针对 FlexLM V11.11 以上版本的 ECC Pacth 方法

By:YangMyron

在本论坛里看到部分朋友对 FlexLM V11.11 以上版本的 ECC Patch 的方法和步骤感到困惑，认为老方法可能不好使了。通过我的研究发现 V11.11 以上版本需要 Patch 两个地方才能达到任意许可都不需要验证的效果。第一个地方大神们早就给出了方法，就是 Patch 函数 l_pubkey_verify 的返回值；第二个地方有部分坛友提到，但没有总结出好的定位办法。我就不谈原理，直接把两个 Patch 地址的定位方法做一总结，与大家分享！

工具材料：IDA Pro V7.0， UltraEdit

参考资料：各位大神在本论坛的神贴，不一一列举

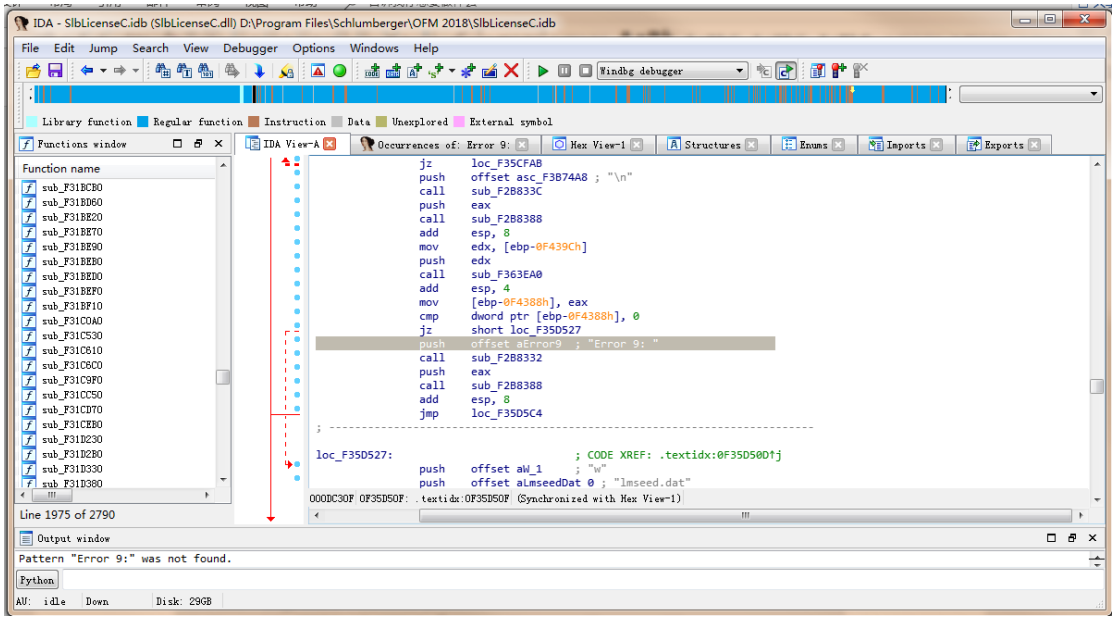
第一个地方：

（1） 在 IDA 中查找对字符串“Error 9:”引用，对应的源代码如，再往上很容易确定出函数 sb_end。（如下图为函数 sub_F363EA0）

代码:

```
    ret  =  sb_end(global_data);
    if  (ret  !=  SB_SUCCESS)
    {
        fprintf(stderr,  "Error  9:  ");
        goto  exit_seed;
    }
/*此处是关键的识别点，往上可找到 sb_end*/
    if  (!(fp  =  fopen("lmseed.dat",  "w")))
        fprintf(stderr,  "Can't  open  lmseed.dat");
    else
    {
        fprintf(fp,
    "\
Once  set,  the  values  for  LM_SEED1-3  must  be  kept  secret  \n\
```

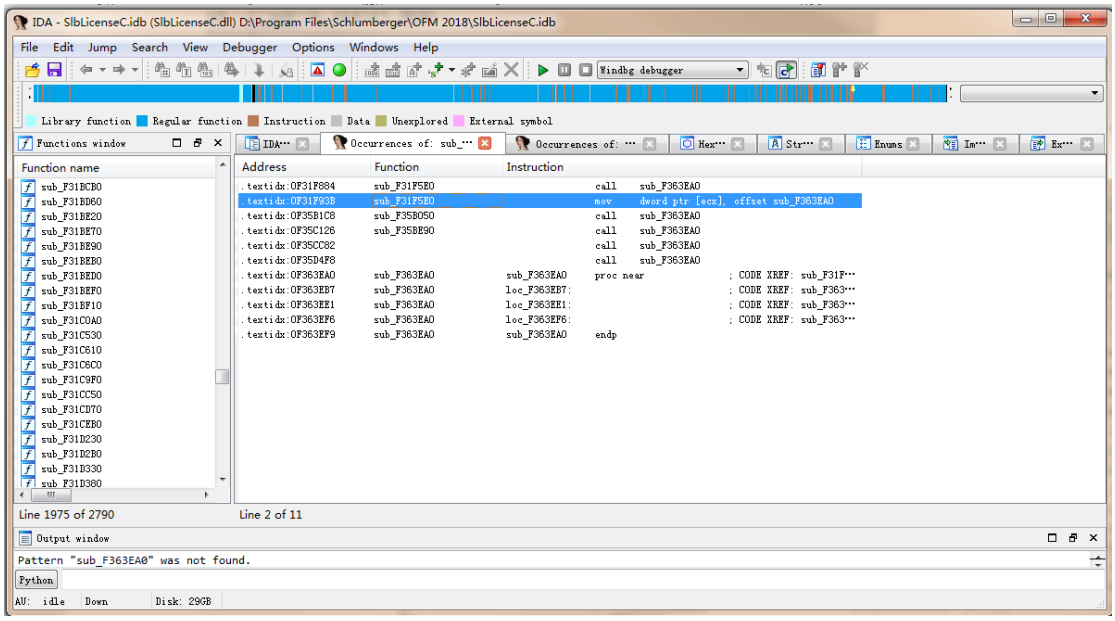
```
and  *never*  change.\n\n\
#define  LM_SEED1  0x%08x\n\
#define  LM_SEED2  0x%08x\n\
#define  LM_SEED3  0x%08x\n",  seed1,  seed2,  seed3);
        fclose(fp);
    }
```

