如何分析感染式病毒

安天安全研究与应急处理中心

www.antiy.com

智者安天下

内容简介

- 病毒的定义
- 感染的对象与模式
- PE文件的感染与分析
- 感染案例
- 展望

病毒的定义

智者安天下

病毒的定义

计算机病毒（Computer Virus）是编制者在计算机程序中插入的破坏计算机功能或者数据的代码，能影响计算机使用，**能自我复制的一组计算机指令或者程序代码**。

病毒是在计算机程序中插入的能自我复制的代码。

病毒是代码！

智者安天下

木马的简化

计算机病毒（Computer Virus）是编制者在计算机程序中插入的**破坏计算机功能或者数据的代码**，能影响计算机使用，能自我复制的一组计算机指令或者程序代码。

木马是获取计算机功能或者数据的代码。

木马是代码！

智者安天下

病毒和木马的关系

- 病毒是一种传播技术
- 木马是目的实现
- 病毒和木马并不冲突，可以相互融合。
  - 木马是病毒的payload。
  - 近年的病毒全部包含木马功能。
  - 臃肿的木马影响了病毒的灵活性。

感染的对象与模式

智者安天下

病毒存在可能性

- 有代码的地方
  - 计算机，手机，路由器，…
- 有插入的机会
  - 发现感染对象
  - 感染对象可写入
  - 对象中的代码能被执行
- 病毒代码在哪里？
  - 二进制代码通过call $获取自己的当前地址
  - 脚本通过访问特殊变量或自我嵌入方法找到

感染的对象

- 引导区、主引导区
- 内存
- 文件
  - 可执行文件
  - 可执行脚本
- 可擦写ROM

智者安天下

感染的对象-可执行文件

- Boot扇区
- DOS COM/EXE
- NE
- PE/PE64
- PE-ARM（WinCE）
- ELF
- …

感染的对象-可执行脚本

- Office（WORD、Excel、PPT…）
- AutoCAD
- Shell
- 编程语言代码（含解释语言和编译语言）

感染的要点

- 获得运行权
  - 优先运行权
    - 可执行文件入口点
    - 自动运行宏
  - 隐秘运行权
    - 深层代码替换
    - 溢出
- 不影响宿主运行，病毒运行后执行权交回宿主。

感染的模式

- 插入
  - 脚本类
- 替换
  - 可执行文件入口点
  - 可执行文件指令序列
  - 内存代码关键点
  - 引导区

可执行文件的感染

- DOS COM/EXE/SYS　　曾经的红火已经不在
- NE　　短暂的火花
- PE/PE64　　从CIH到今天
- PE-ARM（WinCE）　　概念
- ELF　　少、神秘
- Mach-O　　OSX
- …

PE文件的感染

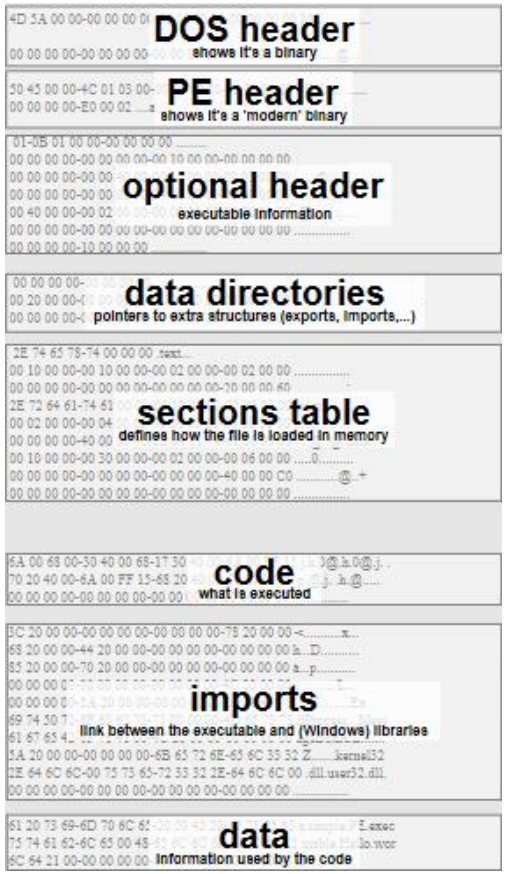
智者安天下

PE的文件结构

- 映像结构

DOS Header
PE Header
.text（代码节）
.rdata（资源节）
.data（数据节）
.idata（输入表）

详细



上图来源：http://pe101.corkami.com

智者安天下

PE文件的感染-病毒代码插入

- 文件尾部
  - 新增加一个节项
  - 修改最后一个节项
- 文件中间
  - 节间隙（CIH）
- 文件替换
  - 保留图标
  - 原始文件成为数据
- 覆盖代码
  - 破坏原功能，新手有BUG作品。

PE文件的感染-获取执行权模式

- 修改入口点
  - 入口点指向跳板代码
- 替换入口代码
- 修改引入表DLL项
- 修改预加载入口表
- 修改中间指令（隐秘）

智者安天下

PE文件的感染-主动免疫

- 病毒为了避免重复运行，会检测一些自定义的标志，通过主动设置这些病毒标志来实现免疫。
- 免疫的类型
  - 内存免疫（效果好，会触发某些专杀假警报）
    - 互斥量　　ChineseHacker-2
  - 注册表免疫
  - 文件免疫，不推荐
    - FunLove （文件长度&0xFF == 3）为已感染
    - 染毒文件杀毒后，如果感染标志未擦除，也会有免疫效果。

PE文件的感染-解毒

- 内存杀毒，切断感染源
  - 终止内存感染线程
  - 卸载内存HOOK代码
- 清除病毒代码，感染逆操作
  - 恢复入口点
  - 恢复被替换的代码
  - 删除增加节
  - 擦除病毒代码，修改节大小
  - *修复引入表（DLL插入）*

PE文件的感染-逃避查杀的办法

- 变形
  - 垃圾代码
  - 等效指令变换
  - 虚拟机
- 攻击杀毒软件
  - 杀死杀毒软件进程
  - 屏蔽杀毒软件升级
- 屏蔽OS安全机制
  - 禁用SFC
  - 禁用UAC

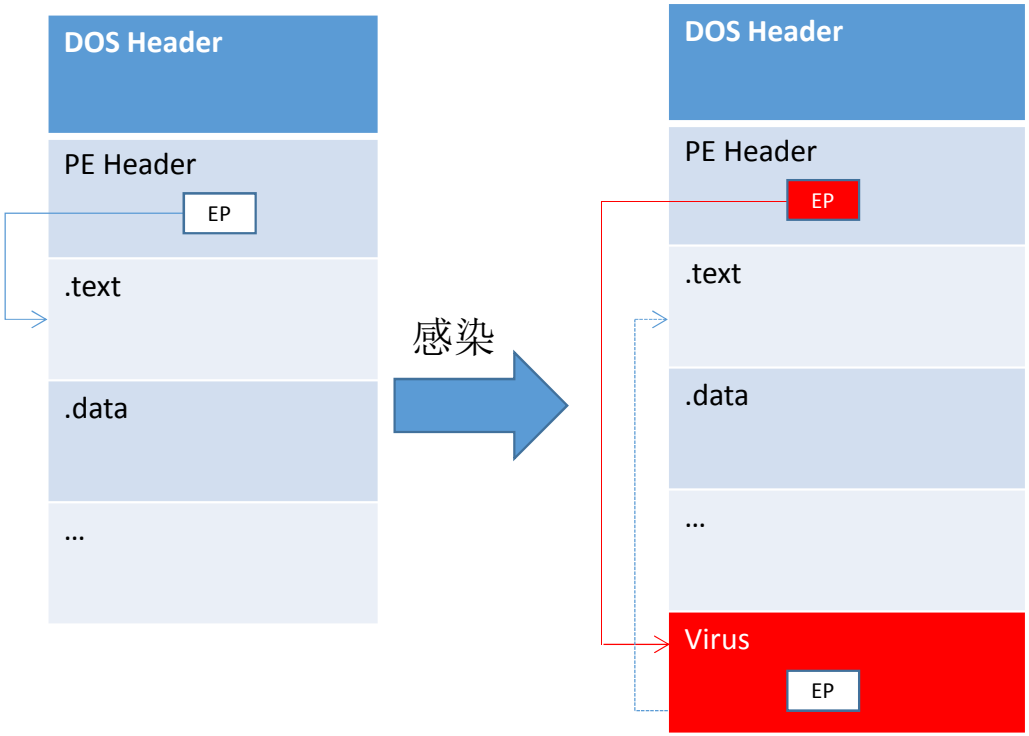
PE文件的感染-逃避查杀的办法

- 隐秘感染
  - 替换CALL指令Win32.Virut,Win32.Sality…
  - 替换子程序　　Win32.XPAJ
- Rootkit/Bootkit

