**进程之间的通讯之IPC对象**

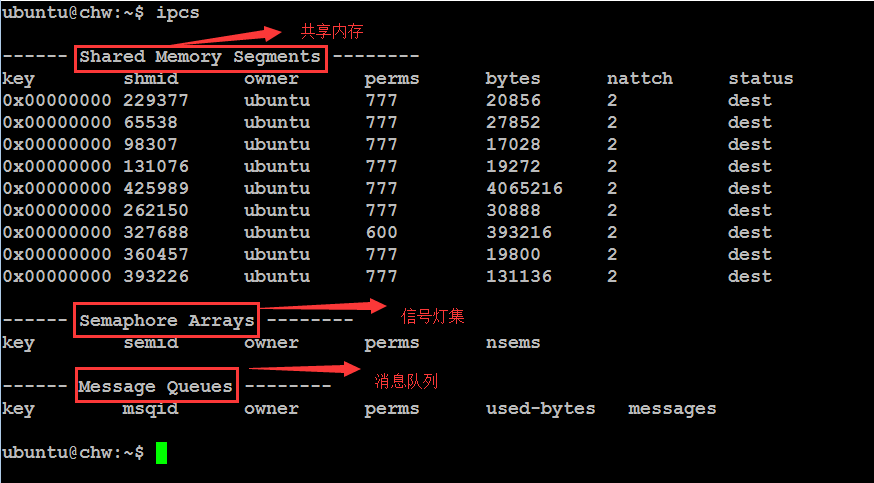
1. **SystemV 进程间的通讯原理讲解**

**我们前面学习了我们传统的进程间通信-----管道和信号。今天我们来学习一下，我们的System V进程间的通信方式。System V IPC(Inter-Process Communication) 对象是由贝尔实验室发展而来的。主要有三种通信方式-----**

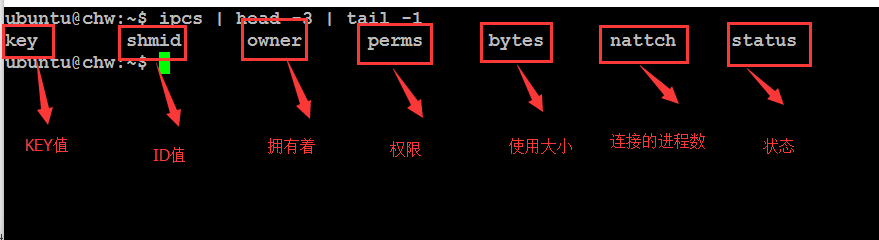
**共享内存,消息队列，信号灯集。Linux对这三种通信机制的****实施大同小异。**

**我们把共享内存,消息队列，信号灯集统称为System V IPC对象。就像我们每次打开一个文件，操作系统会提供过一个inode号一样。我们的每个IPC对象也有一个唯一的标识号。进程可以通过系统调用（操作系统提供的函数）传递的识别号来对我们的IPC对象[共享内存,消息队列，信号灯集]进行操作。与文件的操作一样，我们的每个文件都有权限，我们对象的操作也有一些权限的问题。**

