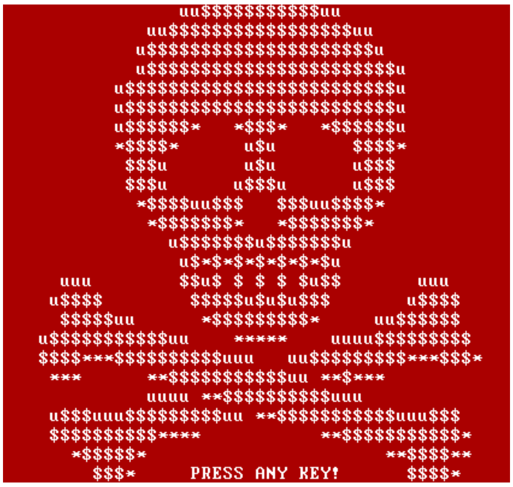
PetyaRansomware
一个具备技术挑战与想象力的勒索软件



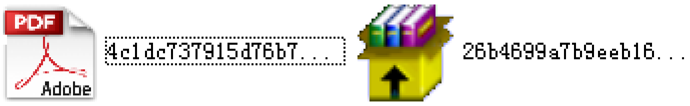
发布事件：2016 年 4 月 14 日

样本信息

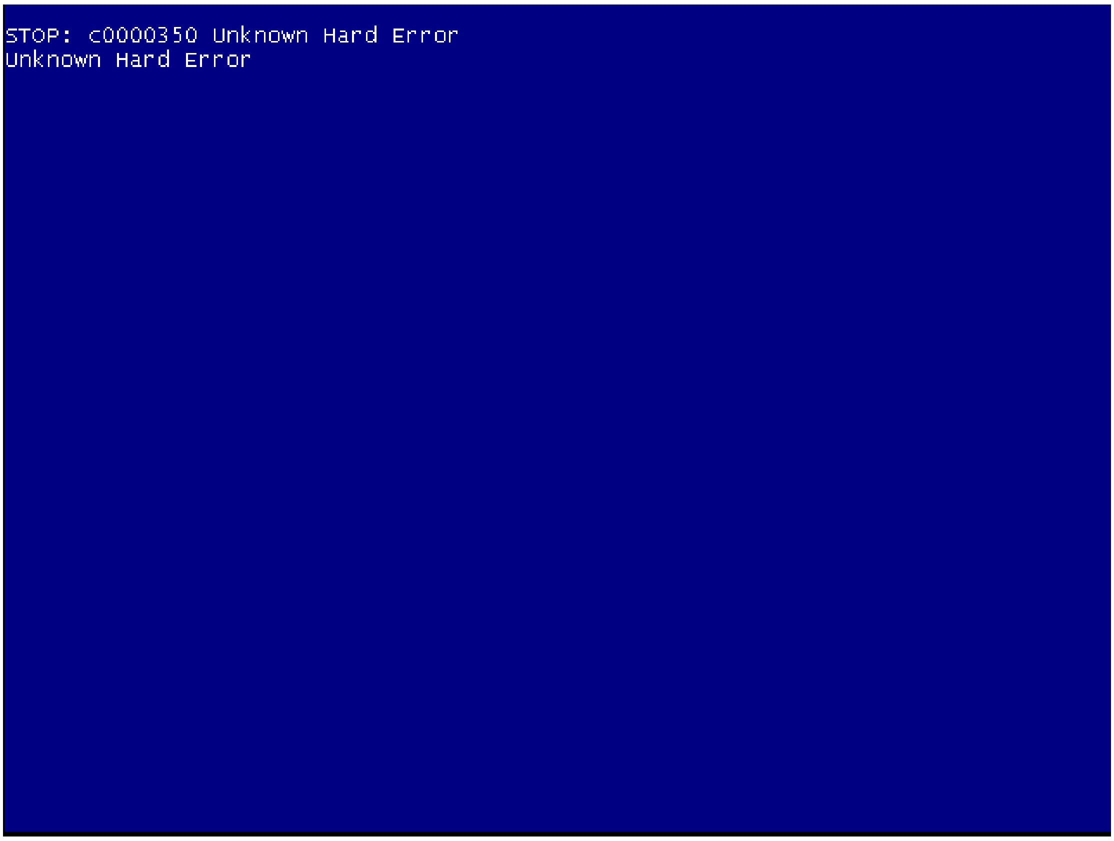
MD5	File
A92F13F3A1B3B39833D3CC336301B713	伪装成 PDF 的 EXE 文件
AF2379CC4D607A45AC44D62135FB7015	伪装成 RAR 的 EXE 文件

行为分析

样本将自己的图标伪装成 PDF 和 RAR 自解压的可执行文件，攻击者通过邮件将恶意代码发送给攻击目标，利用社会工程学引诱攻击者进行运行。

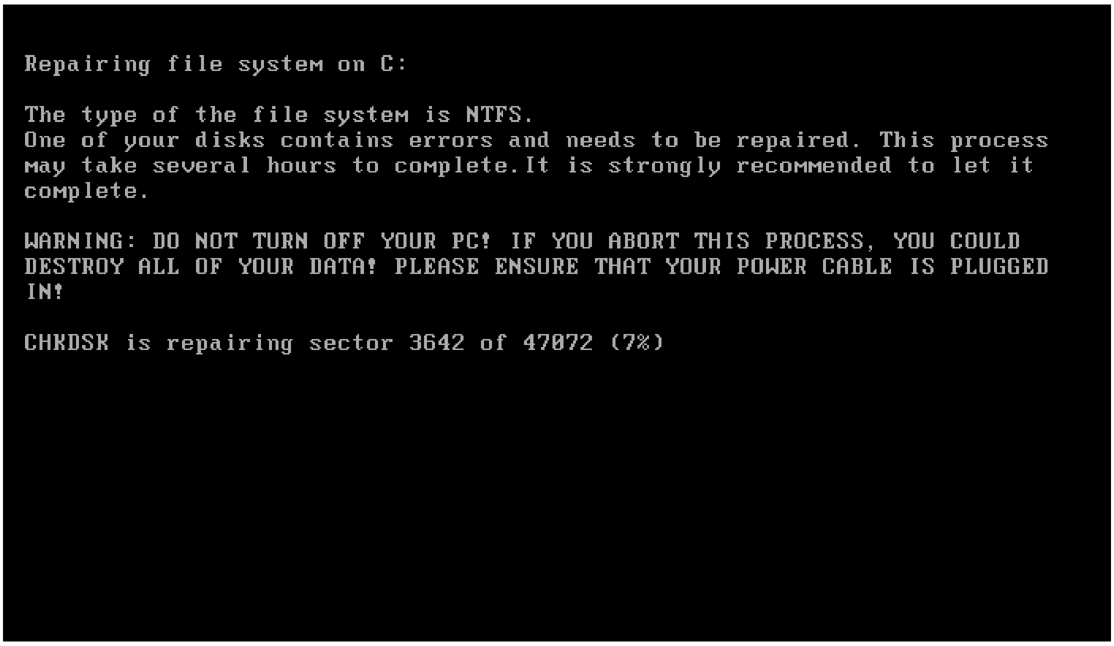


木马运行后通过内部调用系统硬件异常，导致系统蓝屏重启。



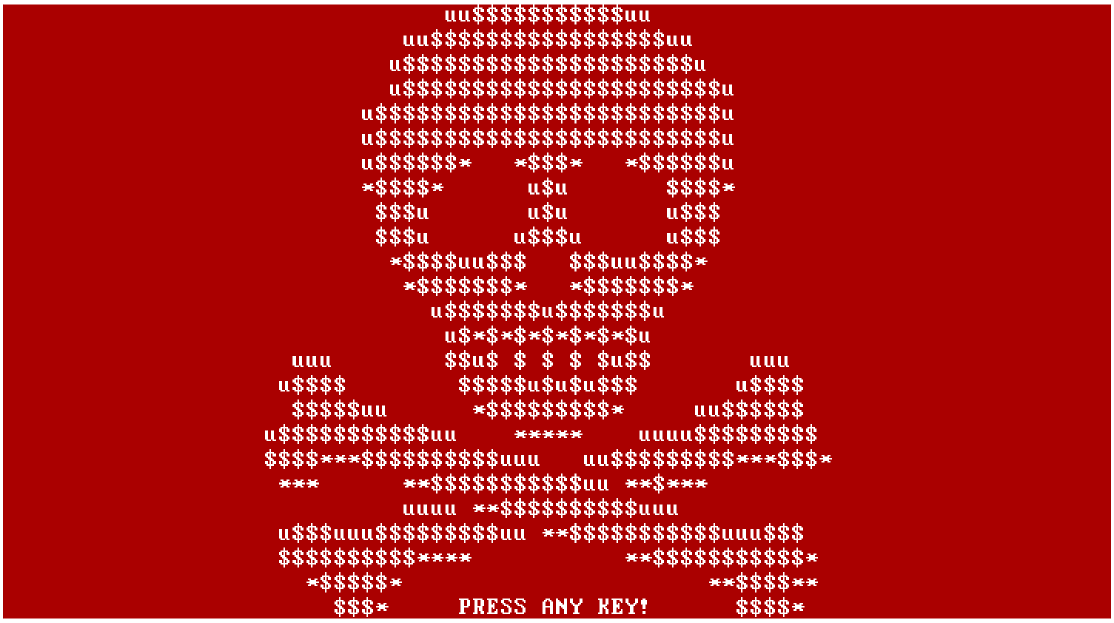
```
STOP: c0000350 Unknown Hard Error
Unknown Hard Error
```

系统重启后会提示用户进行磁盘检查，实际上此时在执行磁盘加密功能。



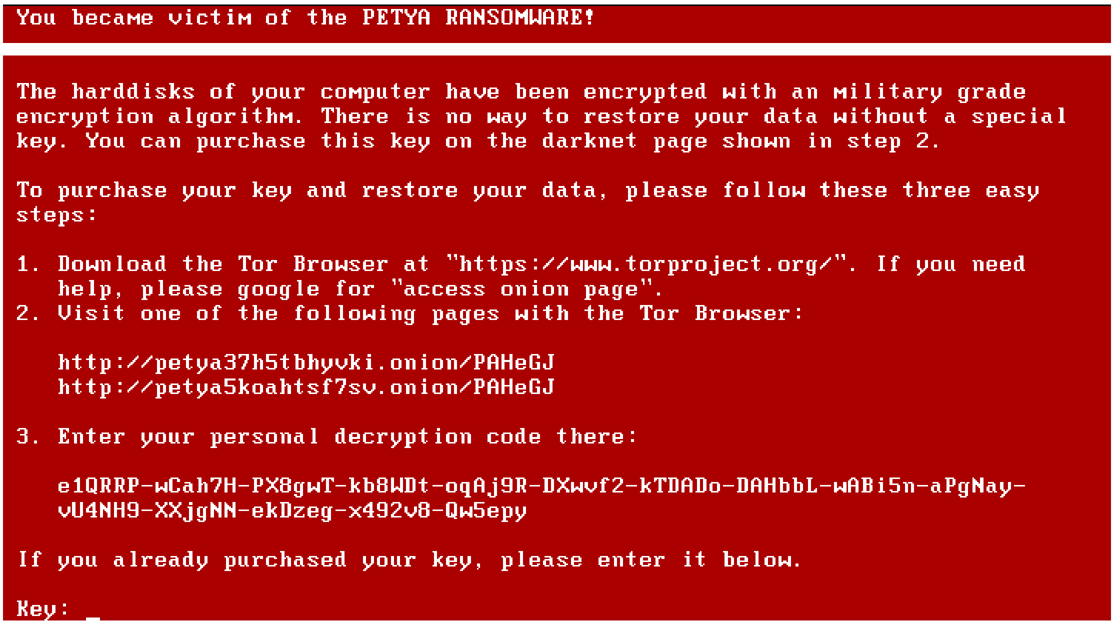
```
Repairing file system on C:

The type of the file system is NTFS.
One of your disks contains errors and needs to be repaired. This process
may take several hours to complete.It is strongly recommended to let it
complete.

WARNING: DO NOT TURN OFF YOUR PC! IF YOU ABORT THIS PROCESS, YOU COULD
DESTROY ALL OF YOUR DATA! PLEASE ENSURE THAT YOUR POWER CABLE IS PLUGGED
IN!

CHKDSK is repairing sector 3642 of 47072 (7%)
```

