

Project 2: Android scheduler

Fan Wu and Bo Wang

Department of Computer Science and Engineering

Shanghai Jiao Tong University

Spring 2016

Objectives

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

Enviroment

■ Implementation

- AVD(Android Virtual Devices)
 - ▶ SDK version r24.4.1

■ Development

- Linux (64-bits)
 - ▶ Ubuntu (recommended)
 - ▶ Debian
 - ▶ Fedora

What to Submit

- A “tar” file of your DIRECTORY, containing:
 - All *.c, *.h files you have changed in Linux kernel.
 - Any “readme” or “.pdf” files asked for in the project
 - Screen captures of the scheduler test
 - ▶ If you cannot get your program to work, submit a run of whatever you can get to work as you can get partial credit

- **DO NOT SUBMIT** your object or executable files, **REMOVE** them before you pack your directory.

How to Submit

- Pack your code in a project directory
`tar -cvf Prj2+StudentID.tar project1`
- Send your `Prj2+StudentID.tar` file to
`cs356.sjtu@gmail.com`

For Help?

■ Teaching Assistant

- Bo Wang

- ▶ Email: wangbo0727@outlook.com, wangbo0727@126.com
- ▶ WeChat: hadesghost727

- Jiapeng Xie

- ▶ Email: jsxiejp@163.com
- ▶ WeChat: xjp18248794518

Compile the Linux Kernel

- Make sure that your environment variable is correct.

```
export JAVA_HOME=/usr/lib/jdk1.8.0_73
export JRE_HOME=/usr/lib/jdk1.8.0_73/jre
export CLASSPATH=.:$CLASSPATH:$JAVA_HOME/lib:$JRE_HOME/lib
export PATH=$PATH:$JAVA_HOME/bin:$JRE_HOME/bin
export PATH=~/.Kit/android-sdk-linux/platform-tools:$PATH
export PATH=~/.Kit/android-sdk-linux/tools:$PATH
export PATH=~/.Kit/android-ndk-linux:$PATH
export PATH=~/.Kit/android-ndk-linux/toolchains/arm-linux-androideabi-4.9/prebuilt/linux-x86_64/bin:$PATH
```

Compile the Linux Kernel (cont.)

■ Modify Makefile in the kernel

● Change

- ▶ ARCH ?= \$(SUBARCH)
- ▶ CROSS_COMPILE ?=

● To

```
export KBUILD_BUILDHOST := $(SUBARCH)
ARCH                ?= arm
CROSS_COMPILE       ?= arm-linux-androideabi-
# Architecture as present in compile.h
```

