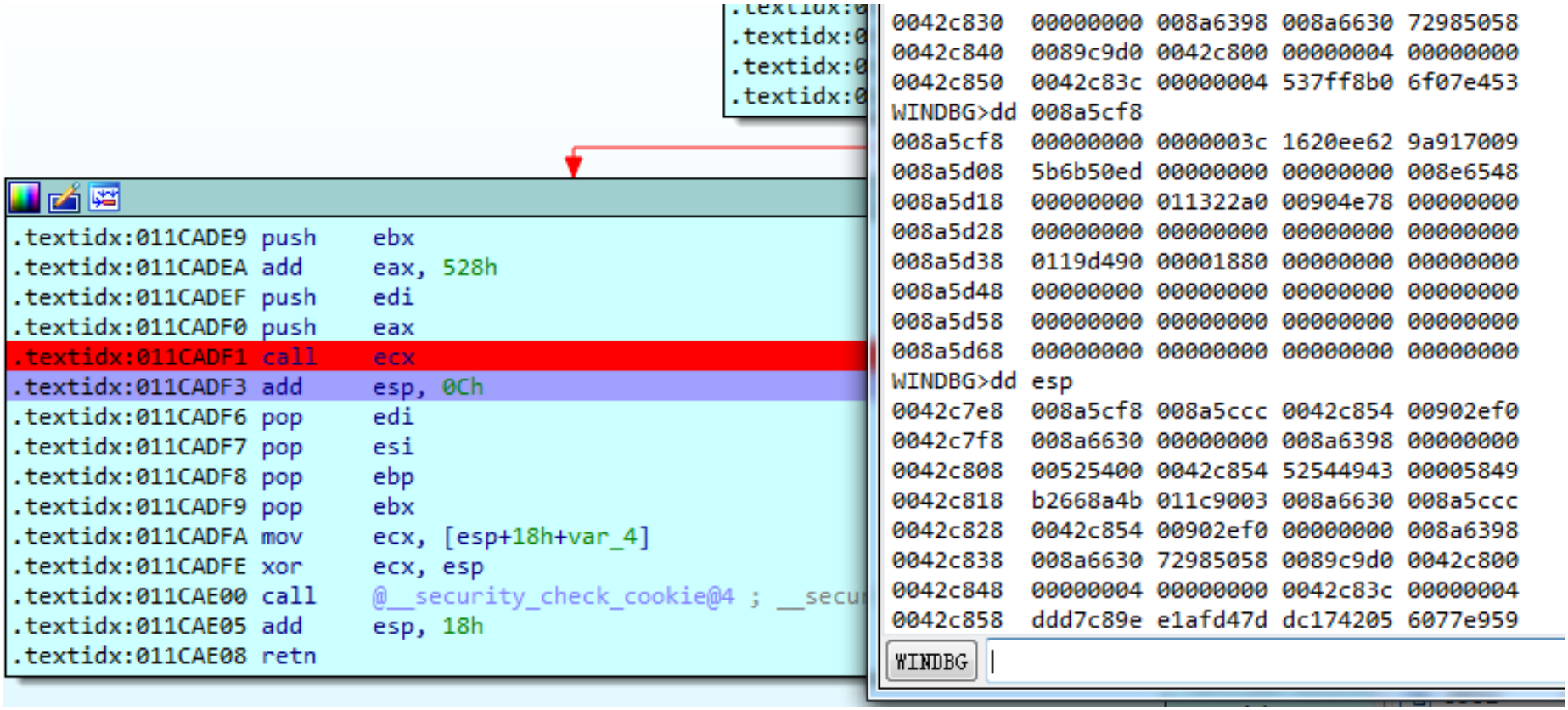
10、按F9继续调试，程序会停在上步设置的断点处。在输出窗口的WinDBG命令输入框中输入命令：dd ebp (dd后面具体要输入什么要根据JMP语句的上一条来确定）



