/*
作者:sysnap
出处:看雪论坛
连接:http://hi.baidu.com/sysnap
*/

看了 combojiang 大侠的 rootkit 专题,发现少了一个导出表钩子,既 EAT HOOK, 刚好前几天自己搞了个 IAT HOOK, 然后就把其中的代码稍做修改,于是有这篇文章,偶学的东西不久,很多东西还不知道,请多指教,呵呵

导出表钩子比导入表钩子感觉好用多,先说下原理吧,函数导入的函数的地址是再运行时候才确定的,比如我们的一个驱动程序导入了 PsGetCurrentProcessId 这个由 ntkrnlpa.exe 导出的函数,那在我们驱动程序加载运行的时候,装载程序会确定 ntkrnlpa.exe 在内存的基地址,接着遍历它的导出表,在 AddressOfNames 指向的"函数名字表"中找到 PsGetCurrentProcessId 的位置,也就是如果在 AddressOfNames[i] 中找到 PsGetCurrentProcessId,那就用 i 在 AddressOfNameOrdinals 中索引,假使得到是 X,那么 AddressOfFunctions[index]的值就是 PsGetCurrentProcessId 的 RVA 了,最后就可以知道 PsGetCurrentProcessId 在内存的值是 MM=ntkrnlpa.exe 在内存的基地址 +PsGetCurrentProcessId 的 RVA,然后转载程序就把这个值写到我们驱动程序的 IAT 中,好了知道这些后,EAT HOOK 就是修改 PsGetCurrentProcessId 的 RVA,使得 PsGetCurrentProcessId 的 RVA(修改后的)+ntkrnlpa.exe 在内存的基地址=我们自己函数的值,这样装载程序会把我们的函数的地址写入那些调用了 PsGetCurrentProcessId 的驱动程序的 IAT,那么当那些驱动程序调用 PsGetCurrentProcessId 时,实际上是执行了我们自己的函数...呵呵.是不是比 IAT HOOK 更好用呢

EAT HOOK 可以用来监控系统函数的调用情况,比如我们 EAT HOOK 了 PsGetCurrentProcessId,那谁调用该函数我们就知道了,其实知道了 EAT HOOK 原理后,我们可以修改函数名字表,比如把 PsGetCurrentProcessId 改成其它名字,这样装载程序遍历"函

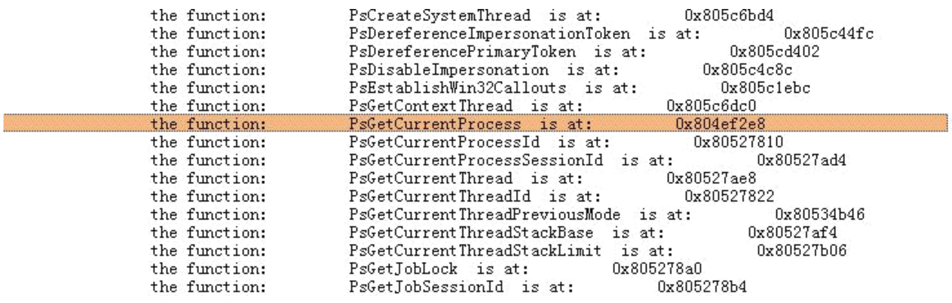
数名字表"就找不到匹对的名字,那驱动程序就宣告装载失败,详细代码请看
<<利用导出表来禁止一些驱动程序的加载>> http://bbs.pediy.com/showthread.php?

t=62531
　那怎么防止 EAT HOOK, 一个方法是自己定位函数在内存地址,请看下面的代码,用于枚举

ntkrnlpa.exe 导出函数在内存的地址

```
   VOID ListKernelFunctionAndAddress()
{
HANDLE      hMod;
      PVOID BaseAddress = NULL;
IMAGE_DOS_HEADER * dosheader;
IMAGE_OPTIONAL_HEADER * opthdr;
      PIMAGE_EXPORT_DIRECTORY exports;

      USHORT      index=0 ;
ULONG addr, i;

PVOID FuncNameRVA;
      PUCHAR pFuncName = NULL;
PULONG pAddressOfFunctions,pAddressOfNames,pAddressOfNameOrdinals;

BaseAddress= GetDriverBaseAdress("ntkrnlpa.exe");
      DbgPrint("Map BaseAddress is:%x\n",BaseAddress);
      hMod = BaseAddress;

dosheader = (IMAGE_DOS_HEADER *)hMod;
      opthdr =(IMAGE_OPTIONAL_HEADER *) ((BYTE*)hMod+dosheader->e_lfanew+24);
      exports = (PIMAGE_EXPORT_DIRECTORY)((BYTE*)dosheader+ opthdr-

>DataDirectory[IMAGE_DIRECTORY_ENTRY_EXPORT].VirtualAddress);

      pAddressOfFunctions=(ULONG*)((BYTE*)hMod+exports->AddressOfFunctions);
pAddressOfNames=(ULONG*)((BYTE*)hMod+exports->AddressOfNames);
pAddressOfNameOrdinals=(USHORT*)((BYTE*)hMod+exports->AddressOfNameOrdinals);


      for (i = 0; i < exports->NumberOfNames; i++)
{

index=pAddressOfNameOrdinals[i];
addr=pAddressOfFunctions[index];
pFuncName = (PUCHAR)( (BYTE*)hMod + pAddressOfNames[i]);
addr = pAddressOfFunctions[index];
DbgPrint("the function:          %s is at:          0x%x\n",pFuncName,addr+

(BYTE*)hMod);

}

}
```

