系统进入0环后，3环的各种寄存器保存在——Trap_Frame结构体中（0环结构体 ）

3环fs为teb，0环fs为KPCR

KiSysTemService 通过中断门进入0环 通过tss任务段保存esp ss段

KiFastCallEntry 快速调用

系统服务表

系统服务表 API调用

如何访问系统服务表

dd KeServiceDescriptorTable

共计四张表

每一张表都是系统服务表

```
dd  KeServiceDescriptorTable
8053f180  80502030 00000000 0000011c 805024a4
8053f190  00000000 00000000 00000000 00000000
8053f1a0  00000000 00000000 00000000 00000000
8053f1b0  00000000 00000000 00000000 00000000
8053f1c0  00002710 bf80db0e 00000000 00000000
8053f1d0  f7ab6a80 f724f9e0 85ec50f0 806e0f40
8053f1e0  00000000 00000000 fdfecf0c ffffffff
8053f1f0  fbba9f4c 01d238a1 00000000 00000000
```

表内成员

第一个成员表示 函数地址表（Ntoskrl导出的内核函数）

第二个成员表示 系统服务表被访问了多少次

第三个成员表示 提供了多少个函数

第四个成员表示 每一个函数的参数个数

Dd KeServiceDescriptorTableShadow

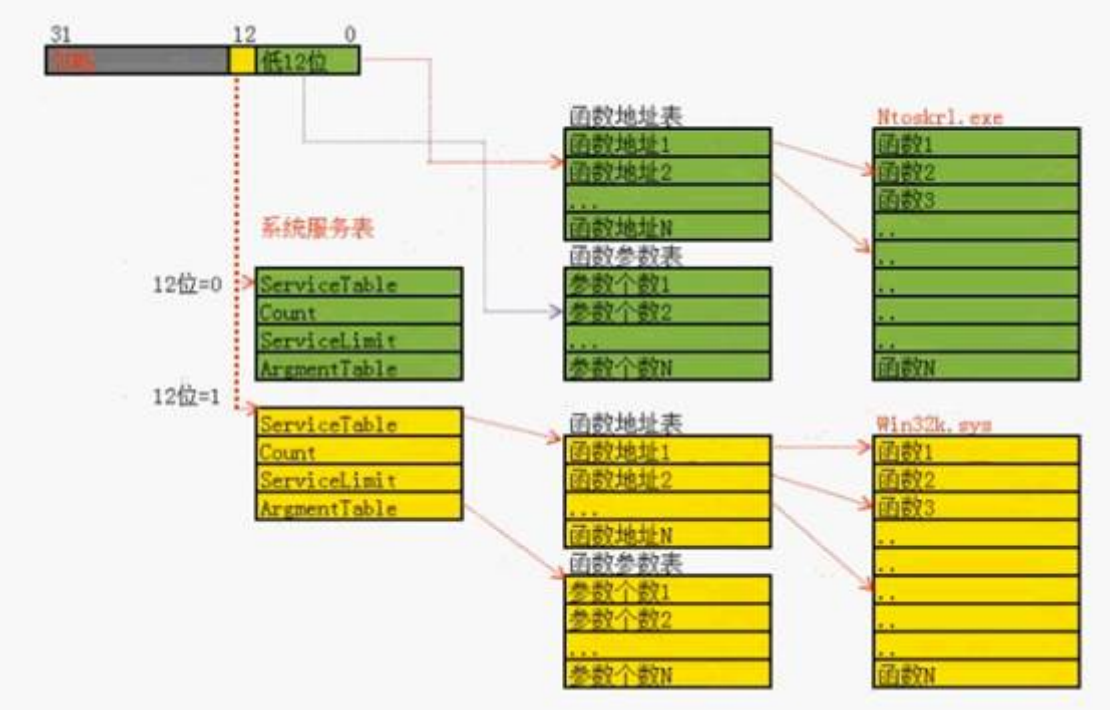
第二张表（Win32k.sys导出的函数）

存储界面等api函数

系统服务表在KTharead结构体 0xE偏移处的地址

在三环传入一个系统服务号

0环根据函数地址表 + 序号*4 找到调用的函数



第13位为 0 第一张表

1 第二张表

SSDT HOOK原理

SSDT ----> Ntoskrl系统服务描述符表 ---->第一个成员函数地址表----->修改函数地址（极容易被发现）

方法

一．修改内存保护属性

1.　修改页属性

通过页表基址直接修改

根据CR4判断分页模式

修改pdt与ptt的属性，从而修改物理内存页

2.修改Cr0寄存器

二．准备用于替换的函数

保证参数一致

三．执行自己的函数，调用原本函数

SYSENTER_HOOK

原理：Kifastentrycall进入0环时，会读取SYSENTER_EIP_MSR寄存器中的值，我们可以在进入0环之前将其替换掉

步骤

A.读取SYSENTER_EIP_MSR寄存器信息，并备份系统KiFastCallEntry函数地址

B.构建一个我们自己的MyKiFastCallEntry函数应以过滤调用信息

C.设置SYSENTER_EIP_MSR寄存器指向我们自己构造函数的地址

D.如果需要摘除钩子，则将备份的系统KiFastCallEntry函数地址写回SYSENTER_EIP_MSR　　寄存器即可

Object-hook

内核对象的组成

每一个内核对象都是由两部分组成

对象头：大家都是一样的

对象体：不同的对象不一样，也是我们经常使用的部分

通过内核对象的头部，我们能够获得此类型内核对象的创建信息。在这些信息中有一组函数指针是非常重要的。我们通过Hook这些函数指针，能够达到监控内核对象操作的目的，这种hook被称之为Object-Hook