执行完毕后主机会看到闪烁的屏幕，由一些 ASCII 码组成。



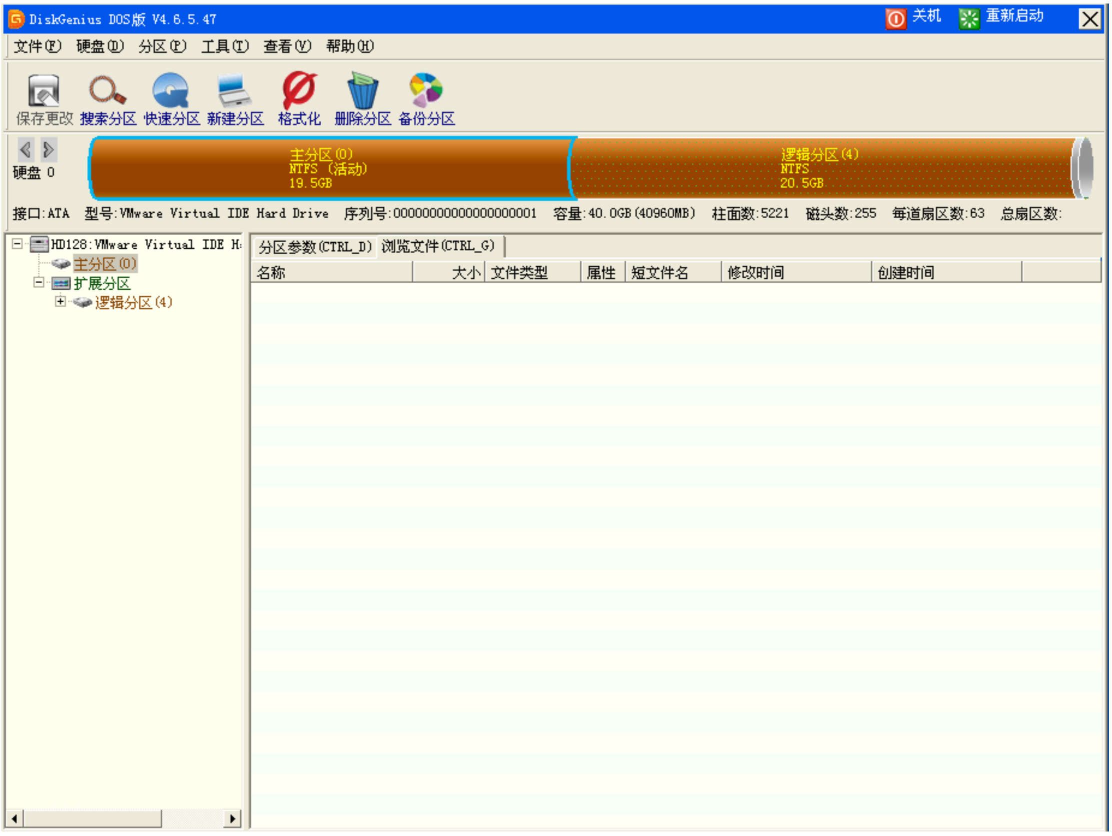
```
              uu$$$$$$$$$$$$uu
           uu$$$$$$$$$$$$$$$$$$uu
          u$$$$$$$$$$$$$$$$$$$$$$$u
          u$$$$$$$$$$$$$$$$$$$$$$$$$u
        u$$$$$$$$$$$$$$$$$$$$$$$$$$$u
        u$$$$$$$$$$$$$$$$$$$$$$$$$$$u
        u$$$$$$*    *$$$*    *$$$$$$u
        *$$$$*       u$u        $$$$*
         $$$u        u$u        u$$$
         $$$u       u$$$u       u$$$
          *$$$$uu$$$    $$$uu$$$$*
           *$$$$$$$*    *$$$$$$$*
             u$$$$$$$u$$$$$$$u
              u$*$*$*$*$*$*$u
   uuu        $$u$ $ $ $ $u$$       uuu
  u$$$$        $$$$$u$u$u$$$       u$$$$
   $$$$$$uu     *$$$$$$$$$*     uu$$$$$$
 u$$$$$$$$$$$uu    *****    uuuu$$$$$$$$$
 $$$$***$$$$$$$$$$uuu   uu$$$$$$$$$***$$$*
  ***      **$$$$$$$$$$$uu **$***
           uuuu **$$$$$$$$$$uuu
  u$$$uuu$$$$$$$$$uu **$$$$$$$$$$$uuu$$$
  $$$$$$$$$$****           **$$$$$$$$$$$*
    *$$$$$*                      **$$$$**
      $$$*     PRESS ANY KEY!      $$$$*
```

根据提示按任意键后，屏幕上回显示勒索信息，按照信息提示支付比特币才能解决问题。



```
You became victim of the PETYA RANSOMWARE!

The harddisks of your computer have been encrypted with an military grade
encryption algorithm. There is no way to restore your data without a special
key. You can purchase this key on the darknet page shown in step 2.

To purchase your key and restore your data, please follow these three easy
steps:

1. Download the Tor Browser at "https://www.torproject.org/". If you need
   help, please google for "access onion page".
2. Visit one of the following pages with the Tor Browser:

   http://petya37h5tbhyvki.onion/PAHeGJ
   http://petya5koahtsf7sv.onion/PAHeGJ

3. Enter your personal decryption code there:

   e1QRRP-wCah7H-PX8gwT-kb8WDt-oqAj9R-DXwvf2-kTDADo-DAHbbL-wABi5n-aPgNay-
   vU4NH9-XXjgNN-ekDzeg-x492v8-Qw5epy

If you already purchased your key, please enter it below.

Key: _
```

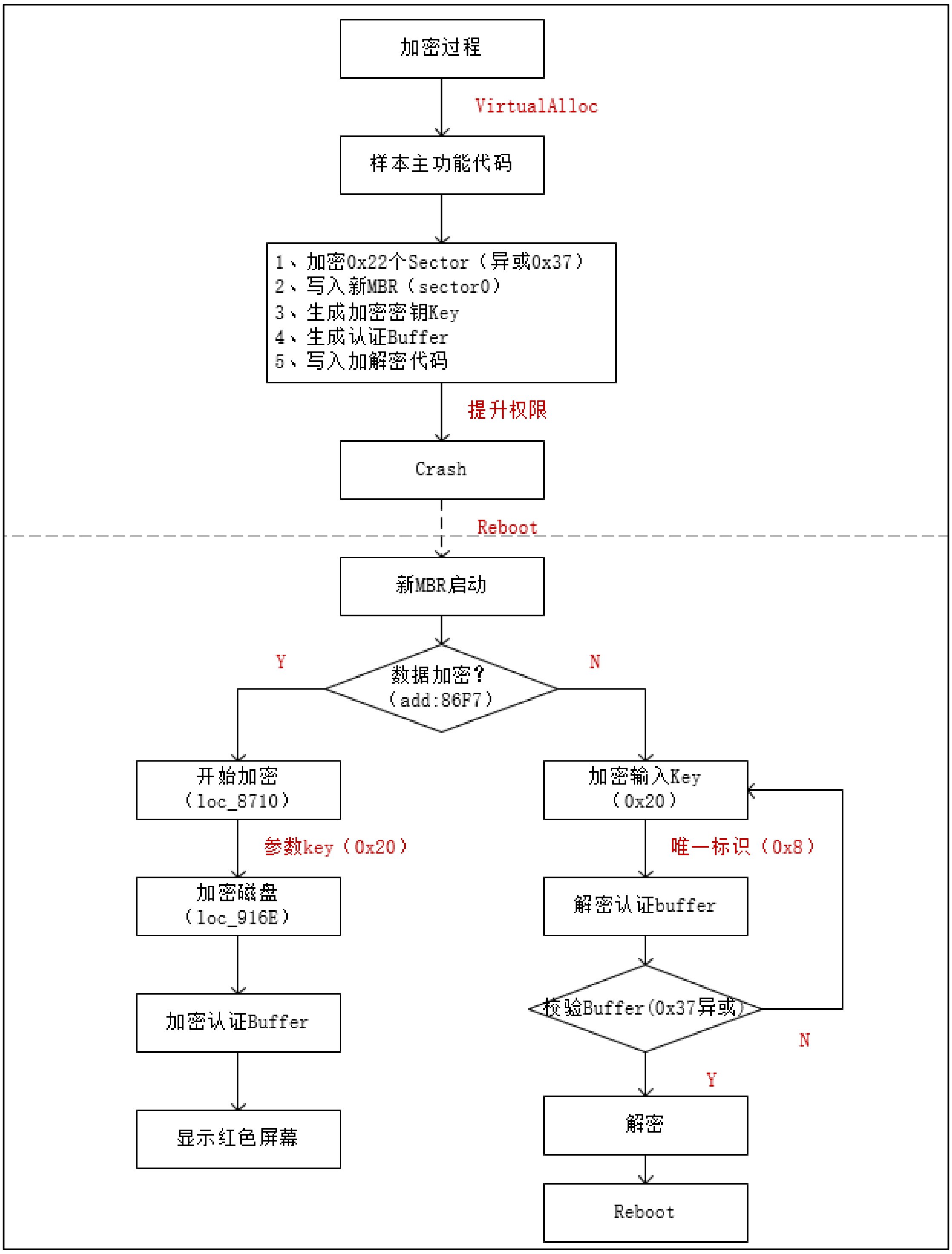
用 diskgenius 查看加密后的情况，发现样本并未进行全盘加密，而是加密了系统分区。


DiskGenius DOS版 V4.6.5.47
关机
重新启动
文件(F) 硬盘(D) 分区(P) 工具(T) 查看(V) 帮助(H)
保存更改 搜索分区 快速分区 新建分区 格式化 删除分区 备份分区
硬盘 0
主分区(0)
NTFS (活动)
19.5GB
逻辑分区(4)
NTFS
20.5GB
HD128:VMware Virtual IDE H:
主分区(0)
扩展分区
逻辑分区(4)
分区参数(CTRL_D) 浏览文件(CTRL_G)
名称 大小 文件类型 属性 短文件名 修改时间 创建时间

接口:ATA 型号:VMware Virtual IDE Hard Drive 序列号:00000000000000000001 容量:40.0GB(40960MB) 柱面数:5221 磁头数:255 每道扇区数:63 总扇区数:

执行概要

该样本主文件是一个外壳程序，静态无法检测到恶意代码，执行过程中会申请新的内存空间，释放主功能代码，写入到物理磁盘的启动位置，修改 MBR，之后强制系统重启。具体流程图如下：



什么是 MBR？

MBR，即主引导记录（Master Boot Record）,是对 IBM 兼容机的硬盘或者可移动设备分区时，在驱动器最前端的一段引导扇区，位于磁盘的 0 柱面、0 磁头、1 扇区（每个扇区为 512 个字节）。

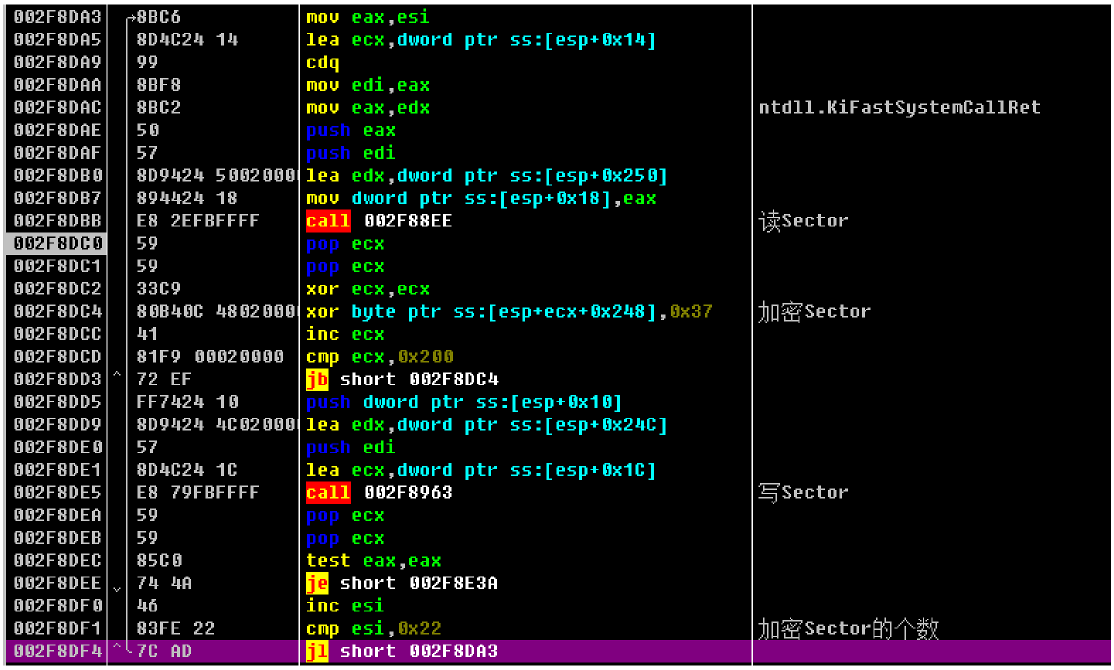
MBR 描述了逻辑分区的信息，包含文件系统和组织方式，以及计算机在启动第二阶段加载操作系统的可执行代码或连接每个分区的引导记录，通常被称为引导程序。

