# 实验五 进程控制与通信

## **一、实验目的**

1、掌握进程的常用终端命令；

2、掌握用system、exec函数簇、fork函数创建进程；

3、掌握wait、waitpid函数的应用；

4、掌握守护进程在监控中的应用并编写守护进程；

5、掌握常用中断的方法；

6、掌握signal函数实现信号处理程序；

7、掌握应用管道实现信号处理的方法。

## 二、实验目标与要求

1、用exec函数创建进程；

2、应用fork( )函数创建子进程，在父子进程中执行不同的任务；

3、守护进程的编写与应用；

4、管道读写程序的编写与应用；

5、应用命名管道实现简单的聊天程序设计；

6、应用消息队列实现简单的聊天程序；

7、共享内存通信机制程序设计。

## 三、实验工具与准备

计算机PC机，Linux Redhat 9.0或者其他的linux操作系统

## **四、实验步骤与操作指导**

**任务一：程序设计**

用exec函数簇中的6个函数替换修改课件6-3.c源程序并调试结果。（源程序给出替换部分的代码即可）

**任务二：程序设计**

（1）在父子进程中分别编写循环程序，应用函数sleep的不同参数等，体现程序中父子进程的并发运行。

（2）在父子进程中分别执行不同的任务，例如在子进程中执行文件编辑任务，在父进程中执行网络连通情况的测试任务，子进程退出后父进程才退出。

**任务三：程序设计**

waitpid函数的应用，要求子进程用sleep等待10秒，父进程用waitpid函数等待子进程正常结束，父进程在等待的时候不阻塞，每1秒在屏幕上输出一行文字，若发现子进程退出，打印出等待进程的进程号(PID)和退出状态。请编写一程序进行调试。

提示：要父进程周期行的检查某个特定的子进程是否终止，在waitpid（pid, &stauts,option）函数的option要如何设置？

**任务四：程序调试**

设计三个并发的守护进程在后台运行，其中第一子进程写守护进程的运行日志记录，第二子进程child2则监控进程中是否有gedit工具调用，第二子进程child3则检查自己是否有新邮件到达，若有则将邮件内容输出到一个主目录下文件。

#include <stdio.h>  
#include <stdlib.h>  
#include <sys/types.h>  
#include <sys/stat.h>  
#include <sys/wait.h>  
#include <unistd.h>  
#include <syslog.h>  
#include <signal.h>  
#include <sys/param.h>  
#include <time.h>  
#include <dirent.h>

int main()  
{  
pid\_t child1,child2,child3;  
struct stat buf;  
int i,check=0,j=0;  
time\_t t;  
DIR \* dir;  
struct dirent \* ptr;

child1=fork();  
if (child1>0)  
exit(0); /\*父进程退出\*/  
else if(child1<0)  
{  
perror("创建子进程失败");  
exit(1);  
}

/\*第一子进程\*/  
setsid();  
chdir("/");  
umask(0);  
for(i=0;i<NOFILE;++i)  
close(i);  
openlog("守护进程程序信息",LOG\_PID,LOG\_DAEMON);

child2=fork();  
if (child2==-1)  
{  
perror("创建子进程失败");  
exit(2);  
}  
else if (child2==0)/\*第二子进程中的child2\*/  
{   
i=0;  
while(i++<100){  
system("ps -ef|grep gedit> /home/king/gedit.log");  
stat("home/king/gedit.log",&buf);

if (buf.st\_size>180 && check==0)

{  
t=time(0);   
syslog(LOG\_INFO,"gedit开始时间为： %s\n",asctime(localtime(&t)));  
check=1;   
}   
if (buf.st\_size<180 && buf.st\_size>0 && check==1)  
{   
t=time(0);  
syslog(LOG\_INFO,"gedit结束时间为： %s\n",asctime(localtime(&t)));  
check=0;  
}  
sleep(1);  
}   
}  
else  
{ /\*在第一子进程下继续创建进程\*/  
child3=fork();  
if (child3<0){  
perror("创建子程序失败");  
exit(3);  
}  
else if (child3==0){ /\*第二子进程child3用来查看邮件\*/  
j=0;  
dir=opendir("/var/spool/mail/king");  
while (j<6){  
j++;  
sleep(10);  
if ((ptr=readdir(dir))!=NULL){   
system("cat /var/spool/mail/king/\* > /home/king/mail.log");  
}   
}  
closedir(dir);  
}  
else   
{ /\*第一子进程写日志来记录守护进程的运行\*/  
t=time(0);  
syslog(LOG\_INFO,"守护进程开始时间为： %s\n",asctime(localtime(&t)));  
waitpid(child2,NULL,0);  
waitpid(child3,NULL,0);  
t=time(0);  
syslog(LOG\_INFO,"守护进程结束时间为： %s\n",asctime(localtime(&t)));  
closelog();  
while (1)  
sleep(10);  
}   
}  
}

写出程序调试测试结果。

**任务五：学习巩固**

（1）使用命令kill，显示linux环境下的信号。分析这些信号的特点。

（2）下列是进程调度中常用的函数，通过网络资源查找这些函数的功能与用法。

|  |  |
| --- | --- |
| 函 数 | 功能与用法 |
| Kill |  |
| Raise |  |
| alarm |  |
| pause |  |
| signal |  |
| sigemptyset |  |
| sigfillset |  |
| sigaddset |  |
| sigdelset |  |
| sigismember |  |
| sigprocmask |  |

**任务六：程序设计**

编写一个C程序，完成以下功能。

（1）父进程创建2个子进程P1、P2；

（2）父进程捕捉从键盘上通过CTRL+C键发来的中断信号；

（3）父进程获得中断信号后使用系统调用kill()向两个子进程分别发终止执行信号SIGUSR1和SIGUSR2；

（4）子进程捕捉到各自的信号后分别输出：

child1 is killed by parent!和child2 is killed by parent!

然后终止执行（无先后次序的要求）；

（5）父进程等待两个子进程终止后输出以下信息，然后终止执行：

parent process is killed!