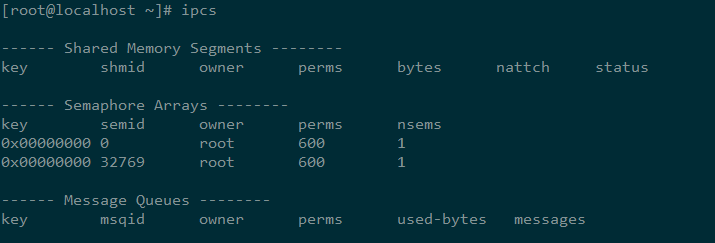
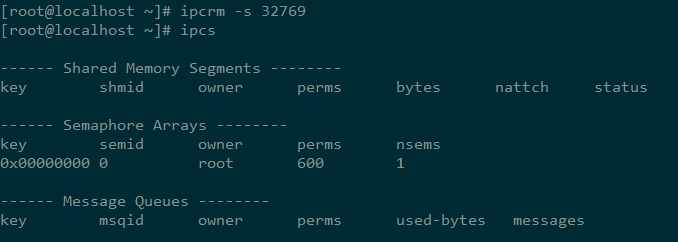
unix/linux下的共享内存、信号量、队列信息管理  
在unix/linux下，经常有因为共享内存、信号量，队列等共享信息没有干净地清楚而引起一些问题。  
查看共享信息的内存的命令是ipcs [-m|-s|-q]。

默认会列出共享内存、信号量，队列信息，-m列出共享内存，-s列出共享信号量，-q列出共享队列。 -a是同时显示共享内存、信号量、队列。Ipcs的默认参数就是-a

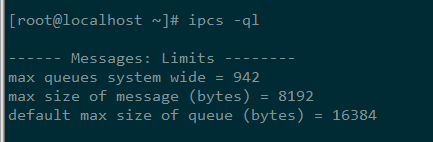
  
清除命令是ipcrm [-m|-s|-q] id。  
-m 删除共享内存，-s删除共享信号量，-q删除共享队列。

ipcrm可用来删除对应的共享内存段、信号量、消息队列；  
命令示例：  
ipcrm -s semid 删除对应的信号量集  
ipcrm -m shmid 删除对应的共享内存段  
ipcrm -q msqid 删除对应的消息队列

  
ipcrm本身只能实现单个资源的删除，利用以下命令可实现批量删除

1.ipcs -s|grep 用户名|cut -d" " -f2|xargs -n1 ipcrm -s  
2.ipcs -s|awk '/用户名/{print $2}'|xargs -n1 ipcrm -s  
3.ipcs -s|awk '/用户名/{system("ipcrm -s "$2)}'  
4.for i in echo `ipcs|grep 用户名|cut -d" " -f2`; do ipcrm -s $i; done

查看消息队列大小 ipcs -ql



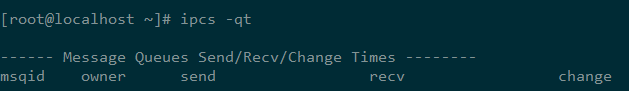
其中：max queues system wide为最大进程数

max size of message (bytes) 队列中最大消息长度

default max size of queue (bytes) 队列默认最大长度

修改消息队列大小：  
  
root：用户： /etc/sysctl.conf  
  
kernel.msgmnb =4203520  
#kernel.msgmnb =3520  
kernel.msgmni = 2878  
  
保存后需要执行 sysctl -p ,然后重建所有消息队列

显示消息队列的创建时间，发送和接收最后一条消息的时间 ipcs -qt



# Msgctl函数

通过msgctl()函数，获取和设置消息队列的属性，我们可以直接控制消息队列的行为，

msgctl系统调用对msqid标识的消息队列执行cmd操作列。

系统调用： msgctl()  
函数声明： int msgctl ( int msgqid, int cmd, struct msqid\_ds \*buf )

msggid 需要处理的消息队列标识符id

### 使用命令 cmd

**IPC\_STAT**

读取[消息队列](http://baike.baidu.com/view/262473.htm" \t "_blank)的数据结构msqid\_ds，并将其存储在b u f指定的地址中。

**IPC\_SET**

设置消息队列的数据结构msqid\_ds中的ipc\_perm元素的值。这个值取自buf参数。

**IPC\_RMID**

从系统内核中移走消息队列。

### 返回值：

0 ，如果成功。

- 1，如果失败：errno = EACCES (没有读的权限同时cmd 是IPC\_STAT )

EFAULT (buf 指向的地址无效)

EIDRM (在读取中队列被删除)

EINVAL (msgqid无效, 或者msgsz 小于0 )

EPERM (IPC\_SET或者IPC\_RMID 命令被使用，但调用程序没有写的权限)

例如：msgctl(0,IPC\_RMID,NULL)

因为IPC\_RMID为移除消息队列，所以此条意思为把消息队列标识符为0的队列从内核中移除。

#include<sys/msg.h>

#include<sys/types.h>

#include<sys/ipc.h>

#include<stdio.h>

#include<stdlib.h>

int main(int argc,char \*argv[]){

int gid;

int k;

if (2!=argc){

puts("USAGE:del\_msgg.c<quene ID>");

exit(1);

}

gid=atoi(argv[1]); //数据交换 atoi是把字符串转换成int型

printf("print gid: %d\n",gid);

system("ipcs -q"); //执行shell ipcs-q

if(0!=(k=msgctl(gid,IPC\_RMID,NULL))){

printf("k=%d\n",k);

perror("msgctl \n"); //将上一个函数发生的错误的原因输出到标准设备stderr上

exit(k);

}

system("ipcs -q");

printf("successfully removed %d queue\n",gid);

exit(0);

}