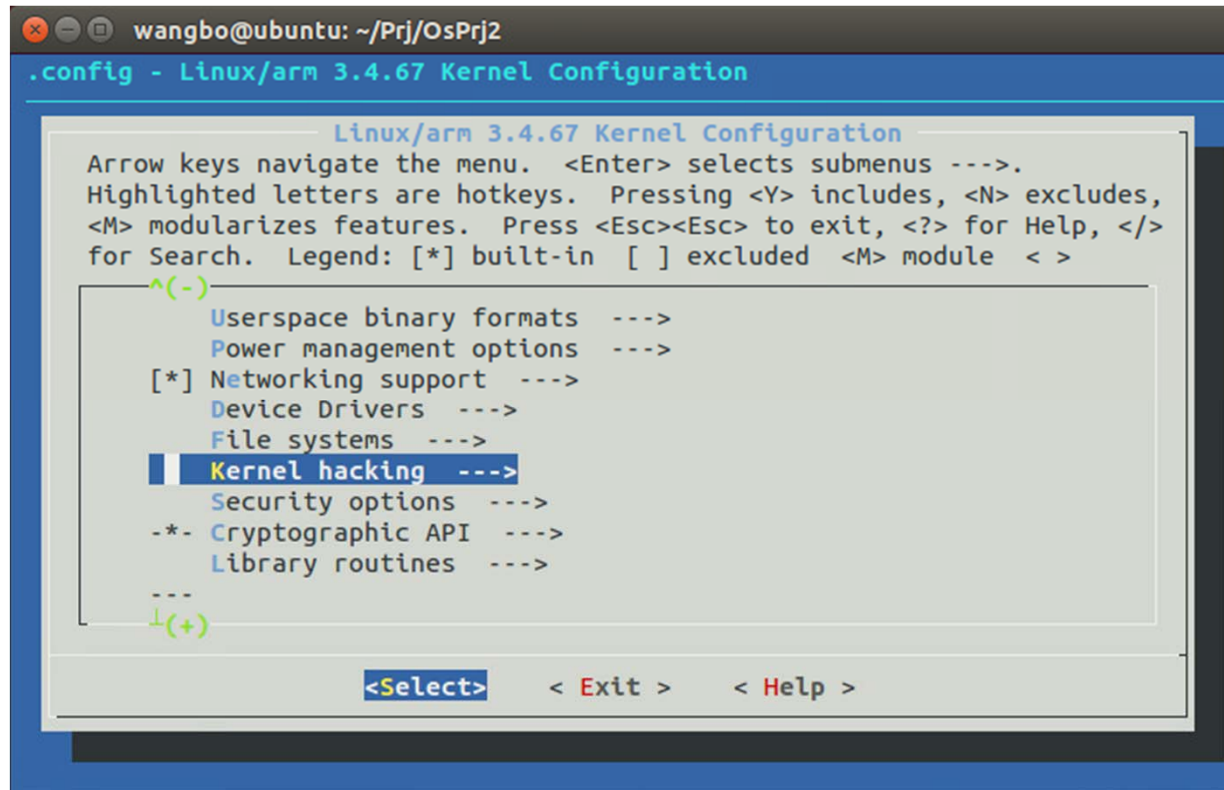

Compile the Linux Kernel (cont.)

- Execute the following command:

```
wangbo@ubuntu:~/Prj/0sPrj2$ make goldfish_armv7_defconfig
#
# configuration written to .config
#
wangbo@ubuntu:~/Prj/0sPrj2$ sudo apt-get install ncurses-dev
wangbo@ubuntu:~/Prj/0sPrj2$ make menuconfig
```

Compile the Linux Kernel (cont.)

- Then you can see:



