《Linux应用程序开发实验报告》

题目: 进程控制开发

班级:\_\_ \_\_ \_\_17计科3班 \_

学号: 10417307 \_\_\_

姓名: \_\_薛冬冬\_ \_\_\_

东南大学成贤学院计算机系

2020年 4月 7日

**实验题目**

1. **实验目的**

1.掌握进程相关的概念；

2.掌握Linux下进程结构；

3.掌握Linux下进程创建相关的系统调用；

4.掌握守护进程的启动方法。

1. **实验内容**
2. 完成实验具体要求内实验；
3. 掌握fork()函数和waitpid()函数的作用并学会使用。
4. **实验具体要求**

**1、以下是对execlp进行调试之后显示的调试信息，根据此信息回答问题**

**（1）执行父进程的进程号是多少？**

**（2）执行子进程的进程号是多少？**

[Detaching after fork from child process 14692. (Try `set detach-on-fork off'.)]

UID       PID         PPID     C         STIME     TTY       TIME          CMD

root     14562      1           0         22:12       ?          00:00:00     gnome-terminal

root     14565     14562     0         22:12      ?           00:00:00     gnome-pty-helper

root     14566     14562     0         22:12     pts/1      00:00:00      bash

root     14675     14566     0         22:15     pts/1      00:00:00      gdb execlp

root     14676     14675     0         22:15     pts/1      00:00:00      /root/7/execlp

root     14692     14676     0         22:15     pts/1       00:00:00      ps -ef

辅助说明：gnome-terminal就是终端进程，bash是shell命令的版本类型，多数Linux发行版本采用bash作为其命令版本；

**答：**

（1）执行父进程的进程号： 14676

（2）执行子进程的进程号： 14692

**2、使用gdb调试waitpid()，如果首先执行的是父进程，会出现什么情况？**

**如果想要调试子进程应该如何做？？？**

**答：**

1. ●若首先执行父进程，且子进程未执行，则窗口将一直显示“The child process has not exited”，需要使用kill使其强制退出。

●若首先执行父进程，且子进程也在执行状态，父进程将输出“The child process has not exited”后暂停1s，直到子进程退出（5s），即输出5条“The child process has not exited”，子进程退出后输出“The father process quit because the child process quit.”和子进程的进程号。

（2）想要调试子进程方法

将“pr = waitpid(pc, NULL, WNOHANG); ”中WNOHANG选项改为0，即“pr = waitpid(pc, NULL, 0); ”

“pr = waitpid(pc, NULL, WNOHANG); ”中WNOHANG选项表示：父进程不阻塞，可以一边执行父进程其他语句一边等待子进程的消息，将WNOHANG选项改为0则表示阻塞父进程，等待子进程执行，这样就可以实现子进程的调试。

**3、已知multi\_proc可执行程序的一个可能的执行结果，如下所示：**

[root@localhost ~]# ./multi\_proc

In child1: execute 'ls -l'

In child2: sleep for 5 seconds and then exit

In father process:

总计 96

-rw------- 1 root root 1155 2014-09-26 anaconda-ks.cfg

drwxr-xr-x 2 root root 4096 2014-09-26 Desktop

-rw-r--r-- 1 root root 35507 2014-09-26 install.log

-rw-r--r-- 1 root root 4399 2014-09-26 install.log.syslog

drwxrwxrwx 10 root root 4096 02-26 08:27 linux

-rwxr-xr-x 1 root root 5760 03-16 12:42 multi\_proc

drwxr-xr-x 3 root root 4096 01-05 08:57 vmtools

Get child1 exit code

The child2 process has not exited!

The child2 process has not exited!

The child2 process has not exited!

The child2 process has not exited!

The child2 process has not exited!

Get child2 exit code

[root@localhost ~]#

**假设CPU是单核，即某时刻只有一个进程在CPU上运行，试说明这些结果是如何运行出来的？**

**是否可以设置进程间的执行顺序，如果可以应该怎么做？**

**答：**

（1）因为CPU是单核，所以某一个时刻仅有一个进程在运行。程序第15行创建了一个子进程child1，运行子进程child1，此时child1==0，首先输出"In child1: execute 'ls -l' "。

然后child1进入休眠等待状态，进入父进程中等待子进程2的退出，子进程2处于执行状态，因此child2=0，首先输出"In child2: sleep for 5 seconds and then exit "，然后输出"In father process:"，阻塞父进程，等待子进程child1退出。

继续执行子进程1，调用execlp函数，一行显示输出一个文件，输出所在目录的文件，输出完成后将waitpid()返回值赋值给child。若child大小和child1相同，则表示进程1执行结束，输出"Get child1 exit code"。

