dm-ioband

A disk IO bandwidth controller
Implemented as a Device-mapper Module

Hirokazu Takahashi, VA Linux Systems Japan

Ryo Tsuruta, VA Linux Systems Japan

Akio Takebe, FUJITSU LIMITED

Agenda

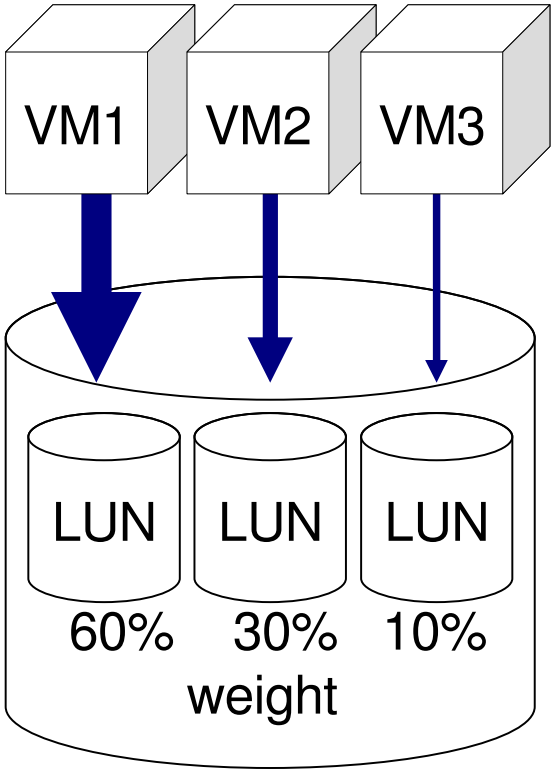
- Motivation & Goals
- Features
- Benchmarks
- Design and Implementation
- IO controller Mini-summit
- Future work

Motivation and Goals

Motivation

As computer hardware becomes more powerful and faster, it makes multiple services run on a single machine and multiple users share the use of it, then we also need to share the bandwidth of one storage with per various kinds of groups.

- per process
  - cgroup
  - user/group
- per filesystem
  - LUN
  - partitions
- per virtual machine



Goals

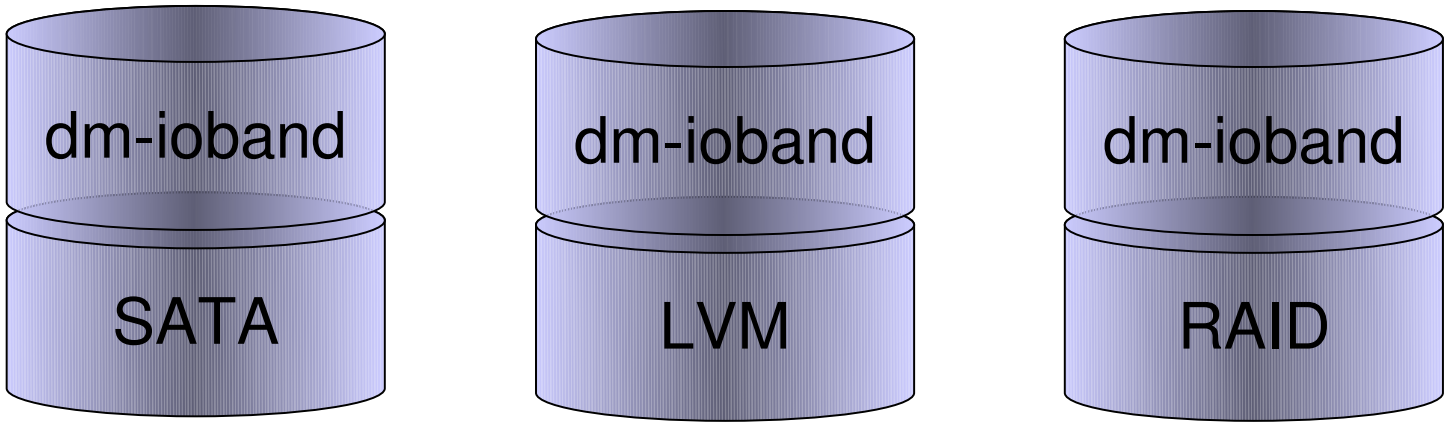
The requirements of the IO controller are:

- Can assign bandwidth per various kinds of groups.
- Can work with any type of block devices.
- Distribute bandwidth for each group in proportion of weight.
- Multiple bandwidth control policies are supported.
- Minimal overhead and throughput decrease, especially against high-end storages and SSD.
- Can work with any type of IO schedulers.

The features of dm-ioband

Stackable on any type of devices

dm-ioband is implemented as a device-mapper driver, so it allows to provide bandwidth control by stacking on any type of block devices.



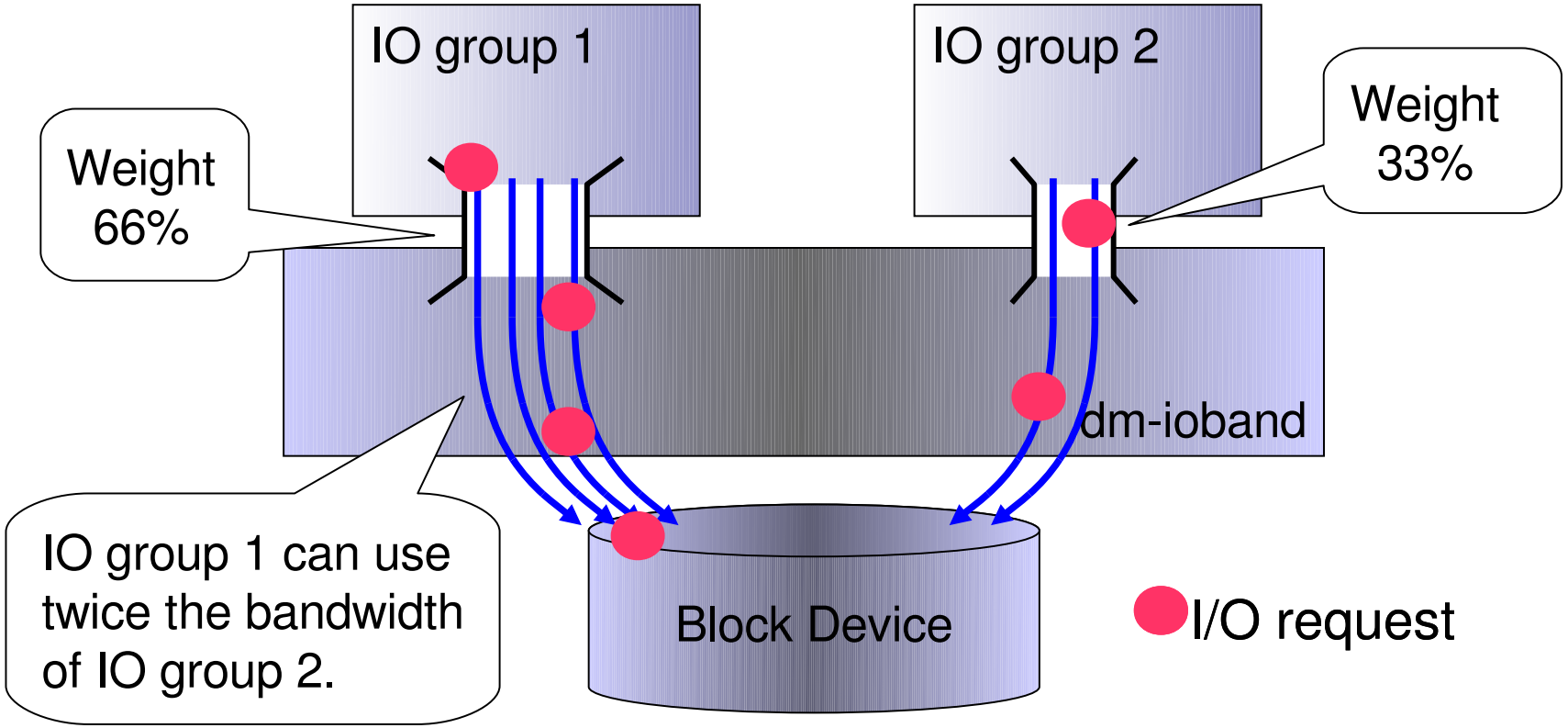
Various grouping

Bandwidth can be assigned on a:

- per partition basis.
- per user ID or group ID basis.
- per process ID or process group ID basis.
- per cgroup basis.