MBR 结构如下：

字节偏移（十六进制）	字节数	描述
0x00-0x1BD	446	引导代码
0x1BE-0x1CD	16	分区表项 1
0x1CE-0x1DD	16	分区表项 2
0x1DE-0x1ED	16	分区表项 3
0x1EE-0x1FD	16	分区表项 4
0x1FE-0x1FF	2	签名值 0xAA55 或者 0x55AA

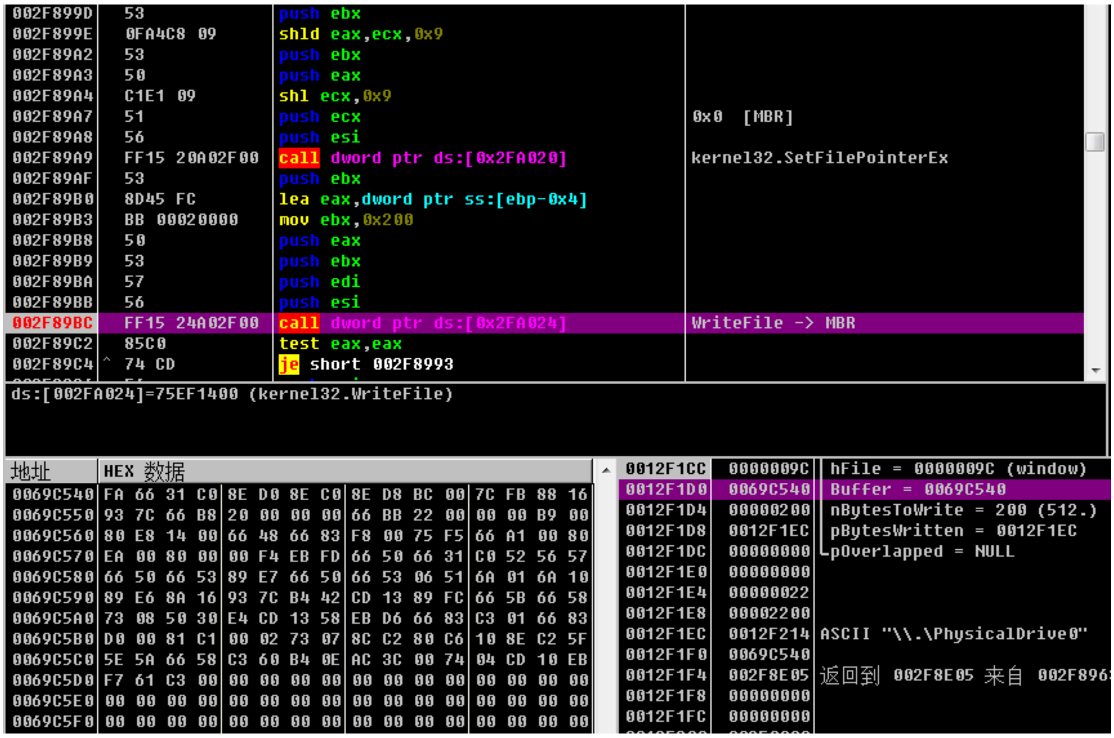
行为分析

- 样本文件的行为



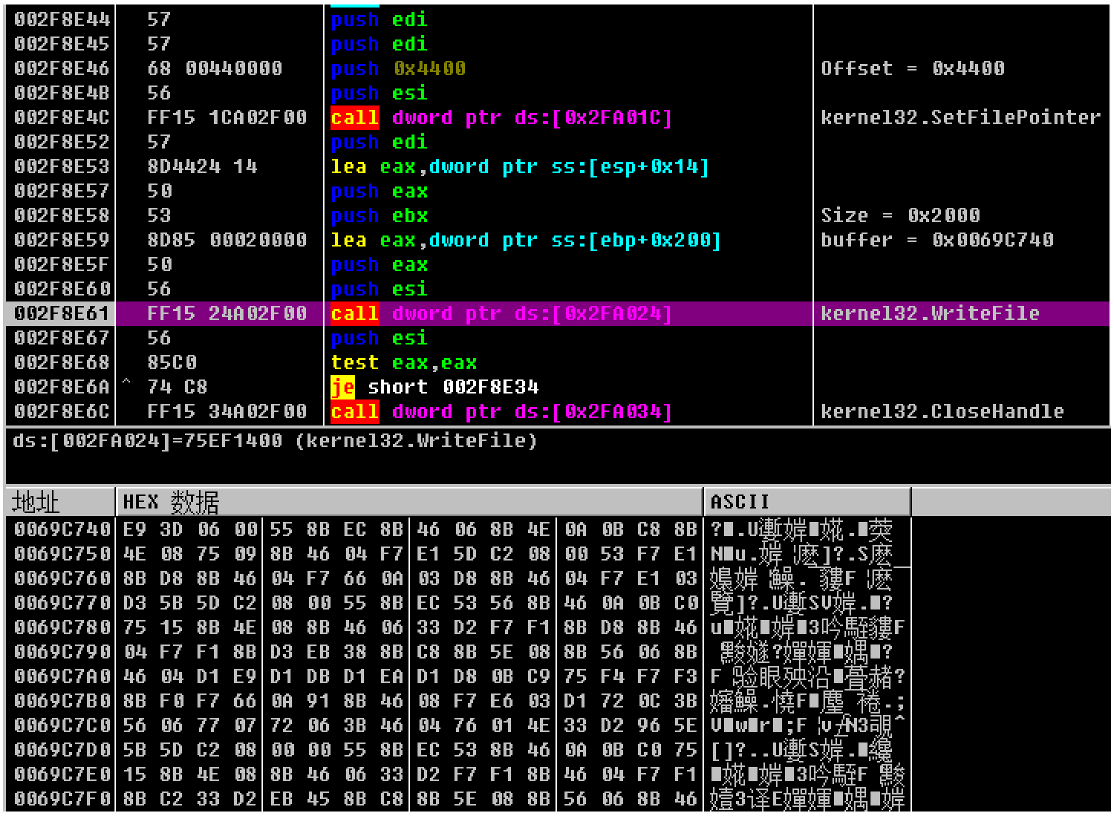
```
002F8DA3  ┌>8BC6           mov eax,esi
002F8DA5  │ 8D4C24 14      lea ecx,dword ptr ss:[esp+0x14]
002F8DA9  │ 99             cdq
002F8DAA  │ 8BF8           mov edi,eax
002F8DAC  │ 8BC2           mov eax,edx                         ntdll.KiFastSystemCallRet
002F8DAE  │ 50             push eax
002F8DAF  │ 57             push edi
002F8DB0  │ 8D9424 5002000 lea edx,dword ptr ss:[esp+0x250]
002F8DB7  │ 894424 18      mov dword ptr ss:[esp+0x18],eax
002F8DBB  │ E8 2EFBFFFF    call 002F88EE                       读Sector
002F8DC0  │ 59             pop ecx
002F8DC1  │ 59             pop ecx
002F8DC2  │ 33C9           xor ecx,ecx
002F8DC4  │ 80B40C 4802000 xor byte ptr ss:[esp+ecx+0x248],0x37 加密Sector
002F8DCC  │ 41             inc ecx
002F8DCD  │ 81F9 00020000  cmp ecx,0x200
002F8DD3 ^│ 72 EF          jb short 002F8DC4
002F8DD5  │ FF7424 10      push dword ptr ss:[esp+0x10]
002F8DD9  │ 8D9424 4C02000 lea edx,dword ptr ss:[esp+0x24C]
002F8DE0  │ 57             push edi
002F8DE1  │ 8D4C24 1C      lea ecx,dword ptr ss:[esp+0x1C]
002F8DE5  │ E8 79FBFFFF    call 002F8963                       写Sector
002F8DEA  │ 59             pop ecx
002F8DEB  │ 59             pop ecx
002F8DEC  │ 85C0           test eax,eax
002F8DEE v│ 74 4A          je short 002F8E3A
002F8DF0  │ 46             inc esi
002F8DF1  │ 83FE 22        cmp esi,0x22                        加密Sector的个数
002F8DF4 ^└7C AD          jl short 002F8DA3
```

加密 0x22 个扇区



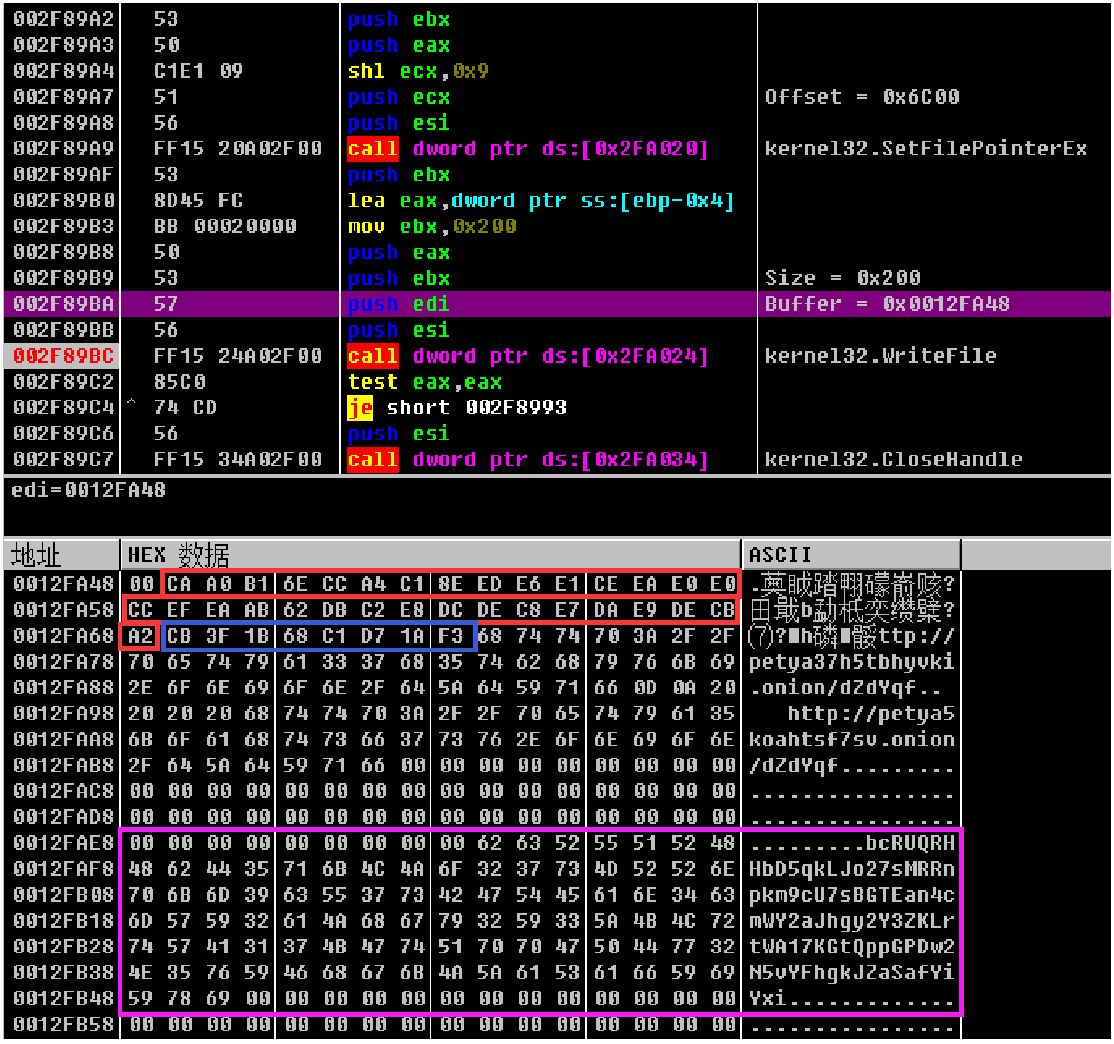
```
002F899D   53              push ebx
002F899E   0FA4C8 09       shld eax,ecx,0x9
002F89A2   53              push ebx
002F89A3   50              push eax
002F89A4   C1E1 09         shl ecx,0x9
002F89A7   51              push ecx                              0x0  [MBR]
002F89A8   56              push esi
002F89A9   FF15 20A02F00   call dword ptr ds:[0x2FA020]          kernel32.SetFilePointerEx
002F89AF   53              push ebx
002F89B0   8D45 FC         lea eax,dword ptr ss:[ebp-0x4]
002F89B3   BB 00020000     mov ebx,0x200
002F89B8   50              push eax
002F89B9   53              push ebx
002F89BA   57              push edi
002F89BB   56              push esi
002F89BC   FF15 24A02F00   call dword ptr ds:[0x2FA024]          WriteFile -> MBR
002F89C2   85C0            test eax,eax
002F89C4 ^ 74 CD           je short 002F8993
ds:[002FA024]=75EF1400 (kernel32.WriteFile)

地址      HEX 数据
0069C540  FA 66 31 C0 8E D0 8E C0 8E D8 BC 00 7C FB 88 16
0069C550  93 7C 66 B8 20 00 00 00 66 BB 22 00 00 00 B9 00
0069C560  80 E8 14 00 66 48 66 83 F8 00 75 F5 66 A1 00 80
0069C570  EA 00 80 00 00 F4 EB FD 66 50 66 31 C0 52 56 57
0069C580  66 50 66 53 89 E7 66 50 66 53 06 51 6A 01 6A 10
0069C590  89 E6 8A 16 93 7C B4 42 CD 13 89 FC 66 5B 66 58
0069C5A0  73 08 50 30 E4 CD 13 58 EB D6 66 83 C3 01 66 83
0069C5B0  D0 00 81 C1 00 02 73 07 8C C2 80 C6 10 8E C2 5F
0069C5C0  5E 5A 66 58 C3 60 B4 0E AC 3C 00 74 04 CD 10 EB
0069C5D0  F7 61 C3 00 00 00 00 00 00 00 00 00 00 00 00 00
0069C5E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0069C5F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0012F1CC   0000009C  ┌hFile = 0000009C (window)
0012F1D0   0069C540  │Buffer = 0069C540
0012F1D4   00000200  │nBytesToWrite = 200 (512.)
0012F1D8   0012F1EC  │pBytesWritten = 0012F1EC
0012F1DC   00000000  └pOverlapped = NULL
0012F1E0   00000000
0012F1E4   00000022
0012F1E8   00002200
0012F1EC   0012F214  ASCII "\\.\PhysicalDrive0"
0012F1F0   0069C540
0012F1F4   002F8E05  返回到 002F8E05 来自 002F896
0012F1F8   00000000
0012F1FC   00000000
```