智者安天下

PE文件的感染-图例1

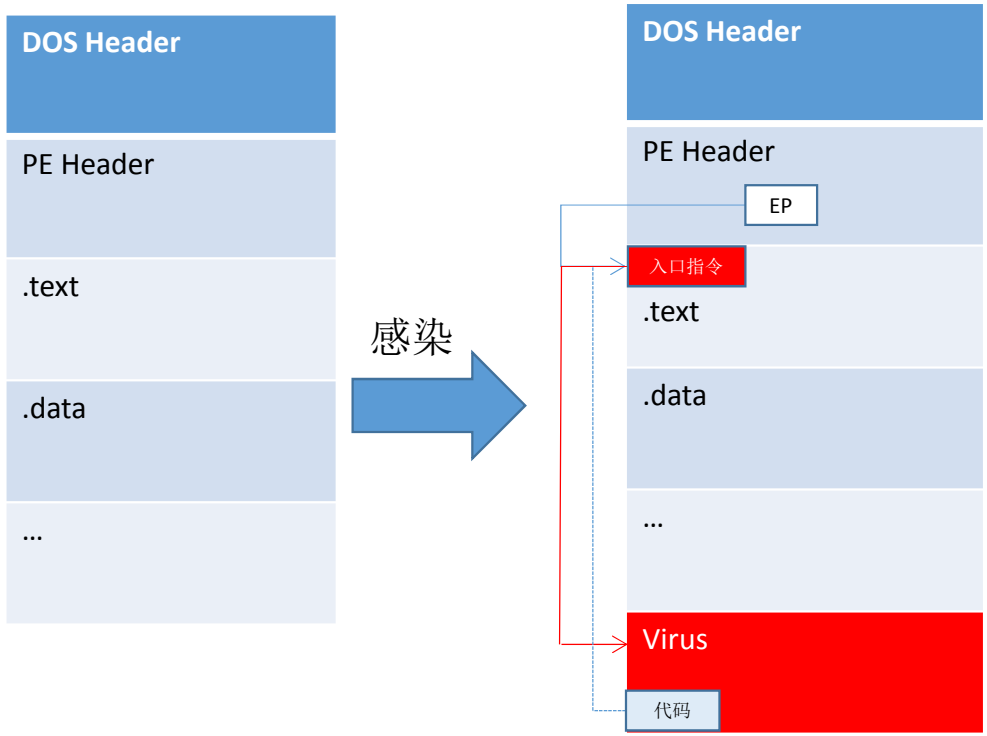
- 修改入口点，添加新节



智者安天下

PE文件的感染-图例2

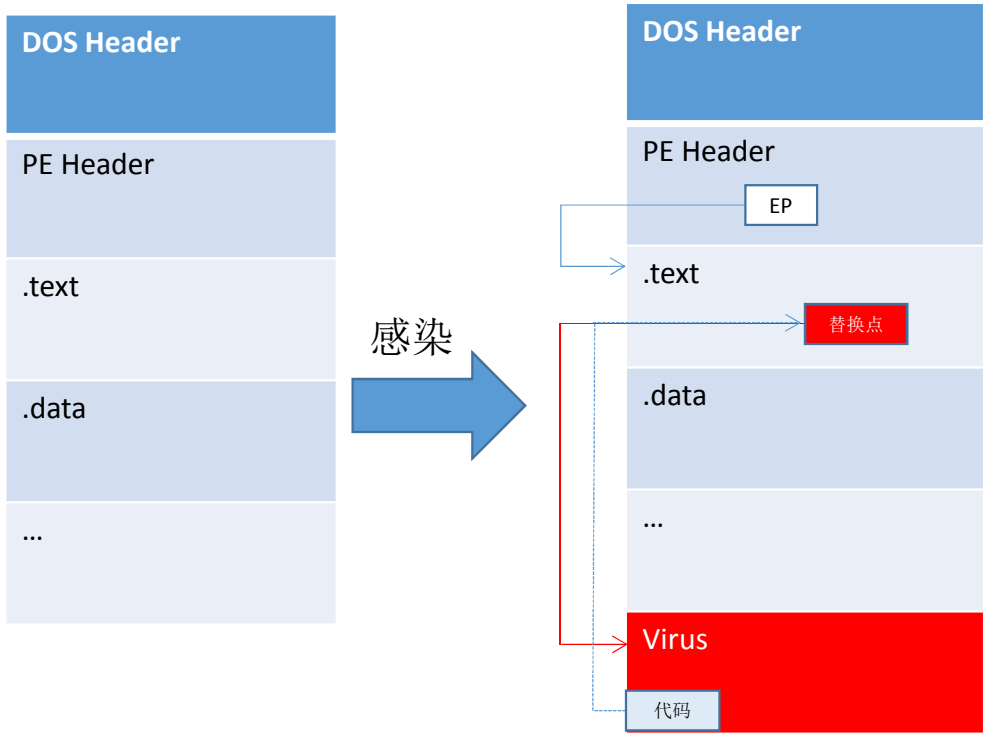
- 替换入口指令，添加新节



智者安天下

PE文件的感染-图例3

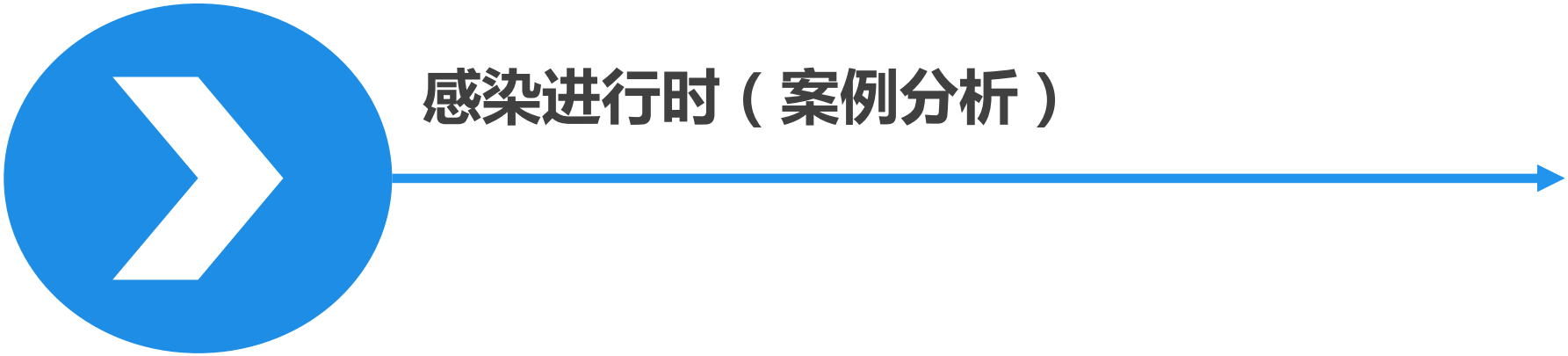
- 替换可能被执行的指令，添加新节



智者安天下

PE文件分析要点

- 保证自身安全，虚拟机或物理隔离。
- 找到关键点
  - 原始入口点
  - 被替换的代码
- 输出其他有价值的信息
  - 免疫相关
  - 网站相关



感染进行时（案例分析）

智者安天下

2014下半年感染情况（VT样本统计）

- 感染一直在持续。
- Parite老变形病毒依然在榜。生命周期超长！

Virus.Win32.Nimnul.a	35.07%
Virus.Win32.Virut.ce	13.86%
Virus.Win32.Sality.gen	7.86%
Virus.Win32.Parite.b	5.96%
Virus.Win32.Elkern.b	4.22%
Virus.Win32.Alman.b	2.96%
Virus.Win32.Crytex.1290	2.51%
Virus.Win32.Small.l	2.37%
Virus.Win32.Virut.n	2.29%
Virus.Win32.Otwycal.a	1.63%

