第四十五章

1. system V IPC：
2. 概述：

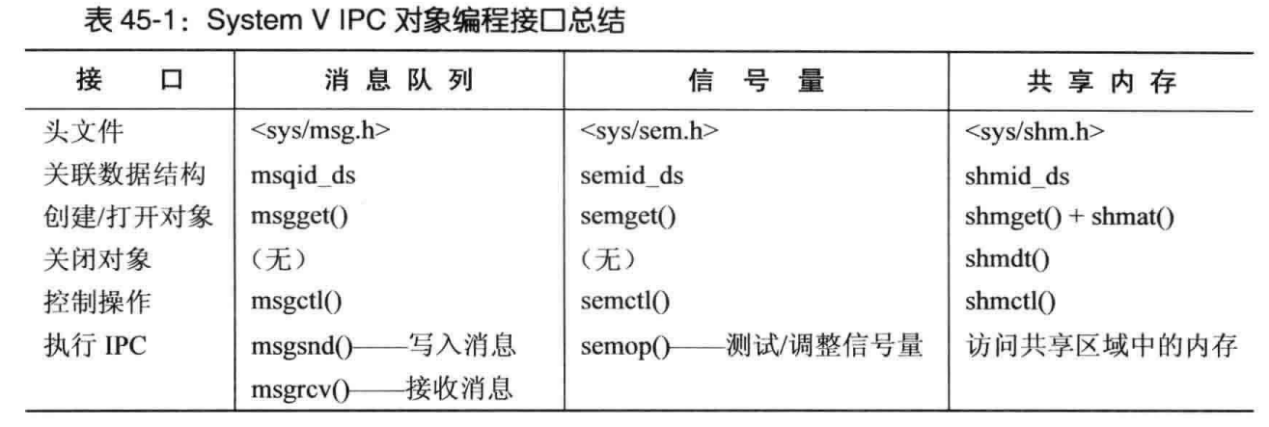
包括三种不同的进程间通信机制：

消息队列，信号量，共享内存。

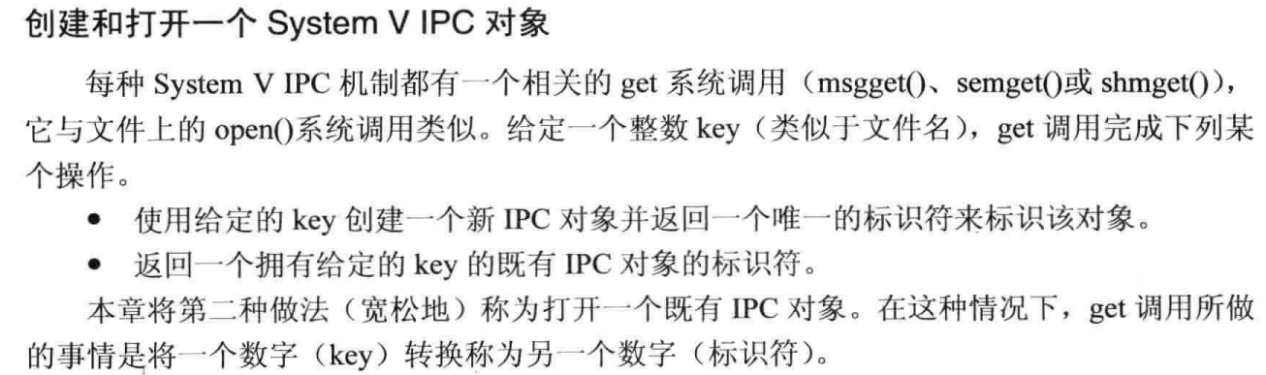
把这三个放在一起讨论的原因：

1. 一同开发出来的。
2. 它们的编程接口具有相同特征。

编程接口总结：



创建和打开一个对象：

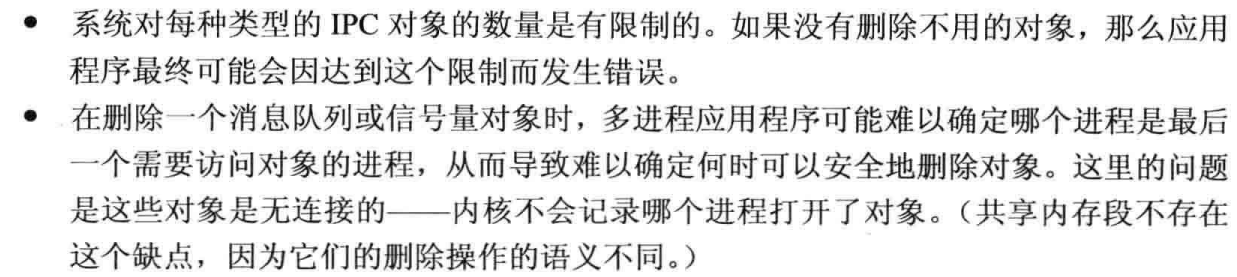


注意，IPC标识符和文件描述符存在一个非常重要的差别：文件描述符是进程特性，而IPC标识符是对象本身属性并且对系统全局可见。这意味着，只要能够获取IPC标识符，就可以跳过get调用。