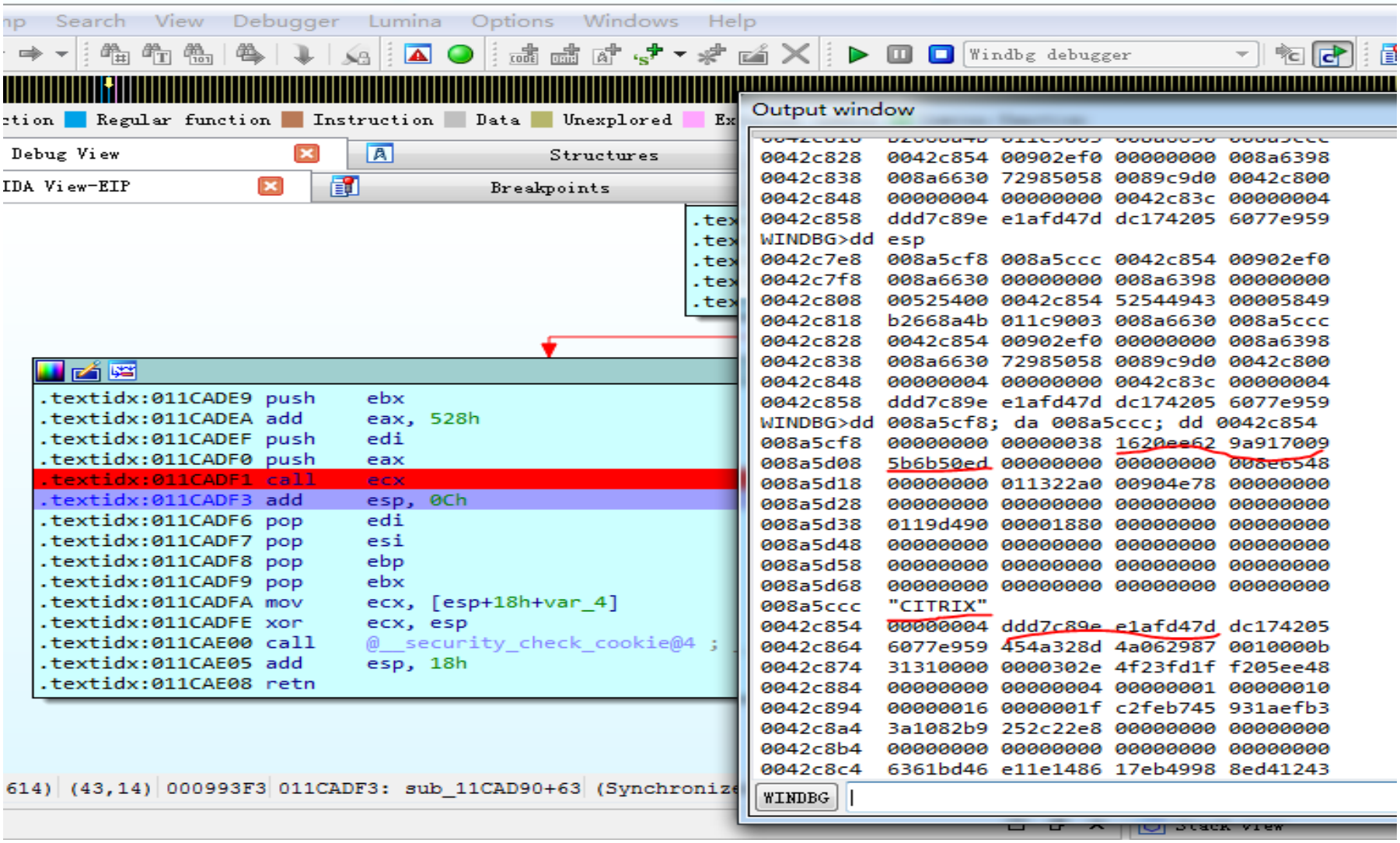
11、现在注意看“dd ebp” 后的输出结果，（结合图示）第三个地址EBP+8（ 008a5cf8 ）里是存着job结构的起始地址。（结合图示）在输出窗口的WinDBG命令输入框中输入命令：“ dd 008a5cf8 ”后就看到了job结构的值，（按顺序从job，job+4, job+8, job+c, job+10h …)。此时我们暂不修改任何值。