替换 MBR 数据



```
002F8E44   57              push edi
002F8E45   57              push edi
002F8E46   68 00440000     push 0x4400                           Offset = 0x4400
002F8E4B   56              push esi
002F8E4C   FF15 1CA02F00   call dword ptr ds:[0x2FA01C]          kernel32.SetFilePointer
002F8E52   57              push edi
002F8E53   8D4424 14       lea eax,dword ptr ss:[esp+0x14]
002F8E57   50              push eax
002F8E58   53              push ebx                              Size = 0x2000
002F8E59   8D85 00020000   lea eax,dword ptr ss:[ebp+0x200]      buffer = 0x0069C740
002F8E5F   50              push eax
002F8E60   56              push esi
002F8E61   FF15 24A02F00   call dword ptr ds:[0x2FA024]          kernel32.WriteFile
002F8E67   56              push esi
002F8E68   85C0            test eax,eax
002F8E6A ^ 74 C8           je short 002F8E34
002F8E6C   FF15 34A02F00   call dword ptr ds:[0x2FA034]          kernel32.CloseHandle
ds:[002FA024]=75EF1400 (kernel32.WriteFile)

地址      HEX 数据                                          ASCII
0069C740  E9 3D 06 00 55 8B EC 8B 46 06 8B 4E 0A 0B C8 8B  ?■.U嫜媂■娍.■葵
0069C750  4E 08 75 09 8B 46 04 F7 E1 5D C2 08 00 53 F7 E1  N■u.媂 鳫]?.S鳫
0069C760  8B D8 8B 46 04 F7 66 0A 03 D8 8B 46 04 F7 E1 03  嬝媂 鱢. 豂F 鳫
0069C770  D3 5B 5D C2 08 00 55 8B EC 53 56 8B 46 0A 0B C0  覽]?.U嫜SV媂.■?
0069C780  75 15 8B 4E 08 8B 46 06 33 D2 F7 F1 8B D8 8B 46  u■娍■媂■3裼駤豂F
0069C790  04 F7 F1 8B D3 EB 38 8B C8 8B 5E 08 8B 56 06 8B   鰊嬘?嬋媈■媀■?
0069C7A0  46 04 D1 E9 D1 DB D1 EA D1 D8 0B C9 75 F4 F7 F3  F 验眼驶沿■蓇緒?
0069C7B0  8B F0 F7 66 0A 91 8B 46 08 F7 E6 03 D1 72 0C 3B  嬸鱢.憢F■鳎 裷.;
0069C7C0  56 06 77 07 72 06 3B 46 04 76 01 4E 33 D2 96 5E  V■w■r■;F ︱v N3覗^
0069C7D0  5B 5D C2 08 00 00 55 8B EC 53 8B 46 0A 0B C0 75  []?..U嫜S媂.■纔
0069C7E0  15 8B 4E 08 8B 46 06 33 D2 F7 F1 8B 46 04 F7 F1  ■娍■媂■3裼駤F 鰊
0069C7F0  8B C2 33 D2 EB 45 8B C8 8B 5E 08 8B 56 06 8B 46  嬄3译E嬋媈■媀■媂
```

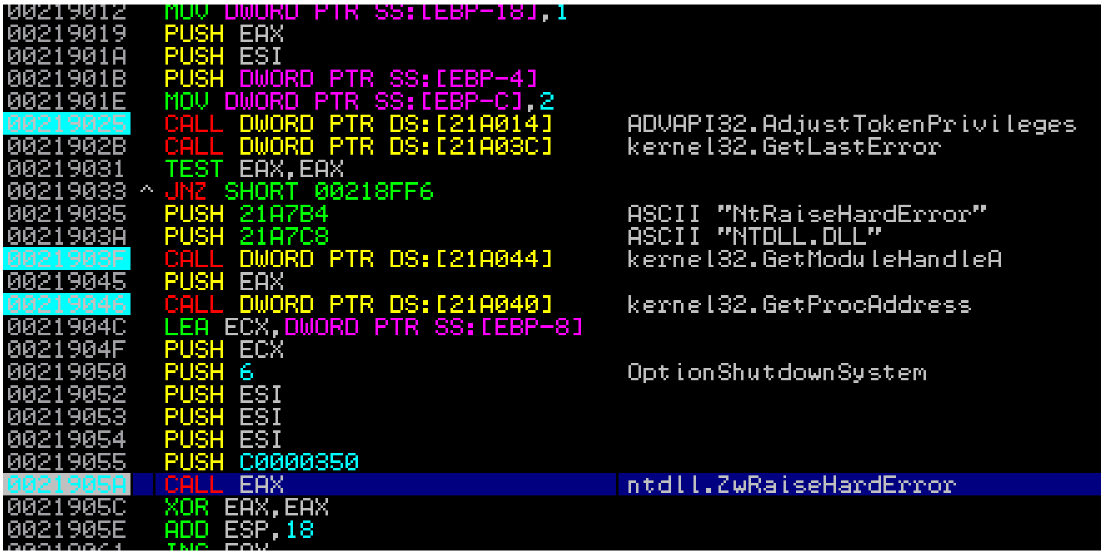
将加解密代码写入磁盘扇区



```
002F89A2    53               push ebx
002F89A3    50               push eax
002F89A4    C1E1 09          shl ecx,0x9
002F89A7    51               push ecx                          Offset = 0x6C00
002F89A8    56               push esi
002F89A9    FF15 20A02F00    call dword ptr ds:[0x2FA020]      kernel32.SetFilePointerEx
002F89AF    53               push ebx
002F89B0    8D45 FC          lea eax,dword ptr ss:[ebp-0x4]
002F89B3    BB 00020000      mov ebx,0x200
002F89B8    50               push eax
002F89B9    53               push ebx                          Size = 0x200
002F89BA    57               push edi                          Buffer = 0x0012FA48
002F89BB    56               push esi
002F89BC    FF15 24A02F00    call dword ptr ds:[0x2FA024]      kernel32.WriteFile
002F89C2    85C0             test eax,eax
002F89C4  ^ 74 CD            je short 002F8993
002F89C6    56               push esi
002F89C7    FF15 34A02F00    call dword ptr ds:[0x2FA034]      kernel32.CloseHandle
edi=0012FA48

地址      HEX 数据                                          ASCII
0012FA48  00 CA A0 B1 6E CC A4 C1 8E ED E6 E1 CE EA E0 E0  .蕒眓踏翢礶萷赅?
0012FA58  CC EF EA AB 62 DB C2 E8 DC DE C8 E7 DA E9 DE CB  田戢b勐柢奕缵髹?
0012FA68  A2 CB 3F 1B 68 C1 D7 1A F3 68 74 74 70 3A 2F 2F  ⑺?■h磷■骹ttp://
0012FA78  70 65 74 79 61 33 37 68 35 74 62 68 79 76 6B 69  petya37h5tbhyvki
0012FA88  2E 6F 6E 69 6F 6E 2F 64 5A 64 59 71 66 0D 0A 20  .onion/dZdYqf..
0012FA98  20 20 20 68 74 74 70 3A 2F 2F 70 65 74 79 61 35     http://petya5
0012FAA8  6B 6F 61 68 74 73 66 37 73 76 2E 6F 6E 69 6F 6E  koahtsf7sv.onion
0012FAB8  2F 64 5A 64 59 71 66 00 00 00 00 00 00 00 00 00  /dZdYqf.........
0012FAC8  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0012FAD8  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0012FAE8  00 00 00 00 00 00 00 00 00 62 63 52 55 51 52 48  .........bcRUQRH
0012FAF8  48 62 44 35 71 6B 4C 4A 6F 32 37 73 4D 52 52 6E  HbD5qkLJo27sMRRn
0012FB08  70 6B 6D 39 63 55 37 73 42 47 54 45 61 6E 34 63  pkm9cU7sBGTEan4c
0012FB18  6D 57 59 32 61 4A 68 67 79 32 59 33 5A 4B 4C 72  mWY2aJhgy2Y3ZKLr
0012FB28  74 57 41 31 37 4B 47 74 51 70 70 47 50 44 77 32  tWA17KGtQppGPDw2
0012FB38  4E 35 76 59 46 68 67 6B 4A 5A 61 53 61 66 59 69  N5vYFhgkJZaSafYi
0012FB48  59 78 69 00 00 00 00 00 00 00 00 00 00 00 00 00  Yxi.............
0012FB58  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

写入 3 个与加解密有关的数据到磁盘中

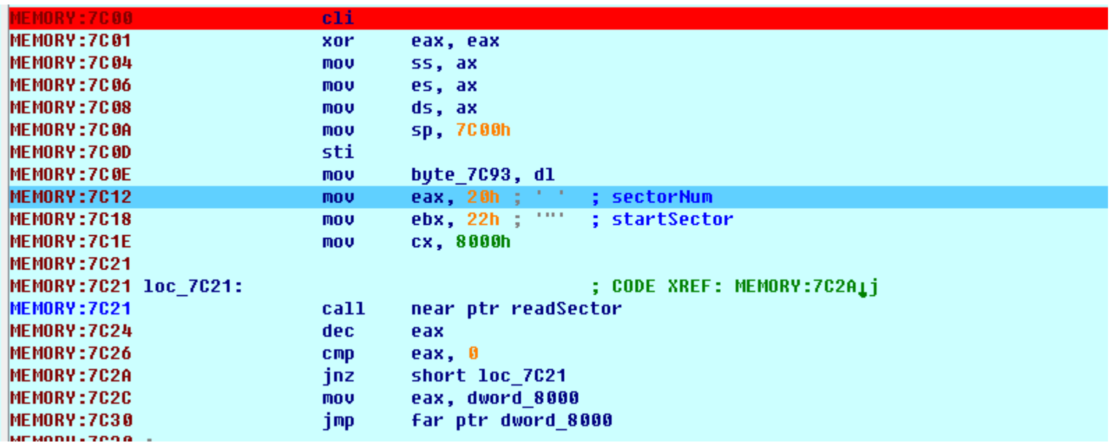
红色部分为 32 个字节经过加密的 KEY，蓝色部分为设备唯一 ID 号，粉色部分为提示用户在勒索网站需要填入的解密字符串。