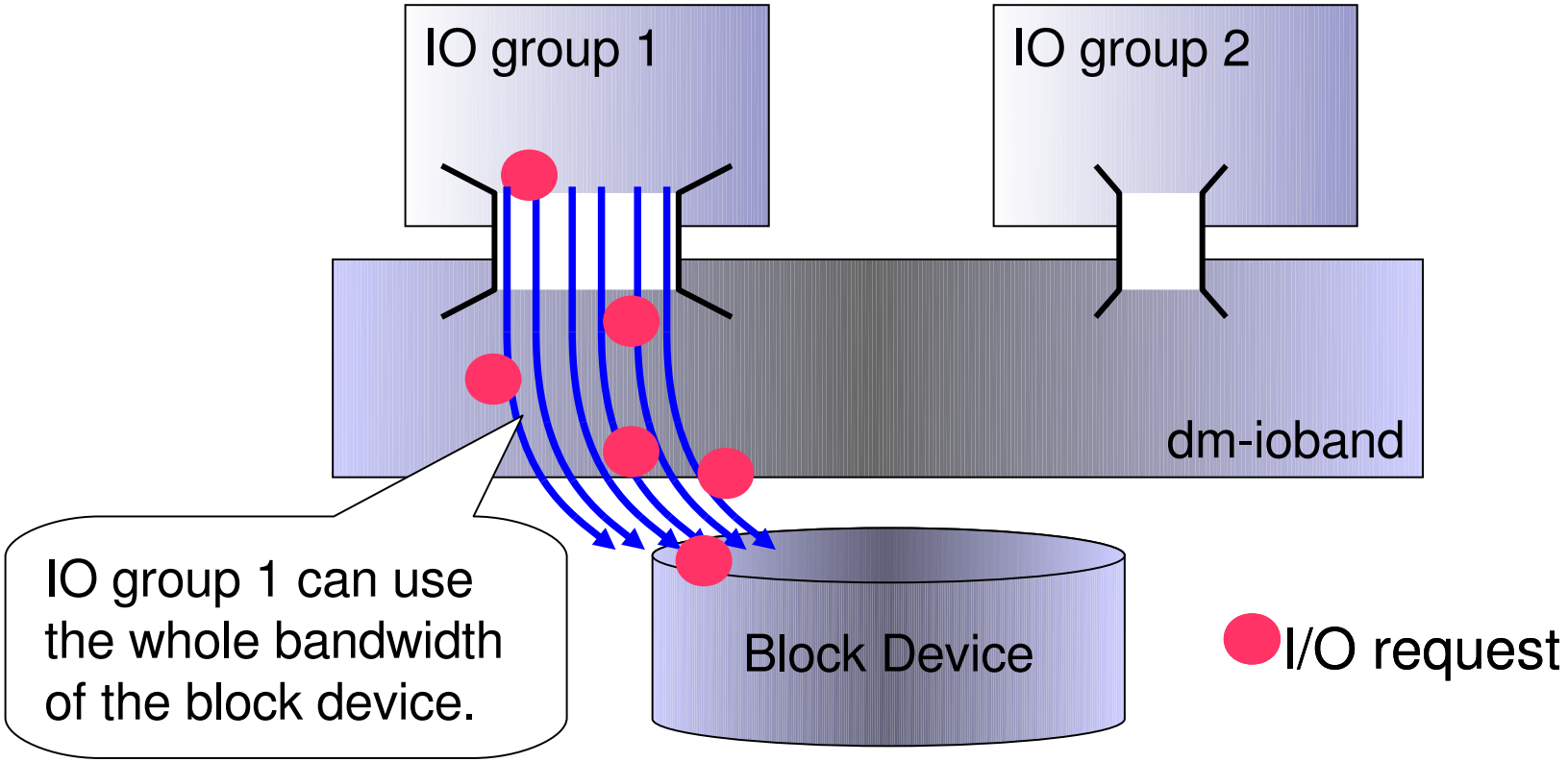
Proportional weight controller

Bandwidth is distributed in proportion of weight of each group.



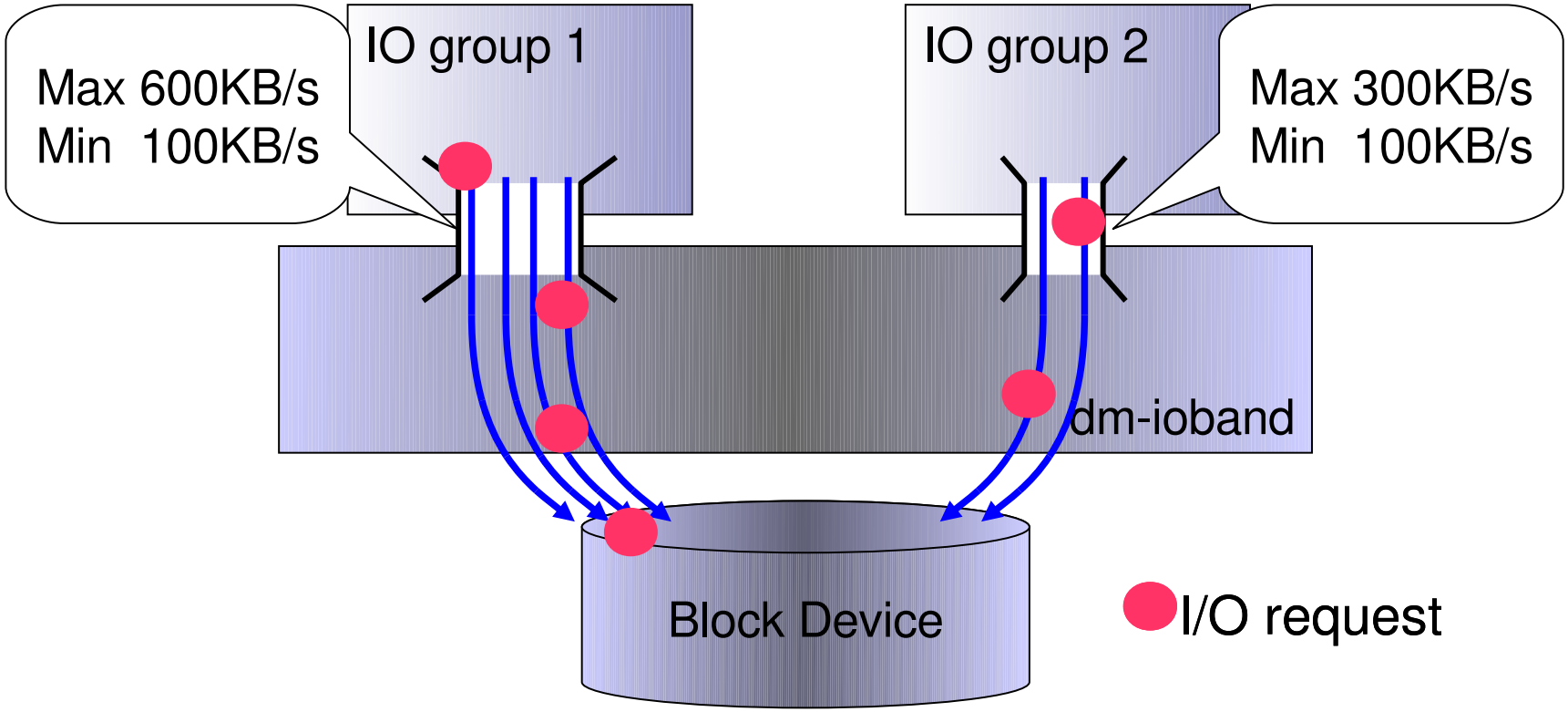
Proportional weight controller

The spare bandwidth of inactive groups is used for active groups.



Min and Max bandwidth controller

Bandwidth control policy is selectable. The rate guarantee and limiting policy is also provided.



Min and Max bandwidth controller

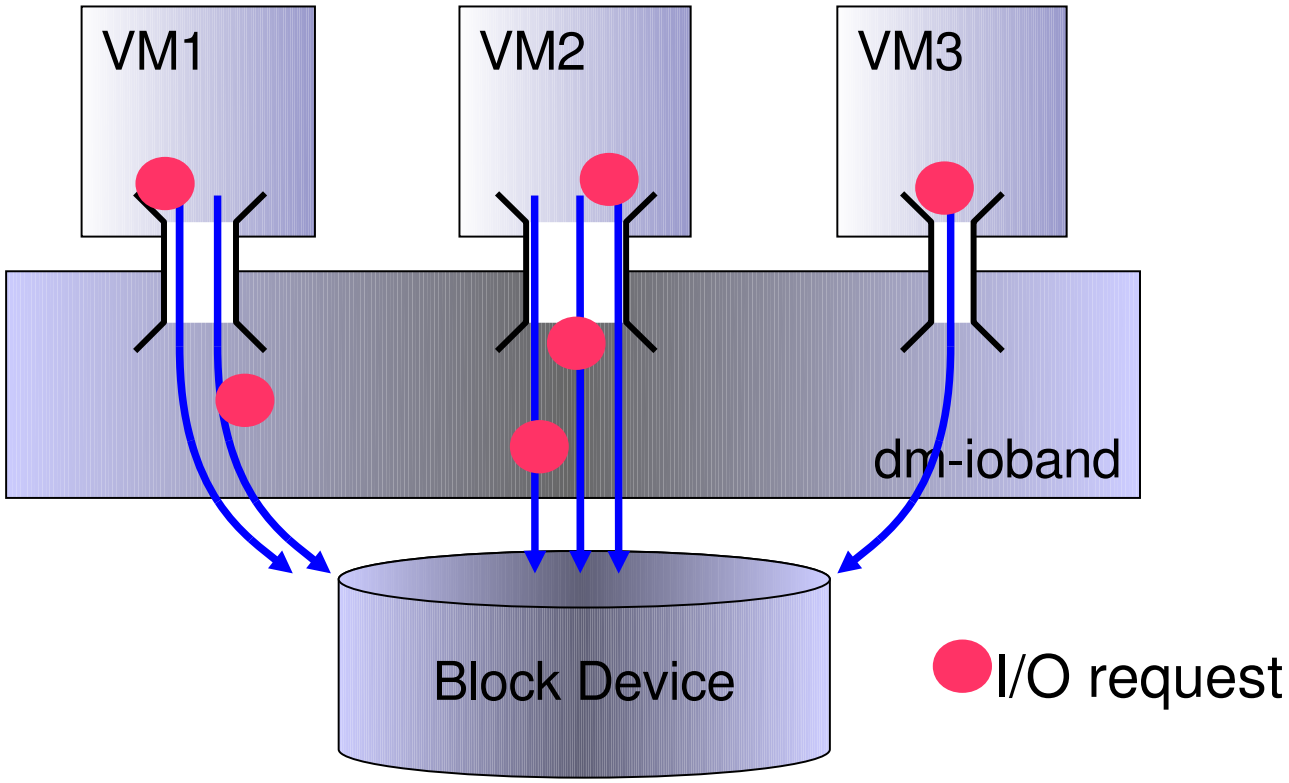
This policy is developed by Dong Jae Kang, an ETRI (Electronics and Telecommunications Research Institute in Korea) researcher, in a part of the corset project.

