使用的是 `aswArPot.sys` 这个驱动，非常简单，熟悉驱动编写的很快会编写出POC，是前两天看到群友发的分析文章：

https://www.aon.com/cyber-solutions/aon_cyber_labs/yours-truly-signed-av-driver-weaponizing-an-antivirus-driver/

```
然后复现了一下，简单的才可以复现，难的就不会了，想上吊了，555...
```

```
总而言之，发送`0x9988c094`这个IOCTL即可，然后`DeviceIoControl`直接给PID即可，他是一个_CLIENT_ID结构，第二个UniqueThread我们不用管，所以直接发一个PID的地址过去就好。
```

对应如下：

```
390         *(_DWORD *)a7 = result;
391         a7[1] = 0i64;
392         return result;
393       case 0x9988C094:
394         if ( a3 != 4 || !a2 )
395           goto LABEL_194;
396         result = ((__int64 (__fastcall *)(_QWORD, __int64))sub_14001DC80)(*(unsigned int *)a2, a8);
397         goto LABEL_148;
398       case 0x9988C098:
399         if ( a3 != 8 || !a2 )
400           goto LABEL_194;
401         result = sub_14001DE10(*(unsigned int *)a2, *((unsigned int *)a2 + 1));
```

```
__int64 __fastcall sub_14001DC80(unsigned int a1)
{
  NTSTATUS v1; // eax
  unsigned int v2; // ebx
  PVOID Object; // [rsp+30h] [rbp-39h] BYREF
  void *ProcessHandle; // [rsp+38h] [rbp-31h] BYREF
  struct _CLIENT_ID ClientId; // [rsp+40h] [rbp-29h] BYREF
  struct _OBJECT_ATTRIBUTES ObjectAttributes; // [rsp+50h] [rbp-19h] BYREF
  struct _KAPC_STATE ApcState; // [rsp+80h] [rbp+17h] BYREF

  memset(&ObjectAttributes.RootDirectory, 0, 20);
  ClientId.UniqueThread = 0i64;
  *(_OWORD *)&ObjectAttributes.SecurityDescriptor = 0i64;
  ClientId.UniqueProcess = (HANDLE)a1;
  ObjectAttributes.Length = 48;
  KeStackAttachProcess(qword_14004CD60, &ApcState);
  v1 = ZwOpenProcess(&ProcessHandle, 1u, &ObjectAttributes, &ClientId);
  v2 = v1 == 0;
  if ( !v1 )
  {
    if ( !ObReferenceObjectByHandle(ProcessHandle, 0, 0i64, 0, &Object, 0i64) )
    {
      switch ( dword_14004D448 )
      {
        case 0x501:
          *((_DWORD *)Object + 146) &= ~0x2000u;
          break;
        case 0x502:
          *((_DWORD *)Object + 144) &= ~0x2000u;
          break;
        case 0x600:
          *((_DWORD *)Object + 138) &= ~0x2000u;
          break;
        case 0x601:
          *((_DWORD *)Object + 156) &= ~0x2000u;
          break;
        case 0x602:
          *((_DWORD *)Object + 154) &= ~0x2000u;
          break;
      }
      ObfDereferenceObject(Object);
    }
    v2 = ZwTerminateProcess(ProcessHandle, 0);
    ZwClose(ProcessHandle);
  }
  KeUnstackDetachProcess(&ApcState);
  return v2;
}
```

调用情况：

```
sub_14001DC80 -> sub_140018FF8 -> sub_140019D54 -> sub_140014890 -> sub_140014B60 -
> sub_1400163F0 -> sub_1400216A0 -> DriverEntry
```

```
这里还有一点，可以看到原始POC在发送`0x9988c094`这个IOCTL的之前还有一个与这个驱动通信的过程使用的是`0x7299c004`.我也去尝试不先发送这个交互但是是没办法KILL掉的。
```

POC的驱动交互代码：

```
__int64 __fastcall sub_140050090(
        __int64 a1,
        unsigned int a2,
        __int64 a3,
        unsigned int a4,
        int a5,
        char a6,
        __int64 a7,
        __int64 a8,
        __int64 a9)
{
  if ( !byte_140028980 )
    sub_140050000();
  return qword_1400289A8(a1, a2, a3, a4, a5, a6, a7, a8, a9);
}
```