智者安天下

调试技巧

- 虚拟机调试（本机调试跑飞了后果严重！）
  - 用虚拟机在最困惑的地方创建快照。反复调试，节省时间。
- 巧用硬件断点
  - 内存执行断点跳过多层解密干扰；
  - 内存写断点发现解密/还原代码；
- 找到病毒自我识别代码/异常出口，跳过感染模块
- 【清除】找到交回控制权的代码
- 【清除】找到还原代码的方式

- 感染模式见图例1
- 最早是非常简单的。
- 5条指令就看到了OEP

```
00409356 push    ebp
00409357 mov     ebp, esp
00409359 push    0FFFFFFFFh
0040935B push    offset unk_40A938
00409360 push    offset  except handler3
00409365 mov     eax, large fs:0
0040936B push    eax
0040936C mov     large fs:0, esp
00409373 sub     esp, 68h
00409376 push    ebx
00409377 push    esi
00409378 push    edi
00409379 mov     [ebp-18h], esp
0040937C xor     ebx, ebx
0040937E mov     [ebp-4], ebx
00409381 push    2
00409383 call    ds:__set_app_type
00409389 pop     ecx
0040938A or      dword_40DCA8, 0FFFFFFFFh
00409391 or      dword_40DCAC, 0FFFFFFFFh
00409398 call    ds:__p__fmode
0040939E mov     ecx, dword 40DC9C
004093A4 mov     [eax], ecx
004093A6 call    ds:__p__commode
```

```
00415000 public start
00415000 start:
00415000 call    $+5
00415005 push    ebp
00415006 mov     ebx, [esp+8]
0041500A mov     ebp, [esp+4]
0041500E sub     dword ptr [esp+4], 0BCAFh        ; 差值，OEP出现
00415016 and     ebx, 0FFFFF000h
0041501C sub     ebp, offset nullsub_1
00415022
00415022 loc_415022:                              ; CODE XREF: .rs
00415022 cmp     dword ptr [ebx+4Eh], 'sihT'
00415029 jnz     short loc_415037
0041502B mov     eax, [ebx+3Ch]
```

415005h - 0BCAFh = 409356h 原始入口地址

```
0018FF88 dd 409356h                               ; 压栈数据-差值=OEP
```

智者安天下

第一个出口
正确返回！

- 下面紧接着就是找kernel32.dll的PE头。
- 然后找GetProcAddress入口，失败则转OEP。
- 宿主代码无修改，EP即为截断地址，无新增节。
- 特征码稳定，有大面积的稳定代码。

```
00415022 cmp     dword ptr [ebx+4Eh], 'sihT'     ; 找PE头
00415029 jnz     short loc_415037
0041502B mov     eax, [ebx+3Ch]
0041502E add     eax, ebx
00415030 cmp     word ptr [eax], 4550h
00415035 jz      short loc_41503F
00415037
00415037 loc_415037:                             ; CODE XREF: .rsr
00415037 sub     ebx, 100h
0041503D jmp     short loc_415022                ; 找PE头
0041503F ; ---------------------------------------------------------
0041503F
0041503F loc_41503F:                             ; CODE XREF: .rsr
0041503F mov     edx, [eax+78h]
00415042 add     edx, ebx
00415044 mov     esi, [edx+20h]
00415047 mov     ecx, [edx+18h]
0041504A add     esi, ebx
0041504C push    ecx
0041504D
0041504D loc_41504D:                             ; CODE XREF: .rsr
0041504D lodsd
0041504E add     eax, ebx
00415050 cmp     dword ptr [eax-1], 74654700h    ; GetProcAddress
00415057 jnz     short loc_415074
00415059 cmp     dword ptr [eax+3], 636F7250h
00415060 jnz     short loc_415074
00415062 cmp     dword ptr [eax+7], 72646441h
00415069 jnz     short loc_415074
0041506B cmp     dword ptr [eax+0Bh], 737365h
00415072 jz      short loc_415079                ; 成功找到GetProc
00415074
00415074 loc_415074:                             ; CODE XREF: .rsr
00415074                                         ; .rsrc:00415060↑
00415074 loop    loc_41504D
00415076 pop     ecx
00415077 pop     ebp
00415078 retn                                    ; 返回OEP
```

病毒感染模块

智者安天下

PE感染式-Virus.Win32.Virut.n

- 随机Stolen Code，见图例3
  - 宿主文件被随机修改了一个CALL，此CALL指向了文件尾的病毒代码。
- 带变形引擎
- 注意：IDA默认不加载资源节。IDA加载文件的时候，一定要勾选Load resources。否则看不到病毒代码。
- 如果在病毒入口设断点，运行后断点不触发怎么办？
  - 强行将IP设定到CALL。

```
01007578    push    [ebp+arg_4]
0100757B    jnz     short loc_100759C
0100757D    push    8
0100757F    call    VirEntry
01007584    add     [eax-1], edx
01007587    adc     eax, offset HeapAlloc
0100758C    test    eax, eax
0100758E    mov     [esi], eax
01007590    jnz     short loc_10075B2
```

原始文件中的一段代码，CALL地址被替换!

尾部的病毒代码

```
01020600 VirEntry   proc near
01020600            pusha
01020601            push    ebp
01020602            mov     ebp, esp
01020604            call    sub_1020613
01020609            call    sub_102068C
0102060E            jmp     loc_102063C
0102060E VirEntry   endp
```

智者安天下

解密代码，执行代码

sub_1020613代码经过反虚拟机等动作后，到达这里，开始解码1。

```
.rsrc:01020648 loc_1020648:                            ; CODE XREF:
.rsrc:01020648 call    $+5
.rsrc:0102064D pop     edx
.rsrc:0102064E add     edx, 4Ch
.rsrc:01020654 sub     ecx, ecx
.rsrc:01020656 xor     ecx, 2394h
.rsrc:0102065C sub     ebx, ebx
.rsrc:0102065E or      ebx, 88h
.rsrc:01020664 push    edx
.rsrc:01020665
.rsrc:01020665 loc_1020665:                            ; CODE XREF:
.rsrc:01020665 xchg    al, ds:(byte_1020699 - 1020699h)[edx]
.rsrc:01020667 sub     ax, bx
.rsrc:0102066A mov     ds:(byte_1020699 - 1020699h)[edx], al
.rsrc:0102066C inc     edx
.rsrc:0102066D dec     ecx
.rsrc:0102066E cmp     ecx, 0
.rsrc:01020671 jnz     short loc_1020665
.rsrc:01020673 pop     edx
.rsrc:01020674 mov     esp, fs:0
.rsrc:0102067A pop     dword ptr fs:0
.rsrc:01020680 lea     ebp, [esp+0Ch+var_8]
.rsrc:01020684 leave
.rsrc:01020685 mov     [esp+8+arg_8], edx
.rsrc:01020689 popa
.rsrc:0102068A jmp     edx
.rsrc:0102068A sub_1020613 endp ; sp-analysis failed
.rsrc:0102068A             edx=.rsrc:byte_1020699
```

解密过程

密钥

解码后的病毒代码，jmp edx到下面。

```
.rsrc:01020699 nop
.rsrc:0102069A call    $+5
.rsrc:0102069F inc     ecx
.rsrc:010206A0 mov     eax, [esp]
.rsrc:010206A3 test    dword ptr [eax+2384h], 80000000h
.rsrc:010206AD mov     [eax+2900h], ebx                ; CODE XREF: sub_1020613+3
.rsrc:010206B3 mov     ebx, [esp+4]
.rsrc:010206B7 jz      short loc_10206E6
.rsrc:010206B9 cld
.rsrc:010206BA pop     ecx
.rsrc:010206BB mov     [eax+2904h], esi
.rsrc:010206C1 mov     [eax+2908h], edi
.rsrc:010206C7 cmp     byte ptr [eax+2388h], 0E8h      ; 替换的指令是FF15还是E8?
.rsrc:010206CE jnz     short loc_10206DD
.rsrc:010206D0 add     ebx, [eax+2389h]
.rsrc:010206D6 mov     ebx, [ebx+2]
.rsrc:010206D9 push    dword ptr [ebx]
.rsrc:010206DB jmp     short loc_10206E5
.rsrc:010206DD ; ---------------------------------------------------------------
.rsrc:010206DD
.rsrc:010206DD loc_10206DD:                            ; CODE XREF: .rsrc:010206D
.rsrc:010206DD mov     ebx, [eax+238Ah]
.rsrc:010206E3 push    dword ptr [ebx]
.rsrc:010206E5
.rsrc:010206E5 loc_10206E5:                            ; CODE XREF: .rsrc:010206D
.rsrc:010206E5 pop     ebx
.rsrc:010206E6
.rsrc:010206E6 loc_10206E6:                            ; CODE XREF: .rsrc:010206E
.rsrc:010206E6 push    ebp
.rsrc:010206E7 mov     ebp, eax
.rsrc:010206E9 sub     dword ptr [esp+4], 5            ; 计算原始地址
.rsrc:010206F1 and     ebx, 0FFFFF000h
.rsrc:010206F7 sub     ebp, 401006h
.rsrc:010206FD mov     edi, [esp+4]                    ; 原始地址
.rsrc:01020701 lea     esi, [ebp+403394h]              ; 被替换代码存放地址
.rsrc:01020707 mov     ecx, 0
.rsrc:0102070C rep movsb
.rsrc:0102070E
.rsrc:0102070E loc_102070E:                            ; CODE XREF: .rsrc:0102072
.rsrc:0102070E cmp     dword ptr [ebx+4Eh], 73696854h
.rsrc:01020715 jnz     short loc_1020724
```