More detailed information is available at:

http://www.corsetproject.net/Wiki/DiskIOController

Virtual machine support

dm-ioband can throttle bandwidth on a per virtual machine basis by creating IO groups which correspond to each virtual machine's IO thread.



cgroup support

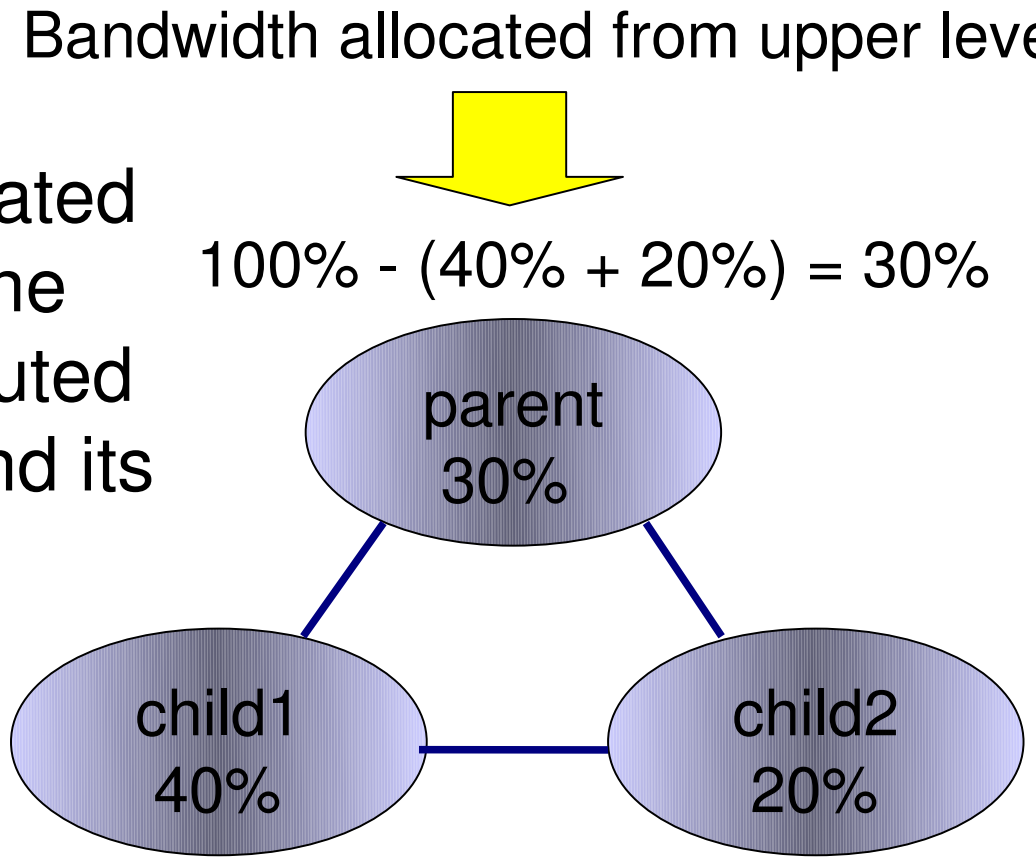
Using dm-ioband with blkio-cgroup can control bandwidth per on a cgroup basis, even if IO requests are buffered write and issued from a kernel thread, such as pdflush, instead of the process which originates the request.

Once the dm-ioband device is created, all settings can be configured through the cgroup interface as well as other cgroup subsystems.

Hierarchical bandwidth allocation

dm-ioband's cgroup support also allows hierarchical bandwidth allocation.

The bandwdith allocated for the parent from the upper level is distributed among the parent and its children.



Benchmarks

Benchmarks

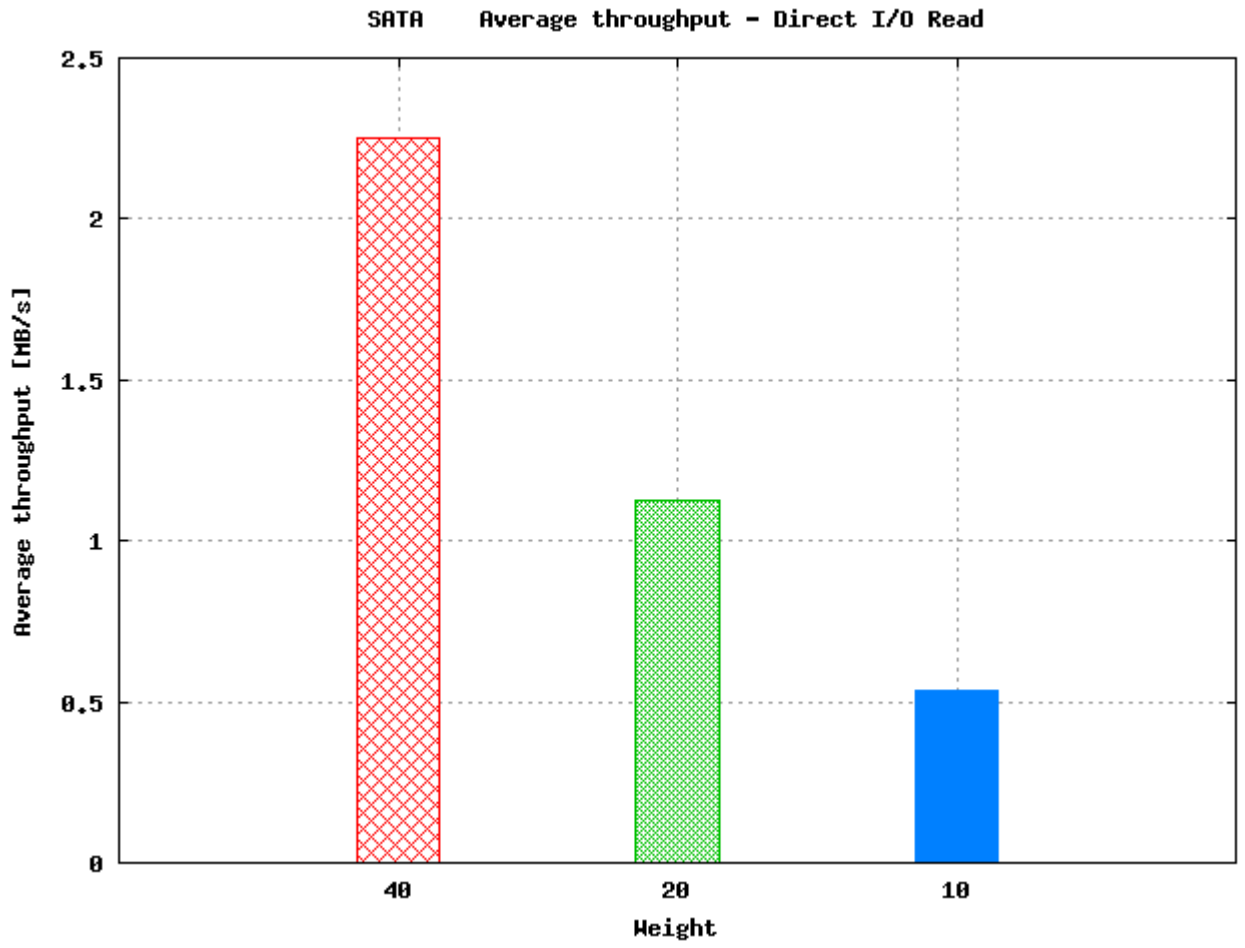
Benchmarked with three different kinds of disk devices.

- An ordinary SATA disk.
- A SAN storage via 4GBps Fiber-channel.
- A high performance SSD via PCI express.

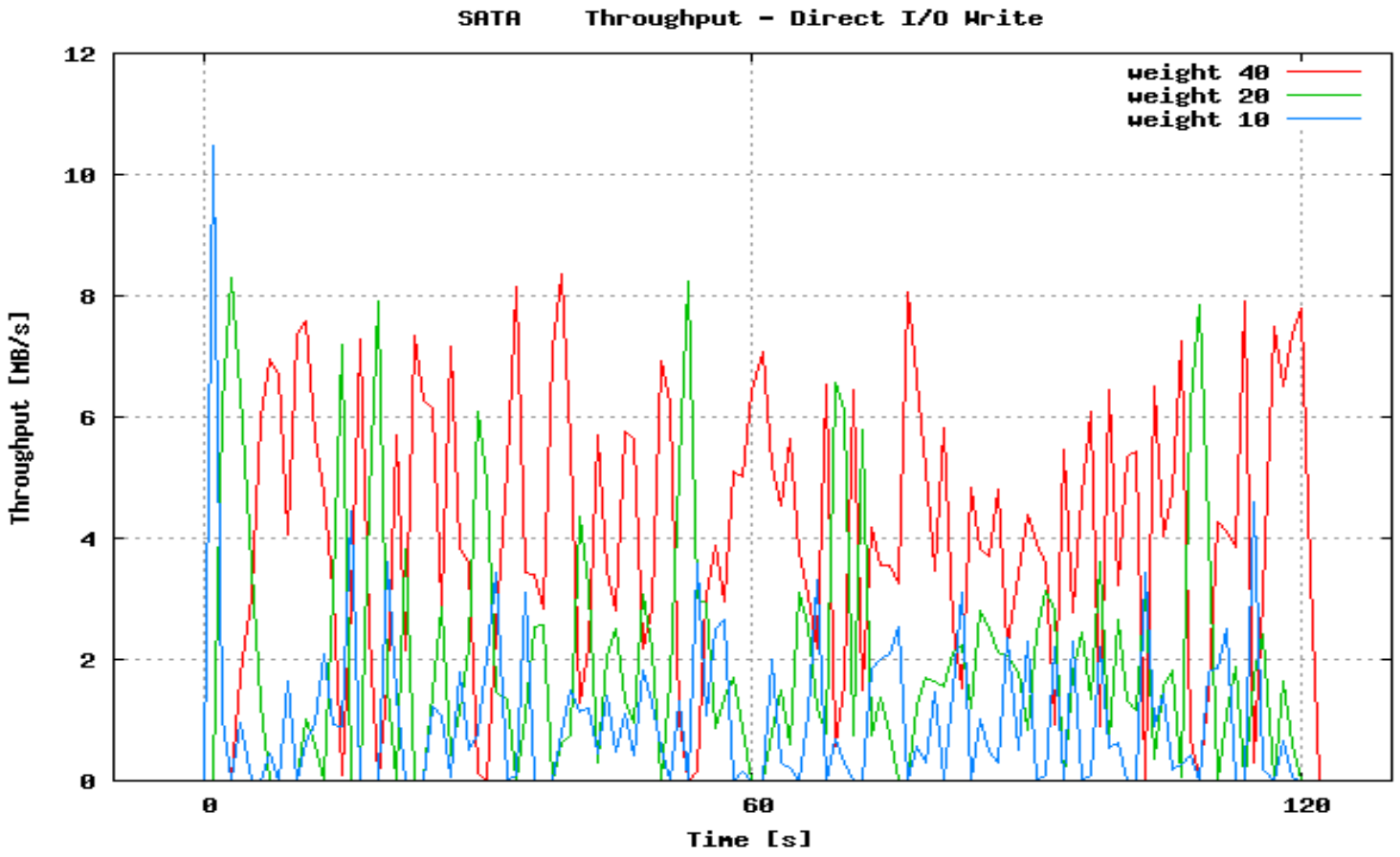
This benchmark did random IOs to three groups at the same time. The groups are assigned weights of 10, 20 and 40 respectively.

