CS356 **Operating System Projects** **Spring 2016**

Project 2: Android scheduler

Objectives:

- Compile the Android kernel.
- Familiarize Android scheduler
- Implement a random policy in round robin scheduler.
- Get experience with software engineering techniques.

Make sure your system is 64-bits system.

Problem Statement:

1. **Compile the Linux kernel.**
   a. Make sure that you have added the following path into your environment variable.

      ANDROID_NDK_HOME/toolchains/arm-linux-androideabi-4.6/prebuilt/linux-x86_64/bin

   b. Open Makefile in KERNEL_SOURCE/goldfish/, and find these:

      ```
      ARCH            ?=  $(SUBARCH)
      CROSS_COMPILE   ?=
      ```

      Change it to:

      ```
      ARCH            ?=  arm
      CROSS_COMPILE   ?=  arm-linux-androideabi-
      ```

      Save it and exit.

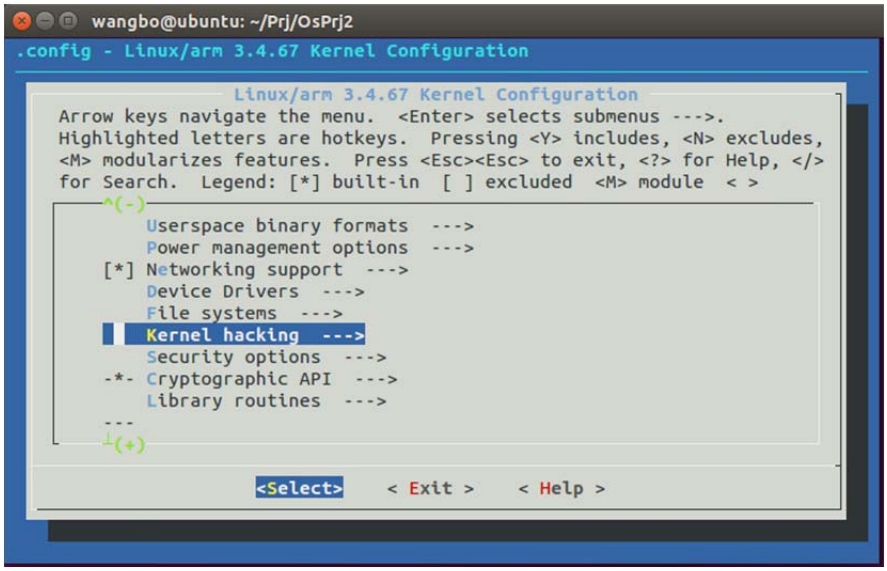
c. Execute the following command in terminal to set compiling config:

```
make goldfish_armv7_defconfig
```

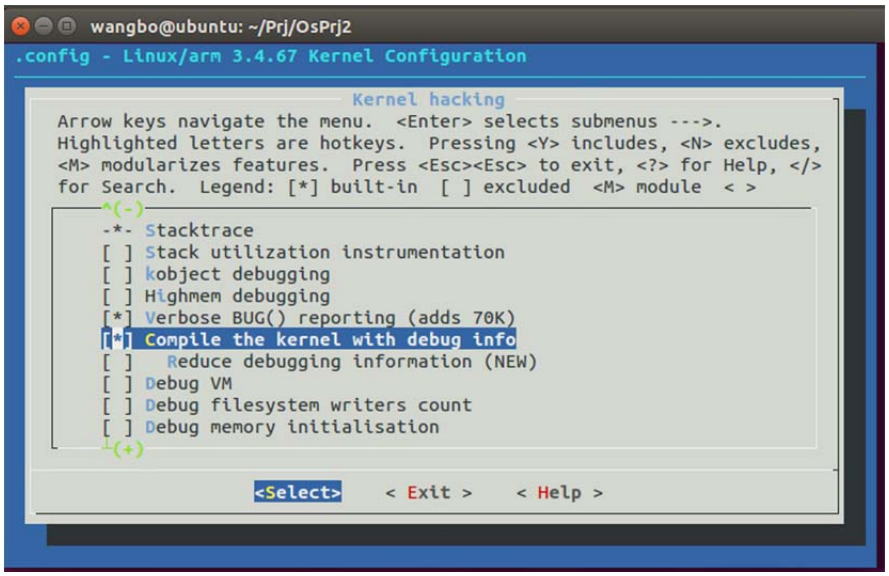
d. Modify compiling config:

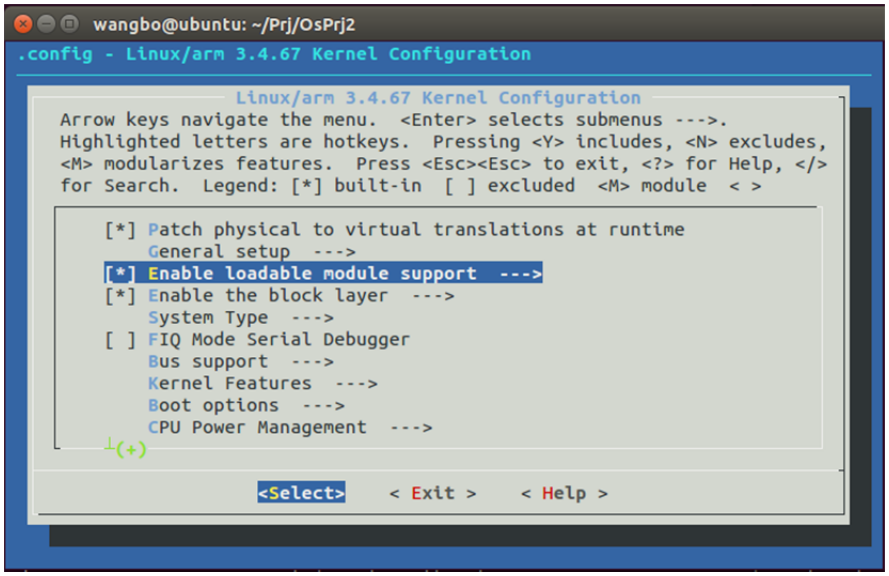
```
sudo apt-get install ncurses-dev
```

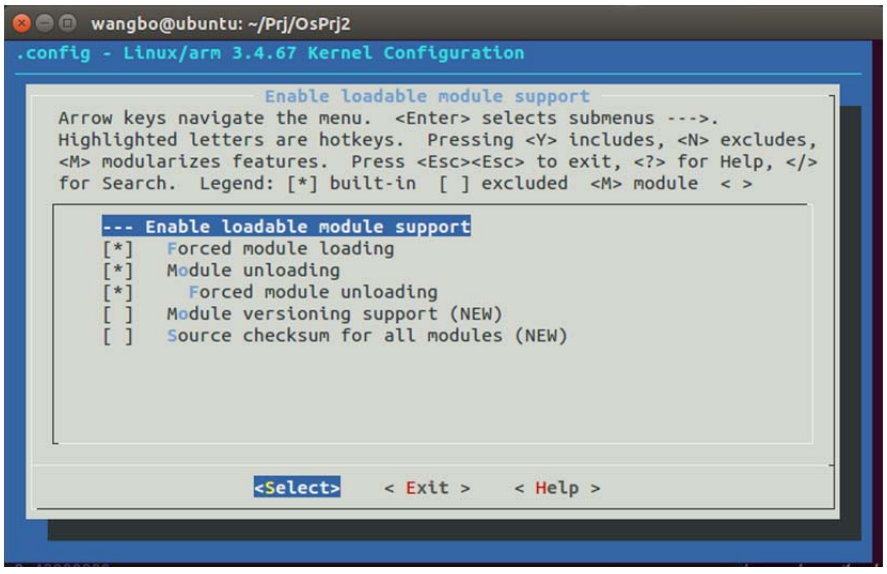
```
make menuconfig
```



Then you can see a GUI config.







