Linux Kernel Block I/O Schedulers

Raj kumar Rampelli

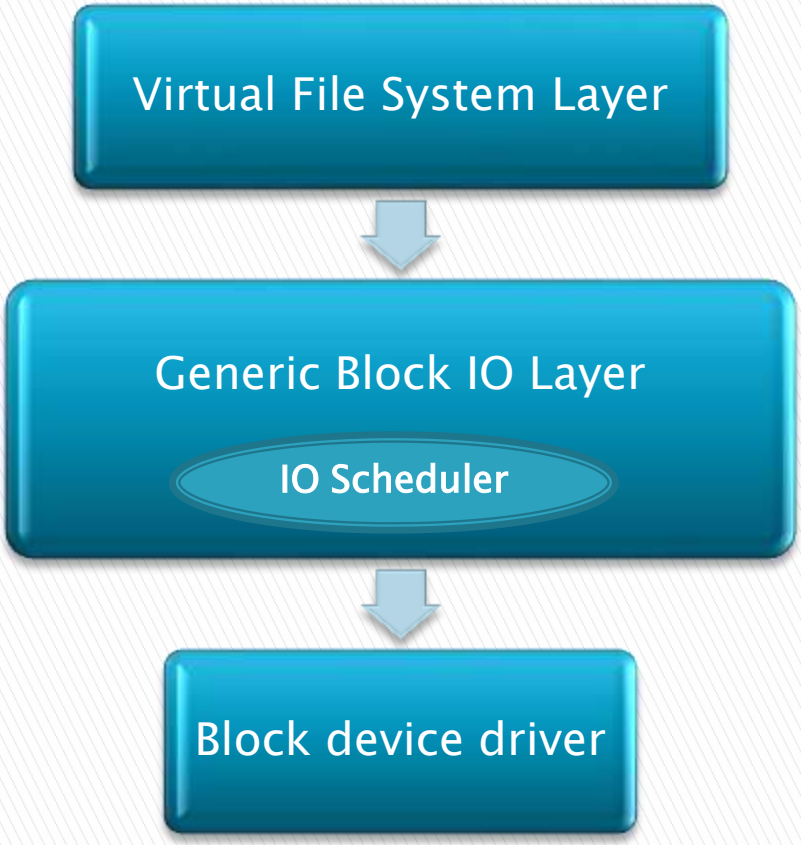
Outline

- I/O Schedulers introduction
- Block I/O subsystem in Linux kernel
- I/O Scheduler primary actions
  - Merging
  - Sorting
- Types of I/O Schedulers
- Conclusion

I/O Schedulers

- It is the subsystem of the kernel which performs "*merging and sorting*" on block I/O requests to improve the performance of the system.
- Process Scheduler, which share the processor among the processes on system.
  - I/O Scheduler and Process Scheduler are two different subsystems in Kernel.
- I/O Scheduler Goal:
  - Minimize disk seeks,
  - ensure optimum disk performance and
  - provide fairness among IO requests

Block I/O subsystem in kernel



- Block IO layer
  - receives IO requests from File System layer
  - Maintains IO requests queue
- I/O Scheduler
  - schedules these IO requests and decides the order of the requests.
  - It selects one request at a time and dispatches it to block device.
  - It perform two actions on request queue
    - Merging
    - Sorting
  - Manages the request queue with goal of reducing seeks and improving the throughput

I/O Scheduler

- Primary actions: Merging and Sorting
- Merging
  - Coalescing of two or more requests into one request operating on one or more adjacent on-disk sectors.
  - Reduces the overhead of multiple requests into a single request
  - Minimizes the seek operations
- Sorting
  - Keeps request queue sorted, sector-wise
  - Minimizes the individual seek, keeping the disk head moving in straight line.

Adding a request to the request queue

- Four operations are possible if any new request to be added into the request queue

1) If a request to an adjacent on-disk sector is in the queue, the existing request and the new requests are merged into single request
2) If a request in the queue is sufficiently old, the new request is inserted at the tail of the queue to prevent starvation of the other, older, requests.
3) If there is a suitable location sector-wise in the queue, the new request is inserted there. Keeping the queue sorted by physical location on the disk
4) Insert new request at the tail of the queue if above scenarios not met.

