目录

[一、基本信息 2](#_Toc55665478)

[二、样本描述 2](#_Toc55665479)

[三、样本分析 2](#_Toc55665480)

[宏代码分析 3](#_Toc55665481)

[DLL文件分析 4](#_Toc55665482)

[Shellcode分析 7](#_Toc55665483)

[参考 11](#_Toc55665484)

## 一、基本信息

|  |  |
| --- | --- |
| 文件名 | \_\_\_-\_\_\_\_2019\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.doc |
| MD5 | 3C3B2CC9FF5D7030FB01496510AC75F2 |
| 文件大小 | 2637824 bytes |
| 病毒类型 | 宏病毒 |
| APT组织 | 海莲花 |

## 二、样本描述

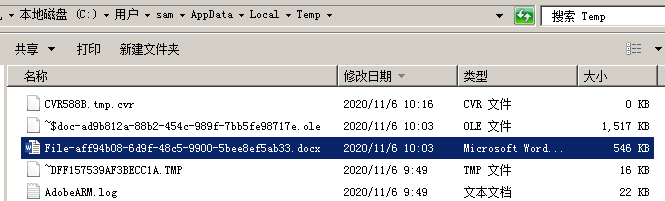
样本的攻击手法符合海莲花的一贯作风，精准投放钓鱼文档诱导用户点击带有恶意宏的Word文档运行，然后调用regsvr32.exe注册释放出来的DLL文件，进一步DLL通过执行其中的shellcode实现恶意下载远控木马等功能。

## 三、样本分析

样本为一个名称是“\_\_\_-\_\_\_\_2019\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.doc”的word文档，打开后显示此文件受保护，诱导用户点击启用内容以执行其恶意宏代码。

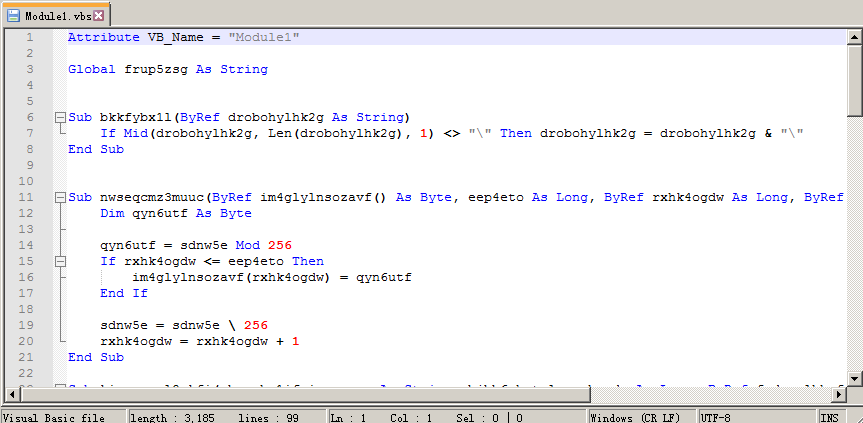


点击启用内容后恶意功能执行起来，关闭当前文档并打开释放出的Word文档，其父进程为regsvr32.exe。

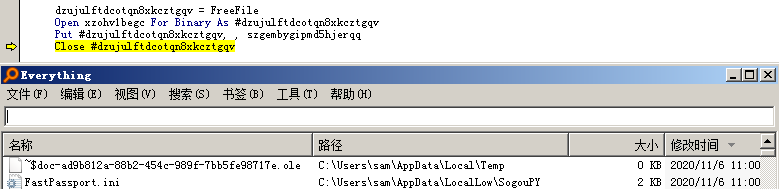


### 宏代码分析

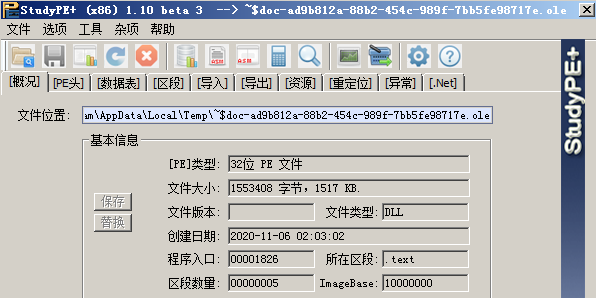
通过 oledump.py 获取其宏代码尝试分析，可以看到为了增加分析难度，其中大量变量为无意义的字符串：