Open the *Compile the kernel with debug info* in *Kernel hacking* and *Enable*

loadable module support with *Forced module loading, Module unloading and Forced module unloading* in it.

Save it and exit.

e. Compile

make -j4

The number of -j* depends on the number of cores of your system.

2. **Change the scheduler of processes.**

Android system has three kind of scheduler: SCHED_NORMAL, SCHED_RR, SCHED_FIFO. The SCHED_RR and SCHED_FIFO are real time scheduler, and SCHED_NORMAL is not. The SCHED_NORMAL is the default scheduler of every processes. In these problem, you should change the scheduler of some process, observe the difference of them by comparing the running time of some Android applications and Linux applications.

To test the performance of different scheduler, an Android application is supported on website [1]. It can show the executing time of it. Some similar executable file of Linux is essential. But it needs to be done by yourself.

You should finish the following task in this problem:

1. Change the scheduler of test applications to SCHED_FIFO, and compare the

executing time of them with the time using SCHED_NORMAL. The priorities of them should be same.

2. Change the scheduler of test applications to SCHED_RR, and compare the executing time of them with the time using SCHED_NORMAL. The priorities of them should be same.
3. Change the scheduler of all descendants of process zygote to SCHED_RR, and compare the executing time of them with the time using SCHED_NORMAL. The priority of any process exclude test application should be same.

You should repeat the above task several times to observe the result, record the result and explain the result in your report.

Tips:

You can use adb install xxx.apk to install application on avd through terminal.

3. Modify Scheduler

Now you have familiar with the schedulers in android. In these problem, you should change the schedule policy in kernel to satisfied the following requirement:

a. Default scheduler of all descendants of process zygote should be SCHED_RR. The priority of process should be $\frac{max\ priority\ of\ SCHED\_RR}{5} \times (PID\ mod\ 5) + 1$.

b. Change the policy of SCHED_RR to pick the next process randomly.

To test the new scheduler, you should set the priority of the android application as a certain number. Then, execute the two applications repeatedly to observe the difference between new scheduler and original scheduler.

Implementation Details:

In general, the execution of any of the programs above will is carried out by specifying the executable program name followed by the command line arguments.

1. When using system or library calls you have to make sure that your program will exit gracefully when the requested call cannot be carried out.
2. One of the dangers of learning about forking processes is leaving unwanted processes active and wasting system time. Make sure each process terminates cleanly when processing is completed. Parent process should wait until the child processes complete, print a message and then quit.
3. Your program should be robust. If any of the calls fail, it should print error message and exit with appropriate error code. Always check for failure when invoking a system or library call.

Material to be submitted:

1. Compress the source code of the programs into **Prj2+StudentID.tar** file. It contains all

*.c, *.h files you have changed in Linux kernel. Use meaningful names for the file so that the contents of the file are obvious. Enclose a README file that lists the files you have submitted along with a one sentence explanation. Call it **Prj2README**.

2. Only internal documentation is needed. Please state clearly the purpose of each program at the start of the program. Add comments to explain your program. (-5 points, if insufficient.)
3. Test runs: Screen captures of the scheduler test to show that your scheduler works. Execution time of every tests.
4. A project report of the project2. In this report, you should compare and figure out the difference between three scheduling class; explain how you complete the project, and analysis the result of all your test.
5. Send your **Prj2+StudentID.tar** file to cs356.sjtu@gmail.com.
6. Due date: **Jun. 24, 2016**, submit on-line **before midnight**.
7. Demo slots: Jun. 25-26, 2016. Demo slots will be posted on the door of East 3-309 SEIEE Building. Please sign your name in one of the available slots.
8. You are encouraged to present your design of the project optionally. The presentation date is Jun. 26, 2016. Please pay attention to the course website and mailing list.

Appendix

[1] http://www.cs.sjtu.edu.cn/~fwu/teaching/res/processtest.apk