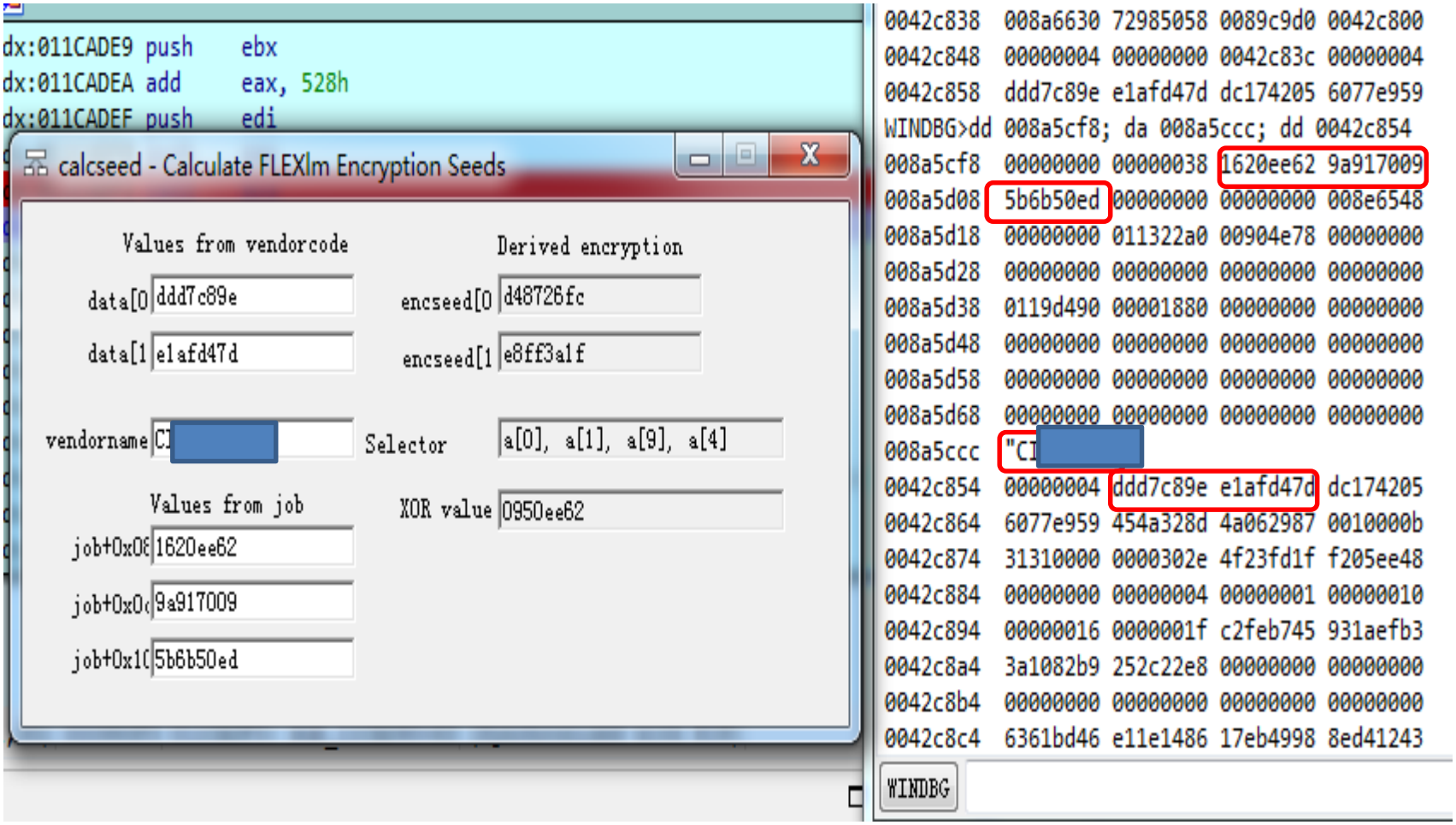
12、按F9继续调试，程序将停在call语句下面的语句，因为我们事先设置了断点。现在注意看WinDBG输入“dd esp” 后的输出结果（注意不是“dd ebp”了，也是在假设没有上一步修改job值的情况下），（结合图示）此时第一个地址ESP（008a5cf8）里是存着Job结构的起始地址，第二个地址ESP+4（008a5ccc）里是存着Vendor名字的地址，第三个地址ESP+8里（0042c854）是存着Data结构的起始地址。



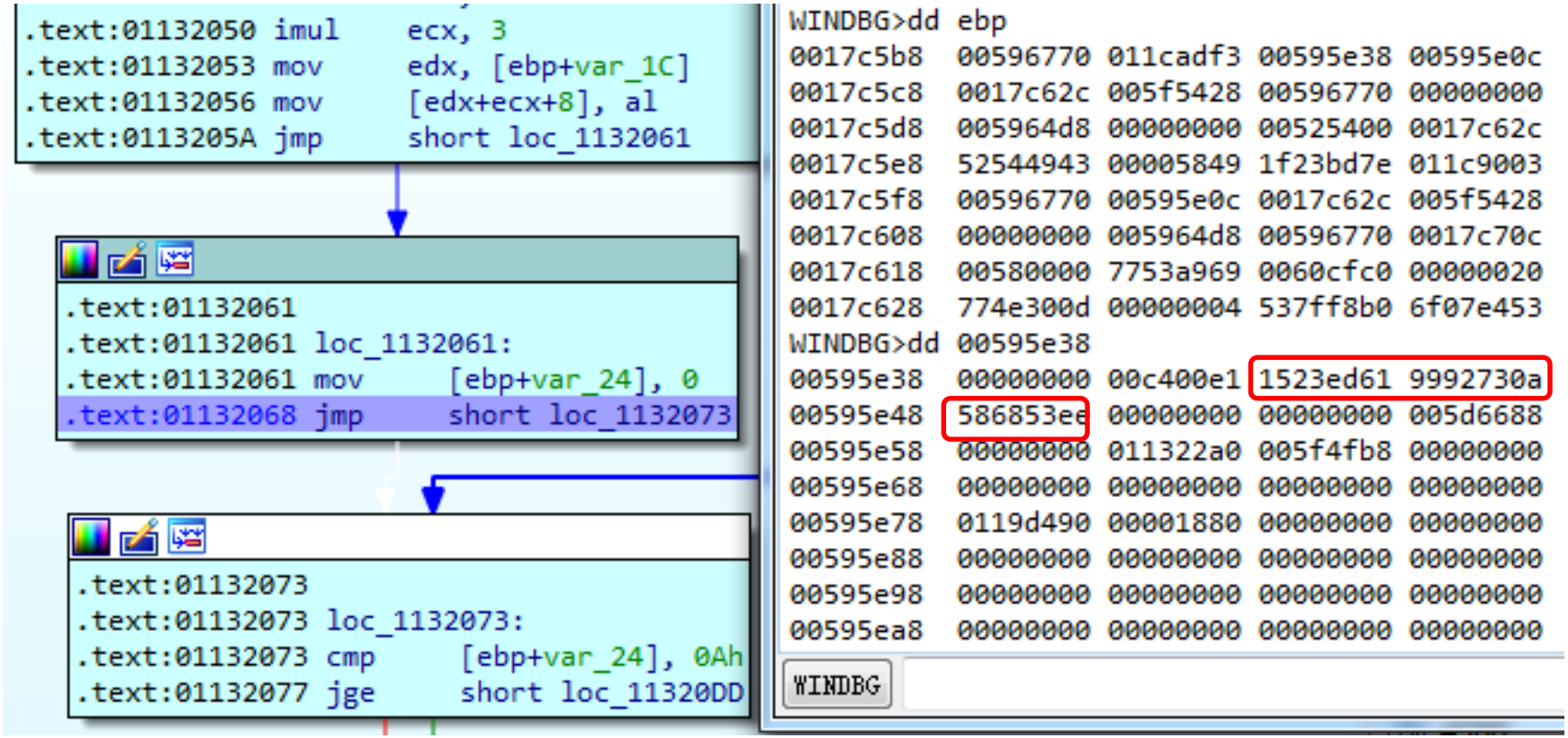
13、现在我们来看看其结构。现在WinDBG输入“dd 008a5cf8; da 008a5ccc; dd 0042c854”后的输出结果。（结合图示）此时第一个地址ESP（008a5cf8）里是存着Job结构的起始地址，第二个地址ESP+4（008a5ccc）里是存着Vendor名字的地址，第三个地址ESP+8里（0042c854）是存着Data结构的地址。