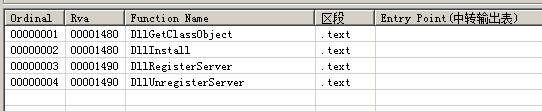
为了进一步了解宏代码的逻辑，按住shift点击启用内容，然后Alt+F11打开宏代码，进行调试。在AutoOpen()中可以看到拼接了 TEMP\~$doc-ad9b812a-88b2-454c-989f-7bb5fe98717e.ole 这样一个文件路径，通过Put函数写入数据：



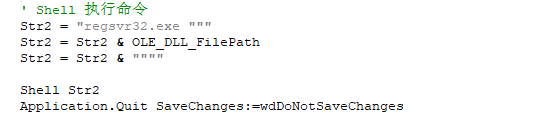
动态捕获查看文件信息可知为一个DLL文件：



同时包括4个与COM 组件和服务相关的导出函数：

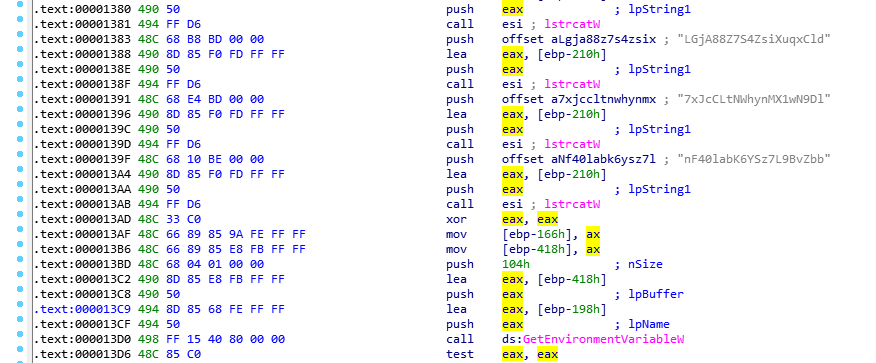


进一步构造一个字符串通过Shell执行：regsvr32.exe "C:\Users\sam\AppData\Local\Temp\~$doc-ad9b812a-88b2-454c-989f-7bb5fe98717e.ole""，执行后注册了该DLL，DLL中打开新的 Word文档。

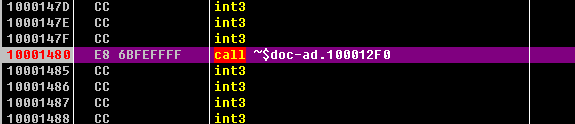


### DLL文件分析

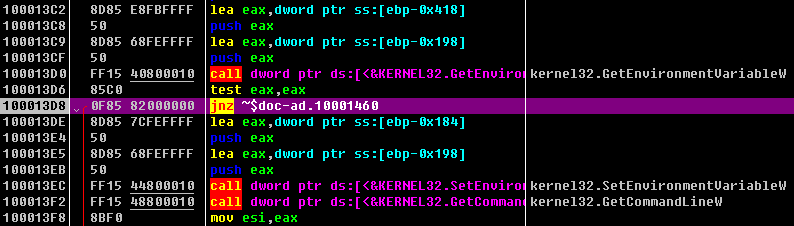
IDA载入文件之后，4个函数内容皆call了DllUnregisterServer\_0，查看内容为拼接了一个长度为200的字符串，继而查看是否存在该环境变量，不存在则进行设置，存在的话则跳转到后续逻辑。