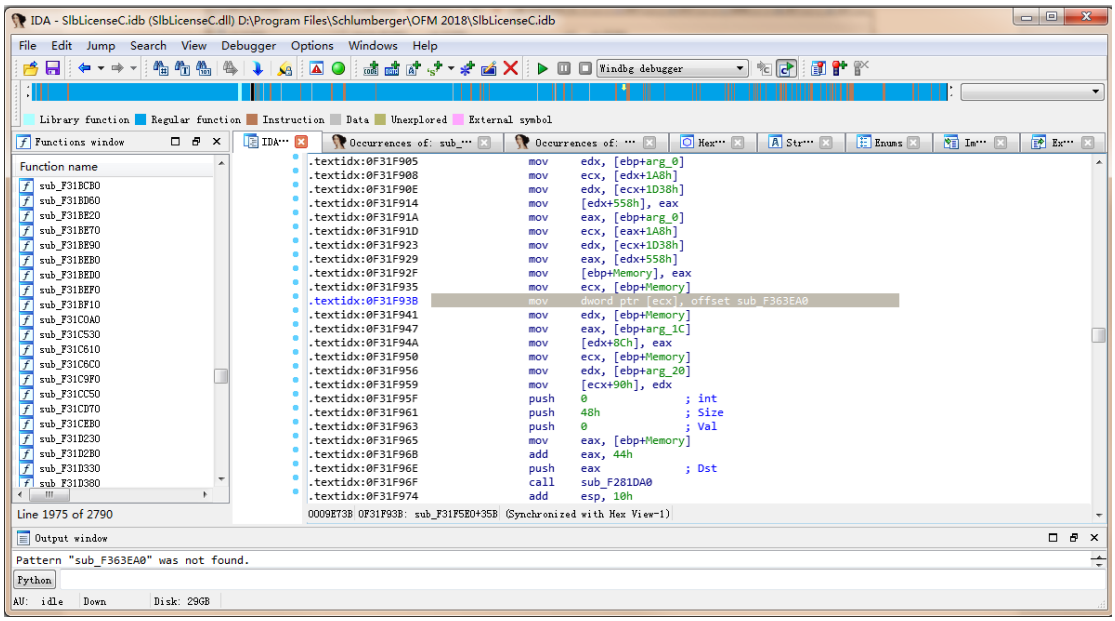


（2）查找对 sb_end 的调用，其中有一个应为 mov [addr], sb_end，（对应 x64 应该是 lea [addr], sb_end, 如 lea rcx, sub_1401E53E0）对应的源代码如下，此处所在的函数即为 l_pubkey_verify

代码:

```
if  (!job->L_VERIFY_MEM)
    {
        typedef  void  (*cleanup)(char  *);
        m  =  (VERIFY_MEM  *)(job->L_VERIFY_MEM  =
            (char  *)l_malloc(job,  sizeof(*m)));
        m->cleanup  =  (cleanup)sb_end;
        m->strength  =  strength;
        m->sign_level  =  sign_level;
```



