消息队列

消息队列，是消息的链接表，存放在内核中。一个消息队列由一个标识符（即队列ID）来标识。无所谓客户和服务器启动先后

1、特点

* 消息队列是面向记录的，其中的消息具有特定的格式以及特定的优先级。
* 消息队列独立于发送与接收进程。进程终止时，消息队列及其内容并不会被删除。
* 消息队列可以实现消息的随机查询,消息不一定要以先进先出的次序读取,也可以按消息的类型读取。

2、原型

#include <sys/msg.h>

// 创建或打开消息队列：成功返回队列ID，失败返回-1

int msgget(key\_t key, int flag);

// 添加消息：成功返回0，失败返回-1

int msgsnd(int msqid, const void \*ptr, size\_t size, int flag);

// 读取消息：成功返回消息数据的长度，失败返回-1

int msgrcv(int msqid, void \*ptr, size\_t size, long type,int flag);

// 控制消息队列：成功返回0，失败返回-1

int msgctl(int msqid, int cmd, struct msqid\_ds \*buf);

*说明：*

在以下两种情况下，msgget将创建一个新的消息队列：

* 如果没有与键值key相对应的消息队列，并且flag中包含了IPC\_CREAT标志位。
* key参数为IPC\_PRIVATE。

函数msgrcv在读取消息队列时，type参数有下面几种情况：

* type == 0，返回队列中的第一个消息；
* type > 0，返回队列中消息类型为 type 的第一个消息；
* type < 0，返回队列中消息类型值小于或等于 type 绝对值的消息，如果有多个，则取类型值最小的消息。

可以看出，type值非 0 时用于以非先进先出次序读消息。也可以把 type 看做优先级的权值。

命名管道相比，消息队列的优势在于，1、消息队列也可以独立于发送和接收进程而存在，从而消除了在同步命名管道的打开和关闭时可能产生的困难。2、同时通过发送消息还可以避免命名管道的同步和阻塞问题，不需要由进程自己来提供同步方法。3、接收程序可以通过消息类型有选择地接收数据，而不是像命名管道中那样，只能默认地接收。

struct msbuf 一般具有两个部分，一个部分是 msgtype， 另一个是msgname消息队列的内容，我们一般采用消息队列传输具有一定结构类型的数据；现在假设需要传送的结构为：

typedef struct \_S{

int a；

int b；

int c；

}TS；

下面是我们的消息结构：

struct msbuf{

long msg\_type；

TS msg\_name;

}

Ipcs -l (get limit)

[root@www ~]# ipcs -l  (-u get staus)  
  
------ Messages: Limits --------  
max queues system wide = 900000            =========MSGMNI //整个系统的最大数量的消息队列。  
max size of message (bytes) = 65536        =========msgmax //单个消息的最大size。  
default max size of queue (bytes) = 655360000  =====msgmnb //每个消息队列的最大字节限制。