SATA Read (Direct I/O)

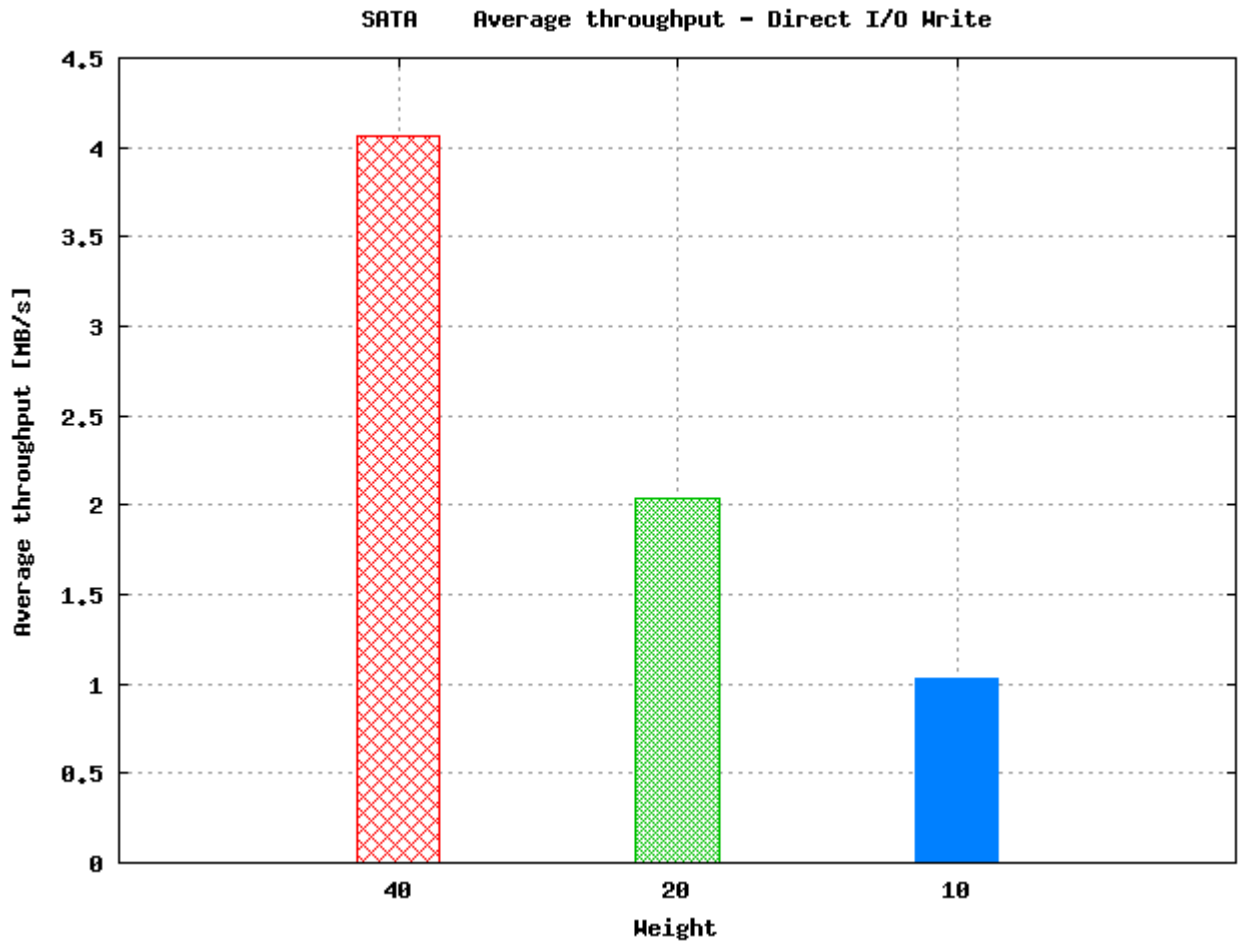
SATA Read (Direct I/O)



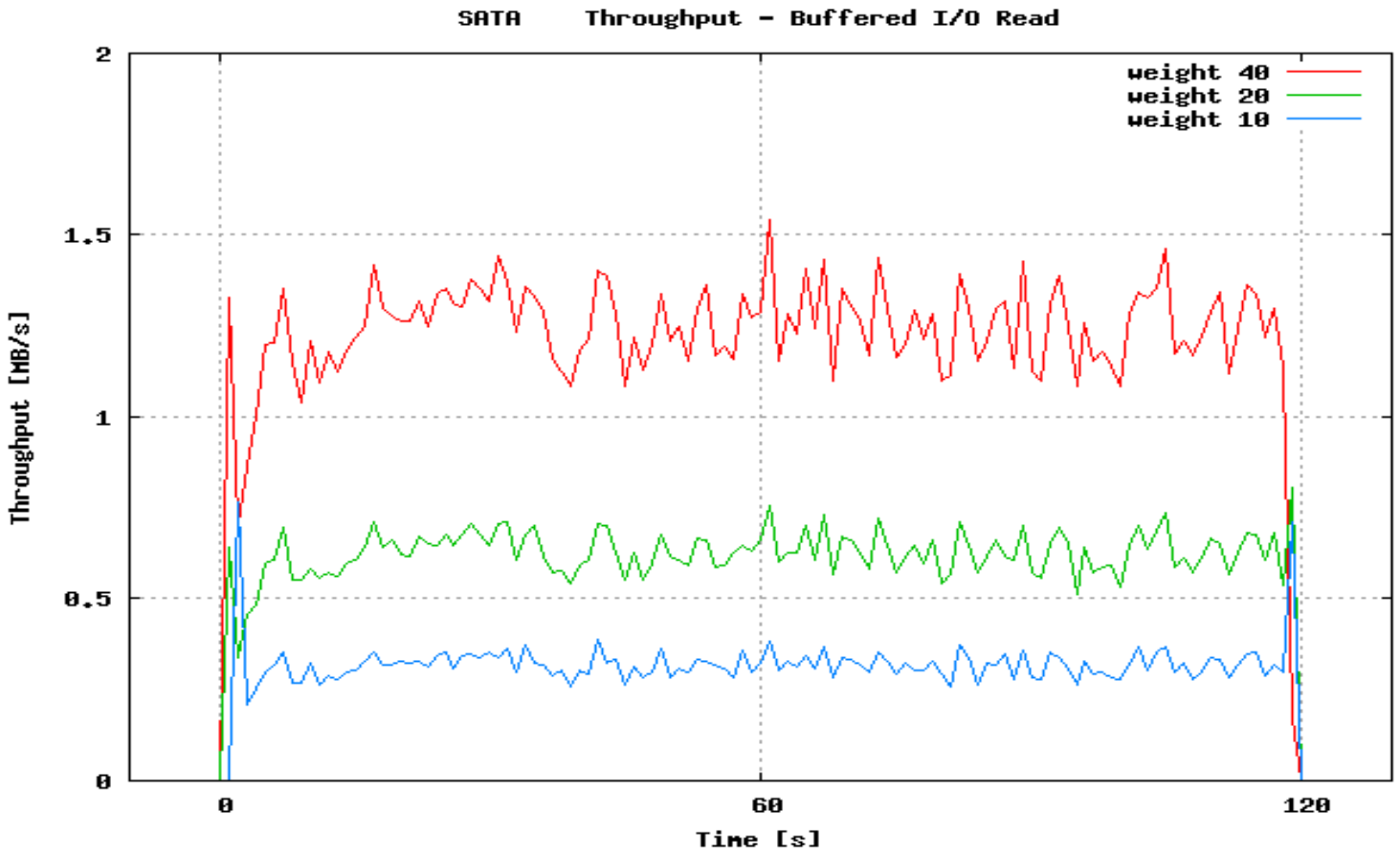
SATA Write (Direct I/O)