The screenshot shows a terminal window titled 'wangbo@ubuntu: ~/Prj/OsPrj2' with a sub-window titled '.config - Linux/arm 3.4.67 Kernel Configuration'. The main window displays the 'Linux/arm 3.4.67 Kernel Configuration' menu. It includes instructions on how to navigate the menu using arrow keys, enter, and other shortcuts. The menu items are listed with their status: 'Userspace binary formats' (--->), 'Power management options' (--->), '[*] Networking support' (--->), 'Device Drivers' (--->), 'File systems' (--->), 'Kernel hacking' (---> and highlighted with a blue bar), 'Security options' (--->), '-*- Cryptographic API' (--->), and 'Library routines' (--->). At the bottom, there are three buttons: '<Select>', '<Exit >', and '<Help >'. Navigation keys like '^(-)' and 'l(+)' are also visible.

```
wangbo@ubuntu: ~/Prj/OsPrj2
.config - Linux/arm 3.4.67 Kernel Configuration

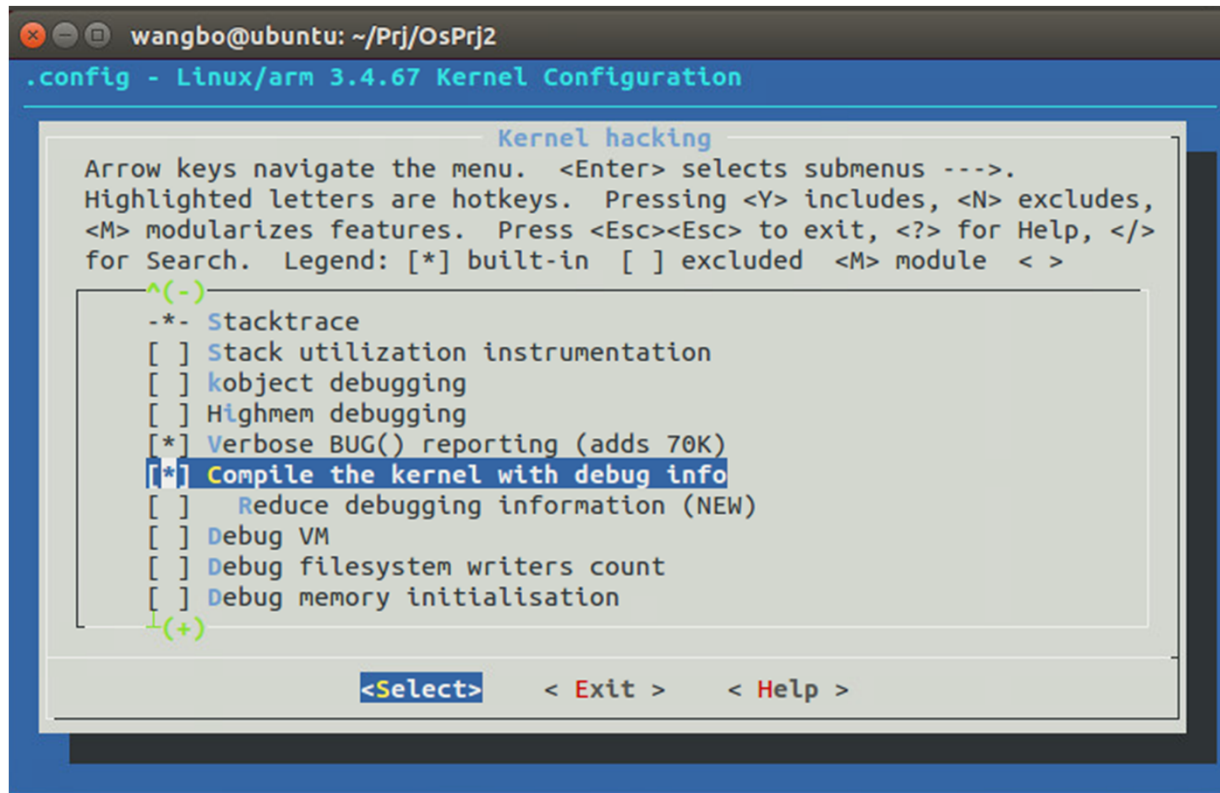
Linux/arm 3.4.67 Kernel Configuration
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes,
<M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </>
for Search. Legend: [*] built-in [ ] excluded <M> module < >

^(-)
  Userspace binary formats --->
  Power management options --->
  [*] Networking support --->
  Device Drivers --->
  File systems --->
  Kernel hacking --->
  Security options --->
  -*- Cryptographic API --->
  Library routines --->
l(+)
```

<Select> <Exit > <Help >

Compile the Linux Kernel (cont.)