JZ假出口

假恢复

继续

智者安天下

假出口，真出口？

- 假出口后面的代码好熟悉，.a变种里有的，右图。
- 不能像.a一样，“近路”反而是陷阱，对抗就是真真假假。
- 在获取GetProcAddress地址后，定位病毒使用WinAPI，然后检测病毒驻留标志，如下：

```
0102070E cmp     dword ptr [ebx+4Eh], 'sihT'
01020715 jnz     short loc_1020724
01020717 mov     eax, [ebx+3Ch]
0102071A lea     eax, [eax+ebx]
0102071D cmp     word ptr [eax], 4550h
01020722 jz      short loc_102072C
01020724
01020724 loc_1020724:
01020724 sub     ebx, 100h
0102072A jnz     short loc_102070E
0102072C
0102072C loc_102072C:
0102072C mov     edx, [eax+78h]
0102072F add     edx, ebx
01020731 mov     esi, [edx+20h]
01020734 mov     ecx, [edx+18h]
01020737 add     esi, ebx
01020739 push    ecx
0102073A
0102073A loc_102073A:
0102073A lodsd
0102073B add     eax, ebx
0102073D cmp     dword ptr [eax-1], 74654700h
01020744 jnz     short loc_1020758
01020746 cmp     dword ptr [eax+3], 636F7250h
0102074D jnz     short loc_1020758
0102074F cmp     dword ptr [eax+7], 72646441h
01020756 jz      short loc_102075D
01020758
01020758 loc_1020758:
01020758
01020758 loop    loc_102073A
0102075A pop     ecx
0102075B pop     ebp
0102075C retn
```

```
010207C4 ; 创建VT_3的Event，不成功则表示病毒已经驻留。
010207C4 call    sub_1020839
010207C9 test    eax, eax
010207CB jz      short RET_10207EE          近道
010207CD push    eax
010207CE call    dword ptr [ebp+40349Ch]
010207D4 test    eax, eax
010207D6 jnz     short loc_10207E8
010207D8 lea     eax, [ebp+4011CBh]     ; 病毒功能模块解码
010207DE
010207DE loc_10207DE:
010207DE mov     dl, [eax-1]
010207E1
010207E1 loc_10207E1:
010207E1 call    sub_1020854
010207E6 jmp     short near ptr VirMain_1020864
010207E8 ; ---------------------------------------------------------
010207E8
010207E8 loc_10207E8:                   ; CODE XREF: sub_102
010207E8 call    dword_403494[ebp]
010207EE
010207EE RET_10207EE:                   ; CODE XREF: sub_102
010207EE test    dword ptr [ebp+40338Ah], 80000000h
010207F8 jz      short loc_1020818
010207FA
010207FA loc_10207FA:                   ; 被替换代码保存位置
010207FA lea     esi, [ebp+40338Eh]
01020800
01020800 loc_1020800:
01020800 mov     edi, [esp+8+var_4]
01020804 movsb                          ; 恢复被替换指令
01020805 movsd
01020806 mov     ebx, [ebp+403906h]
0102080C mov     esi, [ebp+40390Ah]
01020812 mov     edi, [ebp+40390Eh]
01020818
01020818 loc_1020818:                   ; CODE XREF: sub_102
01020818 pop     ebp
01020819 retn
```

回到原始位置，执行已经恢复的指令。

智者安天下

修复前后对比

解码前	解码后
.rsrc:01022A27 SaveCode db 87h .rsrc:01022A28 db 9Dh ; .rsrc:01022A29 db 0F8h ; .rsrc:01022A2A db 99h ; .rsrc:01022A2B db 88h ; .rsrc:01022A2C db 89h ;	.rsrc:01022A27 SaveCode db 0FFh .rsrc:01022A28 db 15h .rsrc:01022A29 db 70h ; p .rsrc:01022A2A db 11h .rsrc:01022A2B db 0 .rsrc:01022A2C db 1
修复前	修复后
.text:0100757B jnz short loc_100759C .text:0100757D push 8 .text:0100757F call sub_1020600 .text:01007584 add [eax-1], edx .text:01007587 adc eax, offset HeapAlloc	.text:0100757B jnz short loc_100759C .text:0100757D push 8 .text:0100757F call ds:GetProcessHeap .text:01007584 add [eax-1], edx .text:01007587 adc eax, offset HeapAlloc .text:0100758C test eax, eax

- 在被替换的CALL指令上直接设置硬件写断点，可以中断在恢复代码位置，有助于快速定位（请勿实机联网操作！！！）。
- 此版本没有变形，除了StolenCode有点麻烦外，其他还算容易。

智者安天下

PE感染式- Win32.XPAJ

- 最早出现在2009年，目前4个变种。
- 100%采取Stolen Code感染模式，替换宿主中某几个子程序。在宿主程序调用被替换的子程序的时候，病毒才会被激活。
- 入口采用数据流解密+简单解释器模式+寄存器跳转+分块存储+变形，无固定特征码。根据数据流的规律，可以穷举。
- 解毒过程复杂。

智者安天下

Stolen Code

- 入口点不变
- 入口代码不变
- 文件长度明显变大
- 自带变形引擎
- 被替换的子程序至少3个，多可以更多。

```
00401306 start      proc near
00401306            push    60h
00401308            push    offset unk_407170
0040130D            call    VirEntry1_4022FC
00401312            mov     edi, 94h
00401317            mov     eax, edi
00401319            call    VirEntry5_402F00
0040131E            mov     [ebp-18h], esp
00401321            mov     esi, esp
00401323            mov     [esi], edi
00401325            push    esi             ; lpVersionInformation
00401326            call    ds:GetVersionExA
0040132C            mov     ecx, [esi+10h]
0040132F            mov     dword_4096FC, ecx
00401335            mov     eax, [esi+4]
00401338            mov     dword_409708, eax
0040133D            mov     edx, [esi+8]
00401340            mov     dword_40970C, edx
00401346            mov     esi, [esi+0Ch]
00401349            and     esi, 7FFFh
0040134F            mov     dword_409700, esi
00401355            cmp     ecx, 2
00401358            jz      short loc_401366
0040135A            or      esi, 8000h
00401360            mov     dword_409700, esi
```

CALL指令没有任何改变，只是子程序内容被替换。

智者安天下

被替换的子程序（1）

- VirEntry是病毒真正的入口

```
004022FC VirEntry1   proc near               ; CODE
004022FC                                     ; start
004022FC
004022FC var_48      = dword ptr -48h
004022FC var_40      = dword ptr -40h
004022FC var_38      = dword ptr -38h
004022FC var_2C      = dword ptr -2Ch
004022FC var_24      = dword ptr -24h
004022FC var_20      = dword ptr -20h
004022FC
004022FC             push    ebp
004022FD             mov     ebp, esp
004022FF             push    edx
00402300             call    VirEntry
00402305
00402305 loc_402305:                         ; CODE
00402305             mov     ecx, 32E09BD0h
0040230A             add     [ebp+var_38], ecx
0040230D             add     ecx, eax
0040230F             add     edi, [ebp+var_48]
00402312             xor     ebx, esi
00402314             mov     ebx, [edi]
00402316             add     ecx, [ebp+var_48]
00402319             xor     ebx, [ebp+var_38]
0040231C             xor     [ebp+var_24], 373E04h
00402323             mov     [ebp+var_20], ebx
00402326             and     ecx, [ebp+var_2C]
00402329             sbb     [ebp+var_40], ecx
0040232C             mov     ecx, 87A06676h
00402331             jmp     loc_402F09
00402331 VirEntry1   endp
```