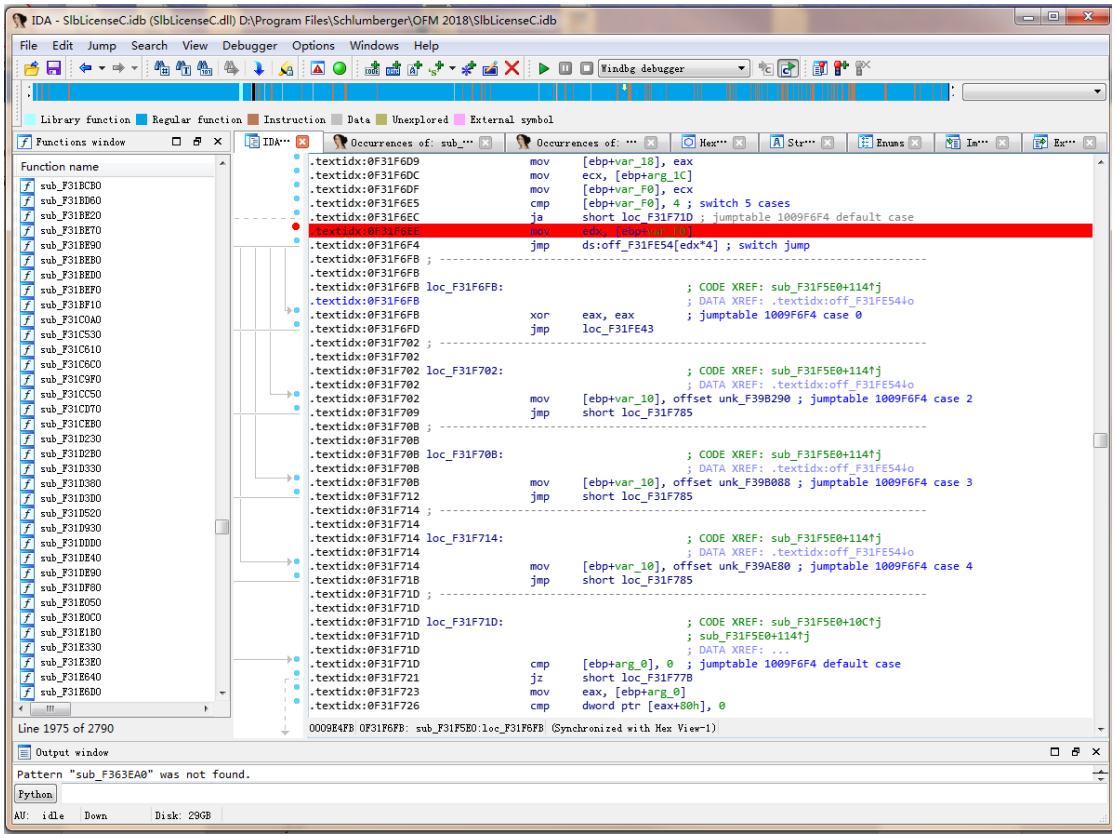


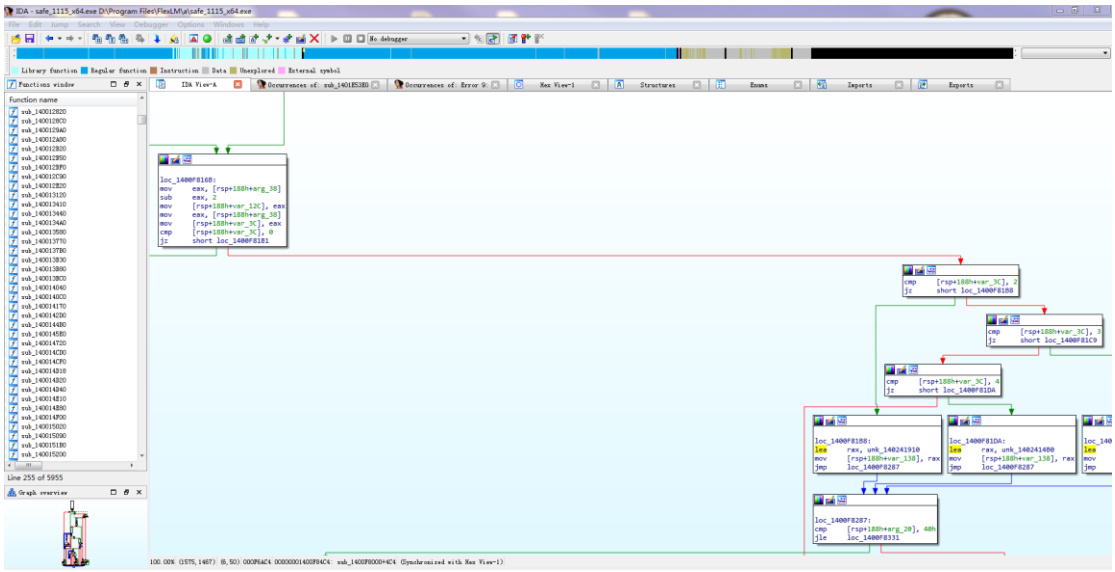
（3）在找到的 mov [addr], sb_end 继续往上，找到选择分支结构，对应的源代码如下。此处的爆破方法很多，可以直接将 strength 置为 0，变成 12 个字符的 SIGN，通过查找加密种子点利用 FlexLM 的 SDK 来产生。（注意 x64 的四个 case 不明显，应该用 Graph 方式看比较方便）

代码：

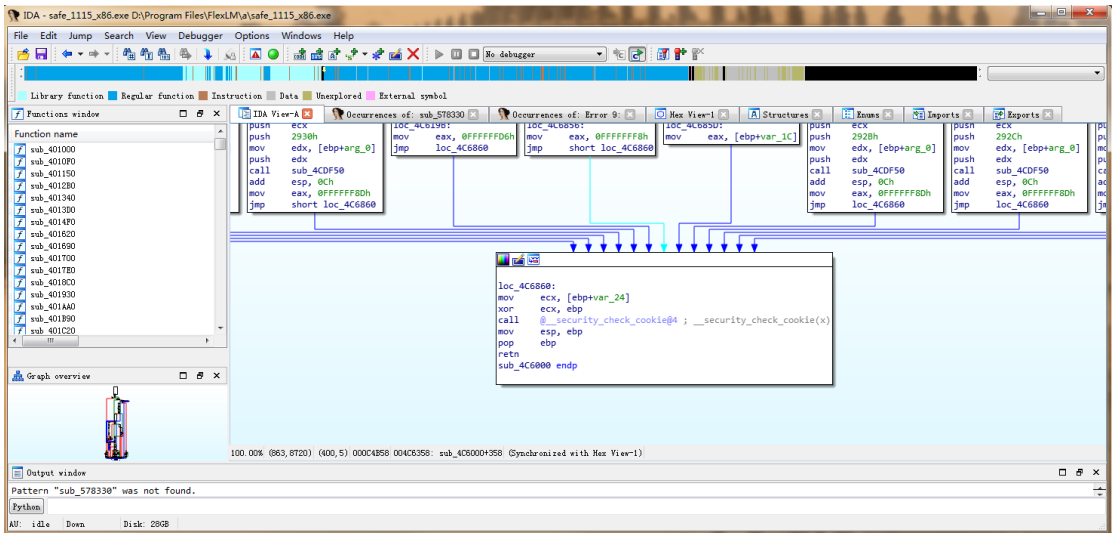
```
    switch(strength)
    {
    case  LM_STRENGTH_LICENSE_KEY:  return  0;
    case  LM_STRENGTH_113BIT:  ellipticCurve  =
                                &LM_PUBKEY_CURVE113BIT;  break;
    case  LM_STRENGTH_163BIT:  ellipticCurve  =
&LM_PUBKEY_CURVE163BIT;  break;
```

```
    case  LM_STRENGTH_239BIT:  ellipticCurve  =
&LM_PUBKEY_CURVE239BIT;  break;
    default:
        {
            LM_SET_ERRNO(job,  LM_BADPARAM,  531,  0);
            ret  =  LM_BADPARAM;
            goto  exit_verify;
        }
    }
```

