**北京邮电大学软件学院**

**2017-2018学年第2学期实验报告**

**课程名称： 操作系统**

**实验名称： 实验二 进程通讯**

**实验完成人：**

**姓名：**\_\_\_\_田宇\_\_\_\_**学号：**\_2016212011\_**成绩：**\_\_\_\_\_\_\_\_

**日 期： 2018 年 3 月 23 日**

1. **实验目的**

（1）消息缓冲队列、共享存储区机制进行进程间的通信；

（2）理解通信机制。

1. **实验内容**

2.1.使用消息缓冲队列来实现 client 进程和 server 进程之间的通信

server 进程先建立一个关键字为 SVKEY（如 75）的消息队列，然后等待接收类型为 REQ （例如 1）的消息；在收到请求消息后，它便显示字符串“serving for client”和接收到 的 client 进程的进程标识数，表示正在为 client 进程服务；然后再向 client 进程发送应答消息，该消息的类型是 client 进程的进程标识数，而正文则是 server 进程自己的标识 ID。client 进程则向消息队列发送类型为 REQ 的消息（消息的正文为自己的进程标识 ID） 以取得 sever 进程的服务， 并等待 server 进程发来的应答；然后显示字符串“receive reply from”和接收到的 server 进程的标识 ID。

2.2. 使用共享存储区来实现两个进程之间的进程通信

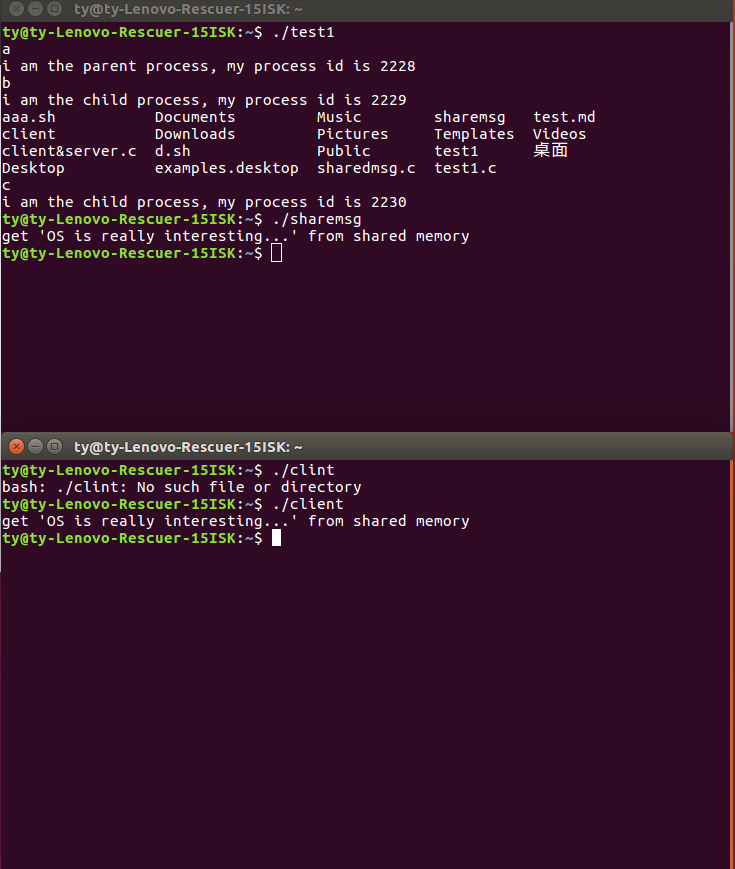
进程 A 创建一个长度为 512 字节的共享内存，并显示写入该共享内存的数据；进程 B 将共享内存附加到自己的地址空间，并向共享内存中写入数据

1. **实验环境**

Ubuntu 16.4

gcc

1. **实验结果**



1. **附件**

**6.1 附件1：源代码**

#include<sys/types.h>

#include<sys/shm.h>

#include<sys/ipc.h>

#include<string.h>

#include<stdio.h>

#include<stdlib.h>

#define SHMKEY 75

int sharemid,i; char \*addr;

char \*argv[ ]={"OS is really interesting..."};

void B( )

{

sharemid=shmget(SHMKEY,512,0777); //打开共享存储区

addr=shmat(sharemid,0,0); //获得共享存储区首地址

memset(addr,'\0',512); //将addr的512字节设置成字符'\0'

strncpy(addr,argv[0],512); //将数组argv的前512字节存入共享区addr

exit(0);

}

void A( )

{

sharemid=shmget(SHMKEY,512,0777|IPC\_CREAT); //创建共享存储区

addr=shmat(sharemid,0,0); //获取首地址

printf("get '%s' from shared memory\n",addr);

exit(0);

}

int main( )

{

while ((i=fork( ))==-1);

if (!i) A();

while ((i=fork( ))==-1);

if (!i) B();

wait(0);

wait(0);

}

#include<sys/types.h>

#include<sys/msg.h>

#include<sys/ipc.h>

#include<stdio.h>

#include<stdlib.h>

#define MSGKEY 75

struct msgform

{

long mtype;

char mtext[250];

}msg;

int msgqid,pid,\*pint,i;

void client()

{

msgqid=msgget(MSGKEY,0777); //打开75#消息队列

pid=getpid();

pint=(int \*)msg.mtext;

\*pint=pid;

msg.mtype=1; //消息类型为1

msgsnd(msgqid,&msg,sizeof(int),0);//添加消息

msgrcv(msgqid,&msg,250,pid,0); //接收消息

printf("(client):receive reply from pid=%d\n",\*pint); //显示server进程标识数

exit(0);

}

void server( )

{

msgqid=msgget(MSGKEY,0777|IPC\_CREAT); //创建75#消息队列

msgrcv(msgqid,&msg,250,1,0); //接收消息

pint=(int \*)msg.mtext; //把正文的内容传给pint，并强制转换类型

pid=\*pint; //获得cilent进程标识数

printf("(server):serving for client pid=%d\n",pid);

msg.mtype=pid; //消息类型为client进程标识数

\*pint=getpid(); //获取server进程标识数

msgsnd(msgqid,&msg,sizeof(int),0); //发送消息

exit(0);

}

int main()

{

while((i=fork())==-1){}; //创建进程1

if(!i)server();

while((i=fork())==-1){}; //创建进程2

if(!i) client();

wait(0);

wait(0);

}