抄代码发现的bug

原作者：layerfsd

参考网址：https://bbs.pediy.com/thread-157472.htm

偷懒抄代码发现的问题总结：

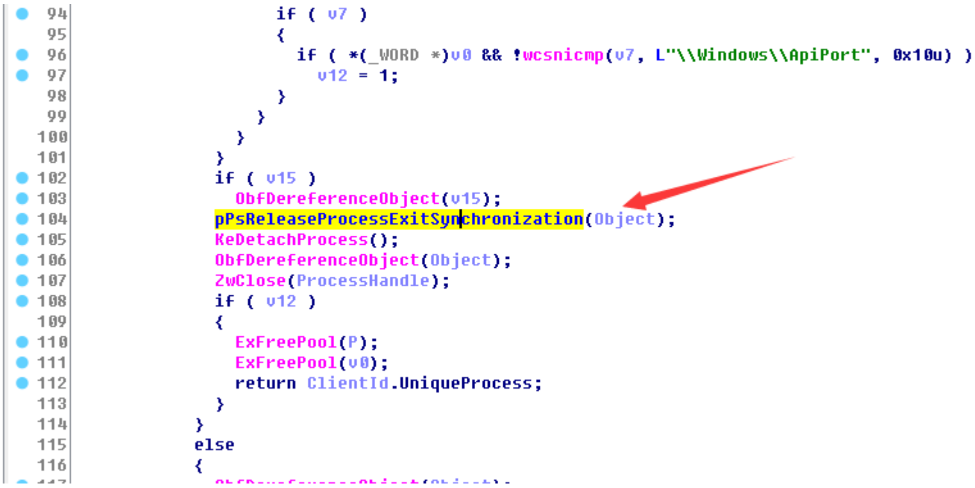
1、关于pPsReleaseProcessExitSynchronization结构的定义

解决疑问的网址：https://processhacker.sourceforge.io/doc/ntfill_8h.html

产生问题的函数：HookPort_GetApiPortProcessId()

产生疑问的地方：

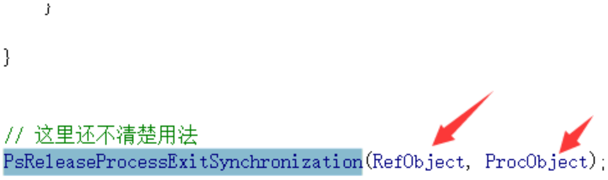
数字.sys



```
              if ( v7 )
              {
                if ( *(_WORD *)v0 && !wcsnicmp(v7, L"\\Windows\\ApiPort", 0x10u) )
                  v12 = 1;
              }
            }
          }
        }
        if ( v15 )
          ObfDereferenceObject(v15);
        pPsReleaseProcessExitSynchronization(Object);
        KeDetachProcess();
        ObfDereferenceObject(Object);
        ZwClose(ProcessHandle);
        if ( v12 )
        {
          ExFreePool(P);
          ExFreePool(v0);
          return ClientId.UniqueProcess;
        }
      }
      else
      {
```

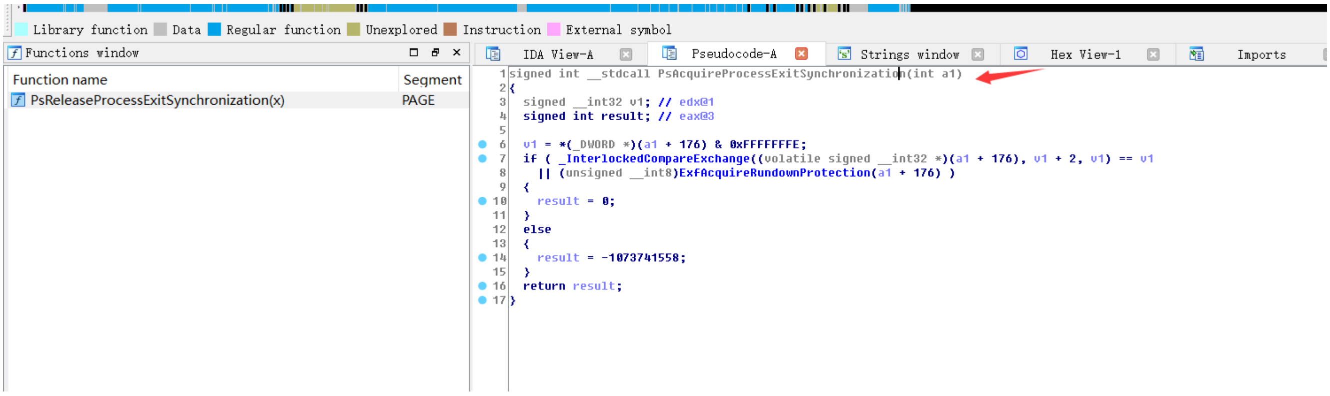
作者定义的函数类型：

typedef NTSTATUS (__stdcall *pPsReleaseProcessExitSynchronization)(HANDLE, HANDLE);



```
        }

    }


    // 这里还不清楚用法
    PsReleaseProcessExitSynchronization(RefObject, ProcObject);
```