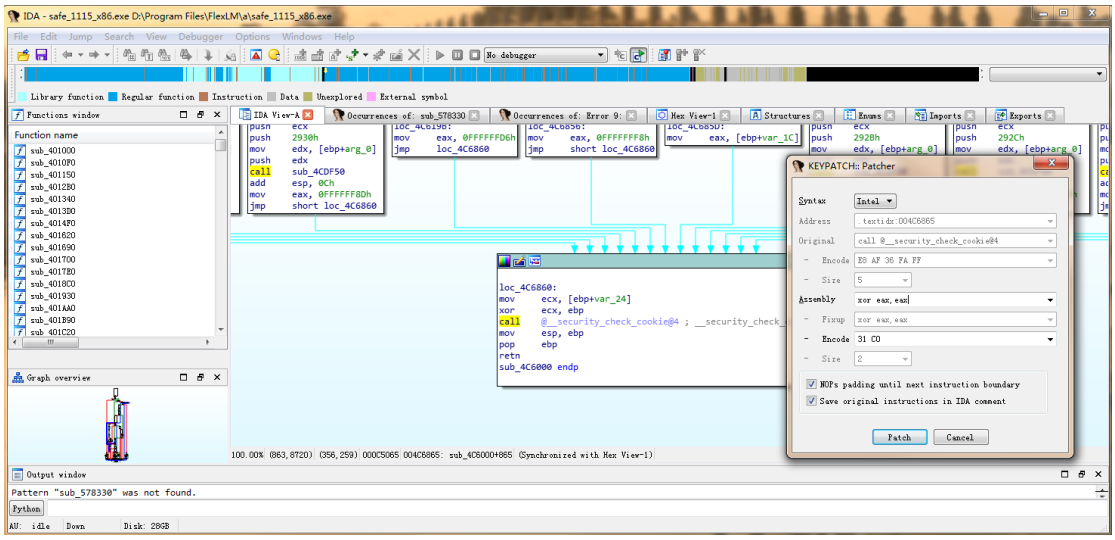


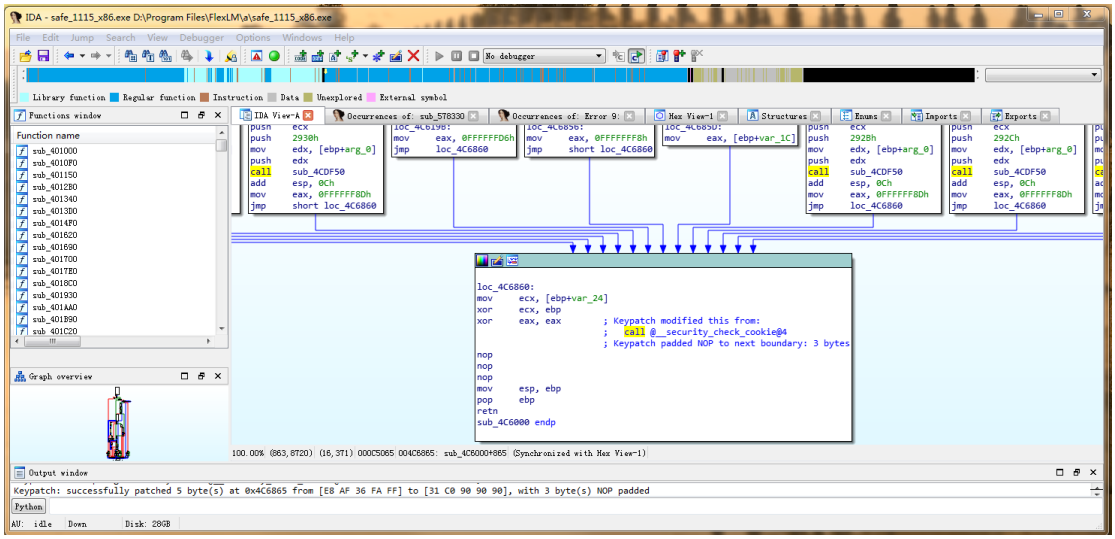


（4）现在演示爆破法，在 IDA-View 窗口按空格键，以 Graph 方式显示，显示该函数的尾部



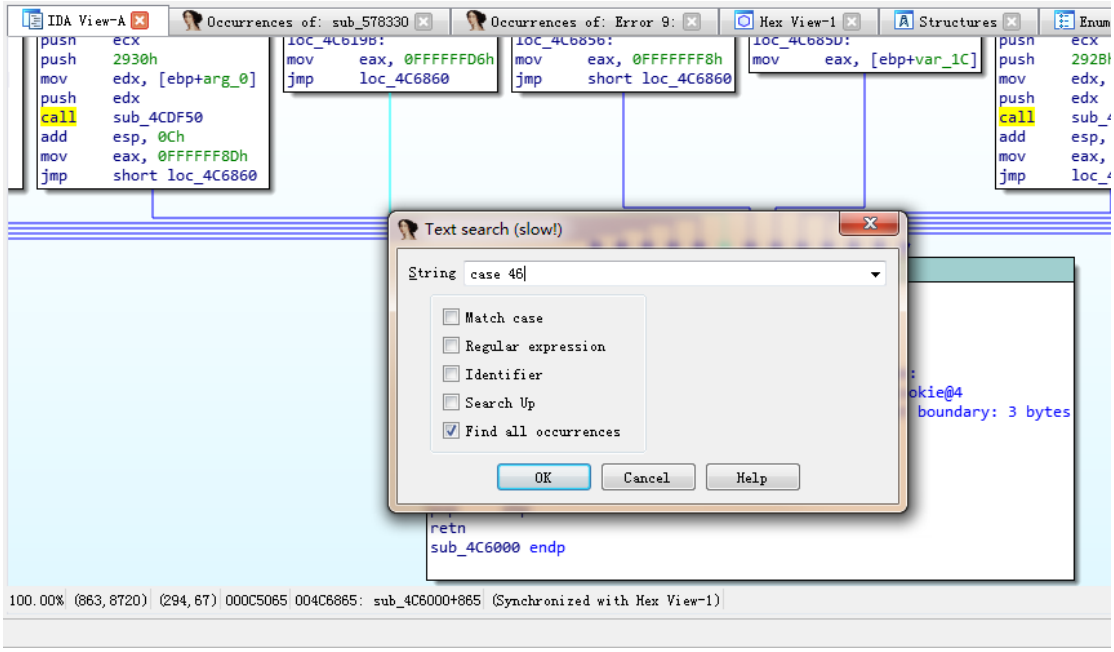
把尾部的 call @__security…修改为”xor eax,eax”和三个 nop