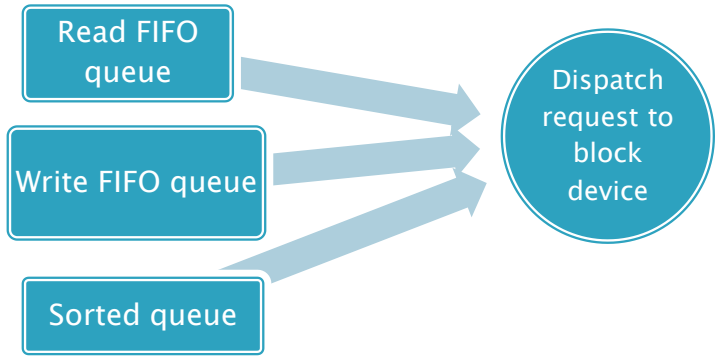
I/O Schedulers Types

- Four different I/O schedulers are present in kernel

1) Deadline I/O Scheduler
2) Anticipatory I/O Scheduler
3) Complete Fair Queuing I/O Scheduler
4) Noop I/O Scheduler

Deadline I/O Scheduler (DIS)

- Each request is associated with an expiration time
  - 500 milli seconds for read requests
  - 5000 milli seconds for write requests
- Maintains 3 queues: Normal sorted queue, read requests queue, write requests queue
- Performs merging/sorting on sorted queue when new request comes.



  - New request is also inserted into either read queue or write queue depends on the type of requests read or write.
- DIS pulls the request from sorted queue and dispatched it to device driver.
  - If any request from read/write queue expires then DIS pulls the request from these queues.
- Ensures that no requests are processes on or before expiration time.
- Prevents requests starvation.
- Code pointer: drivers/block/deadline-iosched.c

Anticipatory I/O Scheduler (AIS)

- Deadline I/O scheduler minimizes the read latency (since more preference has given to read requests) but it compromise on throughput – considering a system undergoing heavy write activity.
- AIS = DIS + Anticipation Heuristic
- After the request is submitted to device driver, AIS sits idle for few milliseconds (default 6 milliseconds), thinking that there is a good chance of receiving new read request which is adjacent to the submitted request. If so, this newly request is immediately served.
- After waiting period elapses, it continue to pull request from request queues similar to DIS.
- Need proper/correctly anticipating the actions of applications and file systems.
- Ideal for servers
  - Perform very poorly on certain uncommon workloads invloving seek-happy databases.
- Code pointer: drivers/block/as-iosched.c

Complete Fair Queuing (CFQ) I/O Scheduler

- Designed for specialized workloads.
- Different from DIS and AIS schedulers
- CFQ maintains one sorted request queue for each process submitting IO requests.
- CFQ services these queues in Round Robin fashion and selects configurable number of requests (default 4) from each queue.
- Provides fairness at a per-process level
- Intended workload is Multimedia and recommended for desktop workloads.
- Code pointer: drivers/block/cfq-iosched.c

Noop I/O Scheduler

- It performs only merging and does not perform sorting.
- It is intended for block devices that are truly random-access, such as flash memory cards.
- If a block device has a little or no overhead associated with “seeking”, then Noop I/O Scheduler is ideal choice.
- Code pointer: drivers/block/noop-iosched.c

Conclusion

- Block devices uses the Anticipatory I/O Scheduler by default.
- Select CFQ for desktop/multimedia workloads
- Select Noop for block devices which are truly random access (flash memory cards) and for block devices which doesn't have seeking overhead.
- It can be overridden through kernel command line with option "elevator=as/cfq/deadline/noop". Choose any one from four schedulers.
- If elevator=noop is set then Noop I/O scheduler will be enabled for all block devices

Reference

- [Book] Linux Kernel Development
  By Robert Love

Thank You ☺

Have a look at
My PPTs @ http://www.slideshare.net/rampalliraj/
My Blog @ http://practicepeople.blogspot.in/