```
00219012    MOV DWORD PTR SS:[EBP-18],1
00219019    PUSH EAX
0021901A    PUSH ESI
0021901B    PUSH DWORD PTR SS:[EBP-4]
0021901E    MOV DWORD PTR SS:[EBP-C],2
00219025    CALL DWORD PTR DS:[21A014]       ADVAPI32.AdjustTokenPrivileges
0021902B    CALL DWORD PTR DS:[21A03C]       kernel32.GetLastError
00219031    TEST EAX,EAX
00219033 ^  JNZ SHORT 00218FF6
00219035    PUSH 21A7B4                      ASCII "NtRaiseHardError"
0021903A    PUSH 21A7C8                      ASCII "NTDLL.DLL"
0021903F    CALL DWORD PTR DS:[21A044]       kernel32.GetModuleHandleA
00219045    PUSH EAX
00219046    CALL DWORD PTR DS:[21A040]       kernel32.GetProcAddress
0021904C    LEA ECX,DWORD PTR SS:[EBP-8]
0021904F    PUSH ECX
00219050    PUSH 6                           OptionShutdownSystem
00219052    PUSH ESI
00219053    PUSH ESI
00219054    PUSH ESI
00219055    PUSH C0000350
0021905B    CALL EAX                         ntdll.ZwRaiseHardError
0021905C    XOR EAX,EAX
0021905E    ADD ESP,18
```

执行硬件错误异常

- MBR 代码

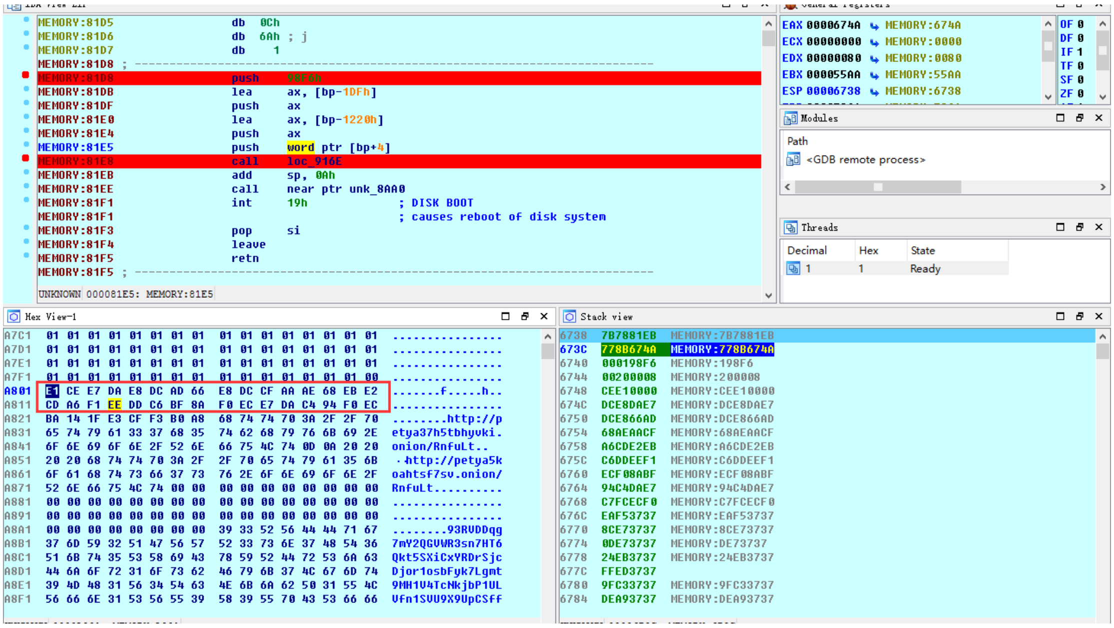


```
MEMORY:7C00                 cli
MEMORY:7C01                 xor     eax, eax
MEMORY:7C04                 mov     ss, ax
MEMORY:7C06                 mov     es, ax
MEMORY:7C08                 mov     ds, ax
MEMORY:7C0A                 mov     sp, 7C00h
MEMORY:7C0D                 sti
MEMORY:7C0E                 mov     byte_7C93, dl
MEMORY:7C12                 mov     eax, 20h ; ' '    ; sectorNum
MEMORY:7C18                 mov     ebx, 22h ; '"'    ; startSector
MEMORY:7C1E                 mov     cx, 8000h
MEMORY:7C21
MEMORY:7C21 loc_7C21:                                 ; CODE XREF: MEMORY:7C2A↓j
MEMORY:7C21                 call    near ptr readSector
MEMORY:7C24                 dec     eax
MEMORY:7C26                 cmp     eax, 0
MEMORY:7C2A                 jnz     short loc_7C21
MEMORY:7C2C                 mov     eax, dword_8000
MEMORY:7C30                 jmp     far ptr dword_8000
```

恶意 MBR 代码

0x7C21 处将样本的主功能代码加在到内存 0x8000 处，然后在 0x7C30 处跳转到恶意代码进行加解密操作。

- 加密代码



```
MEMORY:81D5                 db  0Ch
MEMORY:81D6                 db  6Ah ; j
MEMORY:81D7                 db    1
MEMORY:81D8 ; ---------------------------------------------------------------
MEMORY:81D8                 push    98F6h
MEMORY:81DB                 lea     ax, [bp-1DFh]
MEMORY:81DF                 push    ax
MEMORY:81E0                 lea     ax, [bp-1220h]
MEMORY:81E4                 push    ax
MEMORY:81E5                 push    word ptr [bp+4]
MEMORY:81E8                 call    loc_916E
MEMORY:81EB                 add     sp, 0Ah
MEMORY:81EE                 call    near ptr unk_8AA0
MEMORY:81F1                 int     19h                ; DISK BOOT
MEMORY:81F1                                            ; causes reboot of disk system
MEMORY:81F3                 pop     si
MEMORY:81F4                 leave
MEMORY:81F5                 retn
MEMORY:81F5 ; ---------------------------------------------------------------
UNKNOWN 000081E5: MEMORY:81E5
```

```
EAX 0000674A ↳ MEMORY:674A     OF 0
ECX 00000000 ↳ MEMORY:0000     DF 0
EDX 00000080 ↳ MEMORY:0080     IF 1
EBX 000055AA ↳ MEMORY:55AA     TF 0
ESP 00006738 ↳ MEMORY:6738     SF 0
                               ZF 0
```

Modules

Path
<GDB remote process>

Threads

Decimal	Hex	State
1	1	Ready

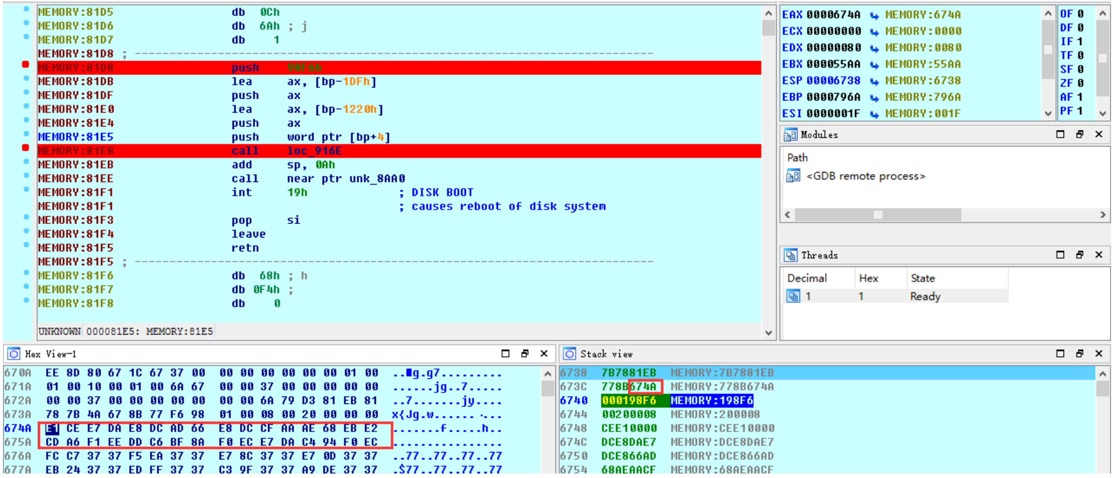
Hex View-1

```
A7C1  01 01 01 01 01 01 01 01  01 01 01 01 01 01 01 01  ................
A7D1  01 01 01 01 01 01 01 01  01 01 01 01 01 01 01 01  ................
A7E1  01 01 01 01 01 01 01 01  01 01 01 01 01 01 01 01  ................
A7F1  01 01 01 01 01 01 01 01  01 01 01 01 01 01 01 00  ................
A801  E1 CE E7 DA E8 DC AD 66  E8 DC CF AA AE 68 EB E2  .......f.....h..
A811  CD A6 F1 EE DD C6 BF 8A  F0 EC E7 DA C4 94 F0 EC  ................
A821  BA 14 1F E3 CF F3 B0 A8  68 74 74 70 3A 2F 2F 70  ........http://p
A831  65 74 79 61 33 37 68 35  74 62 68 79 76 6B 69 2E  etya37h5tbhyvki.
A841  6F 6E 69 6F 6E 2F 52 6E  66 75 4C 74 0D 0A 20 20  onion/RnfuLt..
A851  20 20 68 74 74 70 3A 2F  2F 70 65 74 79 61 35 6B   ·http://petya5k
A861  6F 61 68 74 73 66 37 73  76 2E 6F 6E 69 6F 6E 2F  oahtsf7sv.onion/
A871  52 6E 66 75 4C 74 00 00  00 00 00 00 00 00 00 00  RnfuLt..........
A881  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
A891  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
A8A1  00 00 00 00 00 00 00 00  39 33 52 56 44 44 71 67  ........93RVDDqg
A8B1  37 6D 59 32 51 47 56 57  52 33 73 6E 37 48 54 36  7mY2QGVWR3sn7HT6
A8C1  51 6B 74 35 53 58 69 43  78 59 52 44 72 53 6A 63  Qkt5SXiCxYRDrSjc
A8D1  44 6A 6F 72 31 6F 73 62  46 79 6B 37 4C 67 6D 74  Djor1osbFyk7Lgmt
A8E1  39 4D 48 31 56 34 54 63  4E 6B 6A 62 50 31 55 4C  9MH1V4TcNkjbP1UL
A8F1  56 66 6E 31 53 56 55 39  58 39 55 70 43 53 66 66  Vfn1SVU9X9UpCSff
```

Stack view

```
6738  7B7881EB  MEMORY:7B7881EB
673C  778B674A  MEMORY:778B674A
6740  000198F6  MEMORY:198F6
6744  00200008  MEMORY:200008
6748  CEE10000  MEMORY:CEE10000
674C  DCE8DAE7  MEMORY:DCE8DAE7
6750  DCE866AD  MEMORY:DCE866AD
6754  68AEAACF  MEMORY:68AEAACF
6758  A6CDE2EB  MEMORY:A6CDE2EB
675C  C6DDEEF1  MEMORY:C6DDEEF1
6760  ECF08ABF  MEMORY:ECF08ABF
6764  94C4DAE7  MEMORY:94C4DAE7
6768  C7FCECF0  MEMORY:C7FCECF0
676C  EAF53737  MEMORY:EAF53737
6770  8CE73737  MEMORY:8CE73737
6774  0DE73737  MEMORY:DE73737
6778  24EB3737  MEMORY:24EB3737
677C  FFED3737
6780  9FC33737  MEMORY:9FC33737
6784  DEA93737  MEMORY:DEA93737
```

加密 KEY（0x20 个）数据在内存中的位置



```
MEMORY:81D5                 db  0Ch
MEMORY:81D6                 db  6Ah ; j
MEMORY:81D7                 db    1
MEMORY:81D8 ; ---------------------------------------------------------------
MEMORY:81D8                 push    98F6h
MEMORY:81DB                 lea     ax, [bp-1DFh]
MEMORY:81DF                 push    ax
MEMORY:81E0                 lea     ax, [bp-1220h]
MEMORY:81E4                 push    ax
MEMORY:81E5                 push    word ptr [bp+4]
MEMORY:81E8                 call    loc_916E
MEMORY:81EB                 add     sp, 0Ah
MEMORY:81EE                 call    near ptr unk_8AA0
MEMORY:81F1                 int     19h                ; DISK BOOT
MEMORY:81F1                                            ; causes reboot of disk system
MEMORY:81F3                 pop     si
MEMORY:81F4                 leave
MEMORY:81F5                 retn
MEMORY:81F5 ; ---------------------------------------------------------------
MEMORY:81F6                 db  68h ; h
MEMORY:81F7                 db 0F4h ;
MEMORY:81F8                 db    0
UNKNOWN 000081E5: MEMORY:81E5
```