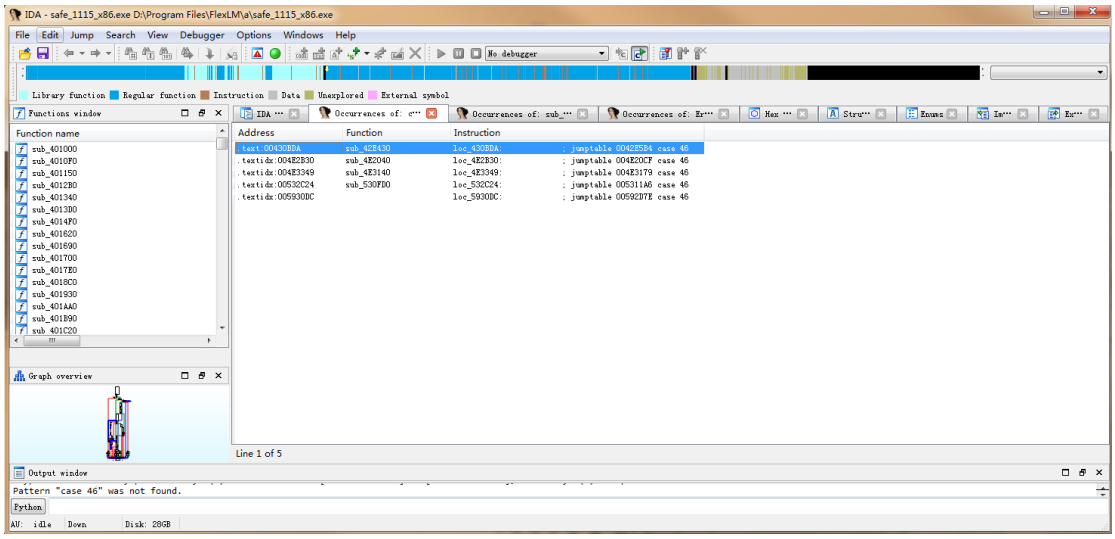




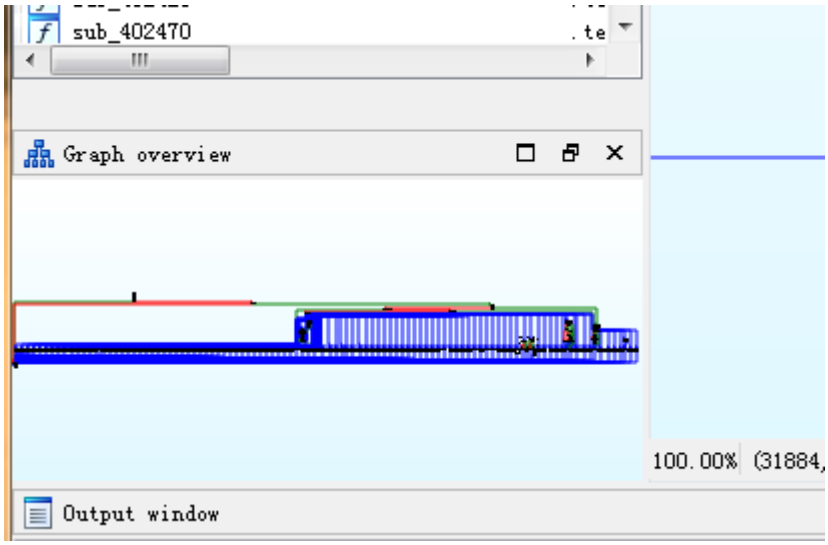
第二个地方：

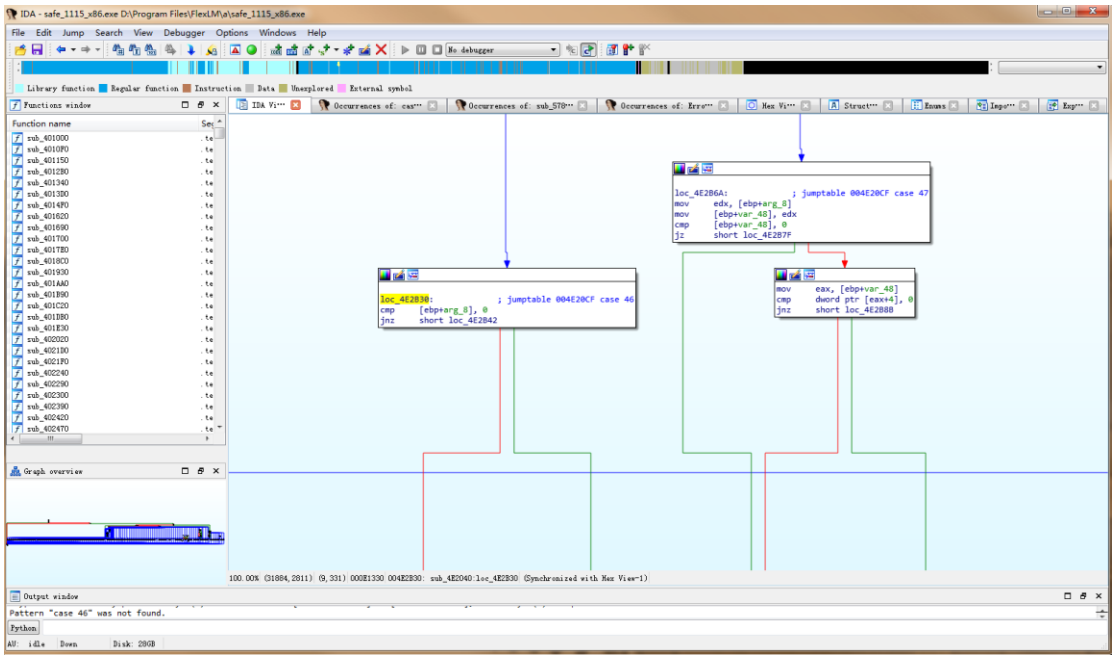
（1）在 IDA 中查找字符串“case 46“，一般会找到 5 个左右