- Then you can see:



```
wangbo@ubuntu: ~/Prj/OsPrj2
.config - Linux/arm 3.4.67 Kernel Configuration

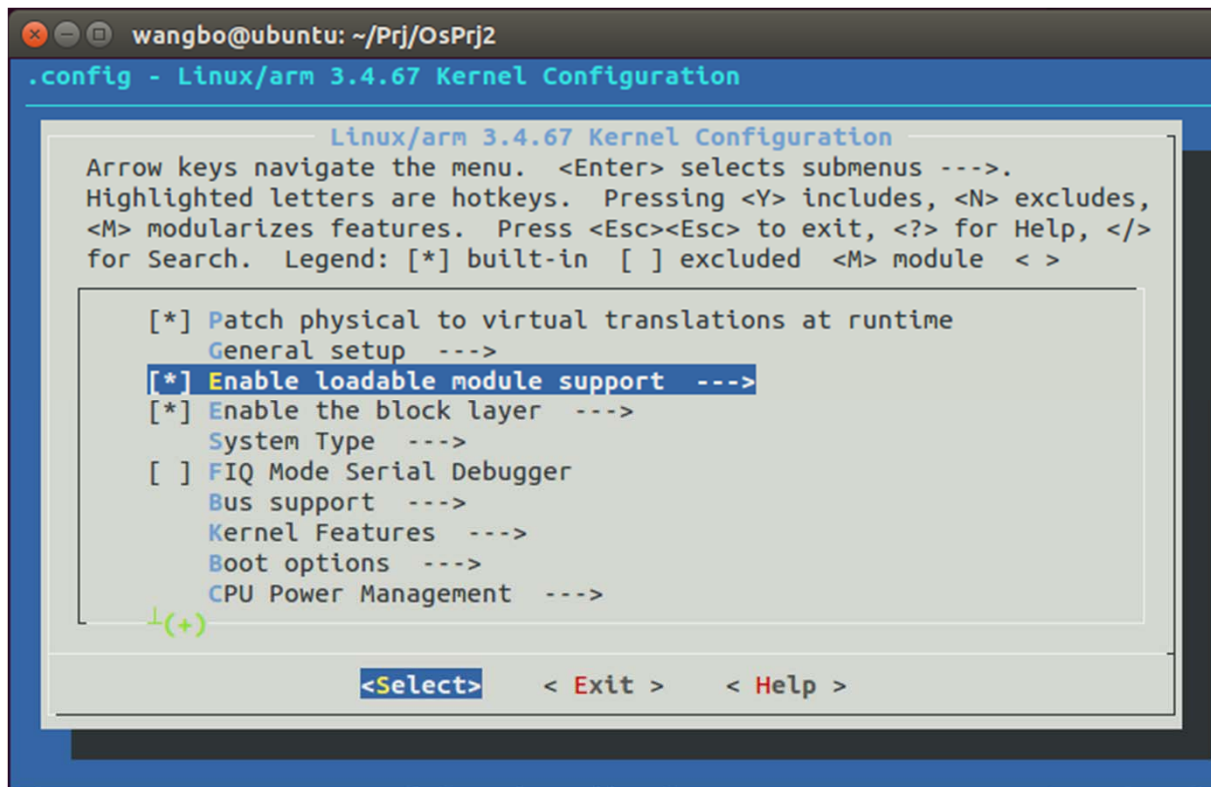
Kernel hacking
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes,
<M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </>
for Search. Legend: [*] built-in [ ] excluded <M> module < >

^(-)
-* Stacktrace
[ ] Stack utilization instrumentation
[ ] kobject debugging
[ ] Highmem debugging
[*] Verbose BUG() reporting (adds 70K)
[*] Compile the kernel with debug info
[ ] Reduce debugging information (NEW)
[ ] Debug VM
[ ] Debug filesystem writers count
[ ] Debug memory initialisation
(+)
```

<Select> < Exit > < Help >

Compile the Linux Kernel (cont.)

- Then you can see:



The screenshot shows a terminal window titled 'wangbo@ubuntu: ~/Prj/OsPrj2'. Inside, the 'Linux/arm 3.4.67 Kernel Configuration' menu is displayed. The menu is a ncurses-style interface with a title bar and a list of configuration options. The current selection is '[*] Enable loadable module support --->'. The menu includes instructions on how to navigate and use the interface. The options listed are: '[*] Patch physical to virtual translations at runtime', 'General setup --->', '[*] Enable loadable module support --->', '[*] Enable the block layer --->', 'System Type --->', '[] FIQ Mode Serial Debugger', 'Bus support --->', 'Kernel Features --->', 'Boot options --->', and 'CPU Power Management --->'. At the bottom, there are three buttons: '<Select>', '<Exit >', and '<Help >'. A green cursor is visible at the bottom left of the menu area.

```
wangbo@ubuntu: ~/Prj/OsPrj2
.config - Linux/arm 3.4.67 Kernel Configuration

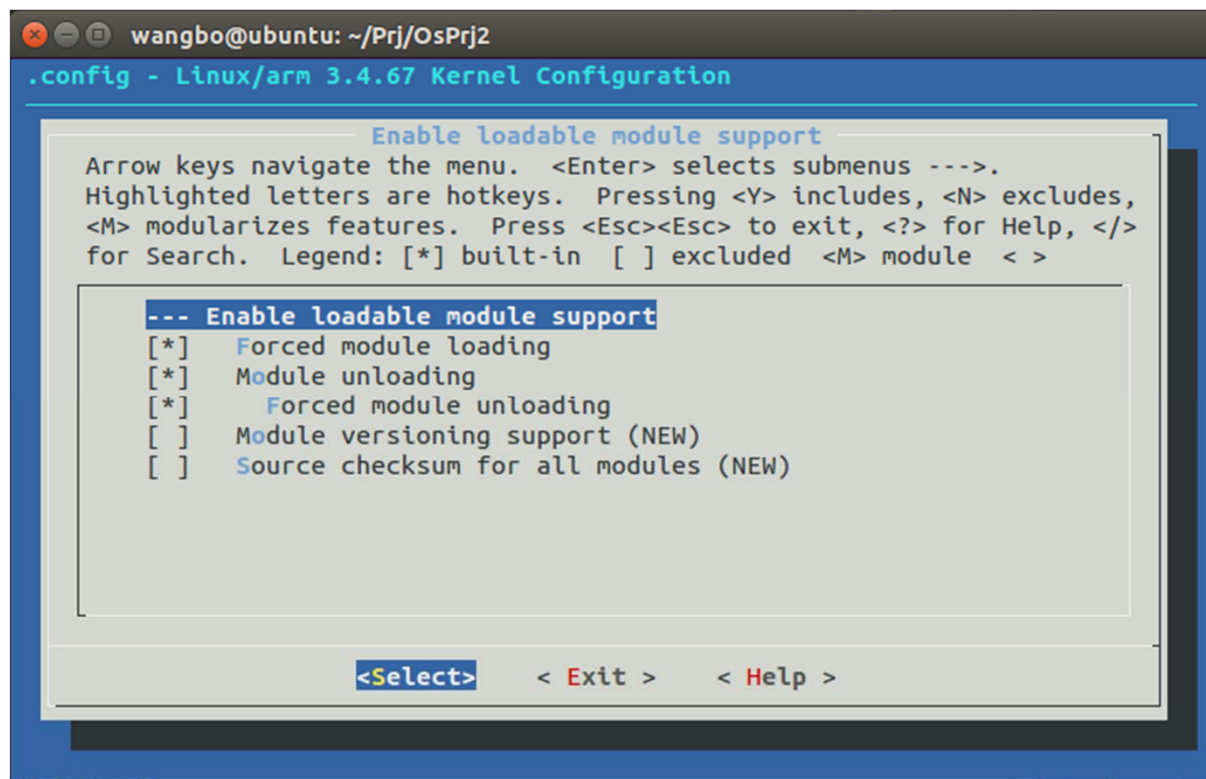
Linux/arm 3.4.67 Kernel Configuration
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes,
<M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </>
for Search. Legend: [*] built-in [ ] excluded <M> module < >

[*] Patch physical to virtual translations at runtime
    General setup --->
[*] Enable loadable module support --->
[*] Enable the block layer --->
    System Type --->
[ ] FIQ Mode Serial Debugger
    Bus support --->
    Kernel Features --->
    Boot options --->
    CPU Power Management --->

<Select>  <Exit >  <Help >
```

Compile the Linux Kernel (cont.)

- Then you can see:



```
wangbo@ubuntu: ~/Prj/OsPrj2
.config - Linux/arm 3.4.67 Kernel Configuration

--- Enable loadable module support
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes,
<M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </>
for Search. Legend: [*] built-in [ ] excluded <M> module < >

--- Enable loadable module support
[*] Forced module loading
[*] Module unloading
[*] Forced module unloading
[ ] Module versioning support (NEW)
[ ] Source checksum for all modules (NEW)

<Select> < Exit > < Help >
```

Compile the Linux Kernel (cont.)

■ Compile it

```
wangbo@ubuntu:~/Prj/0sPrj2$ make -j4
SYSMAP    System.map
SYSMAP    .tmp_System.map
OBJCOPY   arch/arm/boot/Image
Kernel:   arch/arm/boot/Image is ready
GZIP      arch/arm/boot/compressed/piggy.gzip
AS         arch/arm/boot/compressed/piggy.gzip.o
LD         arch/arm/boot/compressed/vmlinux
OBJCOPY   arch/arm/boot/zImage
Kernel:   arch/arm/boot/zImage is ready
wangbo@ubuntu:~/Prj/0sPrj2$
```