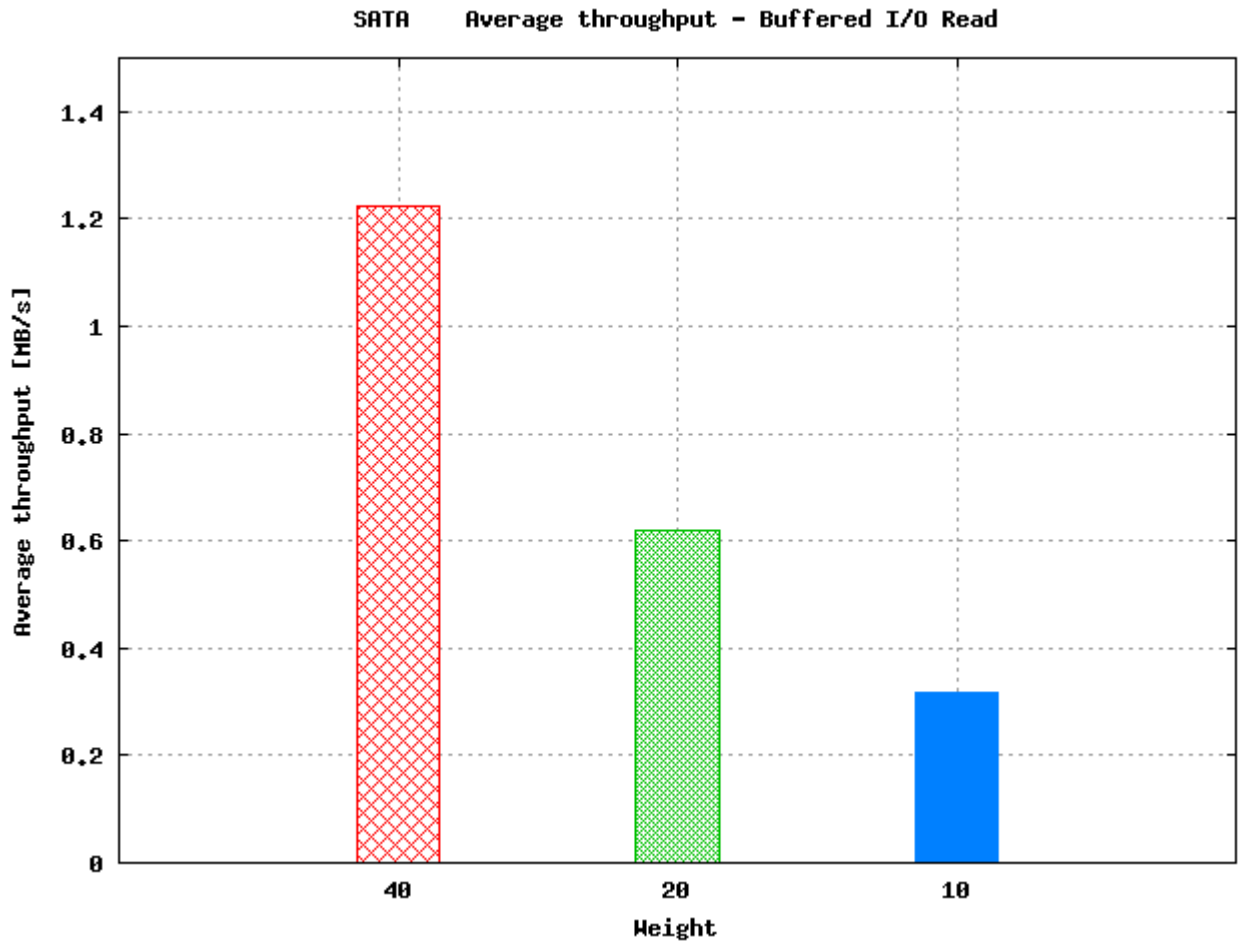
SATA Write (Direct I/O)



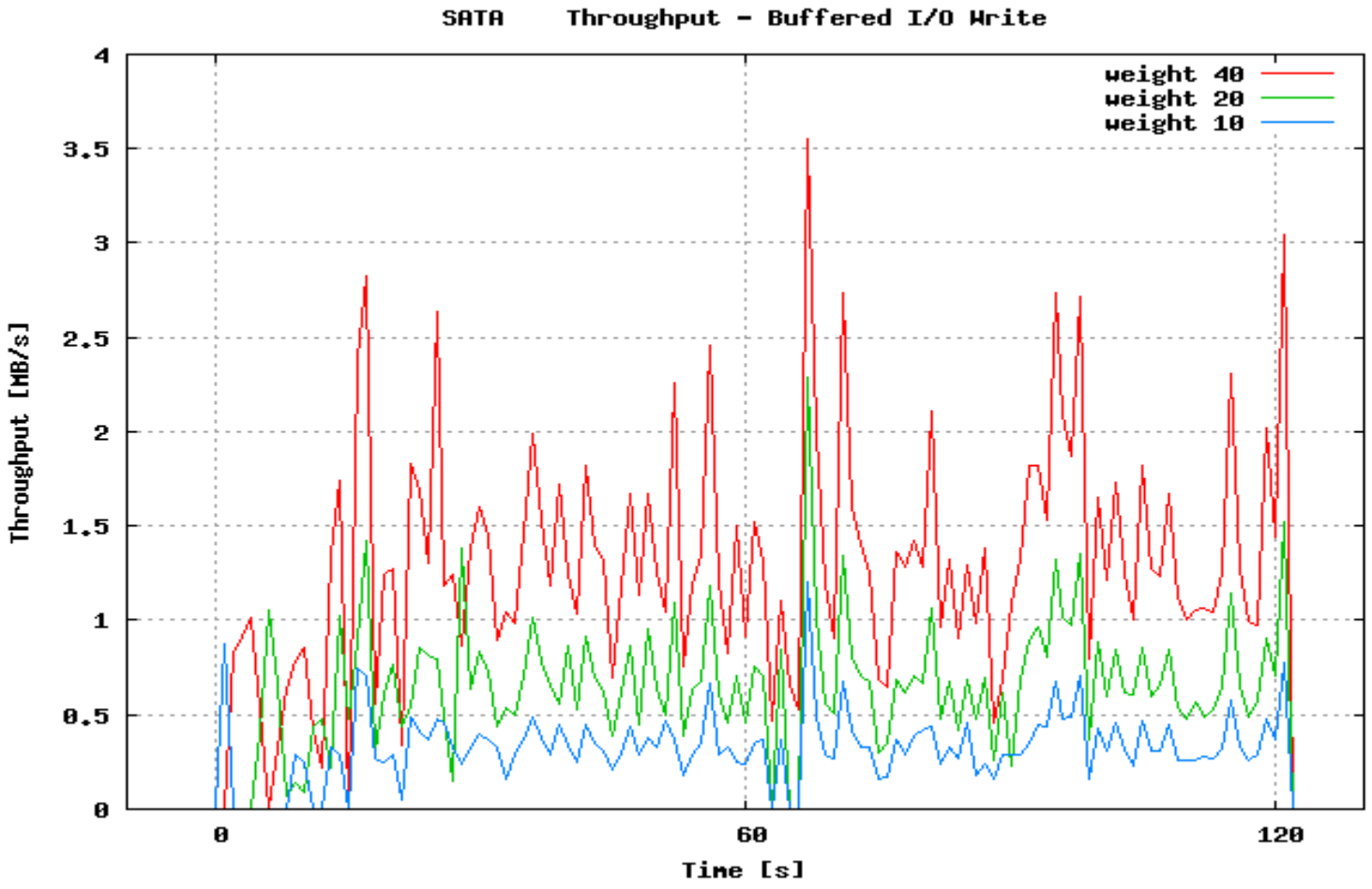
SATA Read (Buffered I/O)



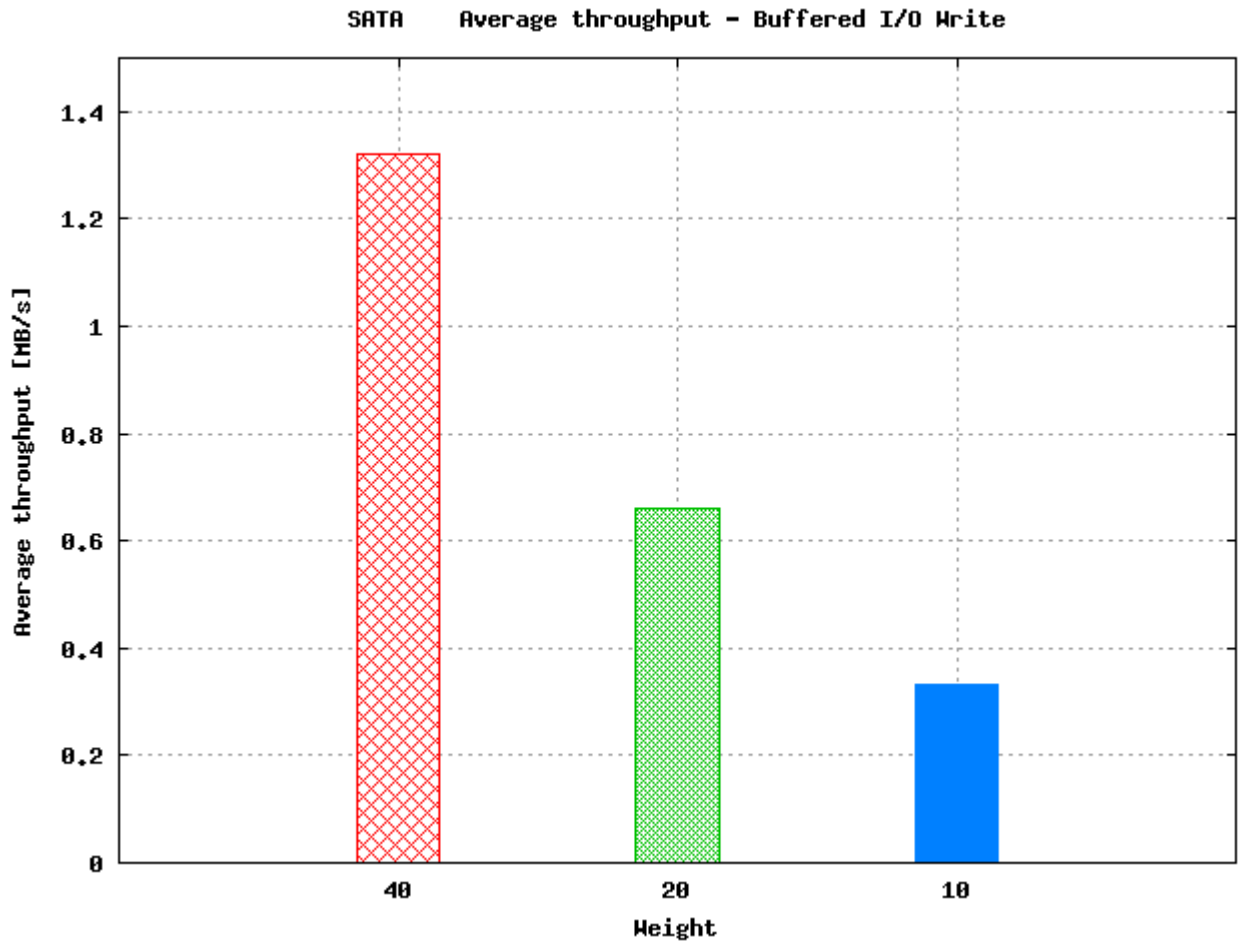
SATA Read (Buffered I/O)