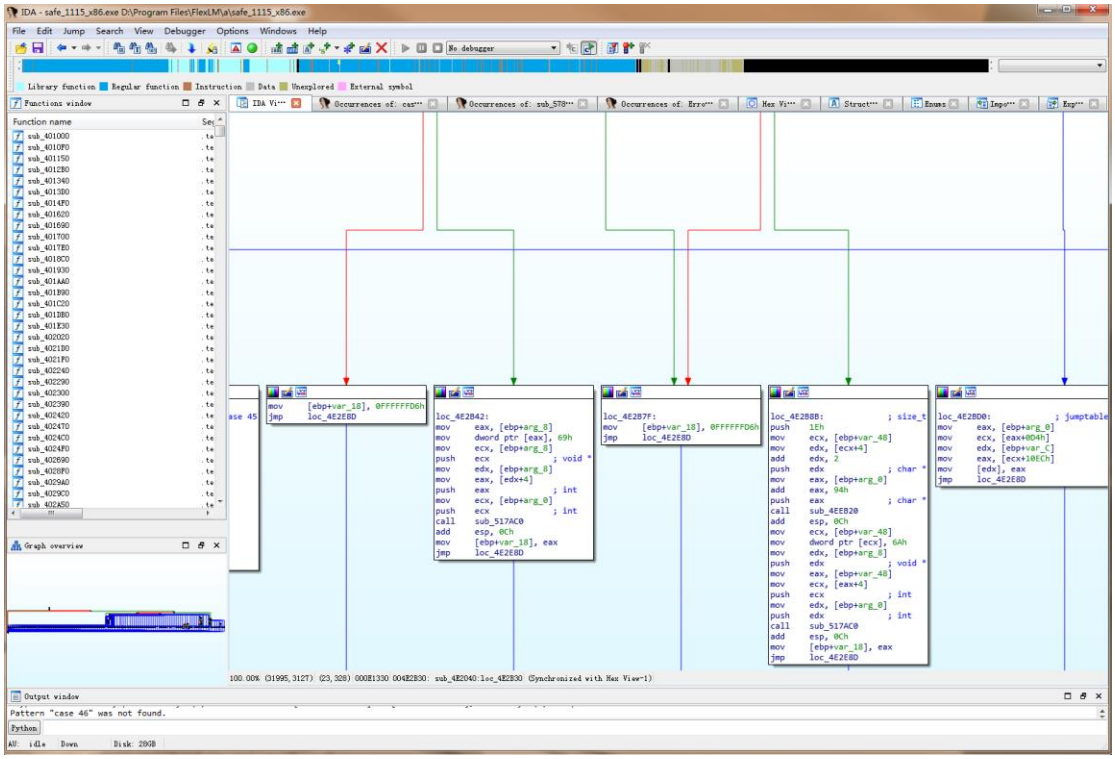


（2）双击每一个找到的“case 46“，在小 Graph Overview 窗口观察显示，细长型的那个才是我们要的，这点很重要！而且在大 IDA-View 窗口能同时看到附近有”case 47“存在。一般对应上面的第二个“case 46“（不排除有例外情况）。

