上一节课讲了使用 ObRegisterCallbacks 实现保护进程，其实稍微 PATCH 下内核，这个函数还能实现文件操作监视。但可惜只能在 WIN7X64 上用。因为在 WIN7X64 上 PATCH 对象结构的成员（ObjectType->TypeInfo.SupportsObjectCallbacks）是合法的，在 WIN8X64 以及之后系统上会触发 PATCHGUARD。

要监控文件，首先要文件对象支持对象回调：

```
//参数传入 IoFileObjectType
VOID EnableObType(POBJECT_TYPE ObjectType)
{
	PMY_OBJECT_TYPE myobtype = (PMY_OBJECT_TYPE)ObjectType;
	myobtype->TypeInfo.SupportsObjectCallbacks = 1;
}
```

剩下的地方就和上一节课的代码差不多了，不同的是注册参数那里设置的对象类为 IoFileObjectType：

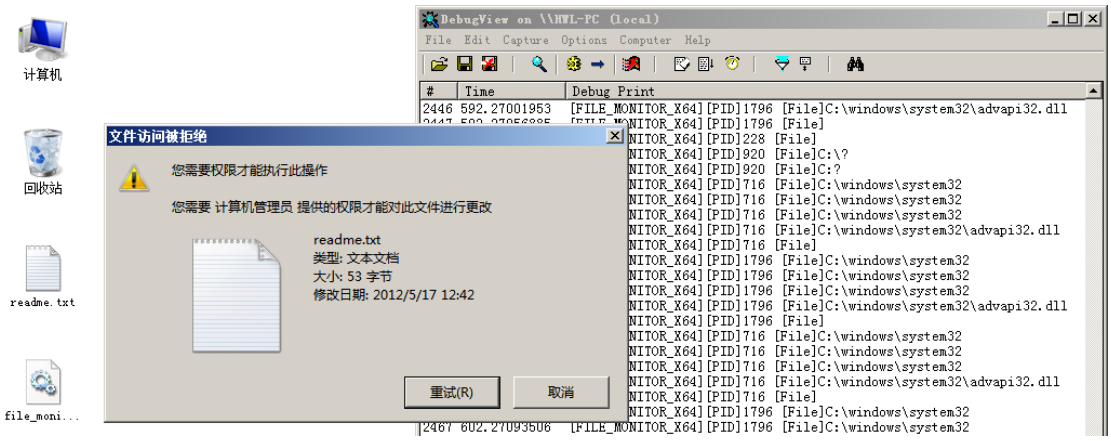
```
	// init callbacks
	memset(&obReg, 0, sizeof(obReg));
	obReg.Version = ObGetFilterVersion();
	obReg.OperationRegistrationCount = 1;
	obReg.RegistrationContext = NULL;
	RtlInitUnicodeString(&obReg.Altitude, L"321000");
	obReg.OperationRegistration = &opReg;
	memset(&opReg, 0, sizeof(opReg));
	opReg.ObjectType = IoFileObjectType;
	opReg.Operations = OB_OPERATION_HANDLE_CREATE|OB_OPERATION_HANDLE_DUPLICATE;
	opReg.PreOperation = (POB_PRE_OPERATION_CALLBACK)&preCall;
	// register callbacks
	status = ObRegisterCallbacks(&obReg, &obHandle);
```

接下来就是处理的不同了。不同的地方在于，OperationInformation->Object 得到的对象是 FILE_OBJECT（对 FILE_OBJECT 的处理要非常谨慎，稍有不慎就会蓝屏）。同理，对权限作出处理，即可实现阻止读写文件。

```
OB_PREOP_CALLBACK_STATUS preCall(PVOID RegistrationContext, POB_PRE_OPERATION_INFORMATION
OperationInformation)
{
	UNICODE_STRING DosName;
	PFILE_OBJECT fileo = OperationInformation->Object;
	HANDLE CurrentProcessId = PsGetCurrentProcessId();
	UNREFERENCED_PARAMETER(RegistrationContext);
	if( OperationInformation->ObjectType!=*IoFileObjectType )
		return OB_PREOP_SUCCESS;
	//过滤无效指针
	if(	fileo->FileName.Buffer==NULL ||
		!MmIsAddressValid(fileo->FileName.Buffer) ||
```

```
            fileo->DeviceObject==NULL                    ||
            !MmIsAddressValid(fileo->DeviceObject)       )
            return OB_PREOP_SUCCESS;
        //过滤无效路径
        if( !_wcsicmp(fileo->FileName.Buffer,L"\\Endpoint")||
            !_wcsicmp(fileo->FileName.Buffer,L"?")       ||
            !_wcsicmp(fileo->FileName.Buffer,L"\\.\\.")  ||
            !_wcsicmp(fileo->FileName.Buffer,L"\\"))
            return OB_PREOP_SUCCESS;
        //阻止访问 readme.txt
        if(wcsstr(_wcslwr(fileo->FileName.Buffer),L"readme.txt"))
        {
            if (OperationInformation->Operation == OB_OPERATION_HANDLE_CREATE)
            {
                OperationInformation->Parameters->CreateHandleInformation.DesiredAccess=0;
            }
            if(OperationInformation->Operation == OB_OPERATION_HANDLE_DUPLICATE)
            {
                OperationInformation->Parameters->DuplicateHandleInformation.DesiredAccess=0;
            }
        }
        RtlVolumeDeviceToDosName(fileo->DeviceObject, &DosName);
        DbgPrint("[FILE_MONITOR_X64][PID]%ld  [File]%wZ%wZ\n", (ULONG64)CurrentProcessId, &DosName,
&fileo->FileName);
        return OB_PREOP_SUCCESS;
}
```

效果如图所示：



课后作业：把文件操作的详细信息（包括权限、文件名等）打印出来，并实现只阻止写入文件不阻止读取文件。