运行后:



```
the function:        PsCreateSystemThread  is at:        0x805c6bd4
the function:        PsDereferenceImpersonationToken  is at:        0x805c44fc
the function:        PsDereferencePrimaryToken  is at:        0x805cd402
the function:        PsDisableImpersonation  is at:        0x805c4c8c
the function:        PsEstablishWin32Callouts  is at:        0x805c1ebc
the function:        PsGetContextThread  is at:        0x805c6dc0
the function:        PsGetCurrentProcess  is at:        0x804ef2e8
the function:        PsGetCurrentProcessId  is at:        0x80527810
the function:        PsGetCurrentProcessSessionId  is at:        0x80527ad4
the function:        PsGetCurrentThread  is at:        0x80527ae8
the function:        PsGetCurrentThreadId  is at:        0x80527822
the function:        PsGetCurrentThreadPreviousMode  is at:        0x80534b46
the function:        PsGetCurrentThreadStackBase  is at:        0x80527af4
the function:        PsGetCurrentThreadStackLimit  is at:        0x80527b06
the function:        PsGetJobLock  is at:        0x805278a0
the function:        PsGetJobSessionId  is at:        0x805278b4
```


好了,讲了这么多时候进去正题,怎样 EAT HOOK,这里我们以 HOOK ntkrnlpa.exe 导出的

PsGetCurrentProcessId

首先我们是定位 ntkrnlpa.exe 被加载在内存中的什么地方,那就写一个函数吧,

```
PVOID GetModlueBaseAdress(char* ModlueName)
{
     ULONG size,index;
     PULONG buf;
     NTSTATUS status;
     PSYSTEM_MODULE_INFORMATION module;
     PVOID driverAddress=0;

     ZwQuerySystemInformation(SystemModuleInformation,&size, 0, &size);
     if(NULL==(buf = (PULONG)ExAllocatePool(PagedPool, size)))
     {
          DbgPrint("failed alloc memory failed  \n");
          return 0;
     }
     status=ZwQuerySystemInformation(SystemModuleInformation,buf, size , 0);
     if(!NT_SUCCESS( status ))
     {
         DbgPrint("failed    query\n");
         return 0;
     }
     module = (PSYSTEM_MODULE_INFORMATION)(( PULONG )buf + 1);
     for (index = 0; index < *buf; index++)
     if (_stricmp(module[index].ImageName + module[index].ModuleNameOffset,

ModlueName) == 0)
     {
          driverAddress = module[index].Base;
          DbgPrint("Module found at:%x\n",driverAddress);
     }
     ExFreePool(buf);
     return driverAddress;
}
```

自己添加点测试代码编译下,没什么问题,这样我们就完成了第一个问题

接着是写自己的函数了,就是替换 PsGetCurrentProcessId 的函数,这里我们很简单的输出点内容就可以了

```
ULONG g_OriginalPsGetCurrentProcessId;
typedef HANDLE (*PSGETCURRENTPROCESSID)();

HANDLE
MyPsGetCurrentProcessId()

{
    HANDLE handle;
    DbgPrint("HOOK_PsGetCurrentProcessId called!\n");
    handle =((PSGETCURRENTPROCESSID)(g_OriginalPsGetCurrentProcessId))();
    return handle;

}
```

