在笔者上一篇文章《驱动开发：内核注册并监控对象回调》介绍了如何运用ObRegisterCallbacks注册进程与线程回调，并通过该回调实现了拦截指定进行运行的效果，本章LyShark将带大家继续探索一个新的回调注册函数，PsSetLoadImageNotifyRoutine常用于注册LoadImage映像监视，当有模块被系统加载时则可以第一时间获取到加载模块信息，需要注意的是该回调函数内无法进行拦截，如需要拦截则需写入返回指令这部分内容将在下一章进行讲解，本章将主要实现对模块的监视功能。

监视模块加载与卸载需要费别使用两个函数，这两个函数的参数传递都是自己的回调地址。

- PsSetLoadImageNotifyRoutine 设置回调
- PsRemoveLoadImageNotifyRoutine 移除回调

此处MyLySharkLoadImageNotifyRoutine回调地址必须有三个参数传递组成，其中FullImageName代表完整路径，ModuleStyle代表模块类型，一般来说ModuleStyle=0表示加载SYS驱动，如果ModuleStyle=1则表示加载的是DLL，最后一个参数ImageInfo则是映像的详细参数结构体。

```
VOID MyLySharkLoadImageNotifyRoutine(PUNICODE_STRING FullImageName, HANDLE
ModuleStyle, PIMAGE_INFO ImageInfo)
```