不阻塞父进程，等待子进程child2退出，同时执行子进程2和父进程，因为此时子进程2处于休眠状态（5s），因此执行父进程，每隔1s输出" The child2 process has not exited!"，共输出5条。5s后子进程2关闭，若waitpid()返回值赋与此时child相同，则代表进程2执行结束，输出"Get child2 exit code "。

（2）不可以，因为用户代码不能保证在使用fork()创建子进程后父子进程的并发的执行顺序。

使用fork()创建子进程后，父子进程的执行顺序fork函数无法确定，但是可以使用wait()和waitpid()函数使父进程阻塞等待子进程退出，确保子进程先结束。

**4、针对GDB多进程调试视频中的问题。**

为什么在GDB中，以默认的调试方式调试waitpid程序；将子进程sleep(5);上下加两条打印语句，如下所示：

printf("The child process is running!\n"); /\*子进程暂停5s\*/

sleep(5);

printf("The child process is quitted!\n"); /\*子进程正常退出\*/

exit(0);

两条printf()之间父进程只打印4条The child process has not exited语句；如下面代码所示：

(gdb) r

Starting program: /root/7/waitpid\_GDB多进程调试 \_子进程有打印语句

Error in re-setting breakpoint 1: Function "waitpid\_GDB多进程调试" not defined.

Detaching after fork from child process 32028.

The child process has not exited

The child process is running!

The child process has not exited

The child process has not exited

The child process has not exited

The child process has not exited

The child process is quitted!

The father process quit because the child process quit.

The return of waitpid() is the child process PID : 32028

Program exited normally.

(gdb)

而在./waitpid\_GDB多进程调试Shell命令行执行中两条printf()之间父进程却打印5条The child process has not exited语句？

[tom@localhost ~]$ ./waitpid\_GDB多进程调试

The child process is running!

The child process has not exited

The child process has not exited

The child process has not exited

The child process has not exited

The child process has not exited

The child process is quitted!

The father process quit because the child process quit.

The return of waitpid() is the child process PID : 7059

**答：**

在GDB调试中，因为以默认的调试方式调试waitpid程序，没有敲入set命令，所以先调试父进程，父进程先输出一句“The child process has not exited”后休眠1s，然后再继续执行子程序，输出“The child process is running!”，刚刚已休眠1s，接着在休息4s，所以接着再次输出4句“The child process has not exited”后，子程序执行结束，输出“The child process is quitted!”。

在使用Shell命令行执行时，子进程和父进程并发执行，谁先抢到CPU谁就先输出，不能确定谁先抢到，不同机器结果不同，题目中结果可见，子进程先抢到了CPU的使用权，因此先输出“The child process is running!”，然后再进入休眠5s，父进程每隔1s输出5句“The child process has not exited”，执行完后输出“The child process is quitted!”，子进程执行结束。

**5、守护进程创建的步骤是什么？它有什么作用？**

**答：**

**守护进程创建步骤：**

（1）创建子进程，父进程退出

子进程变成孤儿进程，然后由1号init进程收养。

1. 在子进程中创建新会话

调用setsid创建新的会话，摆脱原会话，原进程组，原终端的控制，自己成为新会话的组长

1. 改变当前目录为根目录

正在运行的进程文件系统不能卸载，如果目录要回退，则此时进程不能做到，为了避免这种麻烦，以根目录为当前目录

1. 关闭文件描述符，守护进程建立成功。

子进程的文件描述符也是从父进程复制来的，那些不需要的文件描述符永远不会被守护进程使用，会白白的浪费系统资源，还可能导致文件系统无法结束

**作用：**

（1）守护进程是一个生存周期较长的进程，通常独立于控制终端并且周期性的执行某种任务或者等待处理某些待发生的事件。

（2）大多数服务都是通过守护进程实现的。

（3）关闭终端，相应的进程都会被关闭，而守护进程却能够突破这种限制

1. **实验体会**

本次实验课，主要使我掌握了fork()函数和waitpid()函数的结构和作用，巩固了理论知识，理论结合实践，知识掌握的更牢固。

本次实验相较于上次实验，动手实践的实验量少了些许，但是需要思考的地方也很多，需要思考实验流程和几个为什么，需要深入理解代码的意思和作用才能回答得出来，考验较大。其中几个问题，光听老师的讲解和参考教材我觉着我还没有理解透彻，于是我搜索了很多资料，以便更好的深入理解。

本次实验，和前几次实验一样，老师给我们详细讲解了理论知识，并在视频中将重要实验一步一步操作给我们看，将实验代码一句一句透析给我们听含义，使我对代码的理解更深入透彻，在自己动手操作时能深深感悟到老师这样做的好处，既掌握了知识，又利于实践。