# 第 4 章 进程管理

## 4.1 实验 4-1：fork 和 clone

### 1．实验目的

了解和熟悉 Linux 中 fork 系统调用和 clone 系统调用的用法。

### 2．实验要求

1）使用 fork()函数创建一个子进程，然后在父进程和子进程中分别使用 printf 语句来判断谁是父进程和子进程。

2）使用 clone()函数创建一个子进程。如果父进程和子进程共同访问一个全局变量，结果会如何？如果父进程比子进程先消亡，结果会如何？

### 3．实验步骤

（1）fork 例子

（2）clone 例子

下面是本实验的实验步骤。

进入本实验的参考代码目录进行交叉编译。

```
cd /home/lab466/runninglinuxkernel_4.0/rlk_lab/rlk_basic/chapter_8/lab1
export ARCH=arm
export CROSS_COMPILE=arm-linux-gnueabi-
```

编译 test 测试 app。然后把它拷贝到 runninglinuxkernel_4.0/kmodules 目录下面。

```
    arm-linux-gnueabi-gcc clone_test.c -o clone_test  --static
    cp clone_test /home/lab466/runninglinuxkernel_4.0/kmodules
```

启动 QEMU+runninglinuxkernel。最好另外开一个窗口，运行：

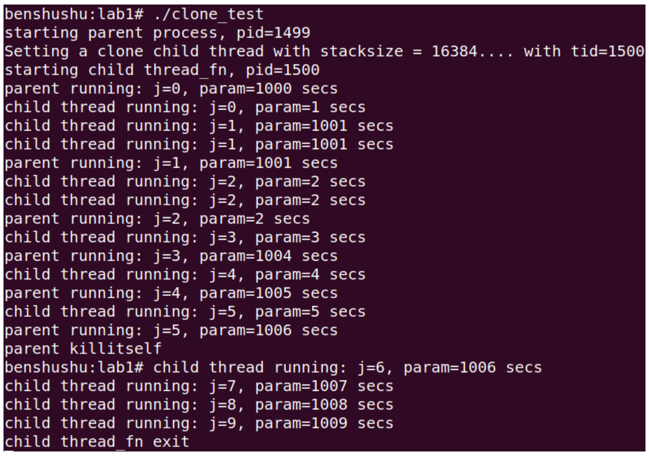
```
    sudo su
cd /home/lab466/runninglinuxkernel_4.0
    sh run.sh arm32
```

进入本实验的参考代码。

```
    # cd /mnt
```

运行 clone_test 程序。

```
benshushu:lab1# ./clone_test
starting parent process, pid=1499
Setting a clone child thread with stacksize = 16384.... with tid=1500
starting child thread_fn, pid=1500
parent running: j=0, param=1000 secs
child thread running: j=0, param=1 secs
child thread running: j=1, param=1001 secs
child thread running: j=1, param=1001 secs
parent running: j=1, param=1001 secs
child thread running: j=2, param=2 secs
child thread running: j=2, param=2 secs
parent running: j=2, param=2 secs
child thread running: j=3, param=3 secs
parent running: j=3, param=1004 secs
child thread running: j=4, param=4 secs
parent running: j=4, param=1005 secs
child thread running: j=5, param=5 secs
parent running: j=5, param=1006 secs
parent killitself
benshushu:lab1# child thread running: j=6, param=1006 secs
child thread running: j=7, param=1007 secs
child thread running: j=8, param=1008 secs
child thread running: j=9, param=1009 secs
child thread_fn exit
```



```
benshushu:lab1# ./clone_test
starting parent process, pid=1499
Setting a clone child thread with stacksize = 16384.... with tid=1500
starting child thread_fn, pid=1500
parent running: j=0, param=1000 secs
child thread running: j=0, param=1 secs
child thread running: j=1, param=1001 secs
child thread running: j=1, param=1001 secs
parent running: j=1, param=1001 secs
child thread running: j=2, param=2 secs
child thread running: j=2, param=2 secs
parent running: j=2, param=2 secs
child thread running: j=3, param=3 secs
parent running: j=3, param=1004 secs
child thread running: j=4, param=4 secs
parent running: j=4, param=1005 secs
child thread running: j=5, param=5 secs
parent running: j=5, param=1006 secs
parent killitself
benshushu:lab1# child thread running: j=6, param=1006 secs
child thread running: j=7, param=1007 secs
child thread running: j=8, param=1008 secs
child thread running: j=9, param=1009 secs
child thread_fn exit
```