```
EAX 0000674A ↳ MEMORY:674A     OF 0
ECX 00000000 ↳ MEMORY:0000     DF 0
EDX 00000080 ↳ MEMORY:0080     IF 1
EBX 000055AA ↳ MEMORY:55AA     TF 0
ESP 00006738 ↳ MEMORY:6738     SF 0
EBP 0000796A ↳ MEMORY:796A     ZF 0
ESI 0000001F ↳ MEMORY:001F     AF 1
                               PF 1
```

Modules

Path
<GDB remote process>

Threads

Decimal	Hex	State
1	1	Ready

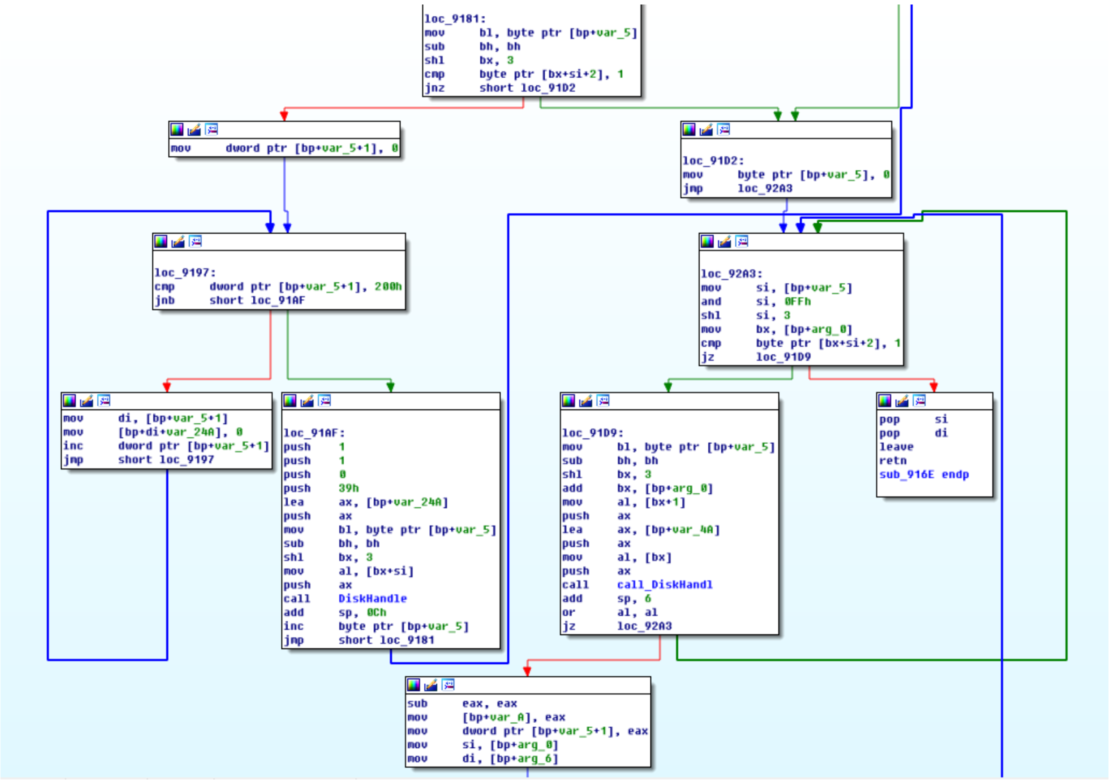
Hex View-1

```
670A  EE 8D 80 67 1C 67 37 00  00 00 00 00 00 00 01 00  ..■g.g7.........
671A  01 00 10 00 01 00 6A 67  00 00 37 00 00 00 00 00  ......jg..7.....
672A  00 00 37 00 00 00 00 00  00 00 6A 79 D3 81 EB 81  ..7.......jy....
673A  78 7B 4A 67 8B 77 F6 98  01 00 08 00 20 00 00 00  x{Jg.w......·...
674A  E1 CE E7 DA E8 DC AD 66  E8 DC CF AA AE 68 EB E2  .......f.....h..
675A  CD A6 F1 EE DD C6 BF 8A  F0 EC E7 DA C4 94 F0 EC  ................
676A  FC C7 37 37 F5 EA 37 37  E7 8C 37 37 E7 0D 37 37  ..77..77..77..77
677A  EB 24 37 37 ED FF 37 37  C3 9F 37 37 A9 DE 37 37  .$77..77..77..77
```

Stack view

```
6738  7B7881EB  MEMORY:7B7881EB
673C  778B674A  MEMORY:778B674A
6740  000198F6  MEMORY:198F6
6744  00200008  MEMORY:200008
6748  CEE10000  MEMORY:CEE10000
674C  DCE8DAE7  MEMORY:DCE8DAE7
6750  DCE866AD  MEMORY:DCE866AD
6754  68AEAACF  MEMORY:68AEAACF
```

加密函数



```
loc_9181:
mov     bl, byte ptr [bp+var_5]
sub     bh, bh
shl     bx, 3
cmp     byte ptr [bx+si+2], 1
jnz     short loc_91D2
```

```
mov     dword ptr [bp+var_5+1], 0
```

```
loc_91D2:
mov     byte ptr [bp+var_5], 0
jmp     loc_92A3
```

```
loc_9197:
cmp     dword ptr [bp+var_5+1], 200h
jnb     short loc_91AF
```

```
loc_92A3:
mov     si, [bp+var_5]
and     si, 0FFh
shl     si, 3
mov     bx, [bp+arg_0]
cmp     byte ptr [bx+si+2], 1
jz      loc_91D9
```

```
mov     di, [bp+var_5+1]
mov     [bp+di+var_24A], 0
inc     dword ptr [bp+var_5+1]
jmp     short loc_9197
```

```
loc_91AF:
push    1
push    1
push    0
push    39h
lea     ax, [bp+var_24A]
push    ax
mov     bl, byte ptr [bp+var_5]
sub     bh, bh
shl     bx, 3
mov     al, [bx+si]
push    ax
call    DiskHandle
add     sp, 0Ch
inc     byte ptr [bp+var_5]
jmp     short loc_9181
```

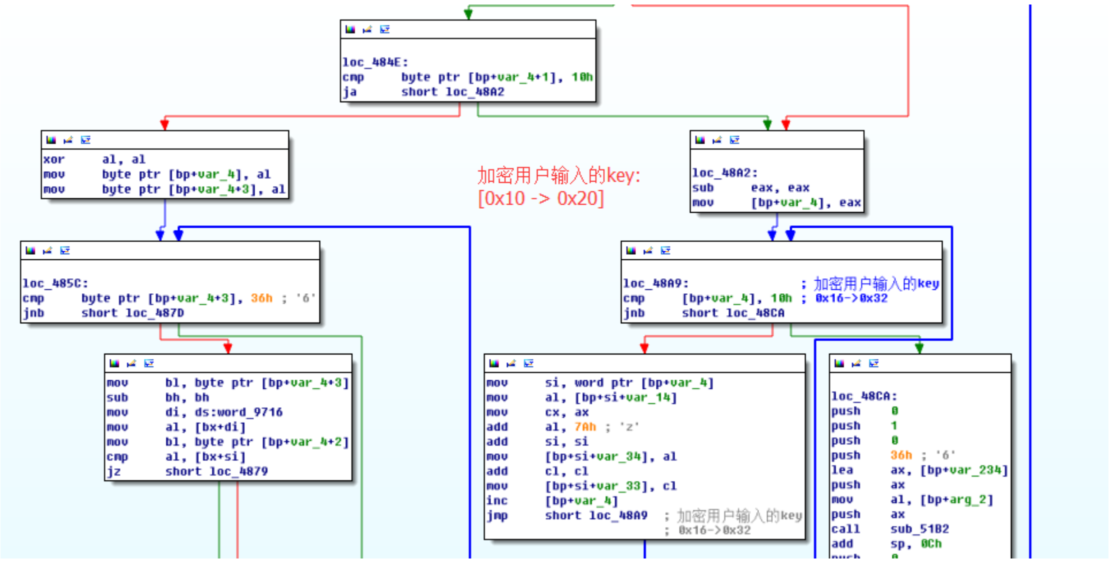
```
loc_91D9:
mov     bl, byte ptr [bp+var_5]
sub     bh, bh
shl     bx, 3
add     bx, [bp+arg_0]
mov     al, [bx+1]
push    ax
lea     ax, [bp+var_4A]
push    ax
mov     al, [bx]
push    ax
call    call_DiskHandl
add     sp, 6
or      al, al
jz      loc_92A3
```

```
pop     si
pop     di
leave
retn
sub_916E endp
```

```
sub     eax, eax
mov     [bp+var_A], eax
mov     dword ptr [bp+var_5+1], eax
mov     si, [bp+arg_0]
mov     di, [bp+arg_6]
```

加密函数的部分流程

- 解密代码



```
loc_484E:
cmp     byte ptr [bp+var_4+1], 10h
ja      short loc_48A2
```

```
xor     al, al
mov     byte ptr [bp+var_4], al
mov     byte ptr [bp+var_4+3], al
```

加密用户输入的key:
[0x10 -> 0x20]

```
loc_48A2:
sub     eax, eax
mov     [bp+var_4], eax
```

```
loc_485C:
cmp     byte ptr [bp+var_4+3], 36h ; '6'
jnb     short loc_487D
```

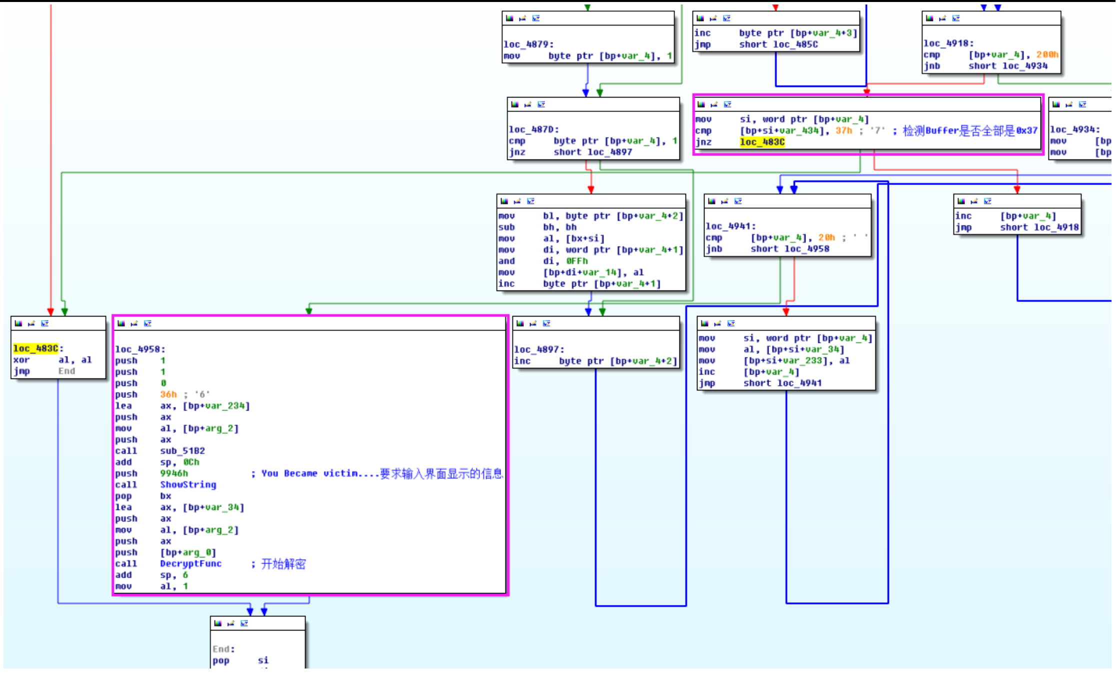
```
loc_48A9:                        ; 加密用户输入的key
cmp     [bp+var_4], 10h ; 0x16->0x32
jnb     short loc_48CA
```

```
mov     bl, byte ptr [bp+var_4+3]
sub     bh, bh
mov     di, ds:word_9716
mov     al, [bx+di]
mov     bl, byte ptr [bp+var_4+2]
cmp     al, [bx+si]
jz      short loc_4879
```

```
mov     si, word ptr [bp+var_4]
mov     al, [bp+si+var_14]
mov     cx, ax
add     al, 7Ah ; 'z'
add     si, si
mov     [bp+si+var_34], al
add     cl, cl
mov     [bp+si+var_33], cl
inc     [bp+var_4]
jmp     short loc_48A9    ; 加密用户输入的key
                          ; 0x16->0x32
```

```
loc_48CA:
push    0
push    1
push    0
push    36h ; '6'
lea     ax, [bp+var_234]
push    ax
mov     al, [bp+arg_2]
push    ax
call    sub_51B2
add     sp, 0Ch
```

加密用户输入 KEY 的流程



```
loc_4879:
mov     byte ptr [bp+var_4], 1
```

```
inc     byte ptr [bp+var_4+3]
jmp     short loc_485C
```

```
loc_4918:
cmp     [bp+var_4], 200h
jnb     short loc_4934
```

```
loc_487D:
cmp     byte ptr [bp+var_4], 1
jnz     short loc_4897
```

```
mov     si, word ptr [bp+var_4]
cmp     [bp+si+var_434], 37h ; '7' ; 检测Buffer是否全部是0x37
jnz     loc_483C
```

```
loc_4934:
mov     [bp
mov     [bp
```

```
mov     bl, byte ptr [bp+var_4+2]
sub     bh, bh
mov     al, [bx+si]
mov     di, word ptr [bp+var_4+1]
and     di, 0FFh
mov     [bp+di+var_14], al
inc     byte ptr [bp+var_4+1]
```

```
loc_4941:
cmp     [bp+var_4], 20h ; ' '
jnb     short loc_4958
```

```
inc     [bp+var_4]
jmp     short loc_4918
```

```
loc_483C:
xor     al, al
jmp     End
```

```
loc_4958:
push    1
push    1
push    0
push    36h ; '6'
lea     ax, [bp+var_234]
push    ax
mov     al, [bp+arg_2]
push    ax
call    sub_51B2
add     sp, 0Ch
push    9946h           ; You Became victim....要求输入界面显示的信息
call    ShowString
pop     bx
lea     ax, [bp+var_34]
push    ax
mov     al, [bp+arg_2]
push    ax
push    [bp+arg_0]
call    DecryptFunc     ; 开始解密
add     sp, 6
mov     al, 1
```

