ECE777Q

IO Scheduler Guideline

Instructor : Dr. Sejun Song

TA : Danny Kim

Wichita State University

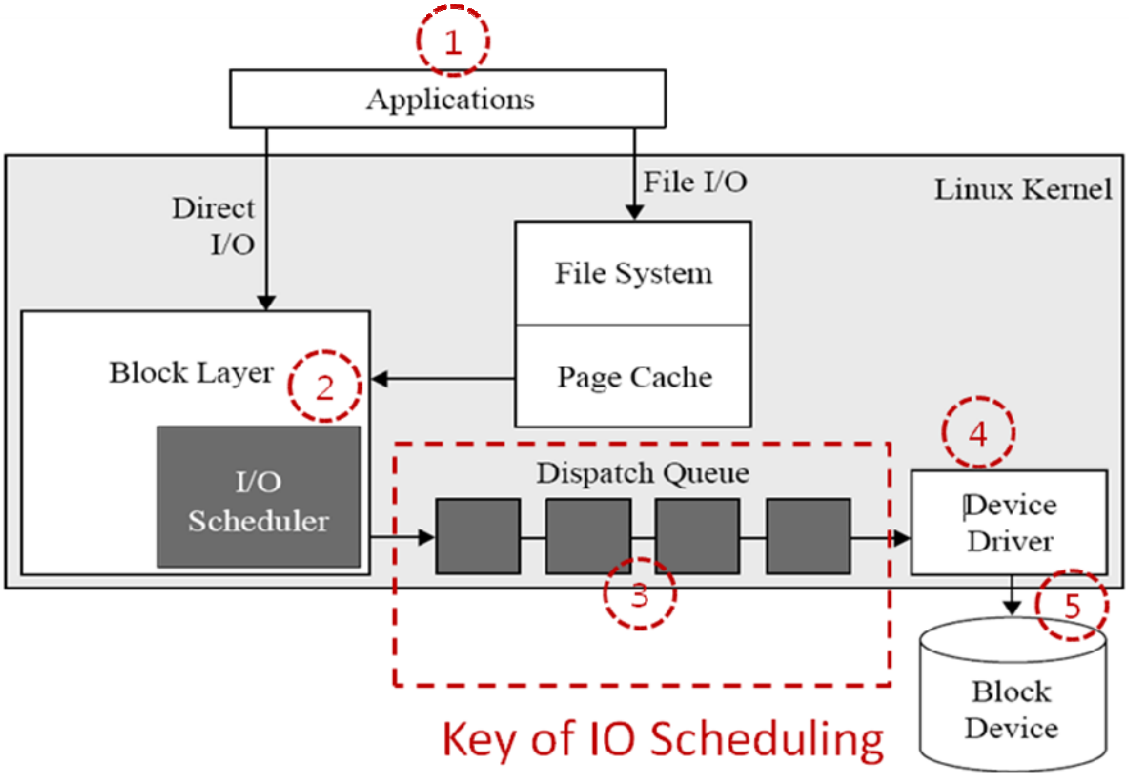
[Purpose]

This document is for students who want to use IO Scheduler for your project, or who want to add enhanced IO scheduler into the scope of your project. This document will show you how to run your IO scheduler into your Linux system, how to check your IO scheduler, and related ideas.
For this guideline, as I mentioned at class, the FIFO IO scheduler which was made by Aaron (for his thesis) will be used.

You can find two IO schedulers which were created by him at following URL.
http://www.gelato.unsw.edu.au/IA64wiki/IOScheduling

[Flow of IO request]

You need to understand the flow of request when you create a file. You have learned how to change device driver program, and where a device driver module (which was already loaded into Linux kernel) gets requests; that is a request queue.



Please remember the upper steps are occurred when data are written into disk and data are read from disk. Read or writing request could be occurred at command line or at your program. For example, when you create a file by using VI editor, your writing request goes to I/O scheduler in Block Layer. Then, I/O scheduler inserts the request into request queue (or dispatch queue). After

that, the device driver for block device (for example, hard disk) retrieves requests from request queue. Then, data are written to device based on the writing request.