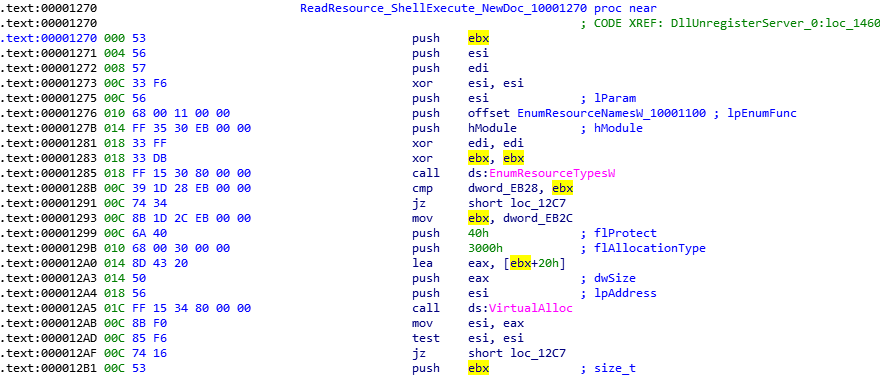


动态调试一下，直接设置eip到该函数入口：

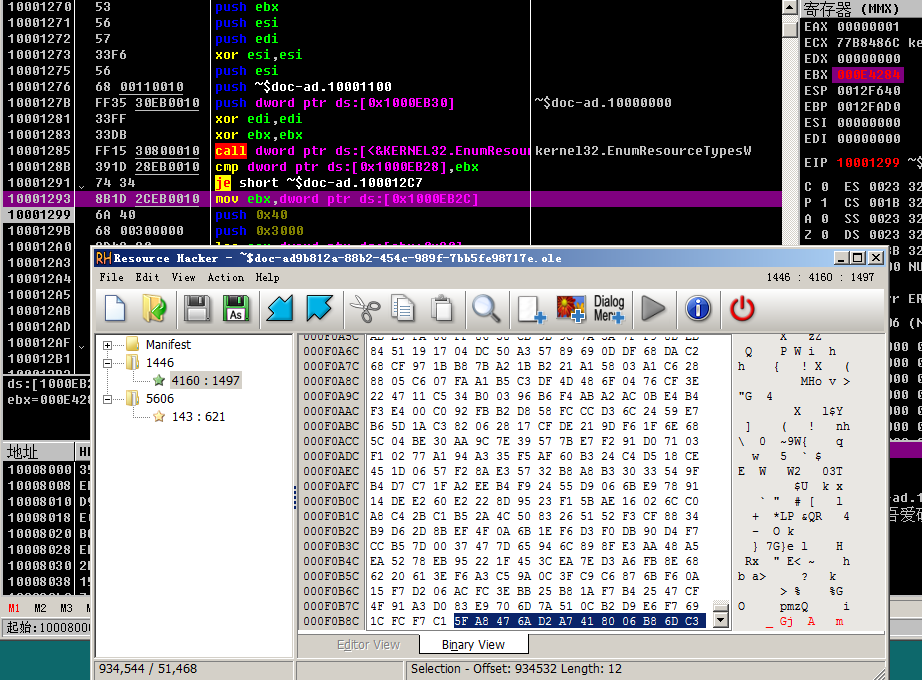


开始的逻辑为检测是否存在 N92KG7KSpA21lGd2OPZA7QwZv 这个环境变量，存在的话说明病毒并非第一次执行，则跳转到后续恶意逻辑，因此在病毒SetEnvironmentVariableW之后，重新在DllUnregisterServer\_0入口点设置 eip 重新运行过来，则实现跳转到 10001270，该函数也是主要功能入口。