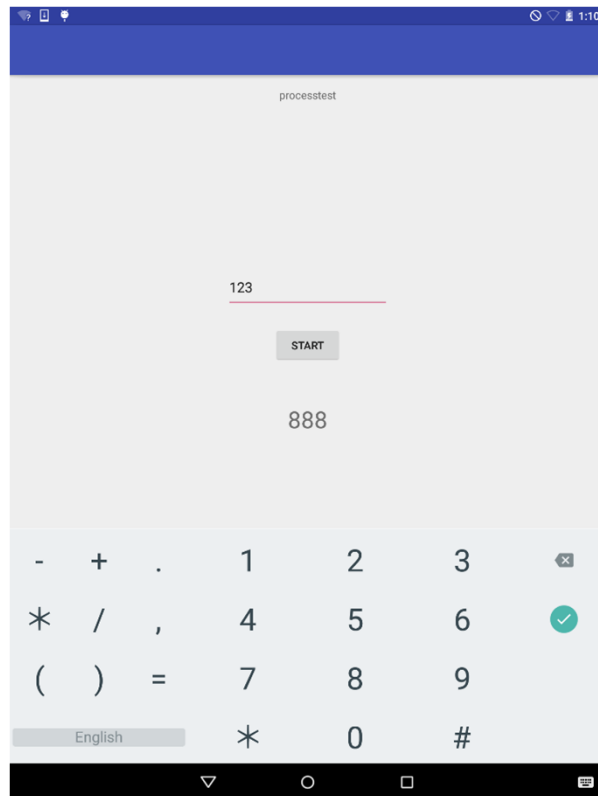
Scheduler in Android

- Android system has three kind of scheduler:
 - SCHED_NORMAL
 - SCHED_RR
 - SCHED_FIFO

Scheduler in Android (cont.)

- You should finish the following task in this problem:
 - 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.
 - 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.
 - 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.
- `sched_getscheduler()` / `sched_setscheduler()`

Scheduler in Android (cont.)



To install:
adb install processtest.apk

To show information:
ps -P

Modify Scheduler

- Change the schedule policy in kernel:
 - **Default scheduler** of all descendants of process zygote should be SCHED_RR. The priority of process should be $\frac{\text{max priority of SCHED_RR}}{5} \times (PID \bmod 5) + 1$.
 - Change the policy of SCHED_RR to pick the next process **randomly**

Modify Scheduler (cont.)

■ Scheduler in kernel

- You can find the scheduling class of SCHED_RR in [kernel/sched/rt.c](#)
- You can learn how the Linux scheduler works in [kernel/sched/core.c](#) and [include/linux/sched.h](#)

Modify Scheduler (cont.)

```
const struct sched_class rt_sched_class = {
    .next          = &fair_sched_class,
    .enqueue_task   = enqueue_task_rt,
    .dequeue_task   = dequeue_task_rt,
    .yield_task     = yield_task_rt,

    .check_preempt_curr = check_preempt_curr_rt,

    .pick_next_task   = pick_next_task_rt,
    .put_prev_task    = put_prev_task_rt,

#ifdef CONFIG_SMP
    .select_task_rq   = select_task_rq_rt,

    .set_cpus_allowed = set_cpus_allowed_rt,
    .rq_online        = rq_online_rt,
    .rq_offline       = rq_offline_rt,
    .pre_schedule     = pre_schedule_rt,
    .post_schedule    = post_schedule_rt,
    .task_woken       = task_woken_rt,
    .switched_from    = switched_from_rt,
#endif

    .set_curr_task    = set_curr_task_rt,
    .task_tick        = task_tick_rt,

    .get_rr_interval  = get_rr_interval_rt,

    .prio_changed     = prio_changed_rt,
    .switched_to      = switched_to_rt,
};
```

Modify Scheduler (cont.)

- Test the performance of new scheduler
 - Set the priority of the android application as a certain number ($>$, $=$, $<$).
 - Execute the two applications repeatedly to observe the difference.

Report

- Compare and figure out the difference between three **scheduling class**.
- Explain how you programs work.
- Analysis the result of your test.

Demo & Presentation

■ Demo:

- **June 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.

■ Presentation:

- You are encouraged to present your design of the project optionally. The presentation date is Jun. 26, 2016.

For Help?

■ Teaching Assistant

- Bo Wang

- ▶ Email: wangbo0727@outlook.com, wangbo0727@126.com
- ▶ WeChat: hadesghost727

- Jiapeng Xie

- ▶ Email: jsxiejp@163.com
- ▶ WeChat: xjp18248794518

■ Some useful website

- <http://www.csdn.net/>
- <http://stackoverflow.com/>
- <http://developer.android.com/>

For Help?

Mid-night, June. 24, 2016
Q&A