Linux Homework 2

姓名: 王笑盈　学号: 1252885　日期: 2014年12月

https://github.com/wangxiaoying/linux-homework2

题目 1

题目描述：Is the COW technique same with running in the parent's address space?

解答：

是不同的。
Copy on Write技术是指在fork的时候产生的与父进程相同的子进程只有在进程空间的各段内容发生变化的时候才会将父进程的内容复制一份给子进程，为的是减少不必要的开销。
在父子进程中有更改的行为发生之前，两个进程用的是相同的物理空间，子进程的代码段，数据段，堆栈都是指向父进程的物理空间，即它们的虚拟空间不同但对应的物理空间实际相同。但是当相应的段发生变化时，内核会给子进程分配相应的物理空间，并将父进程的内容复制过来进行修改。
因此，Copy on Write与在父进程空间上运行是完全不同的。

题目 2

题目描述：What are the differences between fork and vfork? Write a program to verify your answer.

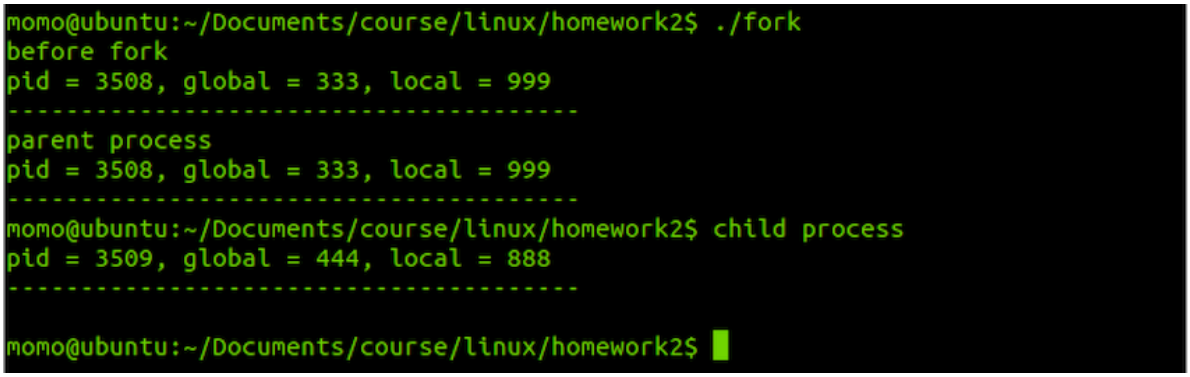
解答：

1. fork的运行先后顺序是不一定的；而vfork保证子进程先运行，在它调用exec或exit之后父进程才可能被调度运行。如果在调用这两个函数之前子进程依赖于父进程的进一步动作，则会导致死锁。
2. fork要拷贝父进程的进程环境资源，这样得到的子进程是独立的，并且拷贝并不会马上进行，如题目1中所讲；而vfork则不需要完全拷贝父进程的进程环境，在子进程没有调用exec和exit之前，子进程与父进程共享进程环境，相当于线程的概念，此时父进程阻塞等待。
　　当需要创建子进程的目的在于调用exec执行一个新程序的时候，如果用fork，那么之前拷贝工作就相当于白费力气了，所以这时就需要用vfork。

提交：

- 源文件：fork.c & vfork.c
- 可执行文件：fork & vfork

执行：



```
momo@ubuntu:~/Documents/course/linux/homework2$ ./vfork
before vfork
pid = 3767, global = 333, local = 999
--------------------------------------
child process
pid = 3768, global = 444, local = 888
--------------------------------------
parent process
pid = 3767, global = 444, local = 888
--------------------------------------
momo@ubuntu:~/Documents/course/linux/homework2$
```

分析证明：

> 由执行情况可知，执行fork程序的时候先执行的是父进程然后才是子进程（实际上在不同环境的linux下先后顺序是不一定的），而执行vfork程序的时候是先执行的子进程再执行的父进程，是因为vfork保证了子进程的先运行。
> 同时vfork中子进程改变了父进程的数据，证明了vfork与父进程共享进程环境，而fork如果子进程先运行的话也是不会改变父进程的数据的，因为fork是拷贝的父进程的环境资源；并且很明显，当fork程序执行时，在执行子进程的时候父进程已经推出，可见子进程是父进程的拷贝，已完全独立于父进程。

题目 3

题目描述：Write a program to show the child process wait the parent process with a block style and a noblock style to terminate. Then show the changing of memory layout.(应该是父进程等子进程吧？)

提交：

- 源文件：wait_block.c & wait_noblock.c
- 可执行文件：wait_block & wait_noblock

执行：

```
momo@ubuntu:~/Documents/course/linux/homework2$ ./wait_block
parent process in
parent start waiting
child process in
.
.
.
.
.

child : pid = 3990, ppid = 3989, pgid = 3989
child process has exited
parent finish waiting
parent : pid = 3989, ppid = 3683, pgid = 3989
momo@ubuntu:~/Documents/course/linux/homework2$
```

```
momo@ubuntu:~/Documents/course/linux/homework2$ ./wait_noblock
parent process in
parent start wait
child process has not exited
child process in
child process has not exited
child process has not exited
child process has not exited
child process has not exited
child : pid = 3995, ppid = 3994, pgid = 3994
child process has exited
parent finish waiting
parent : pid = 3994, ppid = 3683, pgid = 3994
momo@ubuntu:~/Documents/course/linux/homework2$
```

题目 4

题目描述：How can you write a shell? Practice it.

提交：

- 源文件：shell.c
- 可执行文件：shell

功能：

- 能够显示当前登录用户
- 能够显示当前路径
- 可以执行shell下的命令，包括切换工作目录等
- 输入exit退出程序

执行：

```
momo@ubuntu:~/Documents/course/linux/homework2$ ./shell
momo:/home/momo/Documents/course/linux/homework2$ cd ..
momo:/home/momo/Documents/course/linux$ ls
2.fork.c  atexit.c  homework1  jmp    signal    signal.c~  vfork.c
atexit    fork      homework2  jmp.c  signal.c  vfork      vfork.c~
momo:/home/momo/Documents/course/linux$ pwd
/home/momo/Documents/course/linux
momo:/home/momo/Documents/course/linux$ ps
  PID TTY          TIME CMD
 2202 pts/2    00:00:00 bash
 2309 pts/2    00:00:00 shell
 2350 pts/2    00:00:00 sh
 2351 pts/2    00:00:00 ps
momo:/home/momo/Documents/course/linux$ exit
momo@ubuntu:~/Documents/course/linux/homework2$
```