```
loc_4897:
inc     byte ptr [bp+var_4+2]
```

```
mov     si, word ptr [bp+var_4]
mov     al, [bp+si+var_34]
mov     [bp+si+var_233], al
inc     [bp+var_4]
jmp     short loc_4941
```

```
End:
pop     si
```

检测认证缓冲区与解密流程

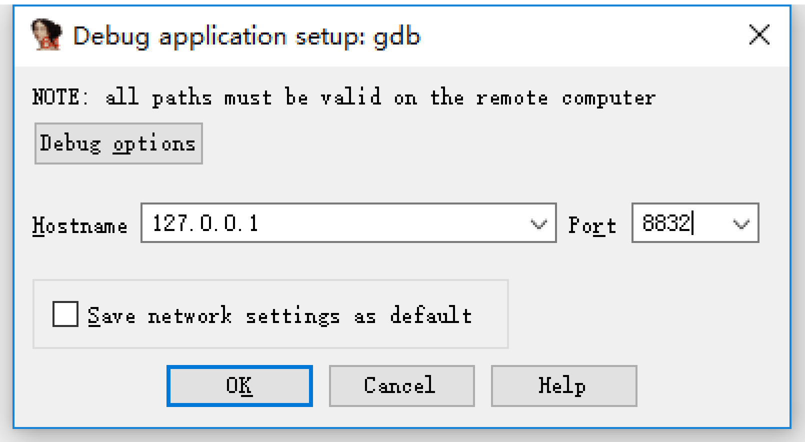
调试方法

此样本利用 MBR 进行攻击，因此针对 MBR 的调试不能在用户层进行调试，需要进行深入的调试，可以利用虚拟机进行 MBR 的调试，这里使用的是 IDA+VMWARE 的解决方案。

VMWARE 提供的 GDB Stub 分两个部分，一个用于支持 X86，一个用于支持 X64。当处于调试状态的 VMWARE 虚拟 CPU 运行在 16/32 位模式下时，32 位支持的 GDB Stub 生效，监听 8832 端口。当处于调试状态的 VMWARE 虚拟 CPU 运行在 Long-Mode 位模式下时，64 位支持的 GDB Stub 生效，监听 8864 端口。当在虚拟机的主配置文件（.VMX）中加入如下代码：

```
debugStub.listen.guest32.remote = “TRUE”
debugStub.listen.guest64.remote = ”TRUE”
monitor.debugOnStartGuest32 = “TRUE”
debugStub.hideBreakpoints = “TRUE”
bios.bootDelay = “3000”
```

启动虚拟机后，IDA 通过附加 Remote GDB debugger，设置如下进行调试：



Debug application setup: gdb

NOTE: all paths must be valid on the remote computer

Hostname 127.0.0.1

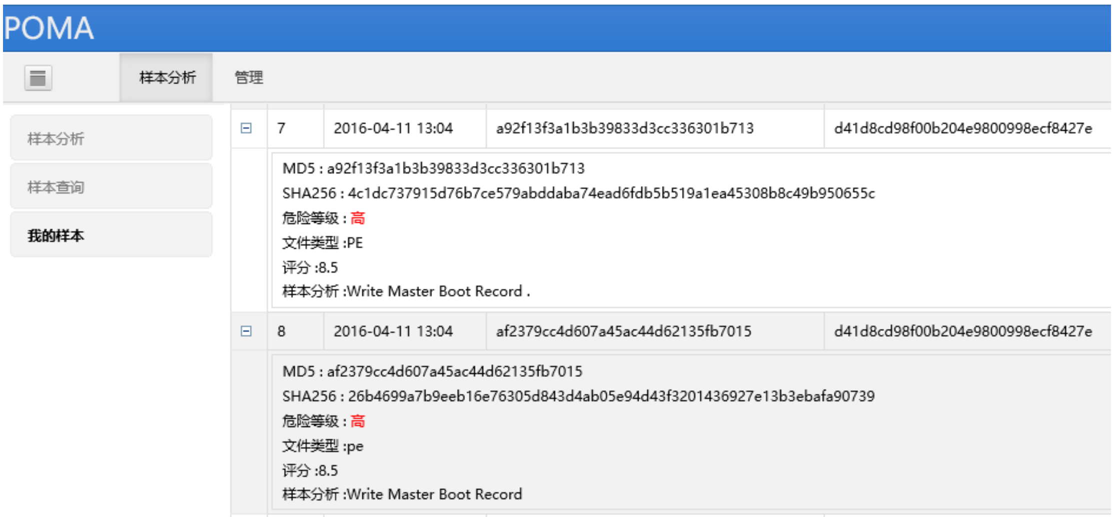
Port 8832

□ Save network settings as default

检测结果

杀毒软件	检测结果
MicroWorld-eScan	Trojan.GenericKD.3132766
nProtect	Trojan/W32.Petr.806912
CAT-QuickHeal	Trojan-Ransom.Petr.r5
McAfee	RDN/Ransom
VIPRE	Trojan.Win32.Generic!BT
K7AntiVirus	Trojan（004e1c831）
BitDefender	Trojan.GenericKD.3132766
K7GW	Trojan（004e1c831）
Cyren	W32/Petya.XMFF-8835
Symantec	Trojan.Cryptolocker.AJ
ESET-NOD32	Win32/Diskcoder.Petya.A
TrendMicro-HouseCall	Ransom_PETYA.E
Kaspersky	Trojan-Ransom.Win32.Petr.l
NANO-Antivirus	Trojan.Win32.AD.ebjjem
ViRobot	Trojan.Win32.S.Petya.806912[h]
AegisLab	Troj.Ransom.W32!c
Rising	PE:Malware.Generic/QRS!1.9E2D [F]
Ad-Aware	Trojan.GenericKD.3132766
Sophos	Troj/Petya-C
F-Secure	Trojan.GenericKD.3132766
DrWeb	Trojan.MBRlock.245
Zillya	Trojan.Petr.Win32.5
TrendMicro	Ransom_PETYA.E
McAfee-GW-Edition	BehavesLike.Win32.Downloader.bh
Emsisoft	Trojan-Ransom.Win32.Petya（A）
F-Prot	W32/Petya.G
Avira	TR/AD.Petya.Y.hhcl
Microsoft	Ransom:Win32/Petya
Arcabit	Trojan.Generic.D2FCD5E
SUPERAntiSpyware	Ransom.Petya/Variant
GData	Trojan.GenericKD.3132766
ALYac	Trojan.GenericKD.3132766
AVware	Trojan.Win32.Generic!BT
Panda	Trj/CryptoPetya.A
Tencent	Win32.Trojan.Petr.Llrb
Yandex	Trojan.Petr!
Ikarus	Trojan-Ransom.PetYa
AVG	Ransomer.LBN
Qihoo-360	Trojan.Generic

杀毒软件检测结果（检测时间：2016-04-12 07:05:29）



7	2016-04-11 13:04	a92f13f3a1b3b39833d3cc336301b713	d41d8cd98f00b204e9800998ecf8427e

MD5 : a92f13f3a1b3b39833d3cc336301b713
SHA256 : 4c1dc737915d76b7ce579abddaba74ead6fdb5b519a1ea45308b8c49b950655c
危险等级 : 高
文件类型 :PE
评分 :8.5
样本分析 :Write Master Boot Record .

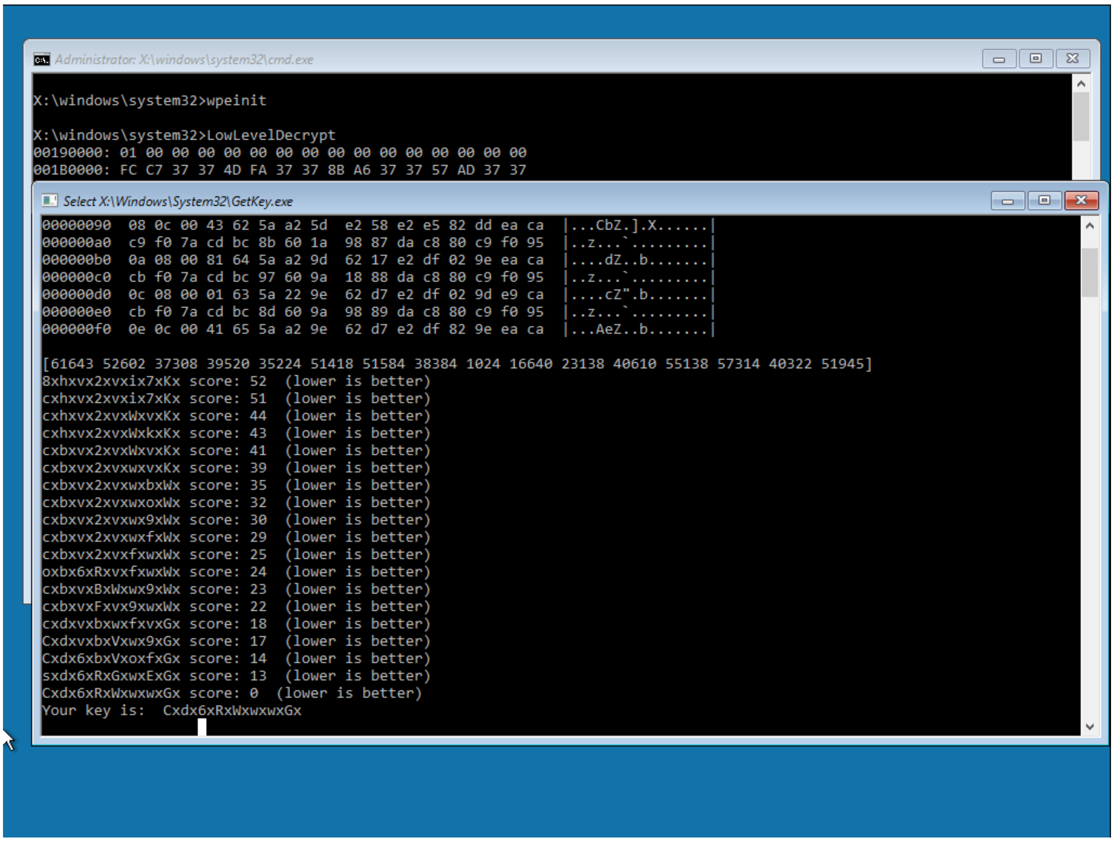
8	2016-04-11 13:04	af2379cc4d607a45ac44d62135fb7015	d41d8cd98f00b204e9800998ecf8427e

MD5 : af2379cc4d607a45ac44d62135fb7015
SHA256 : 26b4699a7b9eeb16e76305d843d4ab05e94d43f3201436927e13b3ebafa90739
危险等级 : 高
文件类型 :pe
评分 :8.5
样本分析 :Write Master Boot Record

绿盟科技 POMA 样本检测结果

数据恢复

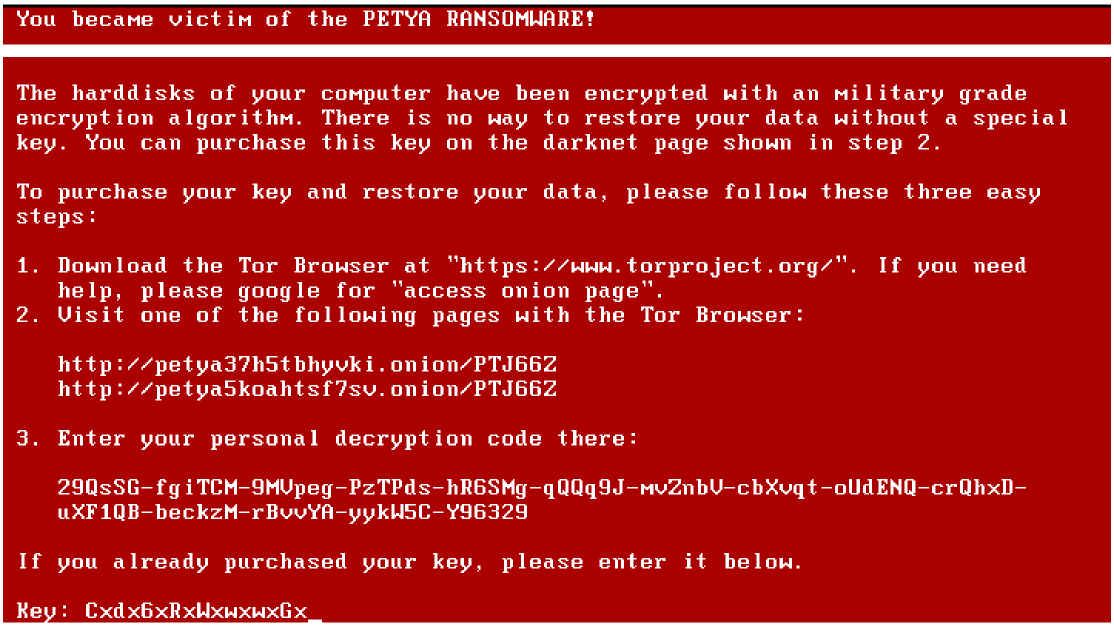
1) 从绿盟科技获取 PetyaRansomware 系统恢复光盘。
2) 从光驱启动或者制作成 U 盘启动。


Administrator: X:\windows\system32\cmd.exe
Select X:\Windows\System32\GetKey.exe