**Ipcs用于显示系统IPC对象[共享内存,消息队列，信号灯集]的信息。**



**由上图所知，一个IPC对象至少包含以下信息：**

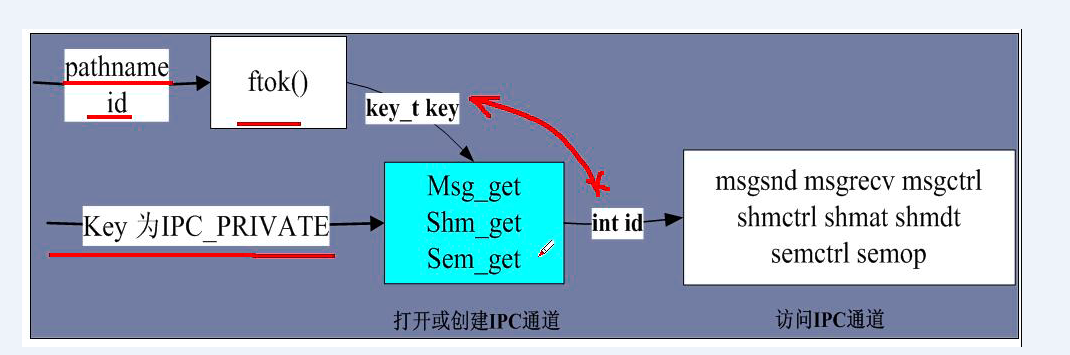


**其他的东西不需要了解，这里我们只需要了解两个东西一个是key值。还有一个就是我们的ID值。**

**ID 资源条目的唯一表示号。**

**KEY 应用程序读取资源使用的参数。**

**每个IPC机制，系统都为其分配了唯一的ID，所有针对该IPC机制的操作都使用对应的ID，因此通信双方都需要通过某个方法来获取ID值。为了解决获取ID值的问题，操作系统引入了key值，约定，IPC创建的时候使用key值作为参数，相同的key值可以得到同一个IPC对象的ID(即一方创建，另一方通过key参数来获得ID)。它保证了双方可以获取用于传递数据的IPC机制的ID值。**



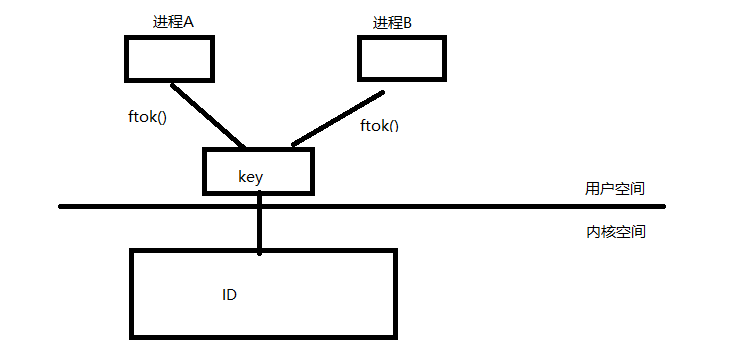
**如上图，为了尽可能与系统信息的载体（文件）相关联。我们的通过文件名和int类型的变量调用ftok（）函数可以获得我们的key值信息。用于获取IPC对象的**

**ID从而访问我们的IPC对象。**

**注意:在IPC的通信模式下，不管是使用消息队列还是共享内存，甚至是信号灯，每个IPC的对象都有唯一的名字,称为"键"(key)。通过"键"，进程能够识别所用的对象。"键"与IPC对象的关系就如同文件名称于文件，通过文件名，进程能够读写文件内的数据，甚至多个进程能够公用一个文件。而在 IPC的通讯模式下，通过"键"的使用也使得一个IPC对象能为多个进程所共用。**

**OK，以上内容为原理的讲解，可能不是很容易理解。总之，大家只需要记住下面的总结就可以了。**

**总结：进程都是通过IPC对象唯一的名字,称为键key,找到IPC对象。但内核还是通过IPC对象的ID来找到它.不同进程只要获得同一IPC对象的键key,就可以实现操作同一IPC对象,从而实际进程间通信.**



**获得key值的方法：**

**<1>调用ftok函数**

**key\_t ftok(const char \*pathname, int proj\_id);**

**功能:获得key值**

**参数:**

**@pathname 一个已经存在的文件路径**

**@proj\_id(子序号) 一般只会使用它的低八位，常常传字符[0-255]**

**返回值：成功返回key值。失败返回-1.**



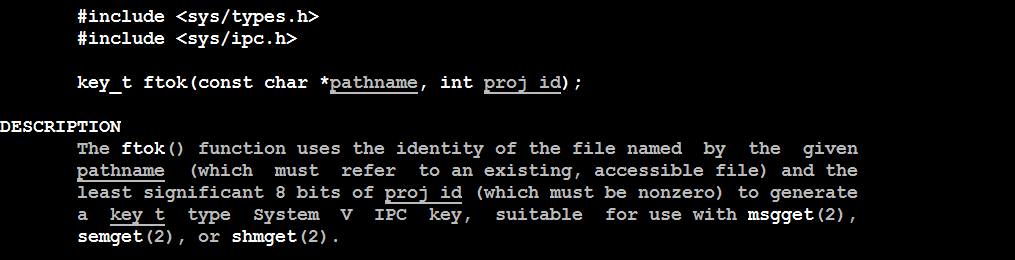
**例如：**

**key\_t key = ftok(“/home/ubuntu”,‘k’);**

**详见man手册**

**这个ftok()函数由我们的pathname参数(这个参数必须存在，并且是一个可进入的文件名)来指定文件名的身份，那个proj\_id(必须不为0)参数，我们只使用了他的低8bit，我们调用fotk函数，结合参数，形成一个key\_t类型的的key值，**