Here, the thing is that I/O scheduler does something with requests in request queue in order to get efficient IO. Please see "Let's make your IO-scheduler" section at lecture note at second device driver class. Something includes merging and sorting. If you want to change the merging and sorting for request queue, and if you want to add some functions for request queue, that means you need to change codes inside IO scheduler. You might need to create your own IO scheduler and add it into Linux kernel

At this practice, the part that you will deal with among the flow of IO requests is the "2. IO Scheduler at Block Layer"

As I told you at class, current Linux has four IO schedulers including Noop (it means no-operation), deadline, anticipatory, and CFQ (complete fair queuing). Four IO schedulers have both sorting and merging functions. The complexity will increase from left to right (^^ Noop → CFQ). Currently, the default Linux IO scheduler for your hard disk is CFQ if you do not change your default IO Scheduler. It might be important for you to understand that you can select a different IO scheduler for a different hard disk; that means, if you have two hard disks, you can assign CFQ IO scheduler for one hard disk, and you can choose anticipatory IO scheduler for the other hard disk. (You will see that how to assign IO scheduler for a hard disk at this practice).

Okay. To see is to believe. Let's start. You will use one (FIFO IO scheduler) of sources which Aaron made. We will assume you already created FIFO IO scheduler which does not have sorting and merging function. You will follow the step "How can you make your own IO scheduler" that I showed you at class.

If you do not understand what I said just before, how about looking at lecture notes as well as Aaron's thesis. Then, just following steps without any understanding might not be useful for you.

[STEPs]

This document consists of followings steps.

- Go to the directory of IO Schedulers
  - You will see what kind of IO Schedulers are there in Linux.

- Copy one to your file & Modify the code
  - We will assume you did this step. Just you will use FIFO IO Scheduler that is given to you.

- Modify configuration
  - You will modify “Kconfig.iosched“ and “Makefile“

- Recompile kernel source
  - You learned how to recompile Linux kernel. Do recompile after modifying configuration based on Linux practice guideline.

[Materials]

You will gain following materials. All materials were gained from Aaron. So, when you use these materials, you need to specify his name at your documents as well as your source codes.

- All source codes and documents for his thesis
- fifo_iosched.c
- Kconfig.iosched
- Makefile

STEP 1. Go to the directory of IO Schedulers

The source codes of all IO schedulers in Linux exist in the directory of Block Layer. You need to move the directory to see which IO schedulers your Linux system does have.

[Check current IO scheduler]

- Before going to the directory of IO schedulers, why don't we check the current IO scheduler of your Linux system?
- Type following command for that
  - **cat /sys/block/<your hard disk device name>/queue/scheduler**

  [expected screen capture]
  [root@ds ~]# cat /sys/block/sda/queue/scheduler
  noop anticipatory deadline **[cfq]**

  ※ What is <your hard disk device name> ?
  You learned about what is device file name. A common hard disk device name in Linux is "sd?" or "hd?". Therefore, first hard disk is named as "sd**a**" or "hd**a**".
  Then, what is the second hard disk name? Oh, right.. "sd**b**" or "hd**b**" …
  Then, what is the third hard disk name? "sd**c**" or "hd**c**" … so on…

  ※ How can you know what hard disk devices are in my Linux system ?
  You can use "df –h"(if the hard disk are mounted) or "iostat" command.

- Then, let's suppose you want to change the IO scheduler for "sda" hard disk to "deadline".
- How can you do that? Type as followings
  - **echo "<wanted IO scheduler>" > /sys/block/<hard disk>/queue/scheduler**

  [expected screen capture]
  [root@ds ~]# echo "deadline" > /sys/block/sda/queue/scheduler
  [root@ds ~]# cat /sys/block/sda/queue/scheduler
  noop anticipatory **[deadline]** cfq

  - The IO scheduler of "sda" hard disk was changed from "cfq" to "deadline"
  - You will use this command later.

[Let's go to the Block Layer directory]

- Oaky, it was warming up. Now, let's get to the point. The directory of Block Layer is

"/usr/src/<your Linux source directory>/block". Now you know what the Linux source directory is. Right?? ^^ You learned about that from Linux practice.

- Go to the directory
  - **[root@ds ~]# cd /usr/src/linux-2.6.25.20/block**
    [**root@ds block]# ls**
    as-iosched.c blk-exec.c blk-map.o blk-sysfs.o bsg.o
    deadline-iosched.o **Kconfig.iosched**
    as-iosched.o blk-exec.o blk-merge.c blk-tag.c built-in.o
    elevator.c genhd.c modules.order blk-barrier.c blk.h blk-merge.o
    blk-tag.o **cfq-iosched.c** elevator.o genhd.o **noop-iosched.c**
    blk-barrier.o blk-ioc.c blk-settings.c blktrace.c cfq-iosched.o
    ioctl.c noop-iosched.o blk-core.c blk-ioc.o
    blk-settings.o blktrace.o compat_ioctl.c ioctl.o scsi_ioctl.c
    blk-core.o blk-map.c blk-sysfs.c bsg.c **deadline-iosched.c** Kconfig **Makefile**
    scsi_ioctl.o

- You see IO scheduler sources for your system.
  - as-iosched.c : anticipatory
  - cfq-iosched.c : CFQ
  - noop-iosched.c : noop
  - deadline-iosched.c : deadline
  - elevator.c : a file for merging and sorting. Merging and sorting functions of this file will be called from upper four IO schedulers
- Also, you see configuration files including Kconfig.iosched as well as Makefile which you will change at later step.