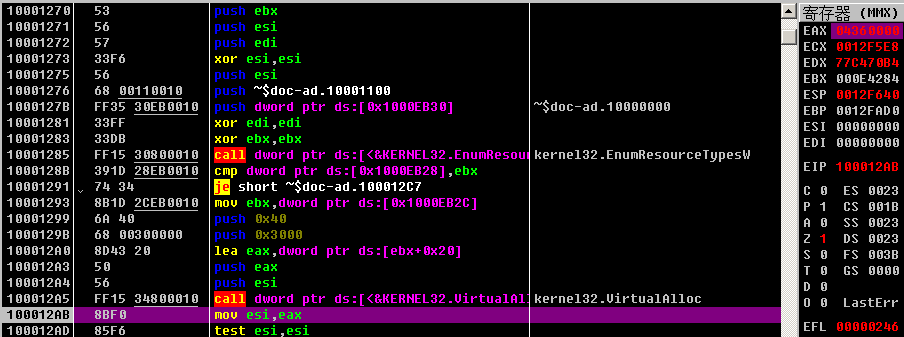




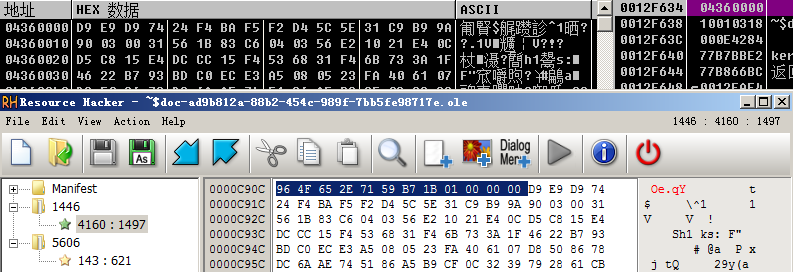
读取自身资源：



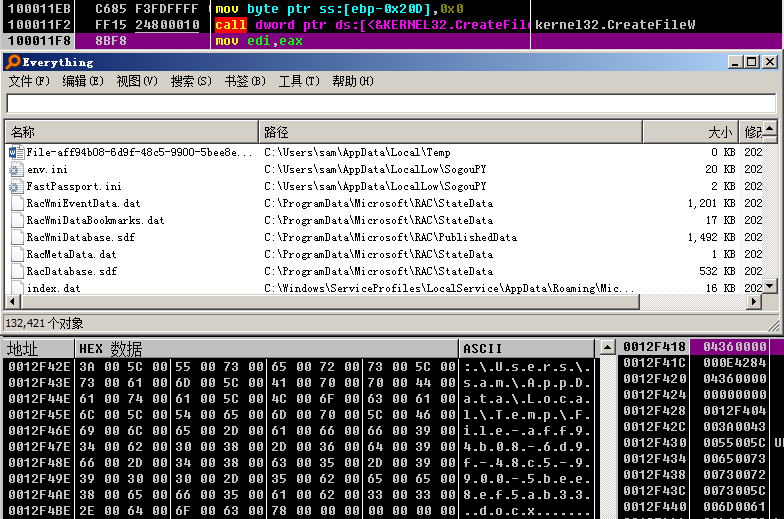
申请一块内存



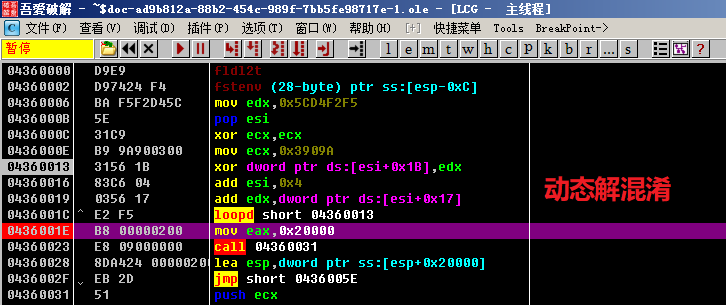
将资源4160:1497偏移12字节开始的所有内容写入申请的内存，实际上是Shellcode：