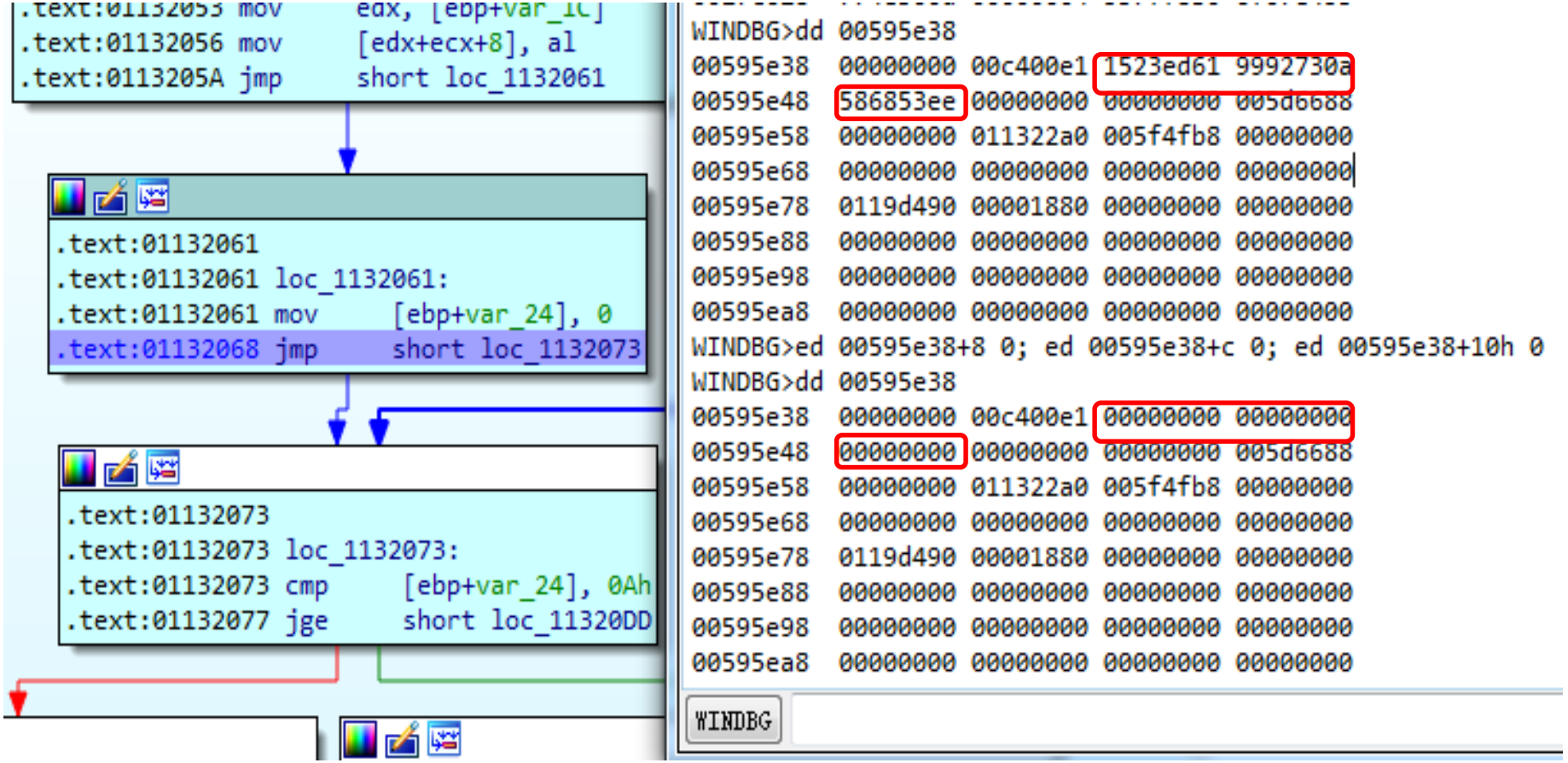
14、有了这些值，我们就可以利用工具calcseed.exe来计算加密种子点的值了。