很明显参数原型不符合，然后自己IDA分析ntkrnlpa.sys发现该函数只有一个参数



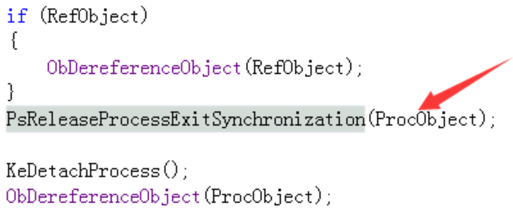
Function name　Segment

PsReleaseProcessExitSynchronization(x)　PAGE

```
signed int __stdcall PsAcquireProcessExitSynchronization(int a1)
{
  signed __int32 v1; // edx@1
  signed int result; // eax@3

  v1 = *(_DWORD *)(a1 + 176) & 0xFFFFFFFE;
  if ( _InterlockedCompareExchange((volatile signed __int32 *)(a1 + 176), v1 + 2, v1) == v1
    || (unsigned __int8)ExfAcquireRundownProtection(a1 + 176) )
  {
    result = 0;
  }
  else
  {
    result = -1073741558;
  }
  return result;
}
```

所以原型应该是如下格式：

typedef NTSTATUS(__stdcall *pPsReleaseProcessExitSynchronization)(__in PEPROCESS Process);



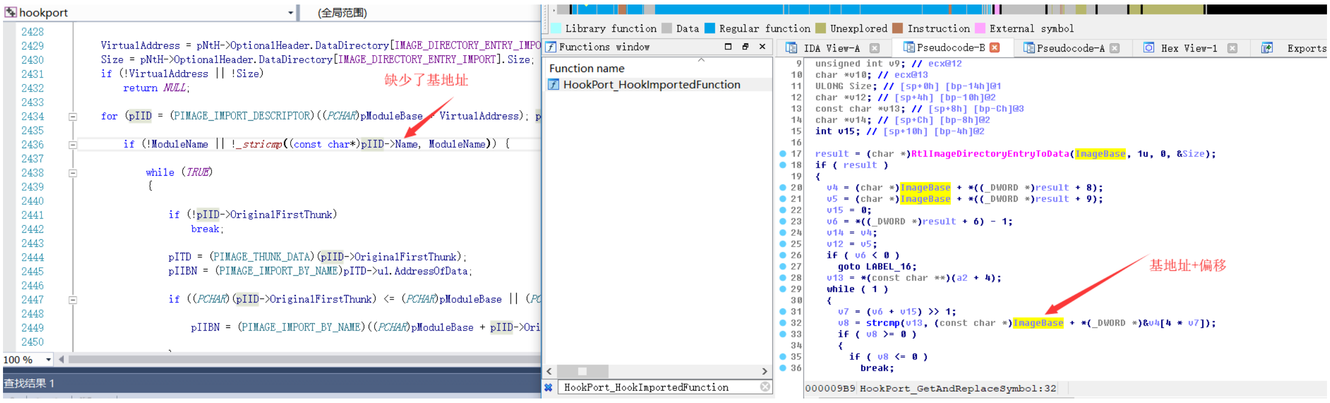
```
if (RefObject)
{
    ObDereferenceObject(RefObject);
}
PsReleaseProcessExitSynchronization(ProcObject);

KeDetachProcess();
ObDereferenceObject(ProcObject);
```

2、重温PE结构(基址+偏移)

产生问题的函数：HookPort_HookImportedFunction()

产生疑问的地方：

错误（左侧）正确（右侧）



```
    VirtualAddress = pNtH->OptionalHeader.DataDirectory[IMAGE_DIRECTORY_ENTRY_IMPO
    Size = pNtH->OptionalHeader.DataDirectory[IMAGE_DIRECTORY_ENTRY_IMPORT].Size;
    if (!VirtualAddress || !Size)
        return NULL;

    for (pIID = (PIMAGE_IMPORT_DESCRIPTOR)((PCHAR)pModuleBase + VirtualAddress); p

        if (!ModuleName || !_stricmp((const char*)pIID->Name, ModuleName)) {

            while (TRUE)
            {

                if (!pIID->OriginalFirstThunk)
                    break;

                pITD = (PIMAGE_THUNK_DATA)(pIID->OriginalFirstThunk);
                pIIBN = (PIMAGE_IMPORT_BY_NAME)pITD->u1.AddressOfData;

                if ((PCHAR)(pIID->OriginalFirstThunk) <= (PCHAR)pModuleBase || (PC

                    pIIBN = (PIMAGE_IMPORT_BY_NAME)((PCHAR)pModuleBase + pIID->Ori
```