对象删除和对象持久：

对于消息队列和信号量来说，删除是立即生效的。而共享内存要等所有使用该内存段的进程与此分离才删除。

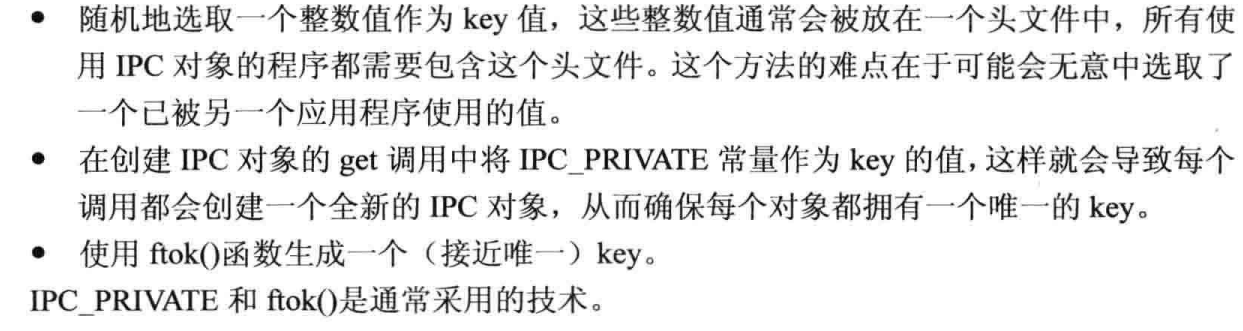
System V IPC是内核持久性，缺点是

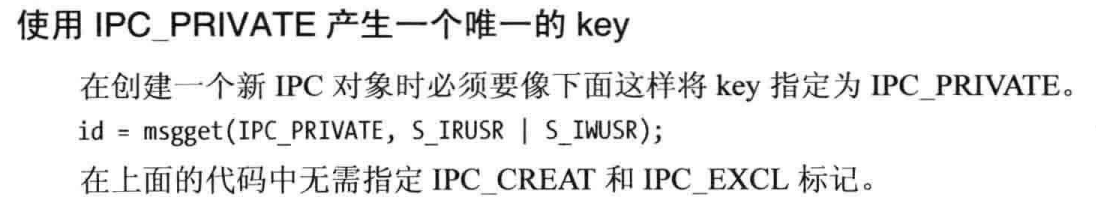