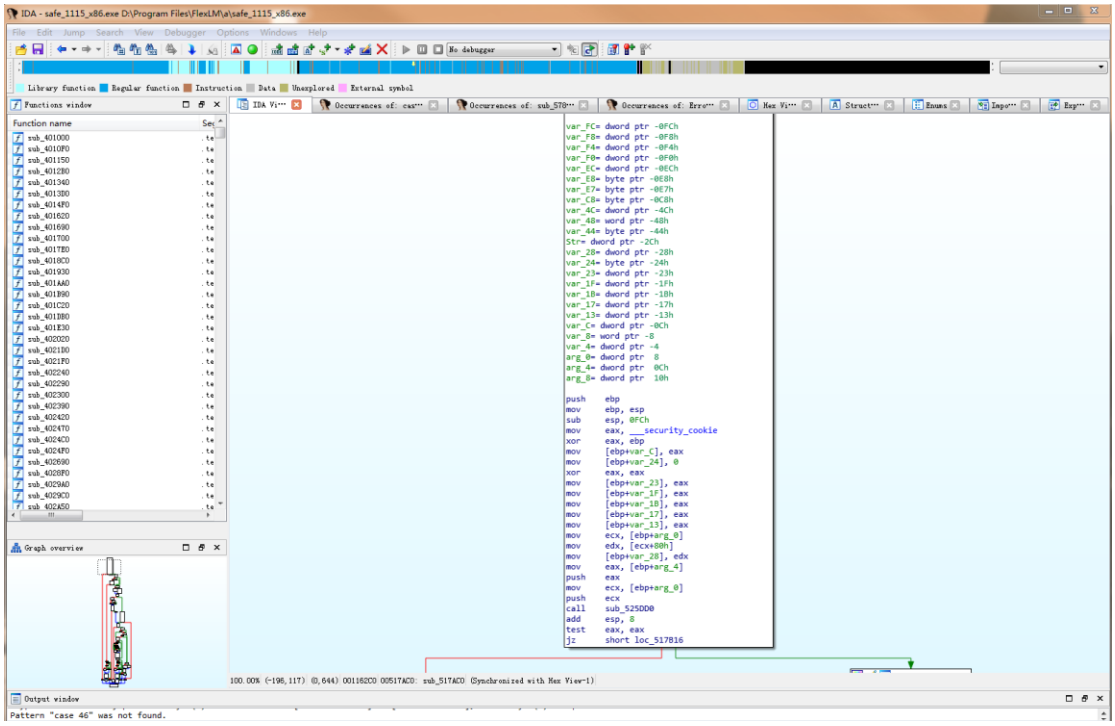


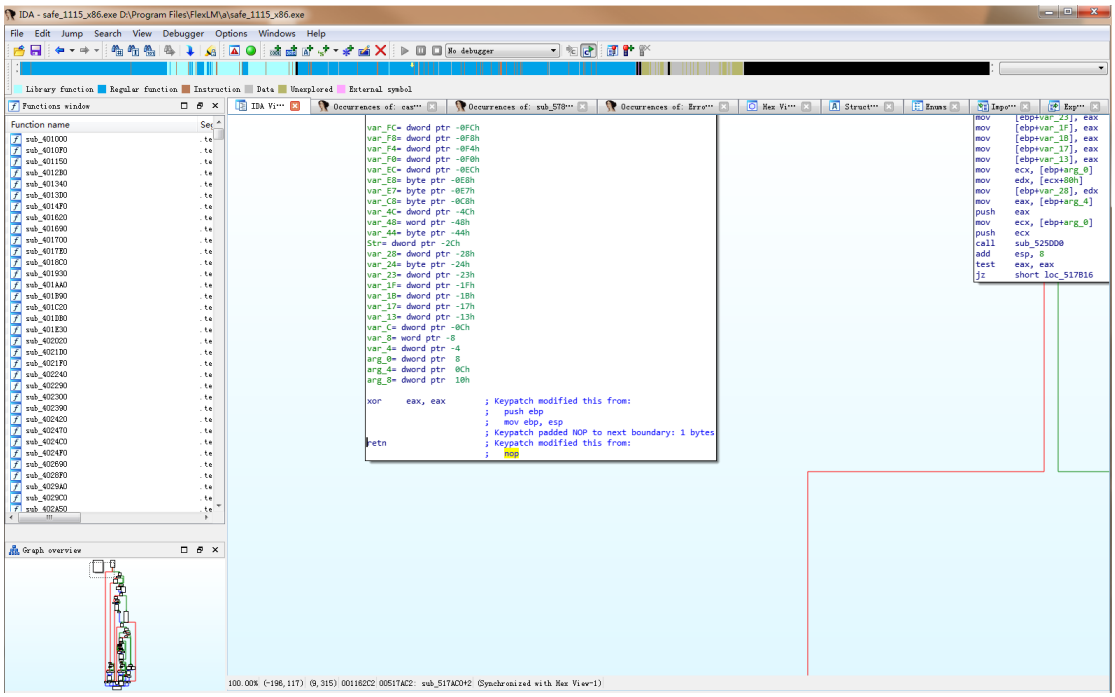


（3）在大 IDA-View 窗口拖动流程图，沿“case 46“的绿色分支观察，其中有一个 call 调用函数。沿“case 47“的的红色再到绿色分支中也有 call 调用这个函数。