原始子程序被病毒替换

```
004022FC        push    offset __except_handler3
00402301        mov     eax, large fs:0
00402307        push    eax
00402308        mov     eax, [esp+8+arg_4]
0040230C        mov     [esp+8+arg_4], ebp
00402310        lea     ebp, [esp+8+arg_4]
00402314        sub     esp, eax
00402316        push    ebx
00402317        push    esi
00402318        push    edi
00402319        mov     eax, [ebp-8]
0040231C        mov     [ebp-18h], esp
0040231F        push    eax
00402320        mov     eax, [ebp-4]
00402323        mov     dword ptr [ebp-4], 0FFFFFFFFh
0040232A        mov     [ebp-8], eax
0040232D        lea     eax, [ebp-10h]
00402330        mov     large fs:0, eax
00402336        retn
```

智者安天下

被替换的子程序（2）

- VirEntry是病毒真正的入口

```
004020D0 VirEntry2       proc near               ; CODE
004020D0                                         ; .text
004020D0
004020D0 var_54          = dword ptr -54h
004020D0 var_50          = dword ptr -50h
004020D0 var_4C          = dword ptr -4Ch
004020D0 var_48          = dword ptr -48h
004020D0 var_44          = dword ptr -44h
004020D0 var_40          = dword ptr -40h
004020D0 var_38          = dword ptr -38h
004020D0 var_34          = dword ptr -34h
004020D0 var_2C          = dword ptr -2Ch
004020D0 var_24          = dword ptr -24h
004020D0 var_1C          = dword ptr -1Ch
004020D0 var_14          = dword ptr -14h
004020D0 var_C           = dword ptr -0Ch
004020D0
004020D0                 push    ebp
004020D1                 mov     ebp, esp
004020D3                 push    ebx
004020D4                 call    VirEntry
004020D9
004020D9 loc_4020D9:                             ; CODE
004020D9                 or      ecx, 0C4C55Ah
004020DF                 mov     [ebp+var_4C], esp
004020E2                 or      [ebp+var_24], 2114EEBh
004020E9                 push    [ebp+var_4C]
004020EC                 push    [ebp+var_1C]
```

正常代码

```
004020D0 _strlen         proc near               ;
004020D0                                         ;
004020D0
004020D0 arg_0           = dword ptr  4
004020D0
004020D0                 mov     ecx, [esp+arg_0]
004020D4                 test    ecx, 3
004020DA                 jz      short main_loop_0
004020DC
004020DC str_misaligned_0:                       ;
004020DC                 mov     al, [ecx]
004020DE                 add     ecx, 1
004020E1                 test    al, al
004020E3                 jz      short loc_402133
004020E5                 test    ecx, 3
004020EB                 jnz     short str_misalign
004020ED                 add     eax, 0
004020F2                 lea     esp, [esp+0]
004020F9                 lea     esp, [esp+0]
00402100
00402100 main_loop_0:                            ;
00402100                                         ;
00402100                 mov     eax, [ecx]
00402102                 mov     edx, 7EFEFEFFh
00402107                 add     edx, eax
00402109                 xor     eax, 0FFFFFFFFh
```

智者安天下

被替换的子程序（3）

- VirEntry是病毒真正的入口

```
00402348 VirEntry3     proc near
00402348
00402348
00402348 var_54        = dword ptr -54h
00402348 var_24        = dword ptr -24h
00402348 var_10        = dword ptr -10h
00402348
00402348               push    ebp
00402349               mov     ebp, esp
0040234B               push    ecx
0040234C               call    VirEntry
00402351
00402351 loc_402351:
00402351               mov     [ebp+var_54], edi
00402354               mov     [ebp+var_24], 195D77Dh
0040235B               mov     edi, ebp
0040235D               mov     ebx, [ebp+var_10]
00402360               jmp     dword ptr [edi-2Ch]
00402360 VirEntry3     endp
```

正常代码

```
00402348               push    offset ModuleName
0040234D               call    ds:GetModuleHandleA
00402353               test    eax, eax
00402355               jz      short loc_40236D
00402357               push    offset ProcName
0040235C               push    eax
0040235D               call    ds:GetProcAddress
00402363               test    eax, eax
00402365               jz      short loc_40236D
00402367               push    [esp+uExitCode]
0040236B               call    eax
0040236D
0040236D loc_40236D:
0040236D
0040236D               push    [esp+uExitCode]
00402371               call    ds:ExitProcess
```

智者安天下

被替换的子程序（4）-主入口

病毒入口，首先获得自身的地址+偏移量，得到数据流的地址。
跳回VirEntry2中的代码，开始解释并处理数据流。

```
0040126E VirEntry       proc near               ; CODE XREF: sub_401000+3B↑p
0040126E                                        ; start+106↓p ...
0040126E
0040126E var_54         = dword ptr -54h
0040126E var_30         = dword ptr -30h
0040126E var_24         = dword ptr -24h
0040126E var_1C         = dword ptr -1Ch
0040126E var_10         = dword ptr -10h
0040126E var_8          = dword ptr -8
0040126E
0040126E                push    ebp
0040126F                mov     ebp, esp
00401271                sub     esp, 5Ch
00401274                push    ecx
00401275                push    ebx
00401276                push    edi
00401277                adc     ecx, [ebp+var_10]
0040127A                call    Get_EIP
0040127F                and     [ebp+var_30], eax
00401282                sbb     [ebp+var_8], 0ECB9FDh
00401289                mov     ebx, [ebp+var_1C]
0040128C                add     ebx, 30D53h
00401292                sub     [ebp+var_24], edi
00401295                mov     [ebp+var_54], ebx
00401298                mov     ebx, [ebp+var_24]
0040129B                jmp     loc_4020D9
0040129B VirEntry       endp
```

```
00402AA7 Get_EIP        proc near
00402AA7                mov     ecx, [ebp-6Ch]
00402AAA                mov     [ebp-1Ch], ecx
00402AAD                adc     [ebp-8], ebp
00402AB0                and     ebx, esp
00402AB2                retn
00402AB2 Get_EIP        endp
```