15、按F9两次继续调试，程序会停在JMP断点处。在输出窗口的WinDBG命令输入框中输入命令：dd ebp (dd后面具体要输入什么要根据JMP语句的上一条来确定）。其中ebp+8中的地址对应的应该是与系统时间有关的随机数，也可能是job结构的值。在输出窗口的WinDBG命令输入框中输入命令“dd 008a5cf8”就看到其值。

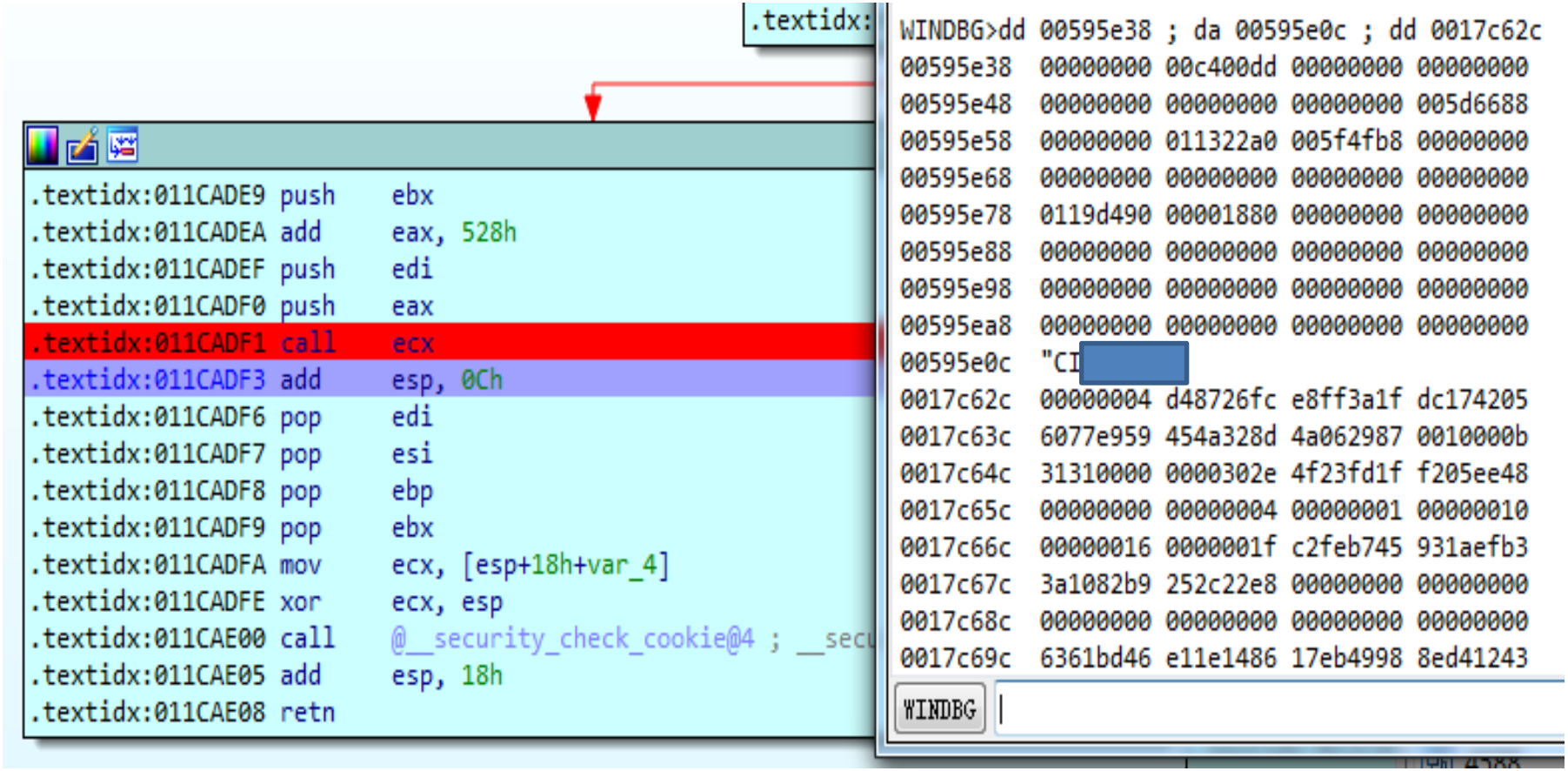


16、按照神人老外的研究成果，其中ebp+8中的地址对应的应该是与系统时间有关的随机数，也可能是job结构的值（本人不确定）。在此直接修改job+8，job+c，job+10h的值为0，在后面一步就直接得到加密种子点的值。根据上步说明的地址，在输出窗口的WinDBG命令输入框中（结合图示）输入命令“ed 00595e38+8 0; ed 00595e38+c 0; ed 00595e38+10h 0”修改其值。然后可以确认修改结果：dd 008a5cf8

