剖析Windows系统服务调用机制(下) PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/140/2021_2022__E5_89_96_E6_9E_90Wind_c100_140137.htm 3. 研究系统内部机制 微软提供的Windows操作系统是一个“ 封闭 ”的系统，很多内部资料都没有公布，我们可以通过Hook技术来探测系统的内部数据结构和运行机制，学习操作系统内部的操作方式。基于Hook的Windows内核黑客技术(Kernel Hacking)是非常之流行和有效，在我们探测系统的一些未公开，未文档化的技术细节时我们都可以使用钩子技术。 4. 其他 其他如我们要调试一个非常麻烦的程序时就可以使用Hook技术，这样就可以更好的帮助我们追踪系统的行动，更好的了解程序内部的执行过程。同样，为了获取系统的一些特殊性能数据，我们也可以在特定的情况下使用Hook技术。 七> Hook系统服务调用的实现 在此我们讨论Hook的对象仅限于由Windows 2000的ntoskrnl.exe提供的系统服务调用。Windows 2000系统服务调用为内核模式的代码，所以我们必须书写设备驱动程序来访问系统服务调度表。如果你对Windows 2000下基本设备驱动程序的书写不太清楚，请查阅相关的书籍，此处不做介绍。我们先回顾一下Win32内核API的实现流程。 Windows 2000系统服务调用向用户提供了经过包装的用户模式的函数接口(由NTDLL.dll提供)。当Kernel32.dll/Advapi32.dll中的函数执行时，先调用NTDLL.dll中对应的相关接口，经过参数检查后使用int 0x2e指令进入内核模式，传递相关的服务号和参数列表。在ntoskrnl.exe中维护着两个表系统服务调度表(System Service Dispath Table)和系统服务参数表(System Service

Parameter Table)，其中int 0x2e指令就是通过服务号在SSDT中查询相关系统服务程序指针的。现在我们已经清楚了每个系统服务调用都对应一个服务号，同时也对应一个服务程序的地址！如果我们修改SSDT中的某个系统服务程序的入口地址为指向我们自定义的函数地址，在执行完我们的代码后再执行原始系统服务地址处的代码，这不就实现了对系统服务调用的了Hook吗？ 对我们来说，定位系统服务调度表是实现Hook的关键。在Windows 2000中有一个未公开的由ntoskrnl.exe导出的单元：KeServiceDescriptorTable，我们可以通过它来完成对SSDT的访问与修改
。KeServiceDescriptorTable对应于一个数据结构，定义如下：
typedef struct SystemServiceDescriptorTable { UINT *ServiceTableBase. UINT *ServiceCounterTableBase. UINT NumberOfService. UCHAR *ParameterTableBase. }SystemServiceDescriptorTable,*PSystemServiceDescriptorTable.
其中ServiceTableBase指向系统服务程序的地址
，ParameterTableBase则指向SSPT中的参数地址，它们都包含了NumberOfService这么多个单元。我们只要由KeServiceDescriptorTable找到了我们关注的系统服务调用程序，就可以修改它的ServiceTableBase参数来实现对相关系统服务调用的Hook了！ 八> T-ProcMon-1.0 关键源码分析 1. 基于CUI的用户模式控制程序 由于在此之前我已经对Win32的系统服务进行了详细的介绍，现在就不做多说了，大家如果有什么疑问请参阅我以前写的文章，你可以到FZ5FZ的主页(http://www.fz5fz.org/)阅读相关文章，或下载相关源代码。 2. 基于设备驱动的Hook代码 定义在用户模式与内核模式程序

间通信的命令代码： #define PROCMON_MONITOR
(ULONG)
CTL_CODE(FILE_DEVICE_PROCMON,0x01,METHOD_BUFF
ERED,FILE_ANY_ACCESS) #define PROCMON_HIDDEN
(ULONG)
CTL_CODE(FILE_DEVICE_PROCMON,0x02,METHOD_BUFF
ERED,FILE_ANY_ACCESS) #define PROCMON_HOOK
(ULONG)
CTL_CODE(FILE_DEVICE_PROCMON,0x03,METHOD_BUFF
ERED,FILE_ANY_ACCESS) #define PROCMON_UNHOOK
(ULONG)
CTL_CODE(FILE_DEVICE_PROCMON,0x04,METHOD_BUFF
ERED,FILE_ANY_ACCESS) 将KeServiceDescriptorTable与相关数据结构联系起来，定义系统调用： __declspec(dllimport)
ServiceDescriptorTableEntry KeServiceDescriptorTable. #define
SYSCALL(_function)
KeServiceDescriptorTable.ServiceTableBase[*(PULONG)((PUCH
AR)_function 1)] 