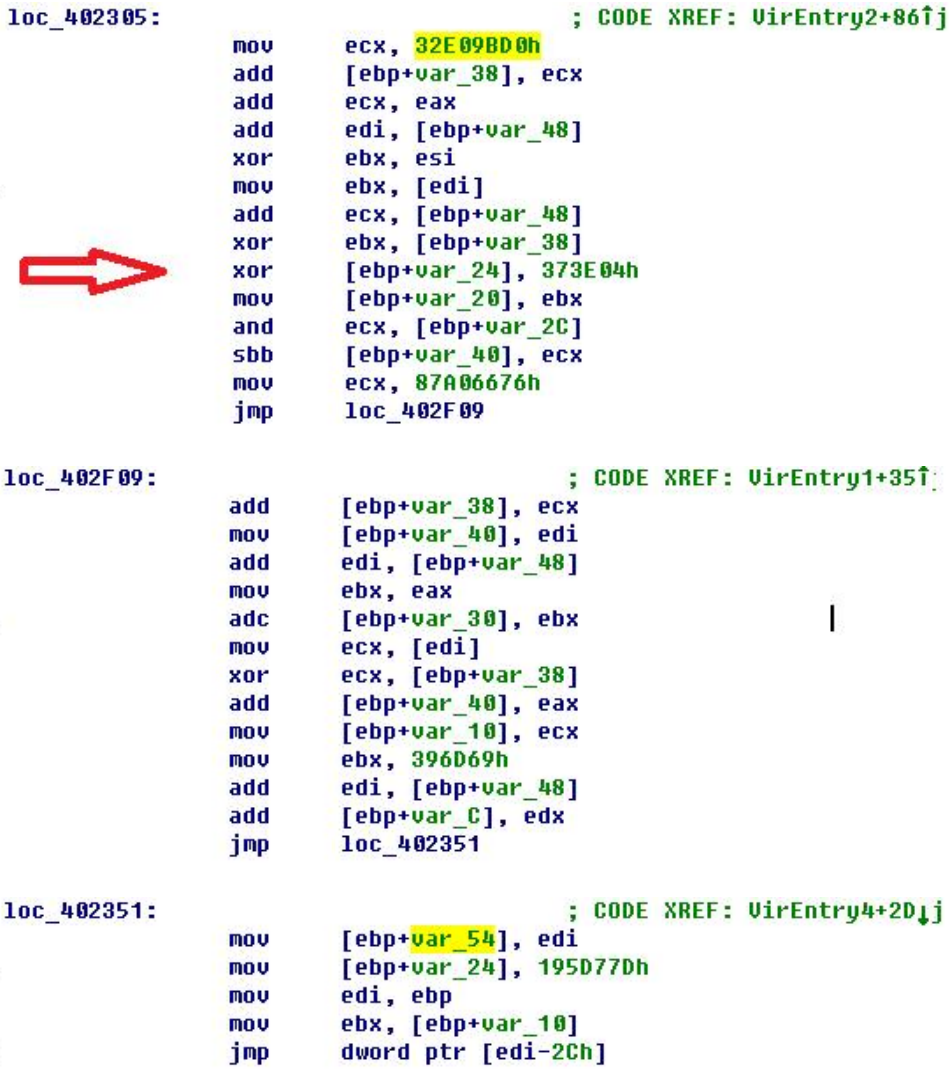
智者安天下

- 解码数据流，看似复杂，其实就是3个XOR。
- 变形引擎生成的代码，跳转采用堆栈，去向难以静态分析。

```
004020D9 loc_4020D9:                             ; CODE XREF: VirEntry+2D↑j
004020D9                or      ecx, 0C4C55Ah
004020DF                mov     [ebp+var_4C], esp
004020E2                or      [ebp+var_24], 2114EEBh
004020E9                push    [ebp+var_4C]
004020EC                push    [ebp+var_1C]
004020EF                sub     [ebp+var_14], edx
004020F2                and     ebx, [ebp+var_14]
004020F5                mov     ecx, [ebp+var_1C]
004020F8                add     ecx, 0E88h
004020FE                mov     [ebp+var_50], ecx
00402101                xor     ecx, [ebp+var_40]
00402104                adc     ecx, [ebp+var_44]
00402107
00402107 loc_402107:                             ; CODE XREF: .text:00402722↓j
00402107                mov     ebx, 4
0040210C                mov     [ebp+var_48], ebx
0040210F                sub     ebx, edi
00402111                mov     ecx, 75C7506Bh
00402116                adc     [ebp+var_34], 1F0BEE0h
0040211D                mov     [ebp+var_38], ecx
00402120                sub     edi, edx
00402122                or      ecx, edi
00402124                mov     edi, [ebp+var_54]
00402127                sub     edi, [ebp+var_48]
0040212A                add     ebx, ecx
0040212C                mov     ebx, [edi]
0040212E                adc     [ebp+var_34], 1AD195Ch
00402135                add     [ebp+var_38], ebx
00402138                mov     ecx, 54FF67h
0040213D                add     edi, [ebp+var_48]
00402140                add     [ebp+var_24], esi
00402143                mov     ecx, [edi]
00402145                mov     [ebp+var_24], edx
00402148                xor     ecx, [ebp+var_38]
0040214B                mov     [ebp+var_C], esi
0040214E                add     ecx, [ebp+var_1C]
00402151                mov     [ebp+var_2C], ecx
00402154                mov     ebx, edx
00402156                jmp     loc_402305
00402156 VirEntry2      endp
```

智者安天下

- 被替换的子程序最少有3个，多则不确定。
- 查、杀都异常困难。



```
00402305 loc_402305:                             ; CODE XREF: VirEntry2+86↑j
00402305                mov     ecx, 32E09BD0h
0040230A                add     [ebp+var_38], ecx
0040230D                add     ecx, eax
0040230F                add     edi, [ebp+var_48]
00402312                xor     ebx, esi
00402314                mov     ebx, [edi]
00402316                add     ecx, [ebp+var_48]
00402319                xor     ebx, [ebp+var_38]
0040231C                xor     [ebp+var_24], 373E04h
00402323                mov     [ebp+var_20], ebx
00402326                and     ecx, [ebp+var_2C]
00402329                sbb     [ebp+var_40], ecx
0040232C                mov     ecx, 87A06676h
00402331                jmp     loc_402F09

00402F09 loc_402F09:                             ; CODE XREF: VirEntry1+35↑j
00402F09                add     [ebp+var_38], ecx
00402F0C                mov     [ebp+var_40], edi
00402F0F                add     edi, [ebp+var_48]
00402F12                mov     ebx, eax
00402F14                adc     [ebp+var_30], ebx
00402F17                mov     ecx, [edi]
00402F19                xor     ecx, [ebp+var_38]
00402F1C                add     [ebp+var_40], eax
00402F1F                mov     [ebp+var_10], ecx
00402F22                mov     ebx, 396D69h
00402F27                add     edi, [ebp+var_48]
00402F2A                add     [ebp+var_C], edx
00402F2D                jmp     loc_402351

00402351 loc_402351:                             ; CODE XREF: VirEntry4+2D↓j
00402351                mov     [ebp+var_54], edi
00402354                mov     [ebp+var_24], 195D77Dh
0040235B                mov     edi, ebp
0040235D                mov     ebx, [ebp+var_10]
00402360                jmp     dword ptr [edi-2Ch]
```

智者安天下

数据流解密结果

- 数据流可以理解为虚拟机指令 。
- 共7条指令。每条指令带2个参数。
  1. 加
  2. 比较
  3. 写参数
  4. 写参数指针
  5. 写回解密数据
  6. 读FS:数
  7. 调用子程序。

地址	指令地址	参数1	参数2
1031fd2	1002b2e	fffffffc	fffffff8
1031fde	1002b14	a0c1	6e0fddd7
1031fea	10026e7	b912d8db	3eb6409b
1031ff6	1002b2e	fffffffc	fffffff4
1032002	1002b14	26c7c	fe54b8b1
103200e	10026e7	f7fb6f39	c595a60e
103201a	1002b14	c179eef2	e10590a6
1032026	1002b14	b49e02ba	366e58fd
1032032	1002b14	e2b03b51	f80aa7a4
103203e	1002b14	3488f69d	582c47ea
103204a	1002b14	9b20	4daf807d
1032056	1002b14	27a93	81fd133d
1032062	1002b2e	fffffffc	fffffff4
103206e	10026e7	908310b1	e503310b
103207a	1002b14	0	251f6d03
1032086	1002b14	0	eed56525
1032092	1002b14	0	a7a451bb
103209e	1002b14	0	9bfbe7c3
10320aa	1002b14	0	463de92f
10320b6	1002b14	0	c85f8c6b
10320c2	1002b14	0	50e9c63
10320ce	1002b14	0	f5a9777d
10320da	1002b14	0	d65435bd
10320e6	1002b14	fffffcc0	8e6b7cc0
10320f2	1002b2e	fffffffc	fffffff4
10320fe	10026e7	80f7ee50	fb3ad63e
103210a	1002b14	29000	5c437680
1032116	1002b2e	fffffff4	0
1032122	1002b2e	fffffffc	ffffffe4
103212e	1002713	24	1549673
103213a	1002b14	0	ff9ef846

智者安天下

7条指令

```
00402AB3 INS_CALL:
00402AB3                add     ebx, 95D3C1h
00402AB9                mov     [ebp-18h], eax
00402ABC                mov     [ebp-38h], ecx
00402ABF                sub     ebx, [ebp-2Ch]
00402AC2                mov     [ebp-48h], edx
00402AC5                and     ebx, 1E802ECh
00402ACB                mov     [ebp-24h], ebx
00402ACE                pop     edi
00402ACF                or      ecx, ecx
00402AD1                call    edi
00402AD3                push    eax
00402AD4                mov     eax, [ebp-18h]
00402AD7                mov     ecx, [ebp-38h]
00402ADA                or      dword ptr [ebp-40h], 1F9508F
00402AE1                mov     edx, [ebp-48h]
00402AE4                adc     edi, [ebp-3Ch]
00402AE7                mov     edi, 0FFFFFB0h
00402AEC                or      [ebp-0Ch], edx
00402AEF                add     edi, ebp
00402AF1                or      ecx, ebx
00402AF3                jmp     dword ptr [edi]
00402AF5 ; ---------------------------------------------------------------
00402AF5                add     ebx, [ebp-34h]
00402AF8                mov     [ebp-28h], esi
00402AFB
```