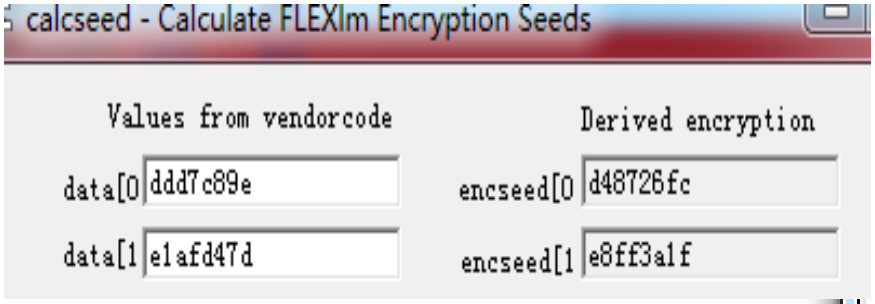


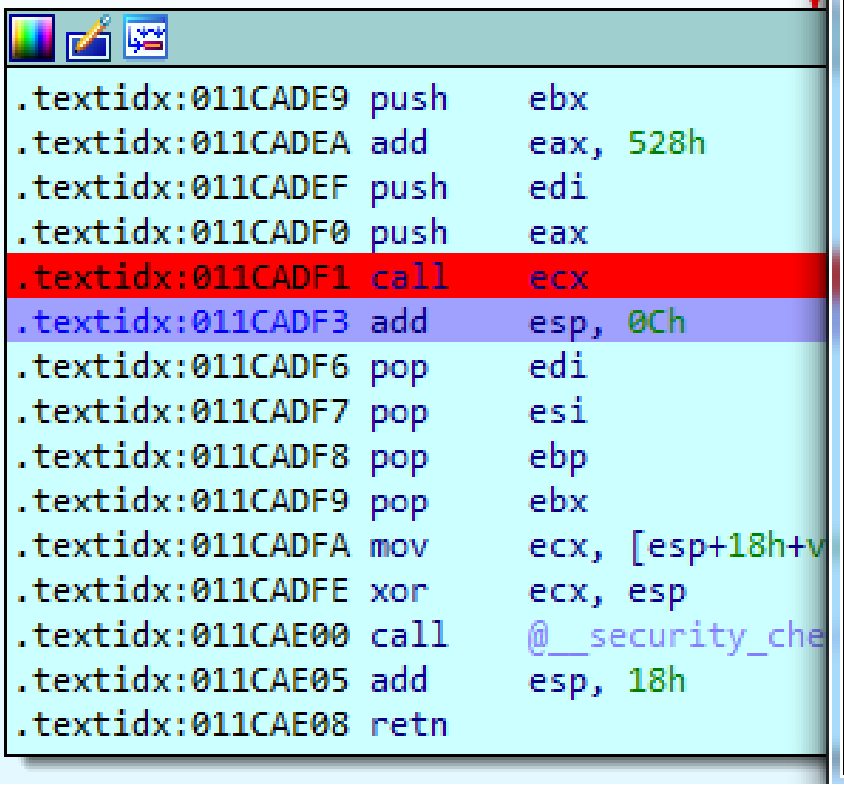
17、确认修改正确后按F9继续调试，程序将停在call语句下面的语句。现在注意看WinDBG输入“dd esp”后的输出结果（注意不是“dd ebp”），（结合图示）根据在第一次没有修改的情况下，理论上此时第一个地址ESP（00595e38）里是存着Job结构的起始地址，第二个地址ESP+4（00595e0c）里是存着Vendor名字的地址，第三个地址ESP+8里（0017c62c）是存着Data结构的起始地址。