```
X:\windows\system32>wpeinit

X:\windows\system32>LowLevelDecrypt
00190000: 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
001B0000: FC C7 37 37 4D FA 37 37 8B A6 37 37 57 AD 37 37
```

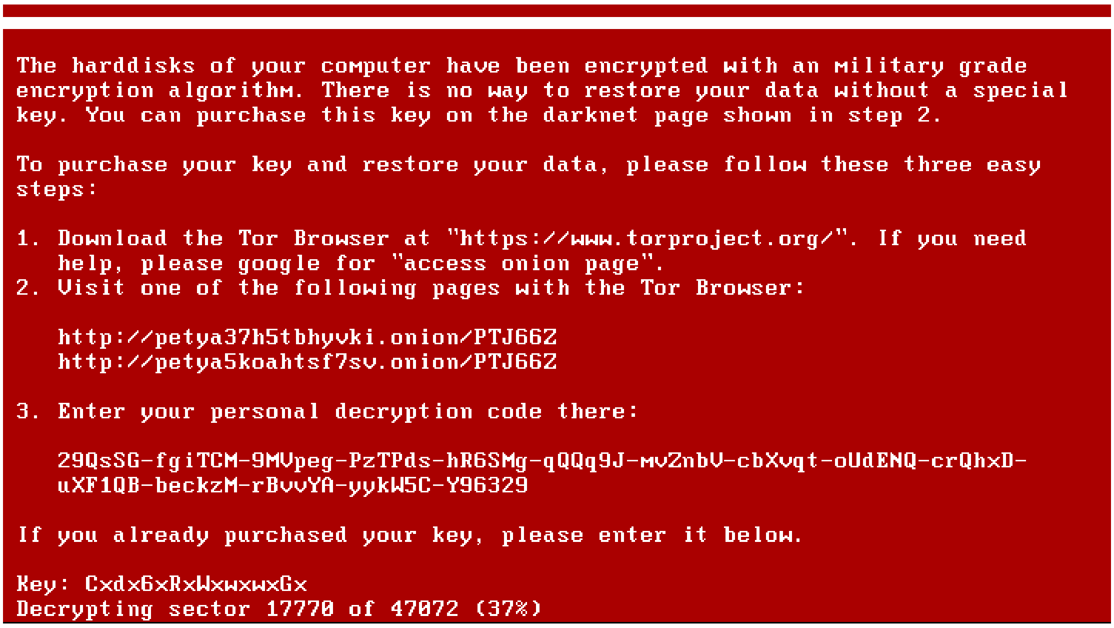
```
00000090  08 0c 00 43 62 5a a2 5d  e2 58 e2 e5 82 dd ea ca  |...CbZ.].X......|
000000a0  c9 f0 7a cd bc 8b 60 1a  98 87 da c8 80 c9 f0 95  |..z...`.........|
000000b0  0a 08 00 81 64 5a a2 9d  62 17 e2 df 02 9e ea ca  |....dZ..b.......|
000000c0  cb f0 7a cd bc 97 60 9a  18 88 da c8 80 c9 f0 95  |..z...`.........|
000000d0  0c 08 00 01 63 5a 22 9e  62 d7 e2 df 02 9d e9 ca  |....cZ".b.......|
000000e0  cb f0 7a cd bc 8d 60 9a  98 89 da c8 80 c9 f0 95  |..z...`.........|
000000f0  0e 0c 00 41 65 5a a2 9e  62 d7 e2 df 82 9e ea ca  |...AeZ..b.......|

[61643 52602 37308 39520 35224 51418 51584 38384 1024 16640 23138 40610 55138 57314 40322 51945]
8xhxvx2xvxix7xKx score: 52  (lower is better)
cxhxvx2xvxix7xKx score: 51  (lower is better)
cxhxvx2xvxWxvxKx score: 44  (lower is better)
cxhxvx2xvxWxkxKx score: 43  (lower is better)
cxbxvx2xvxWxvxKx score: 41  (lower is better)
cxbxvx2xvxwxvxKx score: 39  (lower is better)
cxbxvx2xvxwxbxWx score: 35  (lower is better)
cxbxvx2xvxwxoxWx score: 32  (lower is better)
cxbxvx2xvxwx9xWx score: 30  (lower is better)
cxbxvx2xvxwxfxWx score: 29  (lower is better)
cxbxvx2xvxfxwxWx score: 25  (lower is better)
oxbx6xRxvxfxwxWx score: 24  (lower is better)
cxbxvxBxWxwx9xWx score: 23  (lower is better)
cxbxvxFxvx9xwxWx score: 22  (lower is better)
cxdxvxbxwxfxvxGx score: 18  (lower is better)
CxdxvxbxVxwx9xGx score: 17  (lower is better)
Cxdx6xbxVxoxfxGx score: 14  (lower is better)
sxdx6xRxGxwxExGx score: 13  (lower is better)
Cxdx6xRxWxwxwxGx score: 0  (lower is better)
Your key is:  Cxdx6xRxWxwxwxGx
```

3) 记录下程序提示的 Key，并重启主机，从原始硬盘启动，在提示界面输入之前记录的 Key。



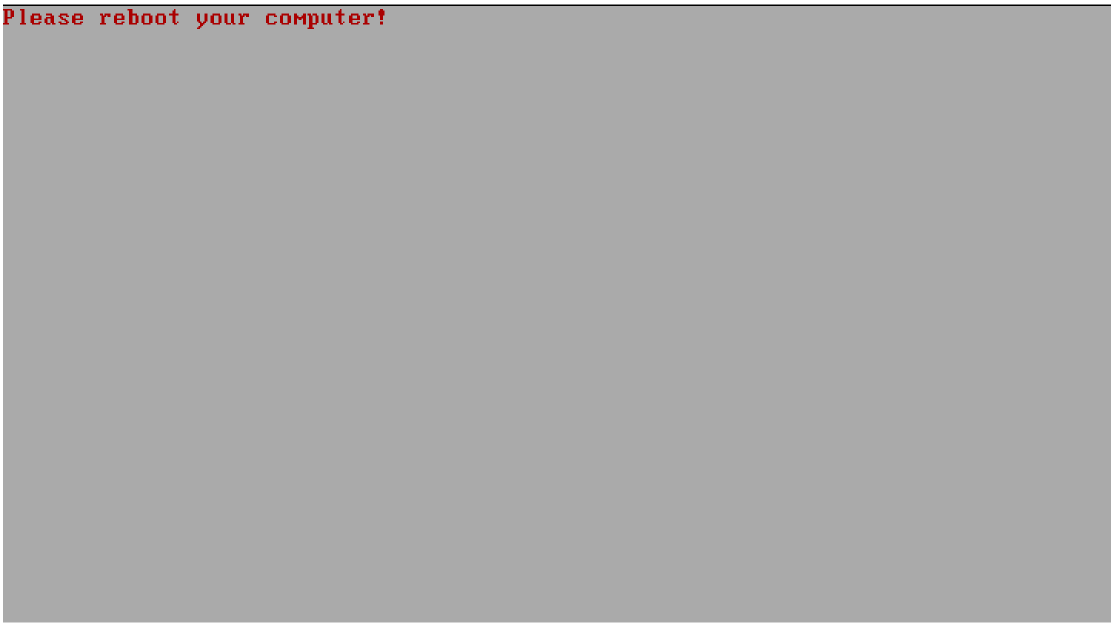
```
You became victim of the PETYA RANSOMWARE!

The harddisks of your computer have been encrypted with an military grade
encryption algorithm. There is no way to restore your data without a special
key. You can purchase this key on the darknet page shown in step 2.

To purchase your key and restore your data, please follow these three easy
steps:

1. Download the Tor Browser at "https://www.torproject.org/". If you need
   help, please google for "access onion page".
2. Visit one of the following pages with the Tor Browser:

   http://petya37h5tbhyvki.onion/PTJ66Z
   http://petya5koahtsf7sv.onion/PTJ66Z

3. Enter your personal decryption code there:

   29QsSG-fgiTCM-9MVpeg-PzTPds-hR6SMg-qQQq9J-mvZnbV-cbXvqt-oUdENQ-crQhxD-
   uXF1QB-beckzM-rBvvYA-yykW5C-Y96329

If you already purchased your key, please enter it below.

Key: Cxdx6xRxWxwxwxGx_
```

4) 输入后系统开始进行解密。



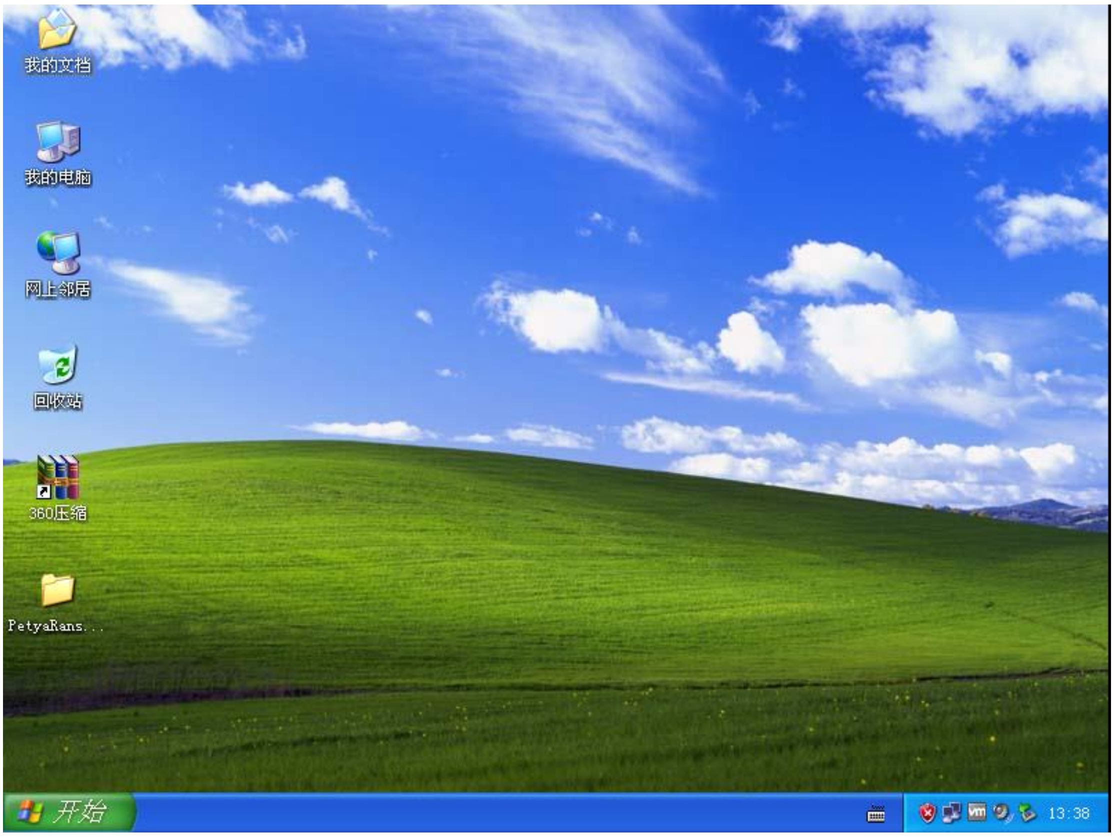
```
The harddisks of your computer have been encrypted with an military grade
encryption algorithm. There is no way to restore your data without a special
key. You can purchase this key on the darknet page shown in step 2.

To purchase your key and restore your data, please follow these three easy
steps:

1. Download the Tor Browser at "https://www.torproject.org/". If you need
   help, please google for "access onion page".
2. Visit one of the following pages with the Tor Browser:

   http://petya37h5tbhyvki.onion/PTJ66Z
   http://petya5koahtsf7sv.onion/PTJ66Z

3. Enter your personal decryption code there:

   29QsSG-fgiTCM-9MVpeg-PzTPds-hR6SMg-qQQq9J-mvZnbV-cbXvqt-oUdENQ-crQhxD-
   uXF1QB-beckzM-rBvvYA-yykW5C-Y96329

If you already purchased your key, please enter it below.

Key: Cxdx6xRxWxwxwxGx
Decrypting sector 17770 of 47072 (37%)
```

5) 解密完成后提示重新启动系统。



Please reboot your computer!

6)　重启后可以正常进入系统。



解决方案

- 针对个人用户
  1) 安装杀毒软件并更新到最新。
  2) 运行绿盟科技 PetyaRansomware 系统恢复软件。
- 针对企业用户
  1) 安装终端安全软件，并更新到最新。
  2) 绿盟科技 TAC+IPS+NGFW 联合解决方案。

3) 绿盟科技安全邮件网关。

4) 绿盟科技 PetyaRansomware 系统恢复软件。