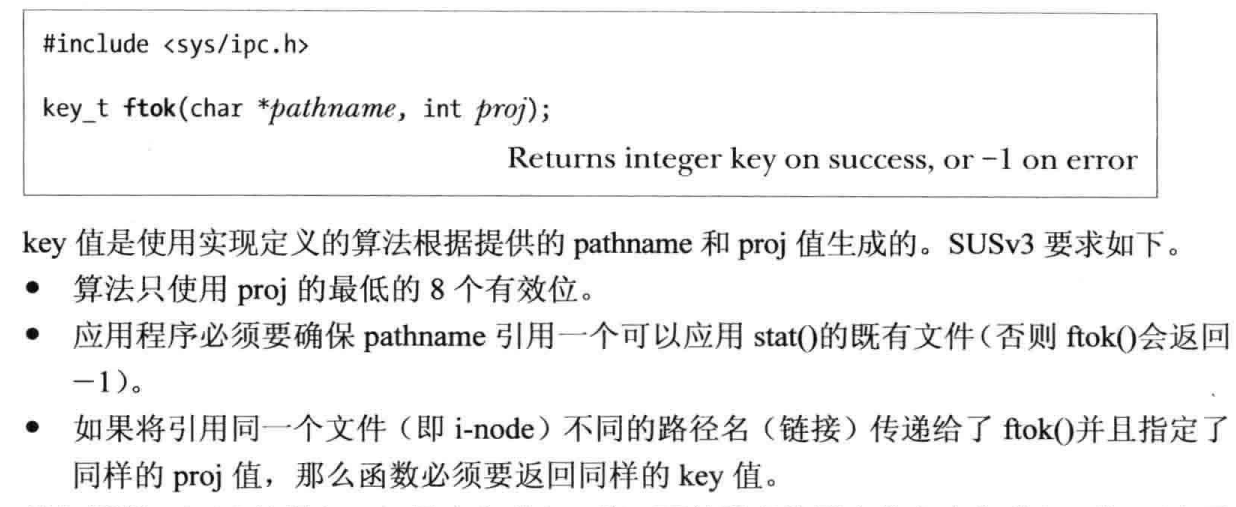
1. IPC KEY：

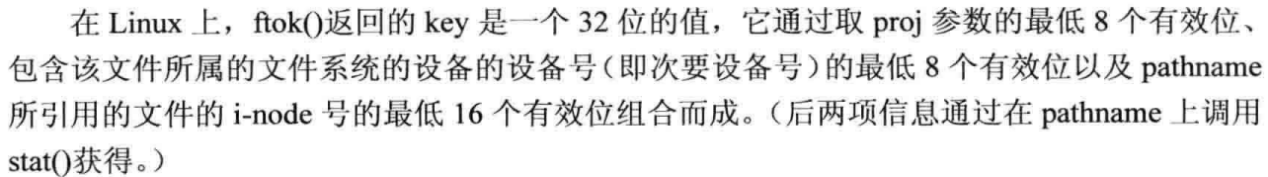
System V IPC里的key是一个整型值，数据类型是key\_t。IPC get将相应的key转换为IPC标识符。新对象对应唯一的标识符，如果指定了一个既有对象的key，那么总会取得该对象的标识符。