好了,那就开始写安装钩子程序吧,因为在卸在钩子时需要用到一些变量,这里我们就把安装和卸载写成一个函数就可以了,注意 IN unsigned int test,传入 1 表示安装钩子,否则表示卸载,IN PCSTR funName 这里我们传入 PsGetCurrentProcessId,好了请看代码

```
VOID StartHook_And_Unhook(IN PCSTR funName, IN unsigned int test)
{
    HANDLE     hMod;
```

```
        PUCHAR BaseAddress = NULL;
        IMAGE_DOS_HEADER * dosheader;
        IMAGE_OPTIONAL_HEADER * opthdr;
        PIMAGE_EXPORT_DIRECTORY exports;

        USHORT      index=0 ;
        ULONG addr ,i;


        PUCHAR pFuncName = NULL;
        PULONG pAddressOfFunctions,pAddressOfNames;
        PUSHORT pAddressOfNameOrdinals;

        BaseAddress=    GetModlueBaseAdress("ntkrnlpa.exe");
        DbgPrint("Map BaseAddress is:%x\n",BaseAddress);
        hMod = BaseAddress;



        dosheader = (IMAGE_DOS_HEADER *)hMod;
        opthdr =(IMAGE_OPTIONAL_HEADER *) ((BYTE*)hMod+dosheader->e_lfanew+24);
        exports = (PIMAGE_EXPORT_DIRECTORY)((BYTE*)dosheader+ opthdr-

>DataDirectory[IMAGE_DIRECTORY_ENTRY_EXPORT].VirtualAddress);

        pAddressOfFunctions=(ULONG*)((BYTE*)hMod+exports->AddressOfFunctions);
        pAddressOfNames=(ULONG*)((BYTE*)hMod+exports->AddressOfNames);


        pAddressOfNameOrdinals=(USHORT*)((BYTE*)hMod+exports-

>AddressOfNameOrdinals);



        for (i = 0; i < exports->NumberOfNames; i++)
{
index=pAddressOfNameOrdinals[i];
pFuncName = (PUCHAR)( (BYTE*)hMod + pAddressOfNames[i]);
if (_stricmp( (char*)pFuncName,funName) == 0)
{
addr=pAddressOfFunctions[index];
break;
}
```

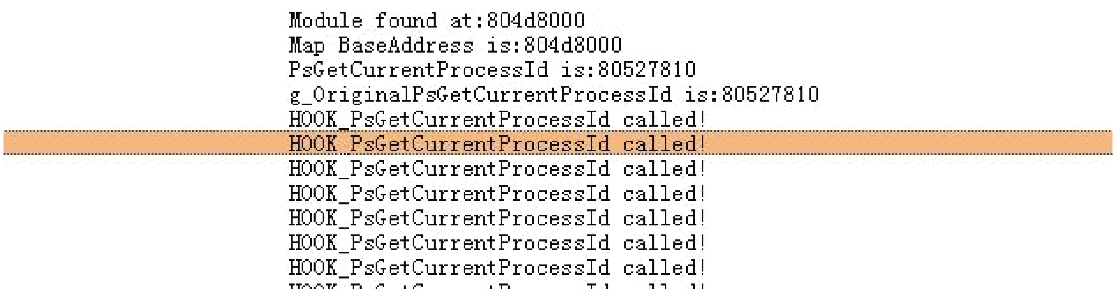
```
}


    if(test==1)     {

    _asm
    {
        CLI
        MOV     EAX, CR0
        AND EAX, NOT 10000H
        MOV     CR0, EAX
    }

DbgPrint("PsGetCurrentProcessId is:%x\n",(PUCHAR)hMod + pAddressOfFunctions

[index]);
pAddressOfFunctions[index] = ( PCHAR )MyPsGetCurrentProcessId - BaseAddress;
DbgPrint("g_OriginalPsGetCurrentProcessId is:%

x\n",g_OriginalPsGetCurrentProcessId);
g_OriginalPsGetCurrentProcessId=   (PUCHAR)hMod + pAddressOfFunctions[index] ;
_asm
    {
        MOV     EAX, CR0
        OR  EAX, 10000H
        MOV     CR0, EAX
        STI
    }
}

    else
    {
      _asm
    {
        CLI
        MOV     EAX, CR0
        AND EAX, NOT 10000H
        MOV     CR0, EAX
    }

pAddressOfFunctions[index] = ( PCHAR )g_OriginalPsGetCurrentProcessId -

BaseAddress;
```

```
_asm
    {
        MOV      EAX, CR0
        OR  EAX, 10000H
        MOV      CR0, EAX
        STI
    }



    }



}
```

好了,基本框架就差不多了,接着就是一些结构的声明,我们把它放在 hookiat.h 这个头文件

里,因为很长就不帖了,可以在附件里看,上面的代码很多地方不是很好,需要自己修改,不

保证在你机器不蓝,呵呵,学习靠思考,在代码里我修改了一处地方,自己改正下再编译吧,
运行之后:



```
Module found at:804d8000
Map BaseAddress is:804d8000
PsGetCurrentProcessId is:80527810
g_OriginalPsGetCurrentProcessId is:80527810
HOOK_PsGetCurrentProcessId called!
HOOK_PsGetCurrentProcessId called!
HOOK_PsGetCurrentProcessId called!
HOOK_PsGetCurrentProcessId called!
HOOK_PsGetCurrentProcessId called!
HOOK_PsGetCurrentProcessId called!
HOOK_PsGetCurrentProcessId called!
```