[Where is the header file for those sources]

You might need to create your own function in upper source codes. In that case, you need to define your own function in header file corresponding to the source file (.c).
When you see "<linux/XXX.h>" at your C file, the header file, XXX.h is located at /usr/src/<your linux kernel source directory>/include/linux.
So, go to the directory and define your own function in the header file. Let's try

- Go to the directory which has Linux header file. Can you see "elevator.h" file?
  - **cd /usr/src/<your linux kernel source directory>/include/linux**

[expected screen shot]
[root@ds ~]# cd /usr/src/linux-2.6.25.20/include/linux
[root@ds linux]# ls elevator.h
elevator.h

[Wicked question]

I said that you can assign different IO scheduler per each hard disk. Then, can you assign different IO schedulers like followings

- "sda" – "deadline" IO scheduler
- "sdb" – "noop" IO scheduler

You might be able to measure the performance by using different IO scheduler per each hard disk.

STEP 2. Copy one to your file & Modify

You will copy fifo-iosched.c that was provided by Aaron into Block Layer Directory

[Copy fifo-iosched.c to Block Layer Directory]

- Now, you need to know about the meaning of Block Layer. When we say about IO Schedulers, they can be found at a directory. The directory which you already saw at STEP 1 is called Block Layer because it contains IO schedulers for block device. Does it make sense? ^^. So, when I mention Block Layer Directory, it means you need to go to the directory that contains IO schedulers.
- If you have done previous step, you must be in the directory.
- **Copy fifo-iosched.c at Block Layer directory**

  [expected results]

  - **[root@ds block]# ls**

    … fifo-iosched.c …

STEP 3. Modify configuration

Now, you need to change configuration for IO Schedulers. You will change Kconfig.iosched and Makefile.

[Modify Kconfig.iosched]

- Now, you copied your IO scheduler (fifo-iosched.c) into Block Layer directory.
- You need to let Block Layer know that new IO scheduler was added into Block Layer
- **Add configuration of FIFO into Kconfig.iosched** as followings (You can refer to(or use) the Kconfig.iosched file that will be given to you)

  [expected modification]

  :

  config IOSCHED_FIFO
  bool
  default y
  ---help---
  A no-op scheduler that really does nothing - FIFO.

  :

- Now, you added configuration of FIFO IO Scheduler into Kconfig.iosched

[Modify Makefile]

- You will modify Makefile so that your IO scheduler can be included into Linux kernel on recompilation of Linux kernel at later step.
- Add one line corresponding to FIFO IO scheduler into Makefile as followings

  [expected modification]
  :
  obj-$(CONFIG_IOSCHED_FIFO) += fifo-iosched.o
  :

- In order to add where you need to add upper one line, you can see the Makefile that will be given to you.
- Okay, by adding one line of fifo-iosched.o, fifo-iosched.c will be compiled and built when you recompile Linux kernel at later step.

STEP 4. Recompile Kernel Source

Okay. Unfortunately, IO schedulers are not part of loadable kernel modules. That means that you need to recompile entire Linux kernel in order to activate your FIFO IO scheduler. Just follow the steps that you learned from Linux practice. What are steps for recompiling Linux kernel? That's right: **make menuconfig** (it might not be necessary because .config already exists) – **make** – **make modules** – **make modules_install** – **make install**

STEP 5. Let's check

Okay. Now, it's time to celebrate you.

[Check compilation]

- Go to your Block Layer directory and let's check whether your fifo-iosched.c was correctly compiled
  - **[root@ds ~]# cd /usr/src/linux-2.6.25.20/block**
    [root@ds block]# ls
    as-iosched.c blk-exec.c blk-map.o blk-sysfs.o bsg.o
    deadline-iosched.o **fifo-iosched.o** Kconfig.iosched
    as-iosched.o blk-exec.o blk-merge.c blk-tag.c built-in.o
    elevator.c genhd.c modules.order blk-barrier.c blk.h blk-merge.o
    blk-tag.o cfq-iosched.c elevator.o genhd.o noop-iosched.c
    blk-barrier.o blk-ioc.c blk-settings.c blktrace.c cfq-iosched.o
    ioctl.c noop-iosched.o blk-core.c blk-ioc.o
    blk-settings.o blktrace.o compat_ioctl.c ioctl.o scsi_ioctl.c
    blk-core.o blk-map.c blk-sysfs.c bsg.c deadline-iosched.c fifo-iosched.c
    Kconfig Makefile scsi_ioctl.o