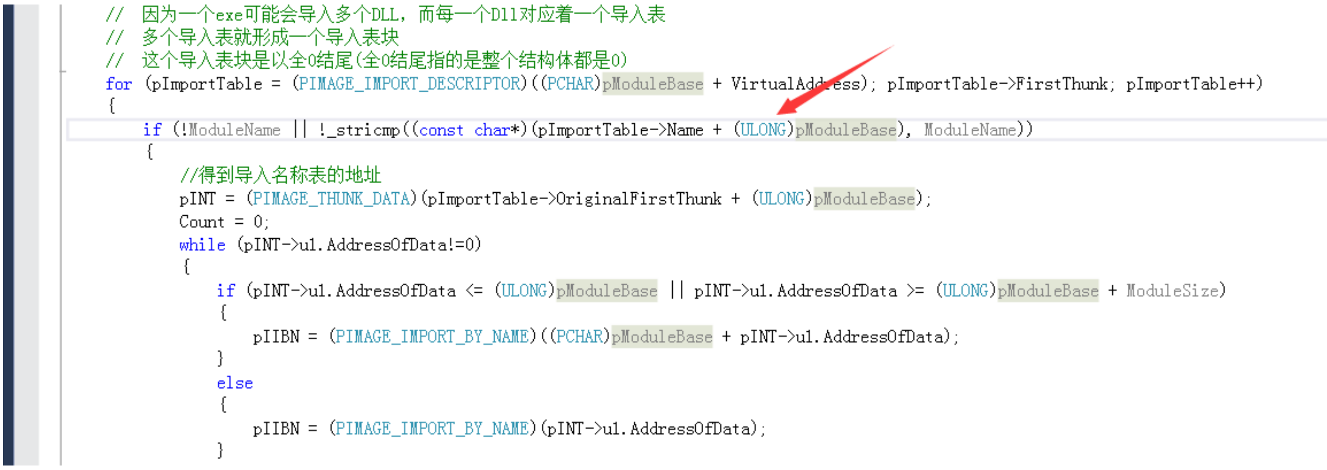
缺少了基地址

```
unsigned int v9; // ecx@12
char *v10; // ecx@13
ULONG Size; // [sp+0h] [bp-14h]@1
char *v12; // [sp+4h] [bp-10h]@2
const char *v13; // [sp+8h] [bp-Ch]@3
char *v14; // [sp+Ch] [bp-8h]@2
int v15; // [sp+10h] [bp-4h]@2

result = (char *)RtlImageDirectoryEntryToData(ImageBase, 1u, 0, &Size);
if ( result )
{
  v4 = (char *)ImageBase + *((_DWORD *)result + 8);
  v5 = (char *)ImageBase + *((_DWORD *)result + 9);
  v15 = 0;
  v6 = *((_DWORD *)result + 6) - 1;
  v14 = v4;
  v12 = v5;
  if ( v6 < 0 )
    goto LABEL_16;
  v13 = *(const char **)(a2 + 4);
  while ( 1 )
  {
    v7 = (v6 + v15) >> 1;
    v8 = strcmp(v13, (const char *)ImageBase + *(_DWORD *)&v4[4 * v7]);
    if ( v8 >= 0 )
    {
      if ( v8 <= 0 )
        break;
```

基地址+偏移

```
000009B9 HookPort_GetAndReplaceSymbol:32
```

应该修正为:



```
    //  因为一个exe可能会导入多个DLL，而每一个Dll对应着一个导入表
    //  多个导入表就形成一个导入表块
    //  这个导入表块是以全0结尾(全0结尾指的是整个结构体都是0)
    for (pImportTable = (PIMAGE_IMPORT_DESCRIPTOR)((PCHAR)pModuleBase + VirtualAddress); pImportTable->FirstThunk; pImportTable++)
    {
        if (!ModuleName || !_stricmp((const char*)(pImportTable->Name + (ULONG)pModuleBase), ModuleName))
        {
            //得到导入名称表的地址
            pINT = (PIMAGE_THUNK_DATA)(pImportTable->OriginalFirstThunk + (ULONG)pModuleBase);
            Count = 0;
            while (pINT->u1.AddressOfData!=0)
            {
                if (pINT->u1.AddressOfData <= (ULONG)pModuleBase || pINT->u1.AddressOfData >= (ULONG)pModuleBase + ModuleSize)
                {
                    pIIBN = (PIMAGE_IMPORT_BY_NAME)((PCHAR)pModuleBase + pINT->u1.AddressOfData);
                }
                else
                {
                    pIIBN = (PIMAGE_IMPORT_BY_NAME)(pINT->u1.AddressOfData);
                }
```