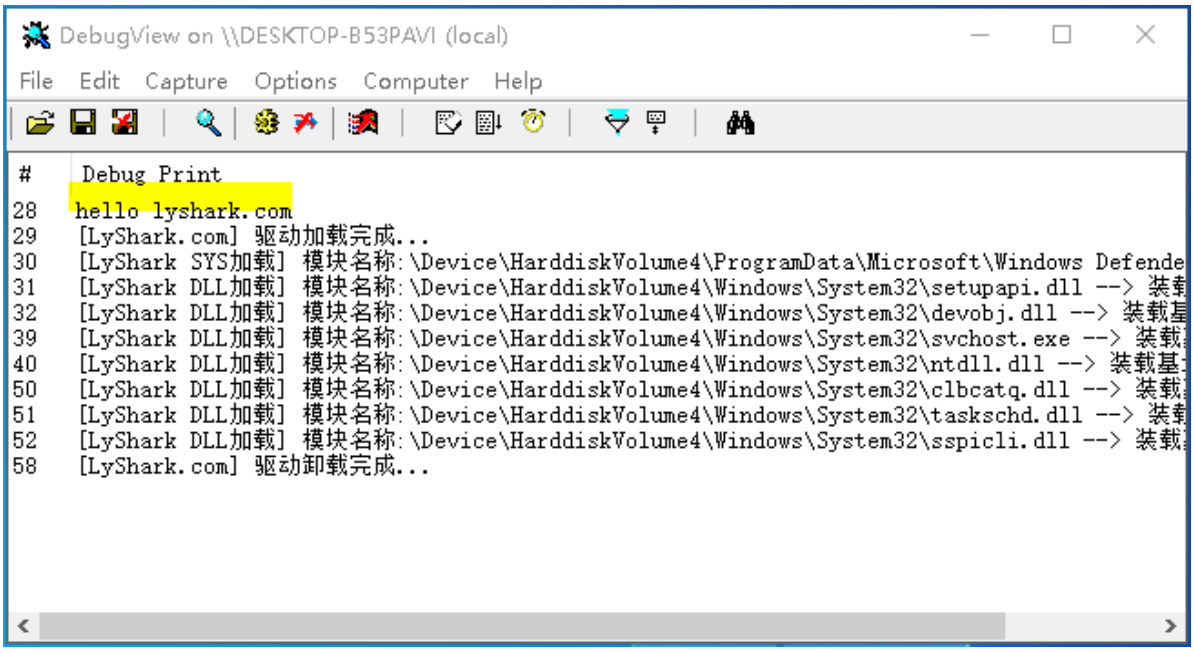
那么如何实现监视映像加载呢，来看如下完整代码片段，首先PsSetLoadImageNotifyRoutine注册回调，当有模块被加载则自动执行MyLySharkLoadImageNotifyRoutine回调函数，其内部首先判断ModuleStyle得出是什么类型的模块，然后再通过GetDriverEntryByImageBase拿到当前进程详细参数并打印输出。

```
// 署名权
// right to sign one's name on a piece of work
// PowerBy: LyShark
// Email: me@lyshark.com
#include <ntddk.h>
#include <ntimage.h>

// 未导出函数声明
PUCHAR PsGetProcessImageFileName(PEPROCESS pEProcess);

// 获取到镜像装载基地址
PVOID GetDriverEntryByImageBase(PVOID ImageBase)
{
    PIMAGE_DOS_HEADER pDOSHeader;
    PIMAGE_NT_HEADERS64 pNTHeader;
    PVOID pEntryPoint;
    pDOSHeader = (PIMAGE_DOS_HEADER)ImageBase;
    pNTHeader = (PIMAGE_NT_HEADERS64)((ULONG64)ImageBase + pDOSHeader-
>e_lfanew);
    pEntryPoint = (PVOID)((ULONG64)ImageBase + pNTHeader-
>OptionalHeader.AddressOfEntryPoint);
    return pEntryPoint;
}

// 获取当前进程名
UCHAR* GetCurrentProcessName()
{
    PEPROCESS pEProcess = PsGetCurrentProcess();
    if (NULL != pEProcess)
```

```
    {
        UCHAR *lpszProcessName = PsGetProcessImageFileName(pEProcess);
        if (NULL != lpszProcessName)
        {
            return lpszProcessName;
        }
    }
    return NULL;
}

// 设置自己的回调函数
VOID MyLySharkLoadImageNotifyRoutine(PUNICODE_STRING FullImageName, HANDLE
ModuleStyle, PIMAGE_INFO ImageInfo)
{
    PVOID pDrvEntry;

    // MmIsAddress 验证地址可用性
    if (FullImageName != NULL && MmIsAddressValid(FullImageName))
    {
        // ModuleStyle为零表示加载sys
        if (ModuleStyle == 0)
        {
            // 得到装载主进程名
            UCHAR *load_name = GetCurrentProcessName();
            pDrvEntry = GetDriverEntryByImageBase(ImageInfo->ImageBase);
            DbgPrint("[LyShark SYS加载] 模块名称:%wZ --> 装载基址:%p --> 镜像长度: %d
--> 装载主进程: %s \n", FullImageName, pDrvEntry, ImageInfo->ImageSize,
load_name);
        }
        // ModuleStyle非零表示加载DLL
        else
        {
            // 得到装载主进程名
            UCHAR *load_name = GetCurrentProcessName();
            pDrvEntry = GetDriverEntryByImageBase(ImageInfo->ImageBase);
            DbgPrint("[LyShark DLL加载] 模块名称:%wZ --> 装载基址:%p --> 镜像长度: %d
--> 装载主进程: %s \n", FullImageName, pDrvEntry, ImageInfo->ImageSize,
load_name);
        }
    }
}

VOID UnDriver(PDRIVER_OBJECT driver)
{

PsRemoveLoadImageNotifyRoutine((PLOAD_IMAGE_NOTIFY_ROUTINE)MyLySharkLoadImageNot
ifyRoutine);
    DbgPrint("[LyShark.com] 驱动卸载完成...");
}

NTSTATUS DriverEntry(IN PDRIVER_OBJECT Driver, PUNICODE_STRING RegistryPath)
{
    DbgPrint("hello lyshark.com \n");
```

```
PsSetLoadImageNotifyRoutine((PLOAD_IMAGE_NOTIFY_ROUTINE)MyLySharkLoadImageNotify
Routine);
    DbgPrint("[LyShark.com] 驱动加载完成...");
    Driver->DriverUnload = UnDriver;
    return STATUS_SUCCESS;
}
```

运行这个驱动程序，则会输出被加载的驱动详细参数。



> 作者： 王瑞 (LyShark)
>
> 作者邮箱： me@lyshark.com