[Check current IO scheduler]

- Which command did you type in order to see the current IO scheduler for a hard disk? Oh!! You remember!!! Thumbs up !!! ^^
- Type the command to see the current IO scheduler for your hard disk

  [expected screen capture]
  [root@ds ~]# cat /sys/block/sda/queue/scheduler
  noop fifo anticipatory deadline [cfq]

- You can see **"fifo" can be selected as the current IO scheduler**
- Now, you can choose "fifo" as the current IO scheduler for "sda" hard disk. How can you do that? ^^ Okay, I know you are excellent. You thought following commands as you did use at STEP 1.

[expected screen capture]
[root@ds ~]# echo "fifo" > /sys/block/sda/queue/scheduler
[root@ds ~]# cat /sys/block/sda/queue/scheduler
noop [fifo] anticipatory deadline cfq

- Now, FIFO IO scheduler will be used for IO scheduler of "sda" hard disk.

[Check running of FIFO IO scheduler]

- How can you believe me? ^^. As usual, it's time to check "/var/log/messages" whether IO scheduler really is running.
- First, can you check one line print function in fifo-iosched.c? Like Linux practices, I added one line that prints a message into fifo-iosched.c. Do you see that?
- Okay, Now, check
  - **tail –f /var/log/messages**

[expected screen capture]
[root@ds block]# tail -f /var/log/messages
:
:
Apr 16 00:58:07 ds kernel: **FIFO SCHEDULER - DISPATCH : DONE**
Apr 16 00:58:07 ds kernel: FIFO SCHEDULER - DISPATCH : DONE
Apr 16 00:58:07 ds kernel: FIFO SCHEDULER - DISPATCH : DONE
Apr 16 00:58:07 ds kernel: FIFO SCHEDULER - DISPATCH : DONE
Apr 16 00:58:31 ds kernel: FIFO SCHEDULER - DISPATCH : DONE
Apr 16 00:58:31 ds kernel: FIFO SCHEDULER - DISPATCH : DONE
Apr 16 00:58:31 ds kernel: FIFO SCHEDULER - DISPATCH : DONE
Apr 16 00:58:31 ds kernel: FIFO SCHEDULER - DISPATCH : DONE
Apr 16 00:58:31 ds kernel: FIFO SCHEDULER - DISPATCH : DONE

- Can you see the message "XXX DISPATCH:DONE" ? The message will be printed at log file when IO scheduler has inserted request into request queue.

- If you can see the message, you finish adjusting new FIFO IO scheduler into your Linux system. Congratulation!!! … But, wait. Just one thing more.

[Start FIFO IO scheduler when system boots up]

- If you reboot, you will see CFQ is selected as default IO scheduler again.
- If you want to start FIFO IO scheduler when system boots up. You can change a file.
- Now, question time. What file did you modify so that you change the default kernel version? Do you remember you changed a file in order to change the default kernel version between two versions? What is that? ^^
  - That's "**/boot/grub/grub/conf**"

- Okay, **let's open the "grub.conf", modify as followings (add "elevator=fifo" at your kernel version), and then save and reboot**. You will see FIFO IO scheduler is default IO scheduler.

[expected screen shot]

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,0)
#          kernel /vmlinuz-version ro root=/dev/sda2
#          initrd /initrd-version.img
#boot=/dev/sda
default=0
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title Fedora (2.6.25.20)
        root (hd0,0)
        kernel /vmlinuz-2.6.25.20 ro root=UUID=6288604d-42c5-4d35-8248-664b4afe0e5e rhgb quiet elevator=fifo
        initrd /initrd-2.6.25.20.img
title Fedora (2.6.25-14.fc9.i686)
        root (hd0,0)
        kernel /vmlinuz-2.6.25-14.fc9.i686 ro root=UUID=6288604d-42c5-4d35-8248-664b4afe0e5e rhgb quiet
        initrd /initrd-2.6.25-14.fc9.i686.img
```

Conclusion

You saw how to adjust your own IO scheduler. Here, just you have used given source code for your practice. However, for your project, you need to modify built-in Linux IO schedulers or create your own source codes.

You can start based on noop IO scheduler source code or here FIFO IO scheduler source code.

Also, you can change settings for each IO scheduler. Whenever you change your current IO scheduler, if you check "/sys/block/<hard disk name>/queue/iosched", you will be able to see the setting variables for each IO scheduler. You might change the settings for performance comparison.

I hope this guide will help you create your own IO scheduler for your project.

Cheer up.