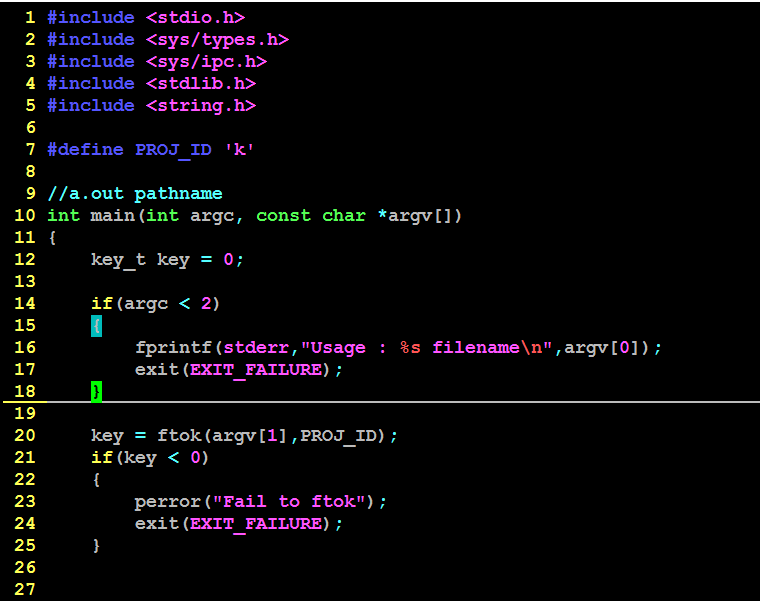
**以用于我们Systrem V的IPC通信中。**

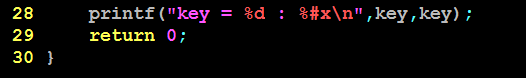


**注意：**

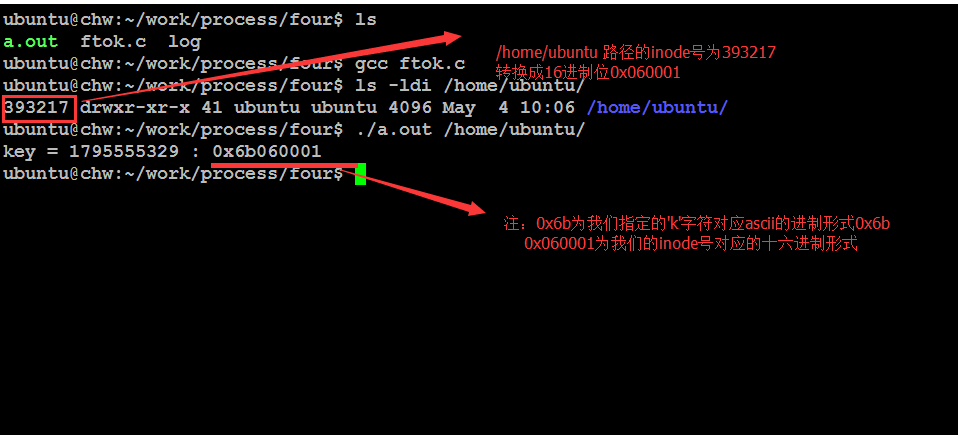
**在一般的UNIX实现中，是将文件的索引节点号取出，前面加上子序号得到key\_t的返回值。如指定文件的索引节点号为65538，换算成16进制为 0x010002，而你指定的ID值为38，换算成16进制为0x26，则最后的key\_t返回值为0x26010002。**

**ftok.c**



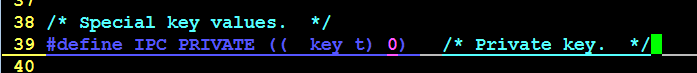


**运行结果：**



**<2>使用IPC\_PRIVATE对象（了解即可）**

**我们使用vi –t IPC\_PRIVATE命令可以查看到下图的信息：**



**使用IPC\_PRIVATE创建的IPC对象, key值属性为0，和IPC对象的编号ID就没有了对应关系。这样毫无关系的进程，就不能通过key值来得到IPC对象的编号（因为这种方式创建的IPC对象的key值都是0）。因此，这种方式产生的IPC对象，和无名管道类似，不能用于毫无关系的进程间通信。但也不是一点用处都没有，仍然可以用于有亲缘关系的进程间通信。**