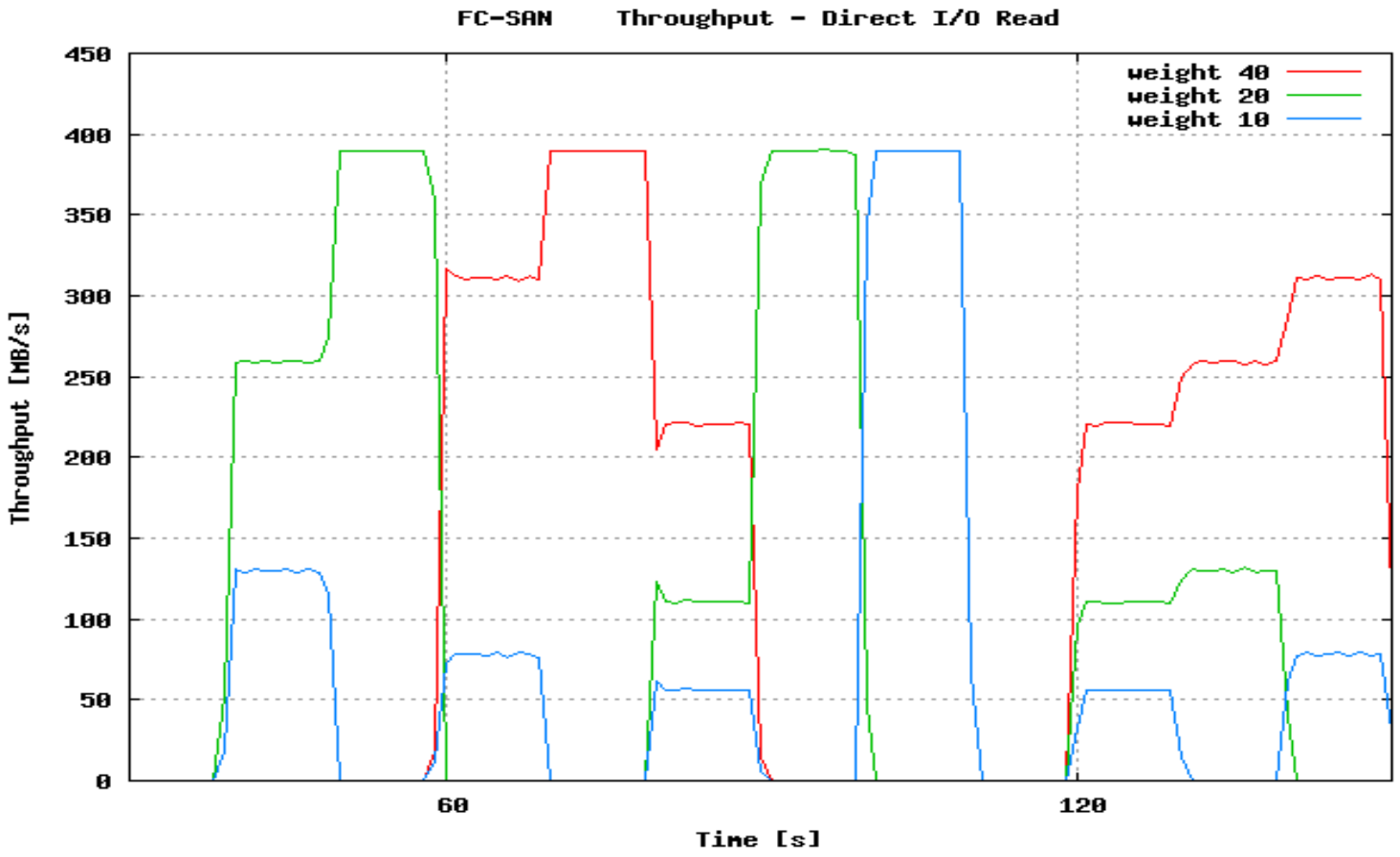
SATA Write (Buffered I/O)



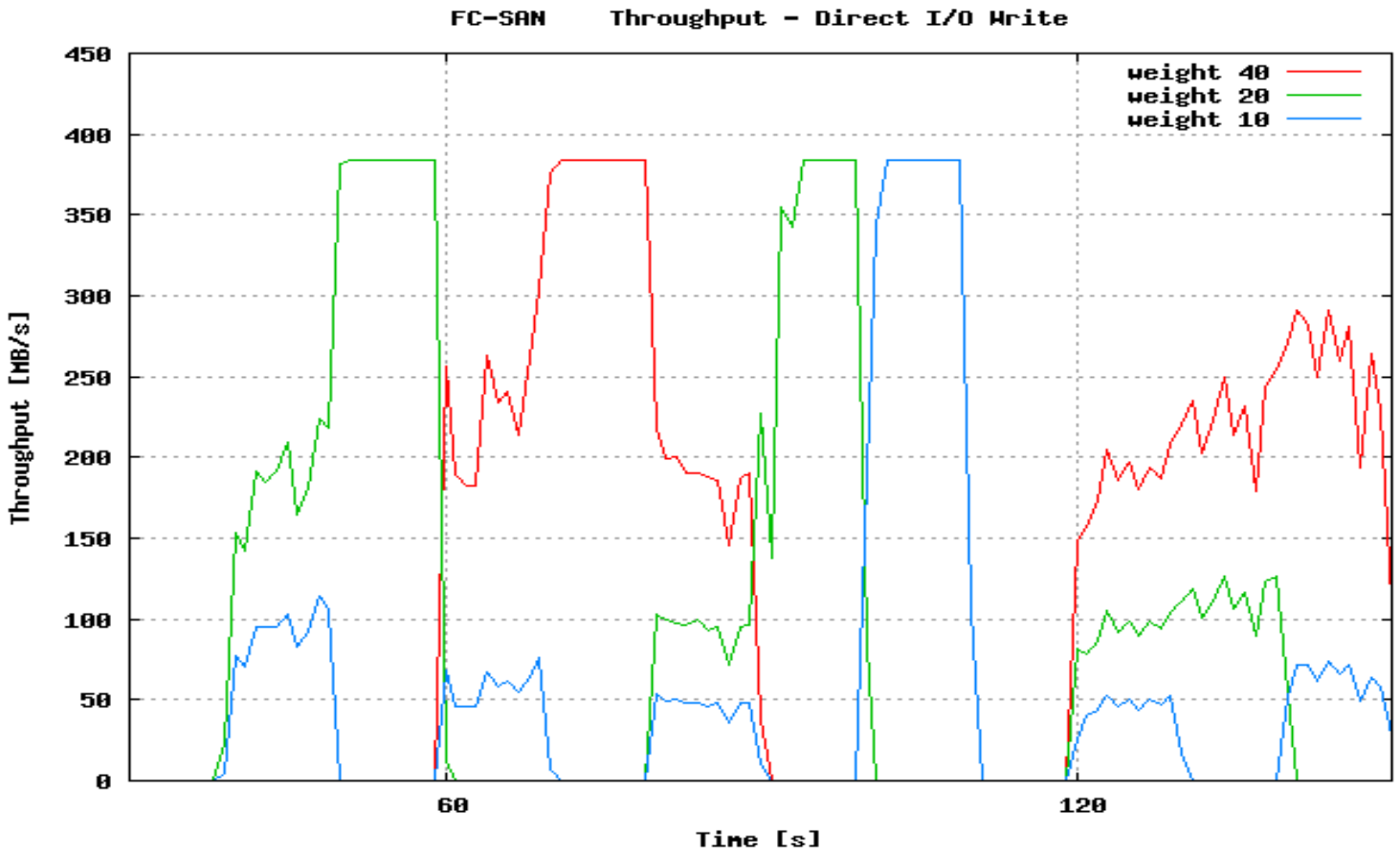
SATA Write (Buffered I/O)