创建伪装文档迷惑用户



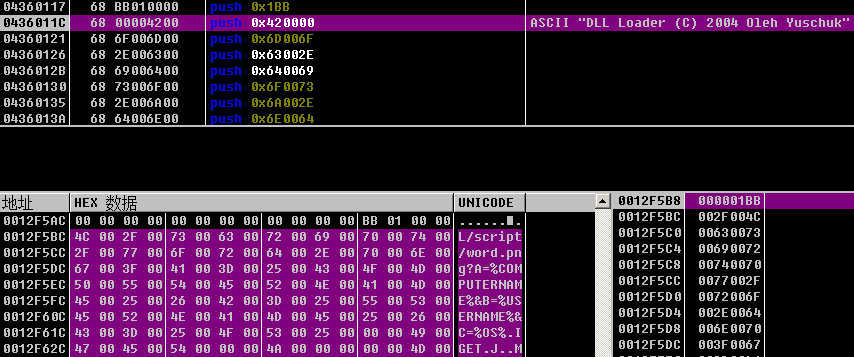
伪装文档打开之后，可以对shellcode内存地址下访问断点，运行过来，shellcode本身存在混淆，通过loop 循环解密，目的是防静态特征码查杀，由于shellcode较长，解密过程时间较长，解密完毕将shellcode保存一下导入IDA分析。



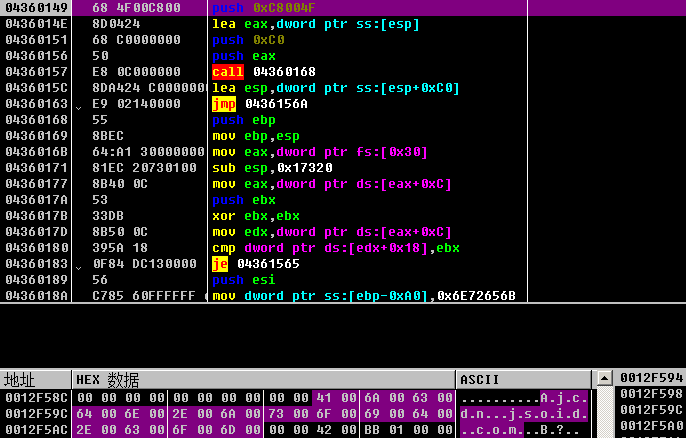
### Shellcode分析

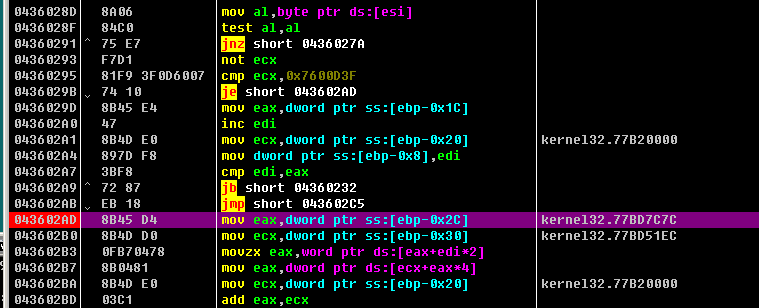
开始是一段解密字符串的逻辑，解出两个字符串：

字符串1：L“/script/word.png?A=%COMPUTERNAME%&B=%USERNAME%&C=%OS%.IGET.J..M“



字符串2：解密网址：A”jcdn.jsoid.com”