如何产生唯一的key呢？





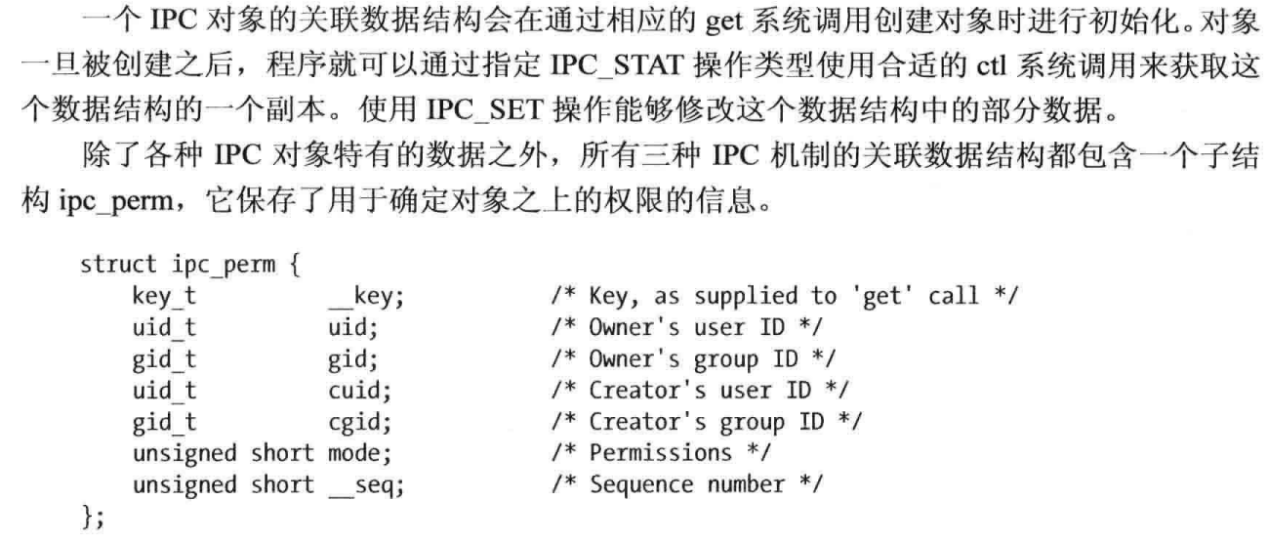




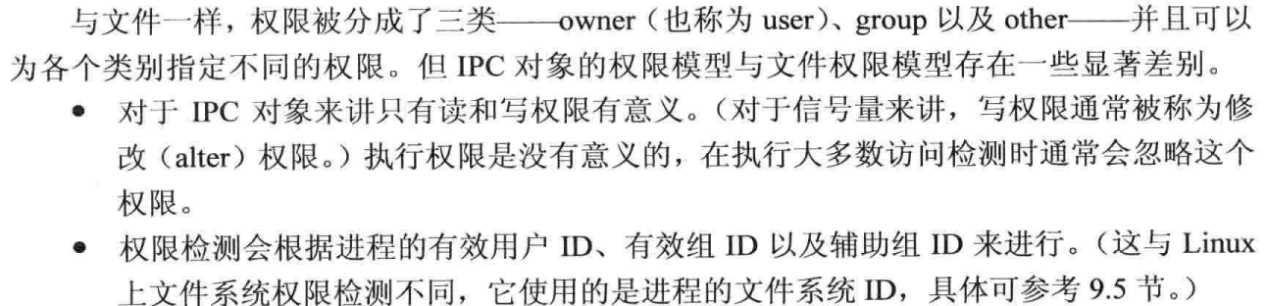


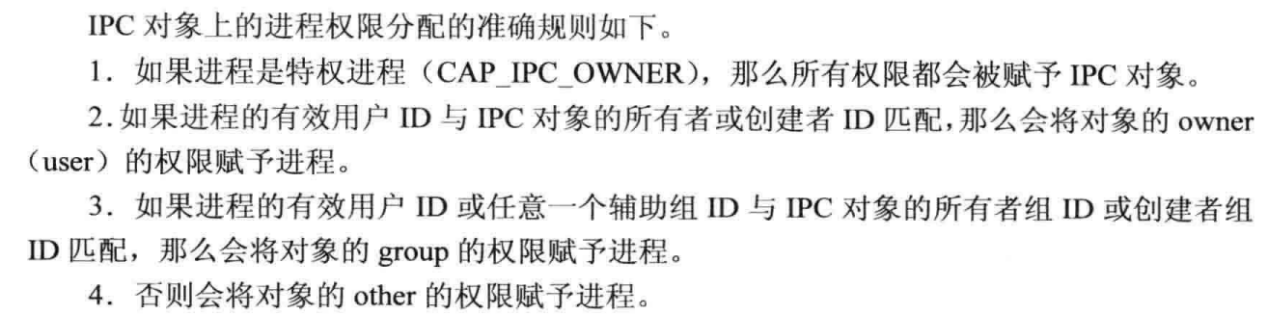
1. 关联数据结构和对象权限：

内核为每个对象的实例都维护了一个关联的数据结构，这个关联数据结构会在get系统调用创建对象的时候进行初始化。