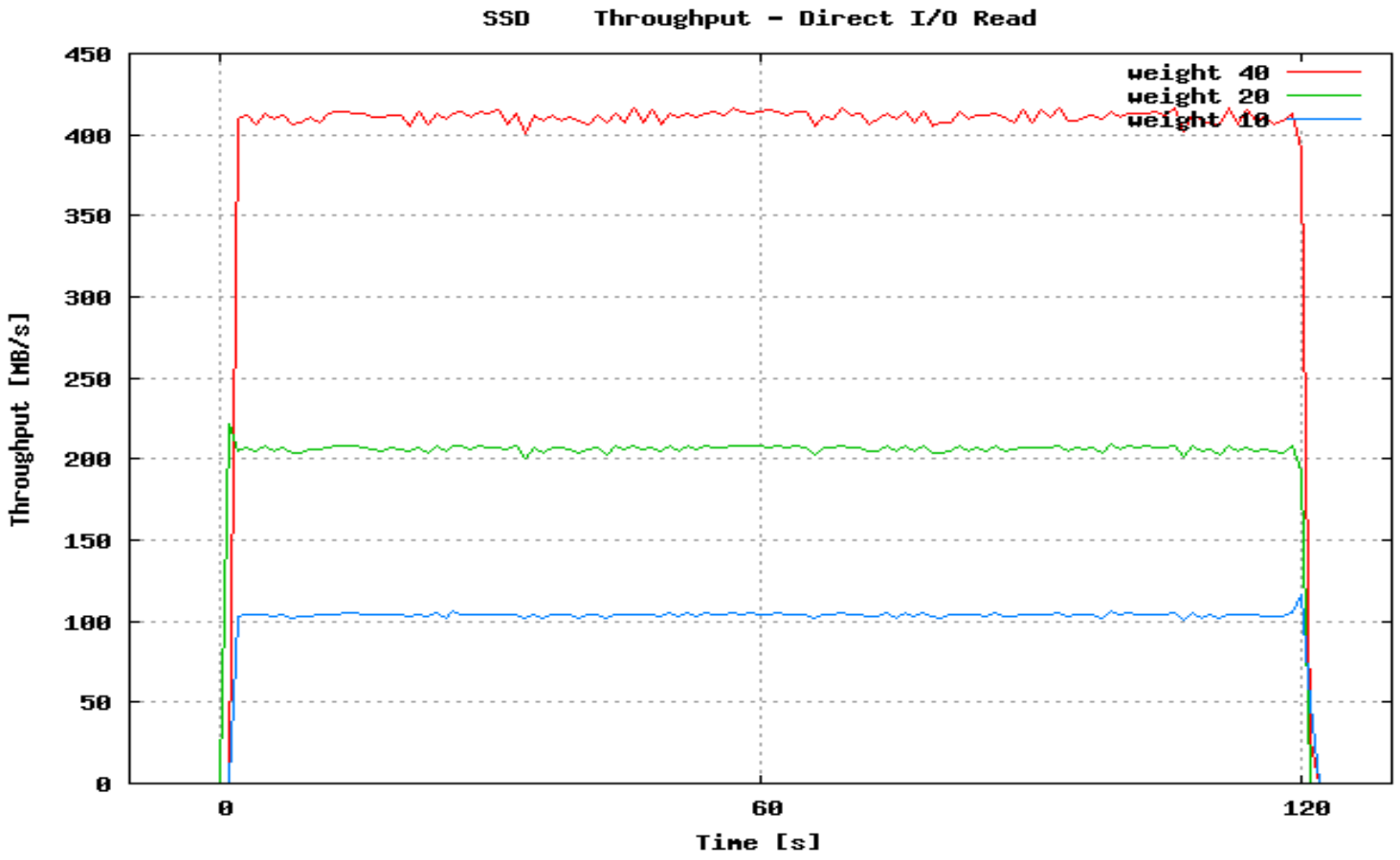
FC-SAN Read (Direct I/O)



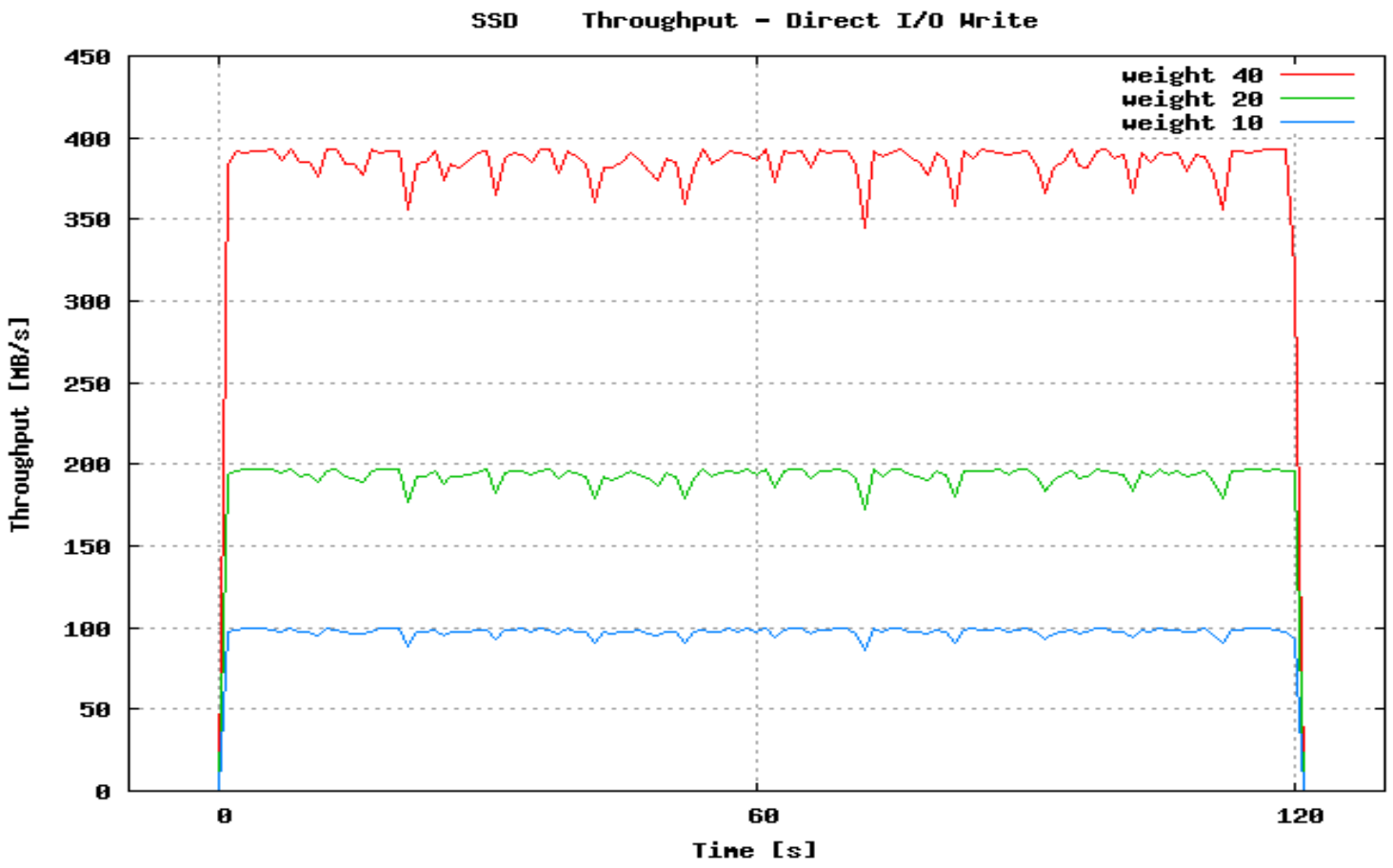
FC-SAN Write (Direct I/O)



SSD Read (Direct I/O)



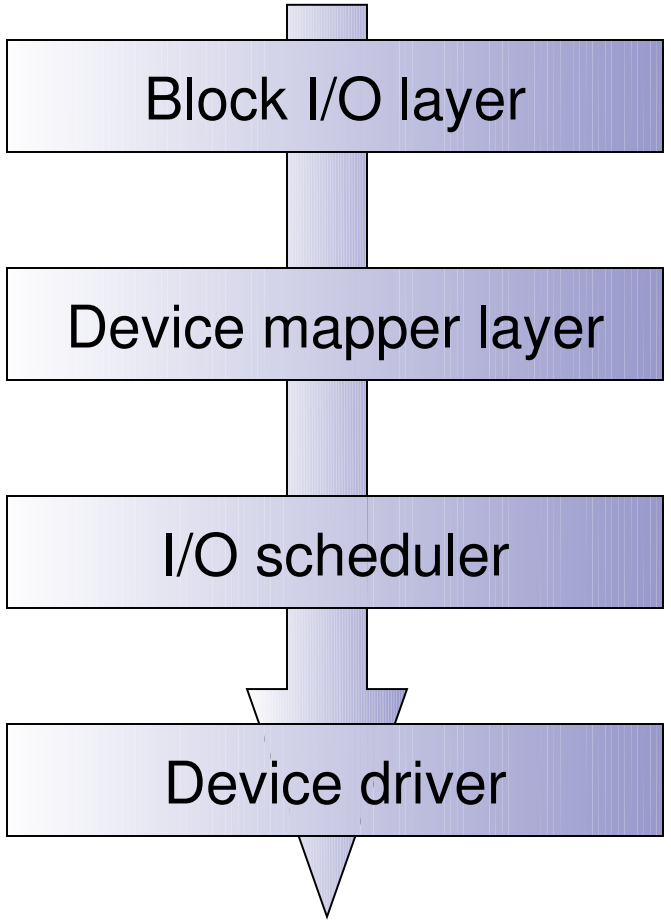
SSD Write (Direct I/O)



Design and Implementation

Which layer is the right place?

- It should be in the block I/O layer or in the device mapper layer to support any type of block devices.
- The I/O scheduler should only focus on I/O performance.
- So we have implemented it at the device mapper layer because the layer is highly independent and it can be a loadable kernel module.