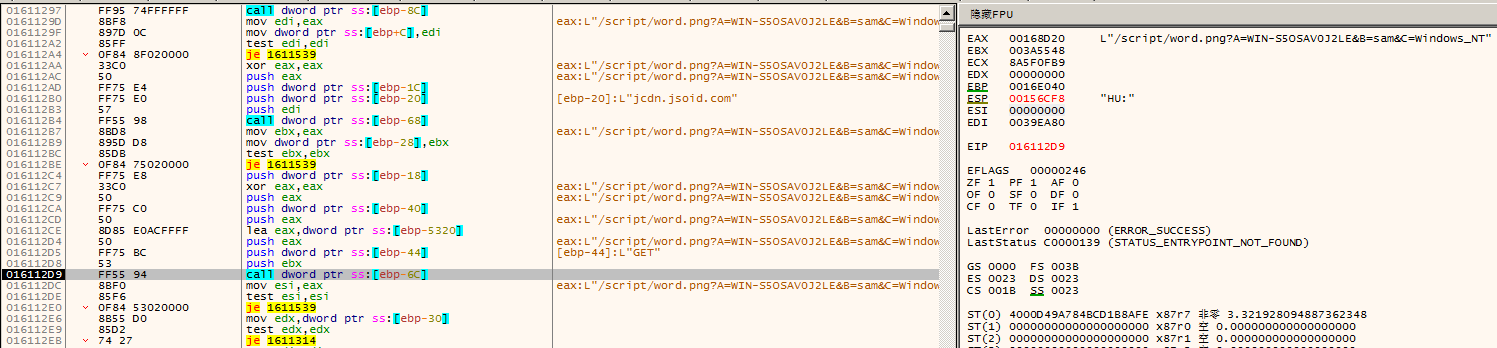


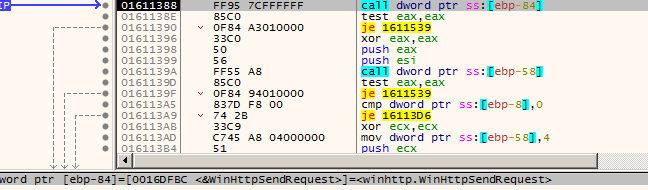
此处存在一个较长的循环，在遍历API的地址，可直接运行到0x43602AD 处。用OD调试时需耐心等待计算完成。

由于OD调试花费时间较多，因此换成了x32dbg进行调试，向下跟踪发现在寻找http请求相关的API比如：winHttpOpen、winHttpConnect、winHttpOpenRequest等。