这些指令并不一定在同一个区域。
此样本分2个部分。

```
00402AFB INS_READ_FS:
00402AFB                mov     ecx, [ebp-3Ch]
00402AFE                pop     edi
00402AFF                mov     ebx, fs:[edi]
00402B02                add     ecx, [ebp-44h]
00402B05                push    ebx
00402B06                mov     ecx, [ebp-50h]
00402B09                sbb     [ebp-4], esi
00402B0C                jmp     ecx
00402B0E ; ---------------------------------------------------------------
00402B0E                adc     edi, 0A492D5h
00402B14
00402B14 INS_WRITE_DW2:
00402B14                add     [ebp-8], esi
00402B17                push    dword ptr [ebp-20h]
00402B1A                mov     ebx, 1B5510Dh
00402B1F                mov     ecx, [ebp-50h]
00402B22                or      [ebp-14h], esp
00402B25                jmp     ecx
00402B27 ; ---------------------------------------------------------------
00402B27                mov     dword ptr [ebp-4], 905F1Dh
00402B2E
00402B2E INS_WRITE_PTR:
00402B2E                mov     [ebp-8], ebx
00402B31                mov     ecx, [ebp-4Ch]
00402B34                add     ecx, [ebp-20h]
00402B37                mov     ecx, [ecx]
00402B39                jmp     loc_402698
```

智者安天下

7条指令


```
004026B6 INS_WRITE_DATA:
004026B6                 add     ecx, esi
004026B8                 mov     ebx, [ebp-4Ch]
004026BB                 add     ebx, [ebp-20h]
004026BE                 mov     [ebp-24h], edx
004026C1                 mov     ebx, [ebx]
004026C3                 add     ebx, [ebp-10h]
004026C6                 mov     edi, esp
004026C8                 pop     dword ptr [ebx]
004026CA                 mov     [ebp-24h], esi
004026CD                 adc     dword ptr [ebp-28h], 3C83A1h
004026D4                 mov     edi, 0FFFFFFB0h
004026D9                 xor     dword ptr [ebp-30h], 9D9CA0h
004026E0                 add     edi, ebp
004026E2                 sbb     [ebp-30h], ebx
004026E5                 jmp     dword ptr [edi]
004026E7 ; ---------------------------------------------------------
004026E7
004026E7 INS_ADD:
004026E7                 pop     ebx
004026E8                 sub     dword ptr [ebp-34h], 0A4BC83h
004026EF                 mov     ecx, 0CBD38Eh
004026F4                 pop     edi
004026F5                 add     ebx, edi
004026F7                 sub     [ebp-40h], edx
004026FA                 mov     edi, edx
004026FC                 push    ebx
004026FD                 mov     ebx, [ebp-3Ch]
00402700                 mov     ebx, ebp
00402702                 sub     dword ptr [ebp-4], 1DC027Fh
00402709                 jmp     dword ptr [ebx-50h]
0040270C ; ---------------------------------------------------------
0040270C                 adc     dword ptr [ebp-24h], 1020B4Eh
```

- 每个指令子程序返回地址由数据流决定。

```
00402713 INS_CMP:
00402713                 adc     [ebp-8], edi
00402716                 pop     ebx
00402717                 and     ecx, [ebp-44h]
0040271A                 pop     edi
0040271B                 adc     ecx, [ebp-18h]
0040271E                 cmp     edi, ebx
00402720                 mov     ecx, ebx
00402722                 jnz     loc_402107
00402728                 mov     ebx, ebx
0040272A                 adc     ebx, 1D64AD9h
00402730                 mov     edi, [ebp-20h]
00402733                 and     [ebp-40h], ebp
00402736                 add     [ebp-54h], edi
00402739                 mov     ecx, [ebp-50h]
0040273C                 and     [ebp-14h], ebp
0040273F                 jmp     ecx
```

智者安天下

7个指令中，CALL是最重要的，前2个CALL是病毒初始化过程。

（1）调用NtProtectVirtualMemory，修改病毒加密数据内存为可读可写可执行；

（2）解码病毒数据，代码见右图。Win32.XPAJ.a的解码最简单。后续版本逐渐复杂。

（3）常态过程，执行病毒其他功能，完成后执行被保存的原始代码，不再返回病毒，直接返回宿主程序。

XPAJ.A解密代码

```
.data:00432DD3 ; ---------------------------
.data:00432DD3 mov     eax, [esp+4]
.data:00432DD7 mov     edx, [esp+14h]
.data:00432DDB xchg    edx, [eax]
.data:00432DDD cmp     [eax], edx
.data:00432DDF jz      locret_432E02
.data:00432DE5 mov     edx, [esp+0Ch]
.data:00432DE9 mov     ecx, [esp+8]
.data:00432DED jmp     loc_432DF4
.data:00432DF2 ; ---------------------------
.data:00432DF2
.data:00432DF2 loc_432DF2:
.data:00432DF2 xor     [eax], edx
.data:00432DF4
.data:00432DF4 loc_432DF4:
.data:00432DF4 add     edx, [esp+10h]
.data:00432DF8 add     eax, 4
.data:00432DFB dec     ecx
.data:00432DFC jnz     loc_432DF2
.data:00432E02
.data:00432E02 locret_432E02:
.data:00432E02 retn    14h
```

智者安天下

第3次CALL


```
00432DD3 add     esp, 4
00432DD6 pop     esp
00432DD7 pop     edi
00432DD8 pop     ebx
00432DD9 pop     ecx
00432DDA leave
00432DDB jmp     loc_41B3B1
```

```
0041B3B1 loc_41B3B1:                ; CODE XREF: .
0041B3B1 call    sub_419082         ; 病毒初始化
0041B3B6 push    dword ptr [esp]
0041B3B9 call    loc_42609F         ; 执行宿主代码
0041B3BE retn
```

```
004286FA sub     edx, ebx
004286FC call    GetKey_40DDFE
00428701 jmp     loc_41F142
```

```
0040DDFE GetKey_40DDFE proc near
0040DDFE mov     eax, 1FD149Eh
0040DE03 retn
0040DE03 GetKey_40DDFE endp
```

执行宿主代码：查表

```
0040C60E mov     eax, [esi+0Ch]                 ; 查表
0040C611 xor     eax, edi
0040C613 cmp     edx, eax
0040C615 ja      loc_410A13
0040C61B ; START OF FUNCTION CHUNK FOR sub_41DE70
0040C61B
0040C61B loc_40C61B:                                    ; CODE
0040C61B mov     ecx, [esi+10h]
0040C61E xor     [esp-24h], edi
0040C622 xor     ecx, edi
0040C624 mov     edx, [esi+1Ch]
0040C627 xor     edx, edi
0040C629 mov     eax, [esi]
0040C62B xor     eax, edi
0040C62D and     edi, edi
0040C62F cmp     eax, 1
0040C632 jz      loc_40B863
0040C638 push    0FFFE6939h
0040C63D call    junk_419937
```

```
004302CF  9E 14 FD 01  00 15 FD 01  11 32 FD 01  9E 3C FD 01
004302DF  E0 15 FD 01  9E 14 FD 01  00 15 FD 01  BE 14 FD 01
004302EF  1D 5A 9A 0E  17 72 C3 0E  CB 15 72 15  CB 96 3F FE
004302FF  0F DE 9E CD  27 7E 0E DE  9E C8 A7 8F  EA 8F 13 AA
0043030F  DE 1D 5F 92  13 AA 2B 7E  0E DE 9E A5  50 EC 75 13
```

下一个表项 被替换的地址

00000000 0000019E 0000268F 000028F0

长度

0000017E 00000000 0000019E 00000020

智者安天下

解密出原始代码


```
004302EF add     esp, 4
004302F2 nop
004302F3 mov     esp, ebp
004302F5 pop     ebp
004302F6 nop
004302F7 push    ebp
004302F8 mov     ebp, esp
004302FA mov     edx, [ebp+8]
004302FD mov     eax, dword_409160
00430302 push    ebx
00430303 mov     ecx, offset unk_4090E0
00430308 push    esi
```

修改堆栈
修复EBP，ESP

```
0043045B push    dword ptr [ebp+0Ch]
0043045E call    ds:UnhandledExceptionFilter
00430464
00430464 loc_430464:
00430464
00430464 pop     esi
00430465 pop     ebx
00430466 pop     ebp
00430467 retn
```