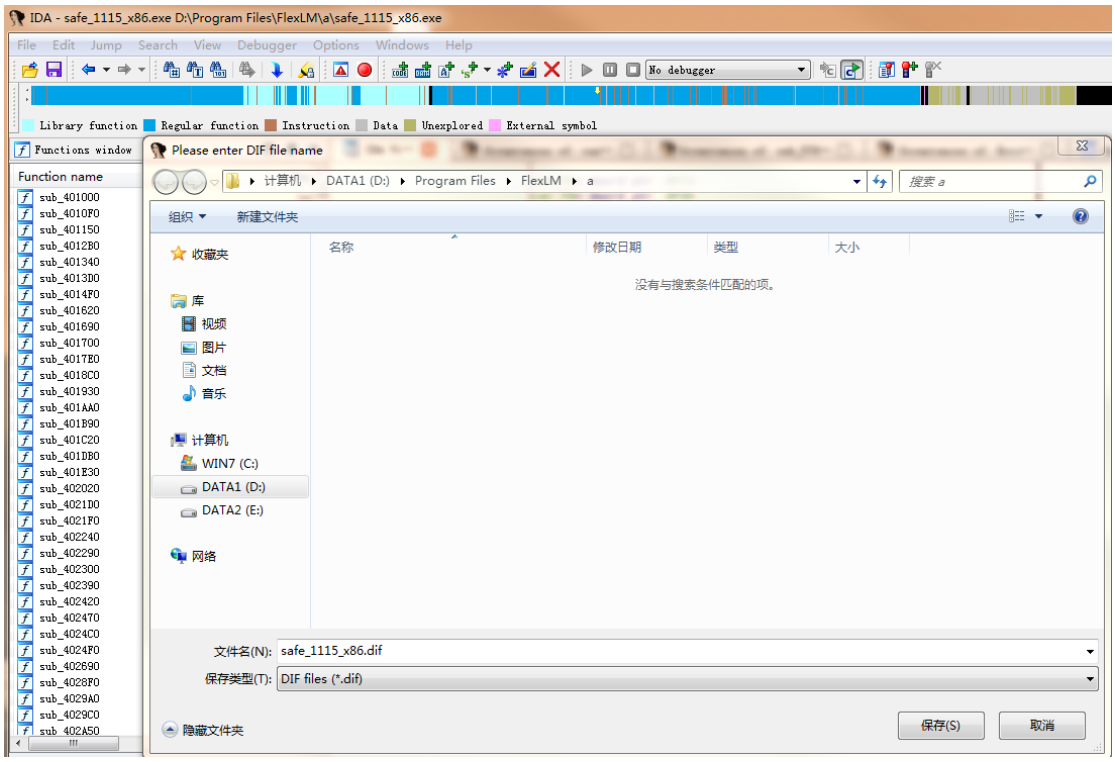


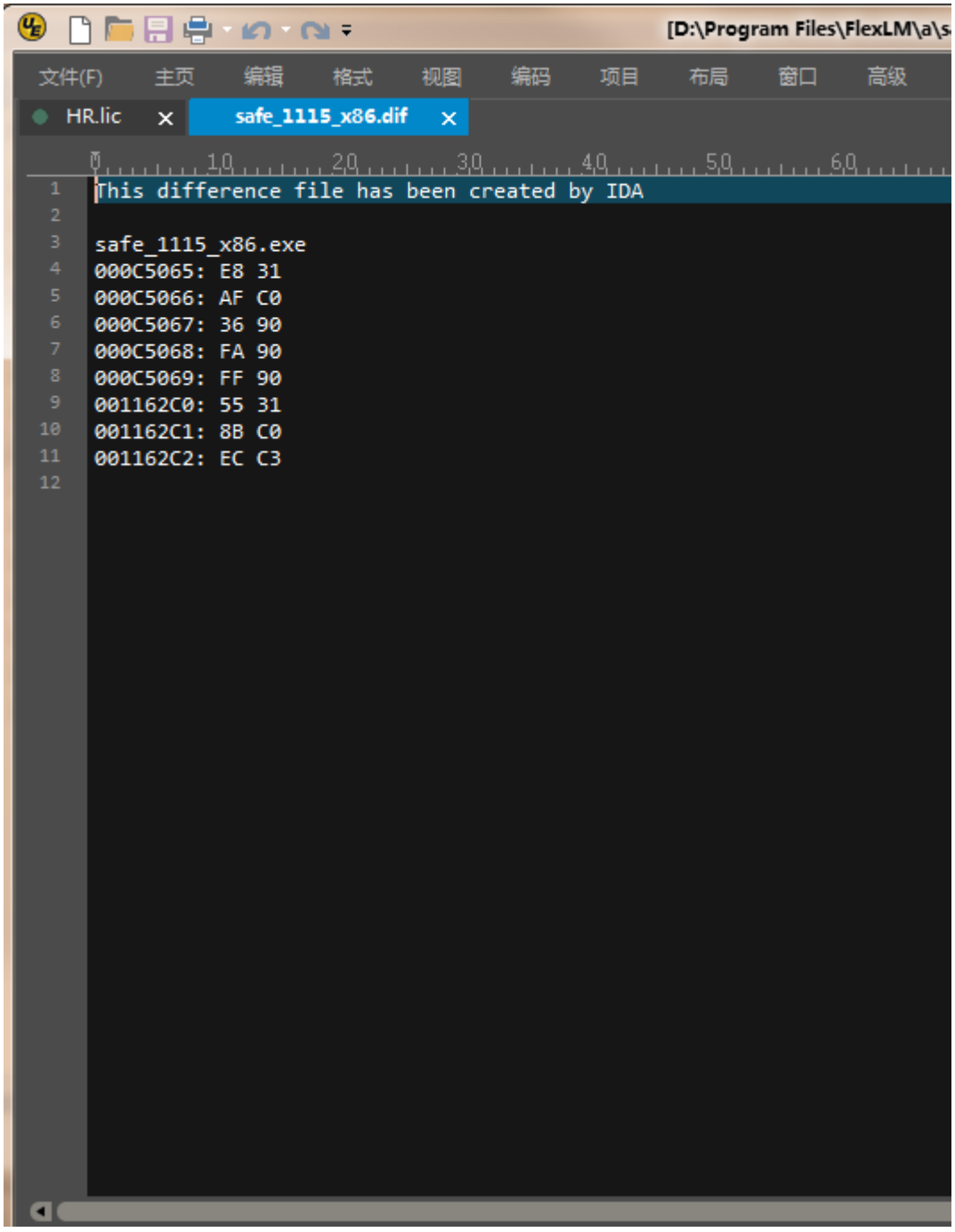
（4）双击该函数，到该函数的头部。直接从 Push ebp 开始修改为”xor eax, eax”和“retn“。（在 x64 里，因为没有 Push 语句，直接从 mov 语句开始修改）





(5)利用 IDA 的 Production File 生成 dif 文件，用 UltraEdit 修改相应的字节值即完成 Pacher。


IDA - safe_1115_x86.exe D:\Program Files\FlexLM\a\safe_1115_x86.exe
File Edit Jump Search View Debugger Options Windows Help
No debugger
Library function Regular function Instruction Data Unexplored External symbol
Functions window
Function name
sub_401000
sub_4010F0
sub_401150
sub_4012B0
sub_401340
sub_4013D0
sub_4014F0
sub_401620
sub_401690
sub_401700
sub_4017E0
sub_4018C0
sub_401930
sub_401AA0
sub_401B90
sub_401C20
sub_401DB0
sub_401E30
sub_402020
sub_4021D0
sub_4021F0
sub_402240
sub_402290
sub_402300
sub_402390
sub_402420
sub_402470
sub_4024C0
sub_4024F0
sub_402690
sub_4028F0
sub_4029A0
sub_4029C0
sub_402A50
Please enter DIF file name
计算机 ▸ DATA1 (D:) ▸ Program Files ▸ FlexLM ▸ a
搜索 a
组织
新建文件夹
收藏夹
库
视频
图片
文档
音乐
计算机
WIN7 (C:)
DATA1 (D:)
DATA2 (E:)
网络
名称
修改日期
类型
大小
没有与搜索条件匹配的项。
文件名(N): safe_1115_x86.dif
保存类型(T): DIF files (*.dif)
隐藏文件夹
保存(S)
取消


[D:\Program Files\FlexLM\a\s
文件(F)
主页
编辑
格式
视图
编码
项目
布局
窗口
高级
HR.lic
safe_1115_x86.dif
This difference file has been created by IDA
safe_1115_x86.exe
000C5065: E8 31
000C5066: AF C0
000C5067: 36 90
000C5068: FA 90
000C5069: FF 90
001162C0: 55 31
001162C1: 8B C0
001162C2: EC C3