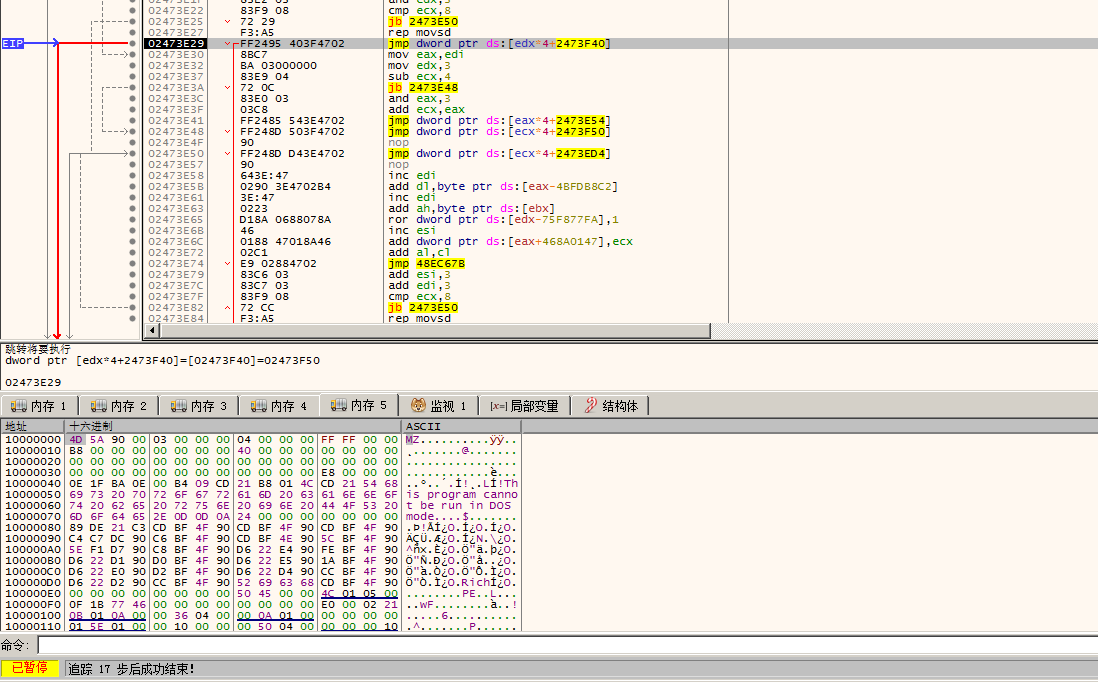


进一步获取：主机名-用户名-系统这样一个URL参数，最终拼接”jcdn.jsoid.com”进行GET请求，向远程发送受害人主机信息。

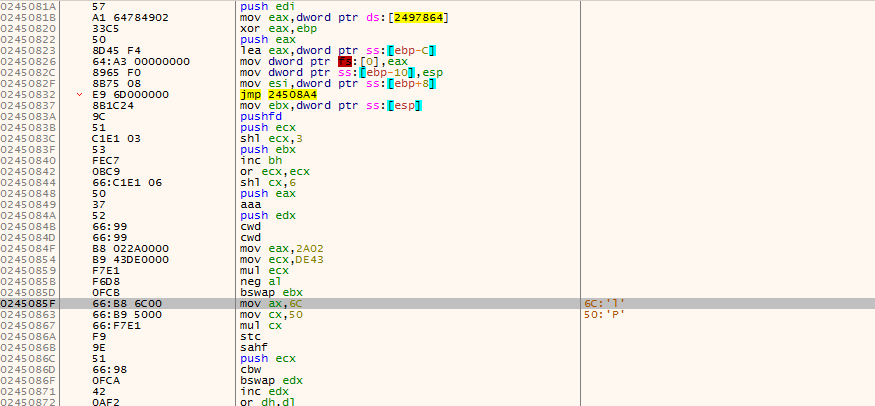




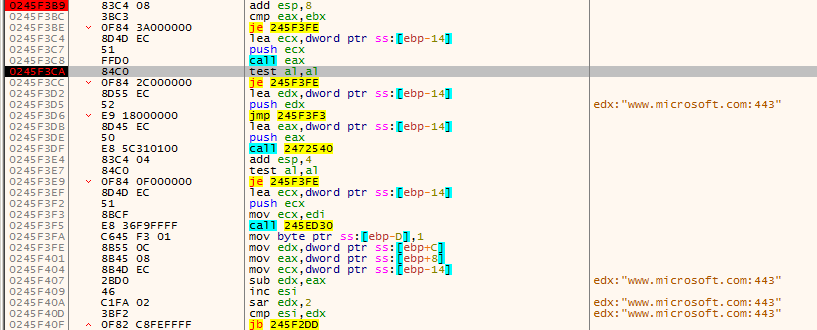
继续向下通过VirtualAlloc申请内存后解密出一个PE文件，每次写1000h字节。

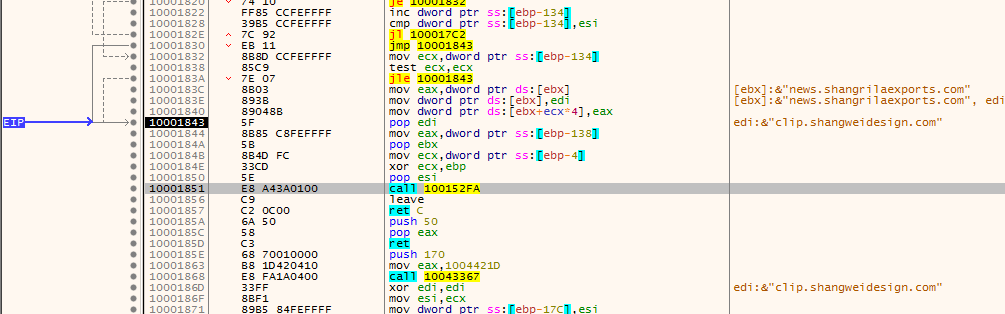


根据特征码可知该PE文件为Denis木马。



后续逻辑会先测试网络连通性，继而不断访问另外两个URL。





调试时可能遇到的坑是最后call较多，刚开始步过的时候容易跑到sleep中。。

## 参考

https://zhuanlan.zhihu.com/p/222086692