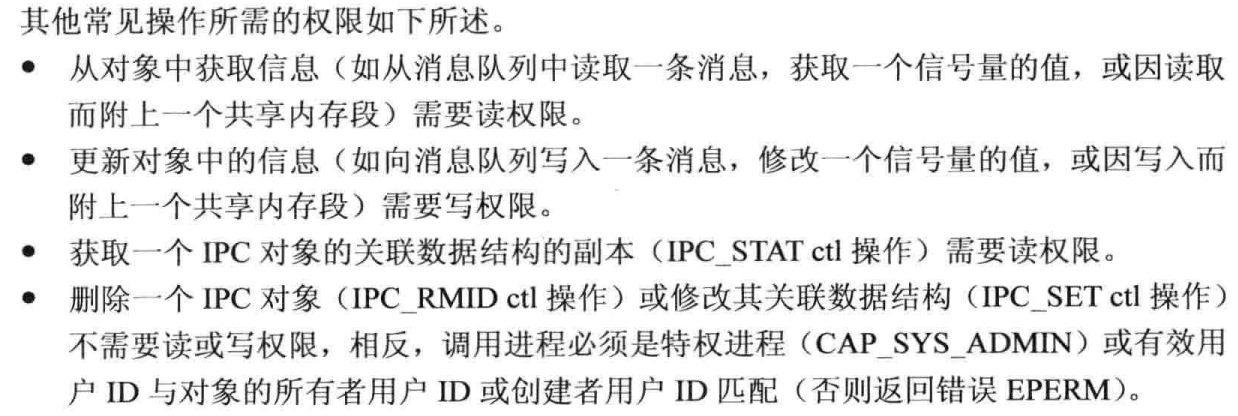
对象的权限：





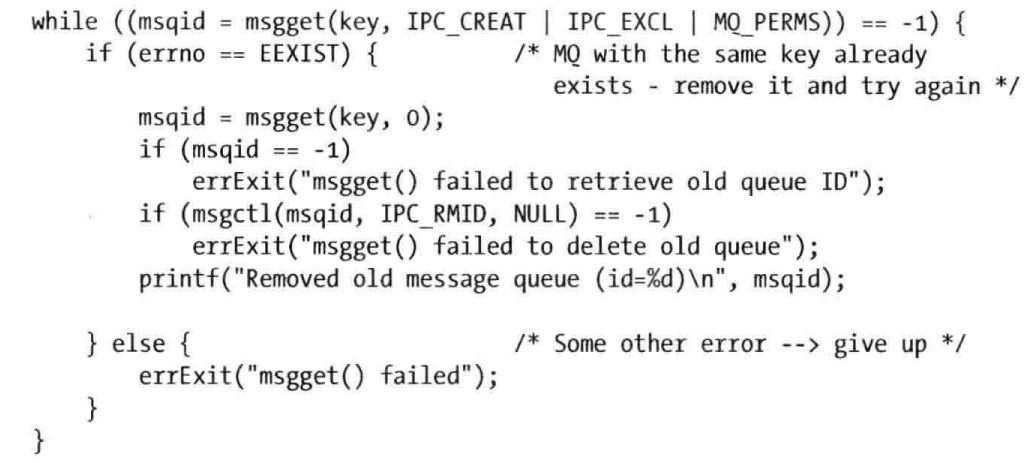
注意，获取一个既有的IPC对象需要进行初次检测。

其他常见操作需要的权限：



1. IPC标识符和客户端/服务器应用程序：

设想一个情景，如果一个会话中，服务器崩溃需要重启，这个时候盲目重启以前的IPC对象没有意义。这个时候需要丢弃现有的客户端，删除上一个服务器进程所创建的IPC对象。示例如下：