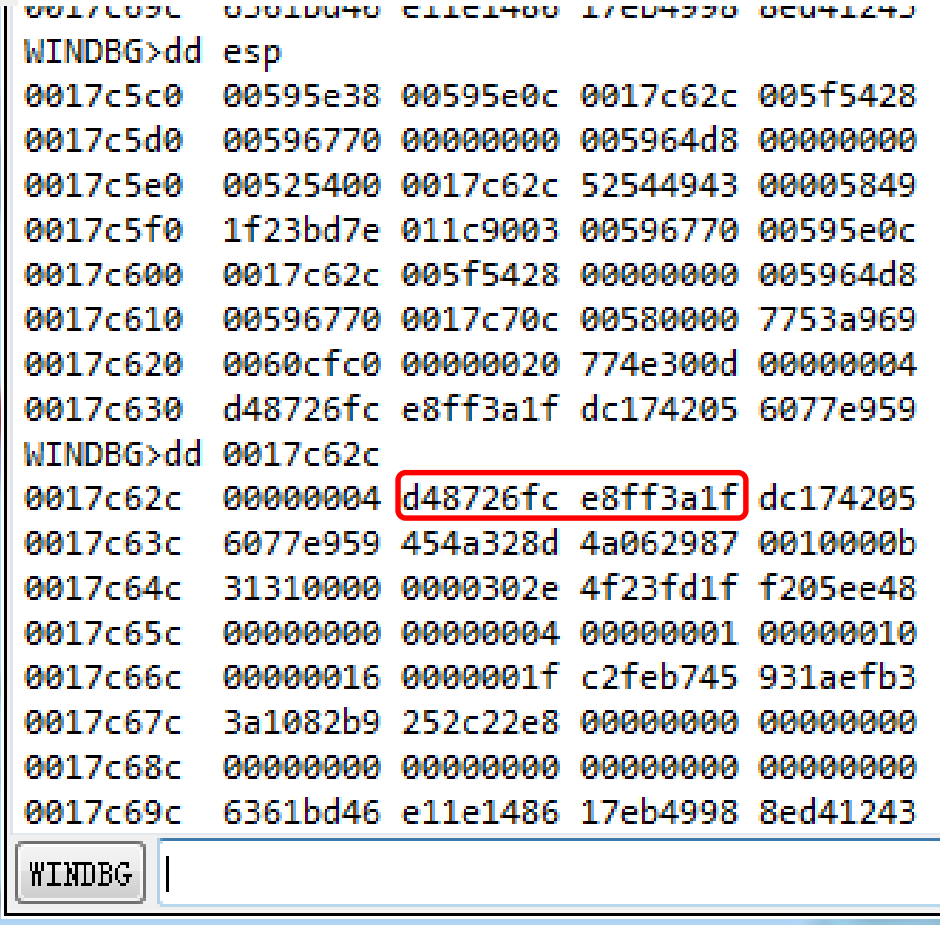
18、实际上由于我们的修改，再次用命令“dd 00595e38 ; da 00595e0c ; dd 0017c62c”查看，它们的值已经发生了改变（尽管其结构没变，这也是其要求的基本条件）。