直接返回宿主程序

```
004302EF  1D 5A 9A 0E 17 72 C3 0E  CB 15 72 15 CB 96 3F FE
004302FF  0F DE 9E CD 27 7E 0E DE  9E C8 A7 8F EA 8F 13 AA
0043030F  DE 1D 5F 92 13 AA 2B 7E  0E DE 9E A5 50 EC 75 13
0043031F  9A DE 13 9A 1B 7E 0E DE  9E A5 56 ED 9A A7 8F EA
0043032F  9C AD 57 1B 57 91 1A BF  9F 9E 9E 15 C7 96 1B 45
```

```
0040B80A loc_40B80A:
0040B80A xor     [esi+edx], al
0040B80D inc     edx
0040B80E dec     ecx
0040B80F jnz     loc_40B80A
```

```
004302EF  83 C4 04 90 89 EC 5D 90  55 8B EC 8B 55 08 A1 60
004302FF  91 40 00 53 B9 E0 90 40  00 56 39 11 74 11 8D 34
0043030F  40 83 C1 0C 8D 34 B5 E0  90 40 00 3B CE 72 EB 8D
0043031F  04 40 8D 04 85 E0 90 40  00 3B C8 73 04 39 11 74
0043032F  02 33 C9 85 C9 0F 84 21  01 00 00 8B 59 08 85 DB
```

- 代码相同，内存地址不同

智者安天下

Win32.XPAJ杀毒

- 找到病毒存放代码、地址对照表。
- 依次恢复每一块的代码，修正可能涉及的重定位项。
- 最后修复节表项，调整文件大小，杀毒完成。

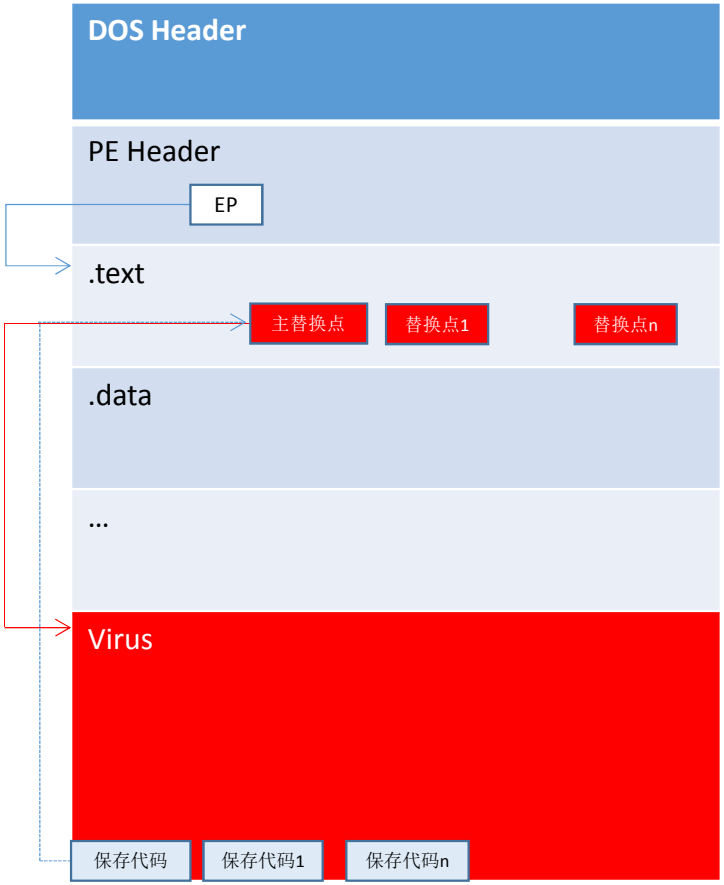
注意事项：

- 硬件写断点无效，病毒不还原被替换的代码。
- CALL指令不返回原处。

智者安天下

Win32.XPAJ感染示意图

特点：
1.代码碎片化，有点像CIH，但不同的是CIH占用的节间隙，而XPAJ是替换子程序代码。
2.无需修改节属性。
3.代码搬移不回写。



主替换点

PUSH EBP
MOV ESP,EBP
SUB ESP, xxx
虚拟机
解码开始

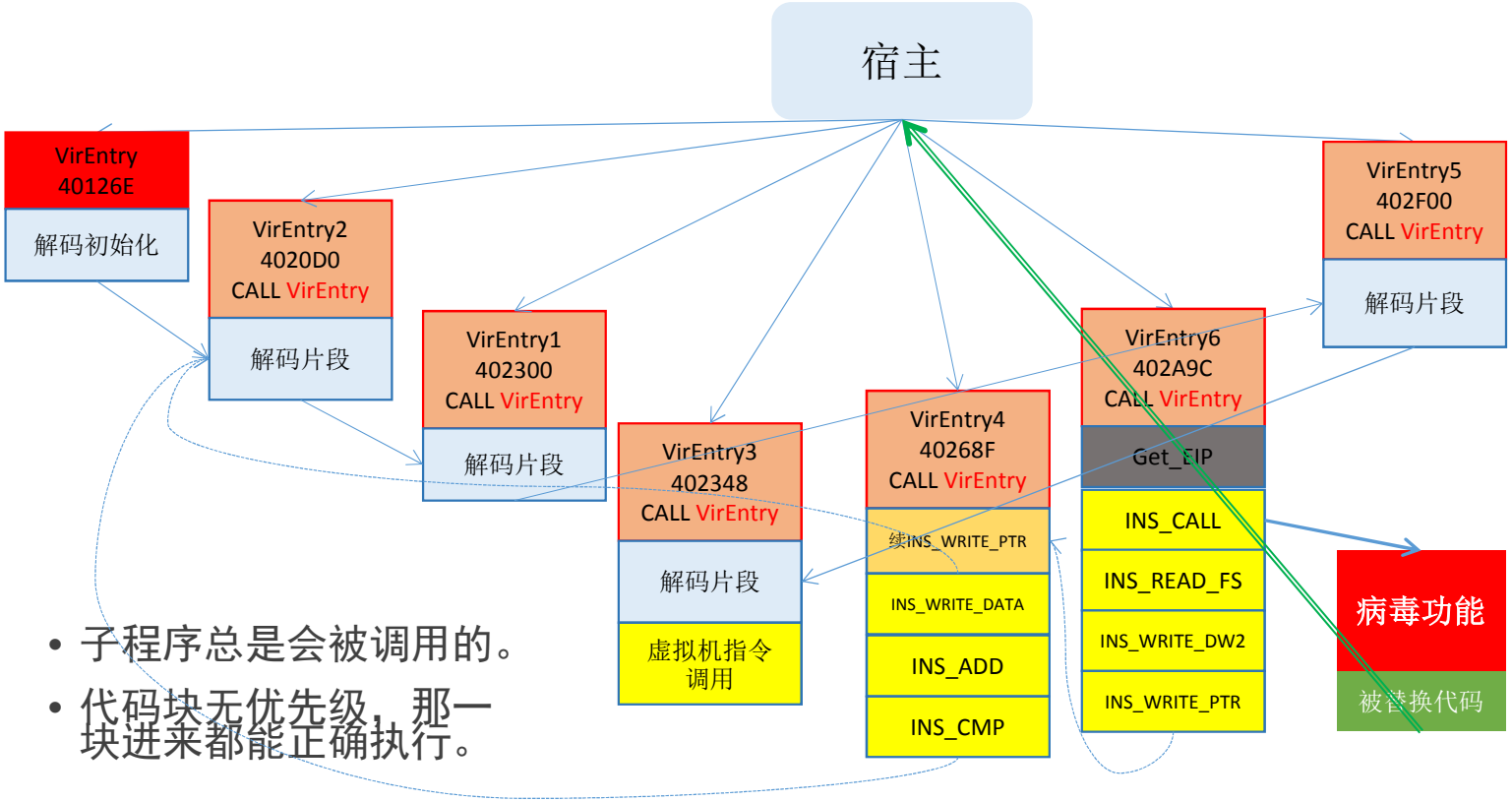
替换点

PUSH EBP
MOV ESP,EBP
CALL VirEntry

代码片段
空间

智者安天下

Win32.XPAJ.a 样本模块关系图



- 子程序总是会被调用的。
- 代码块无优先级，那一块进来都能正确执行。

智者安天下

智者安天下

展望-未来感染式病毒方向

- 技术门坎高，越来越小众，越来越不为人所知。
- 从独立的个性展示，到木马的附庸，走向幕后。
  - APT？
- 互联网走向物联网，方向更广，威胁在扩大。
- 矛盾关系永远存在。

感谢大家参与本次交流！