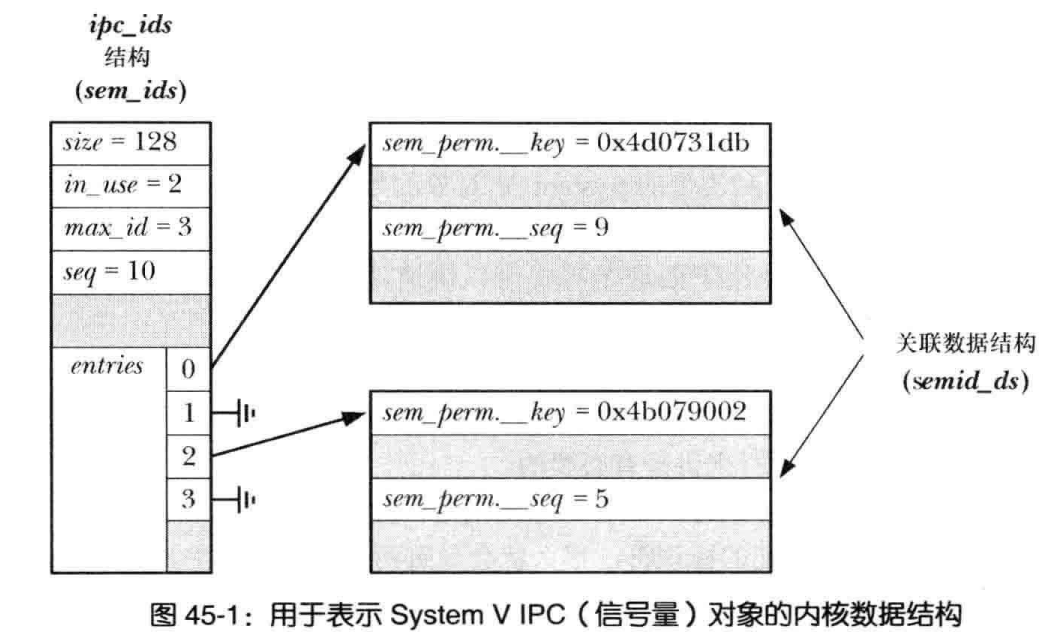


但是有一个问题：

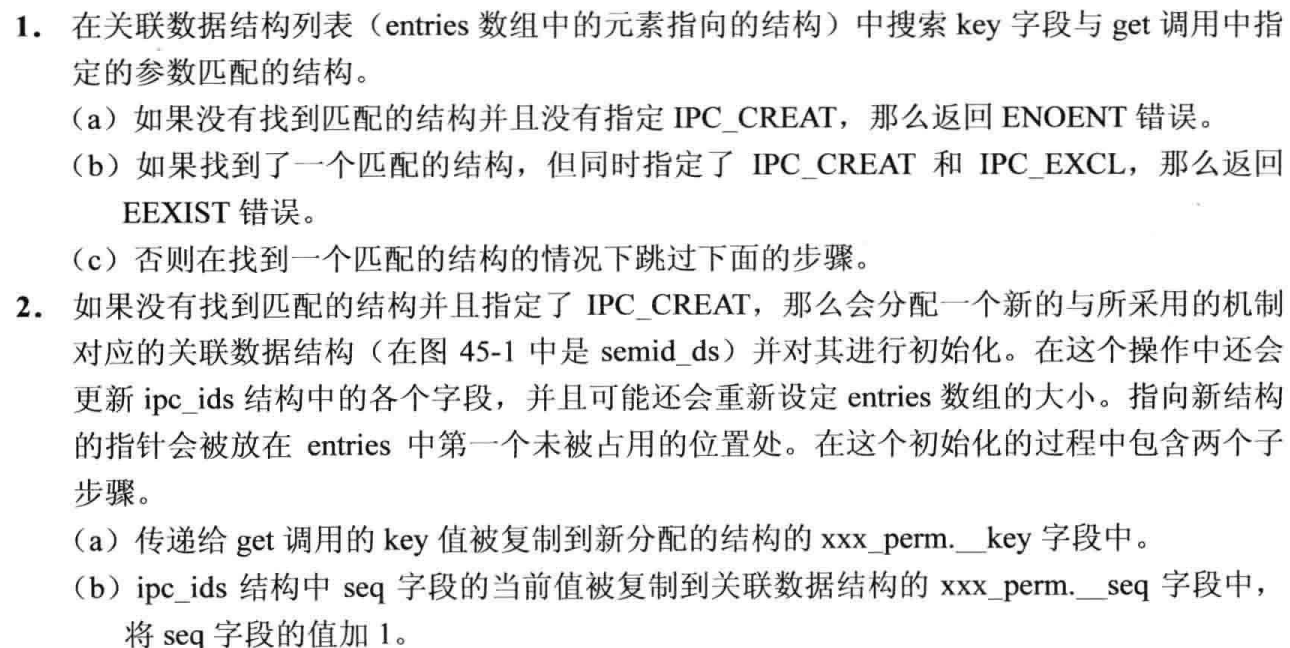
从客户端来看，如果服务器重启IPC对象使用了同样的标识符，客户端就无法感知这一变化。解决这个问题需要用到下一节知识。