19、现在WinDBG输入“dd 0017c62c”后（ESP+8），其data[0]和data[1]位置的结果即为两个加密种子点值。与工具计算出的值一致





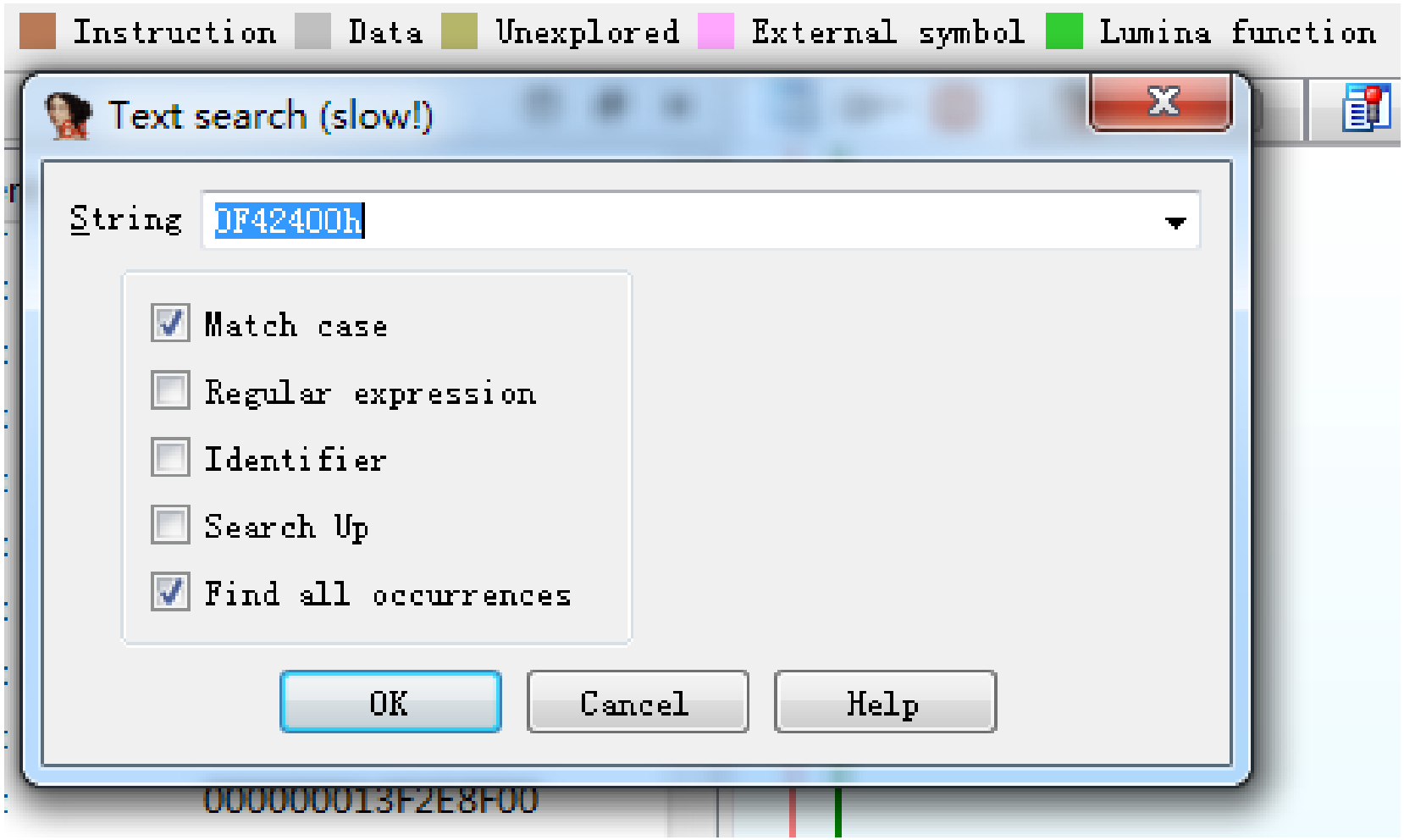


Flexnet Publisher V11.16及其以下版本加密种子点恢复方法

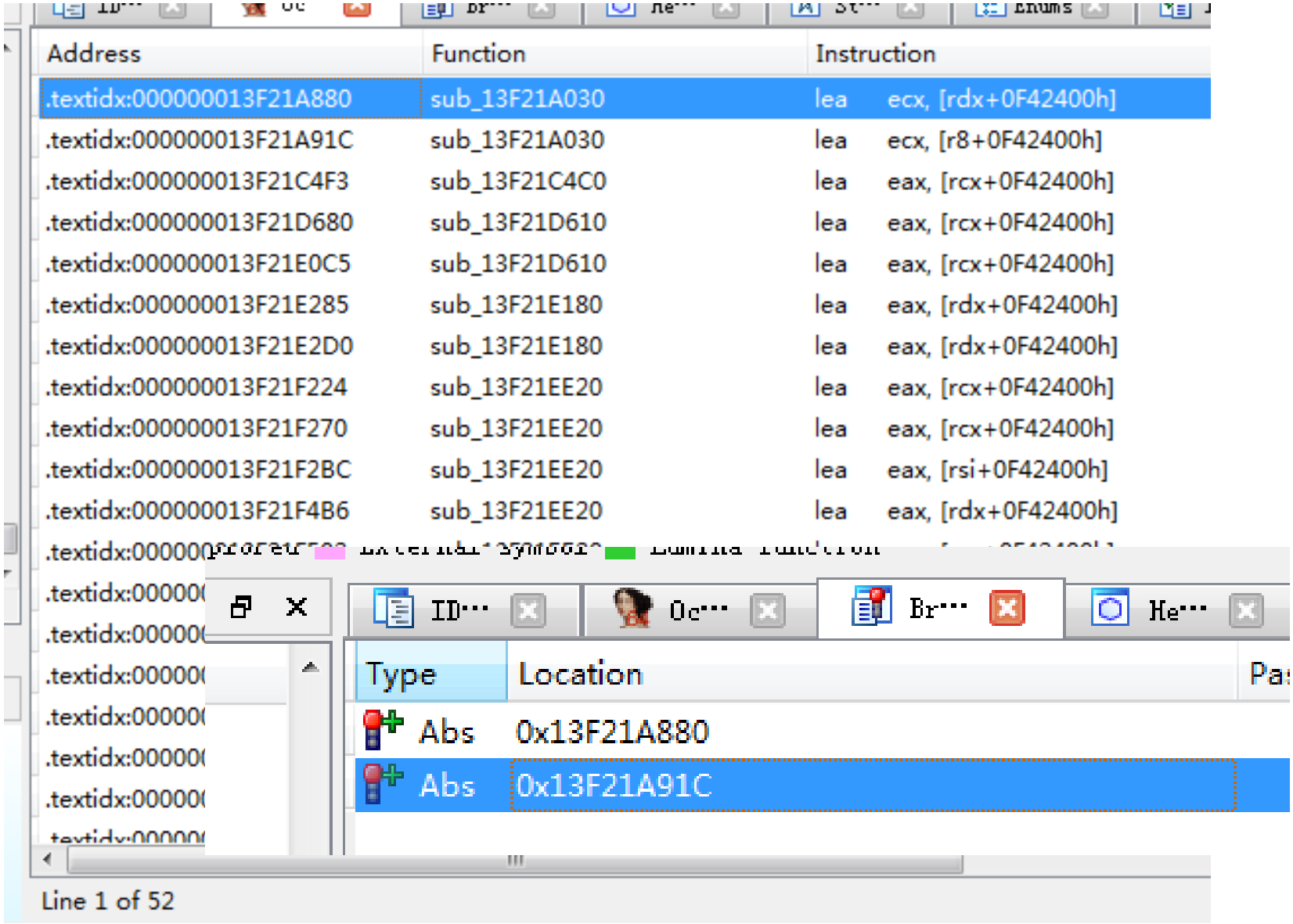
By: YangMyron

标志0F42400h定位法

1、在汇编窗口查找“0F42400h”



2、在搜索结果窗口双击找到的第一和第二两个“0F42400h”，并分别下断点。



3、设置好调试参数，不断F9后，如果能在断点处停下，第一处停下的rdx（x86的edx）值就是加密种子点1；第二处停下的r8d（x86的edx）值就是加密种子点2。（具体情况看本行命令里的寄存器是谁）。如果停不住就是该守护神不支持Non-TRL加密模式。