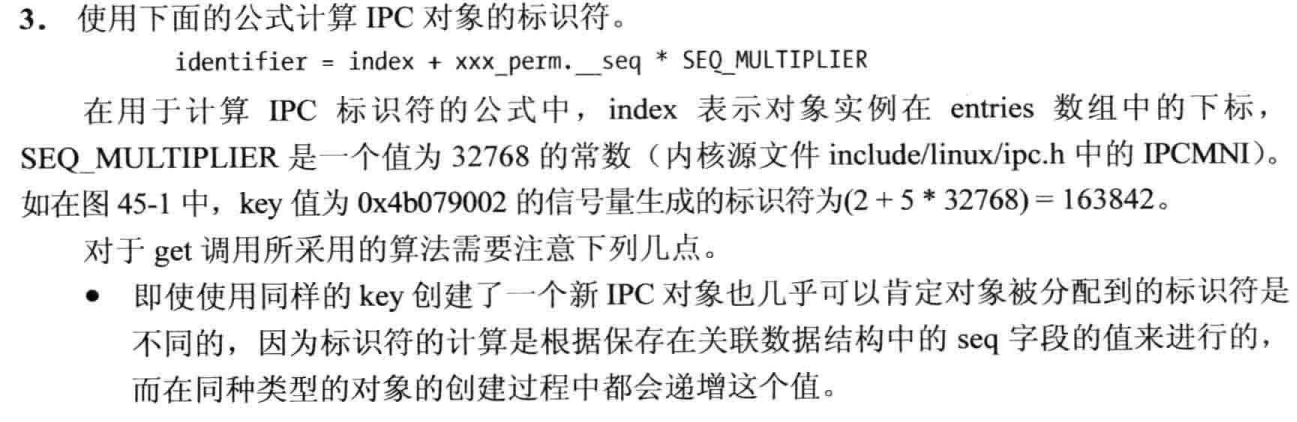
1. System V IPC get调用使用的算法：

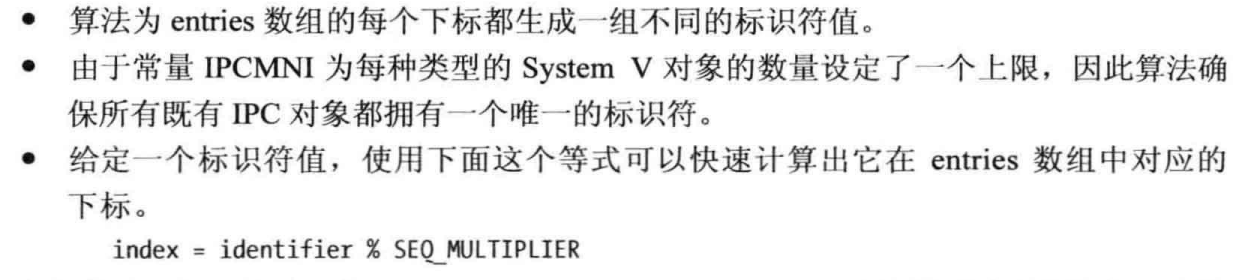
内核内部用来表示System V IPC对象的结构：



执行get调用时采用的算法：





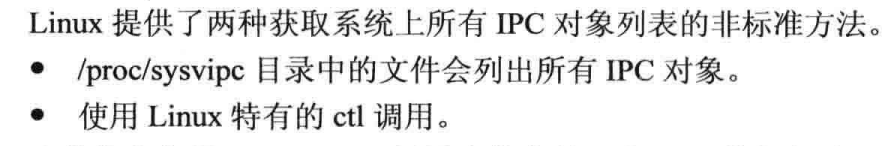


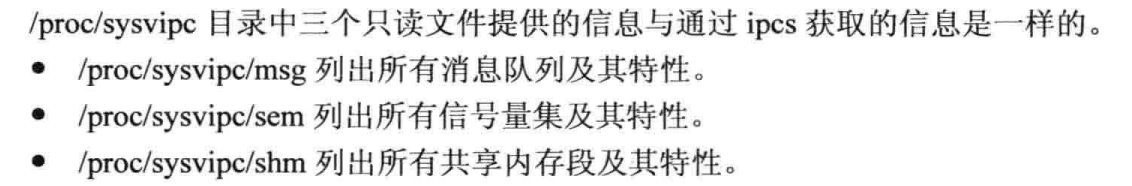
注意，从上面的描述中可以看出，每种类型的对象数量上限由IPCMNI决定，由上面第二点和index的计算可以得出对象数量不超过IPCMNI。

1. Ipcs和ipcrm命令：

类似于ls和rm的文件命令。具体查看手册。

1. 获取所有IPC对象列表：







1. IPC限制：

接下来章节会介绍。