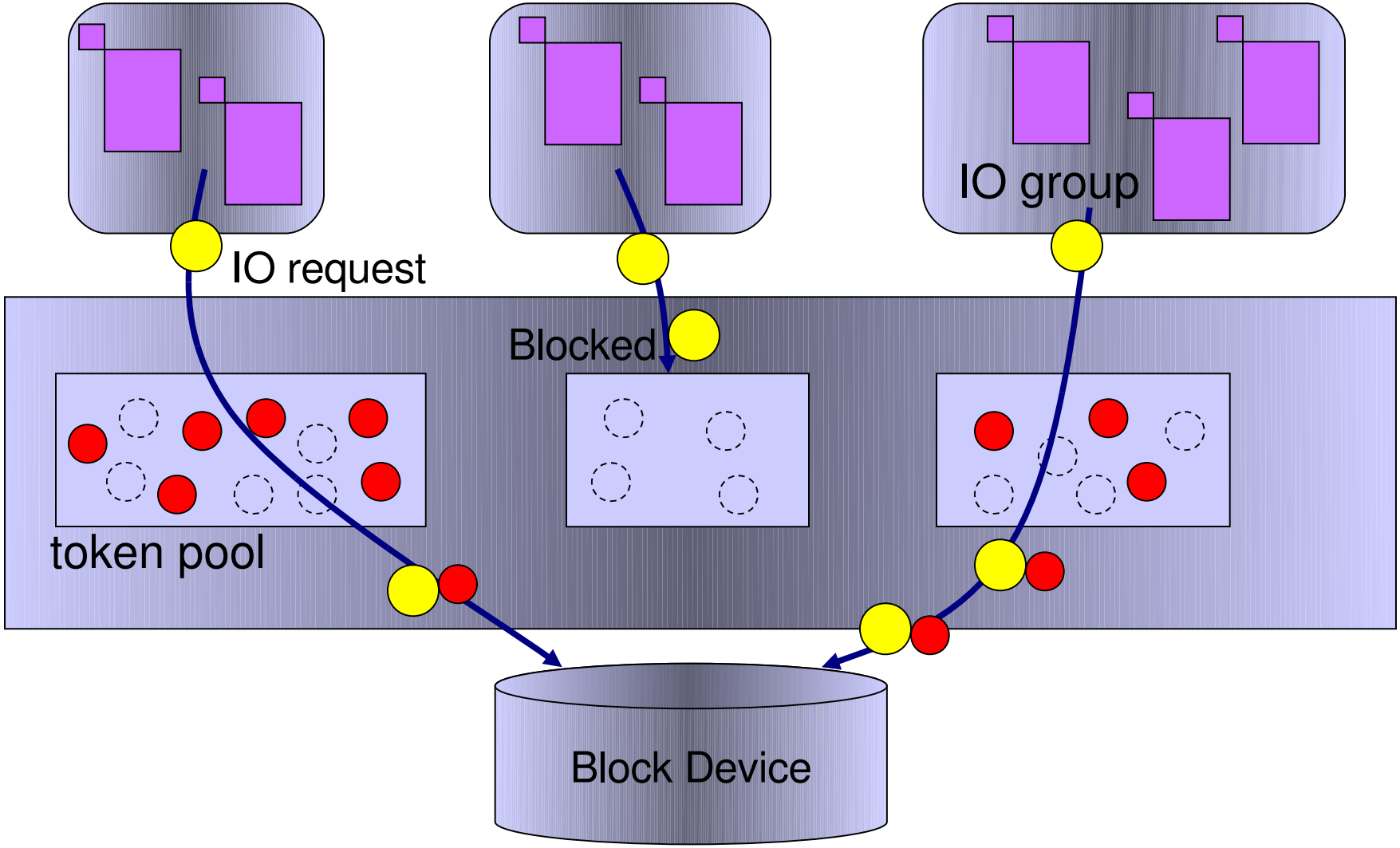


How bandwidth is distributed?

The bandwidth control is done by token bucket algorithm.

- Give tokens to each IO group according to proportion of weight.
- Every I/O request consumes one or more tokens.
- An I/O group is blocked once it used up its own tokens.
- Recharge tokens when all active groups used up their tokens.

Token bucket algorithm



Token bucket algorithm

- To gain throughput, dm-ioband will recharge new tokens even if there are some tokens left.
- Don’t block emergency I/O requests such as page-out requests even when the owner group has used up its tokens.

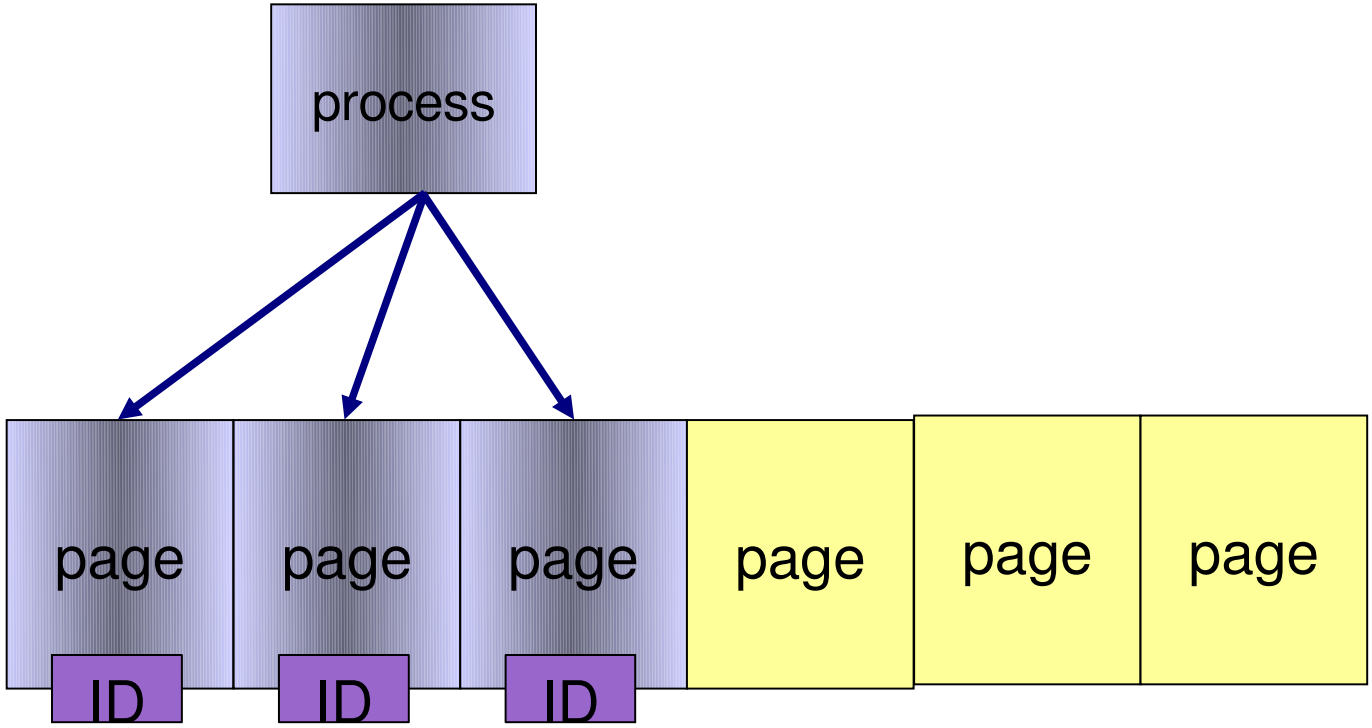
cgroup support

dm-ioband uses blkio-cgroup to determine a cgroup to which an IO request belongs. blkio-cgroup provides two major functions:

- Set a blkio-cgroup ID to a page when a read or write operation is requested to the page. Of course, the buffered write to a page is also handled properly.
- Retrieve a blkio-cgroup ID from a page when dm-ioband gets an IO request.

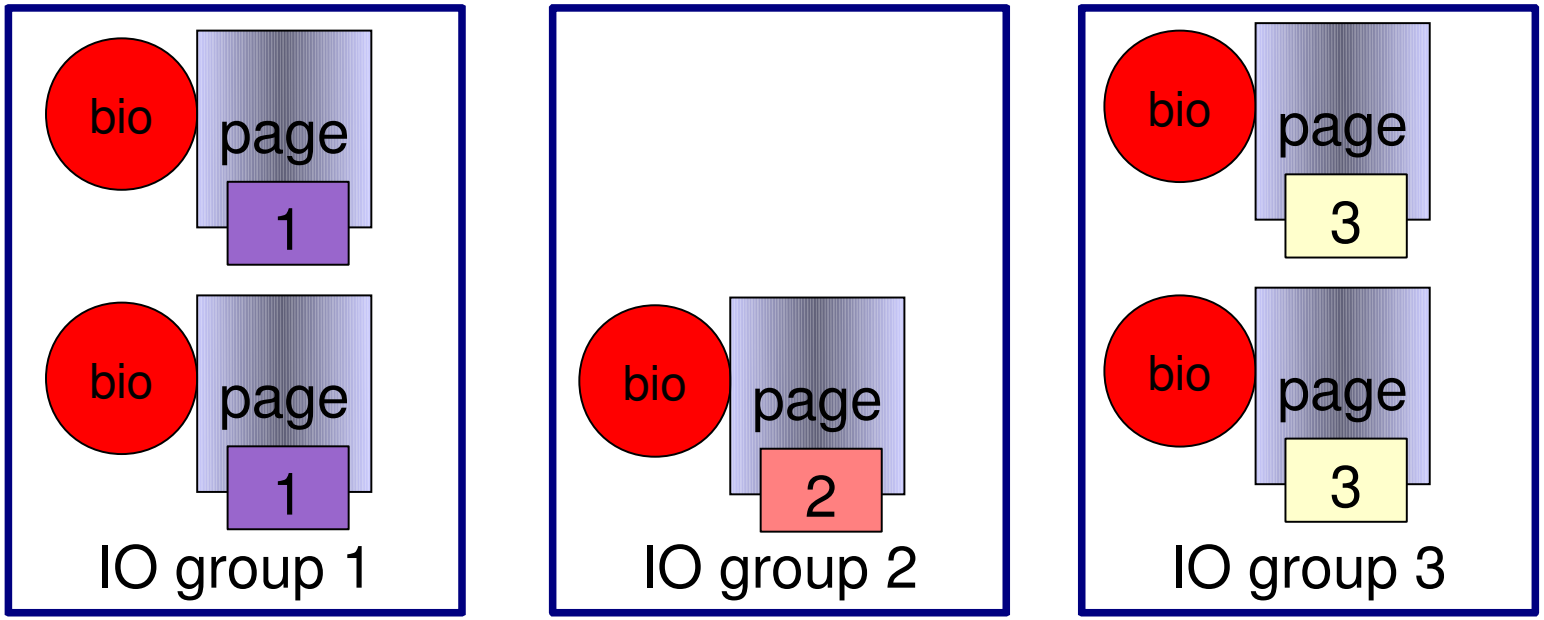
blkio-cgroup

blkio-cgroup sets an ID to a page when a process requests a page or the page is marked dirty.



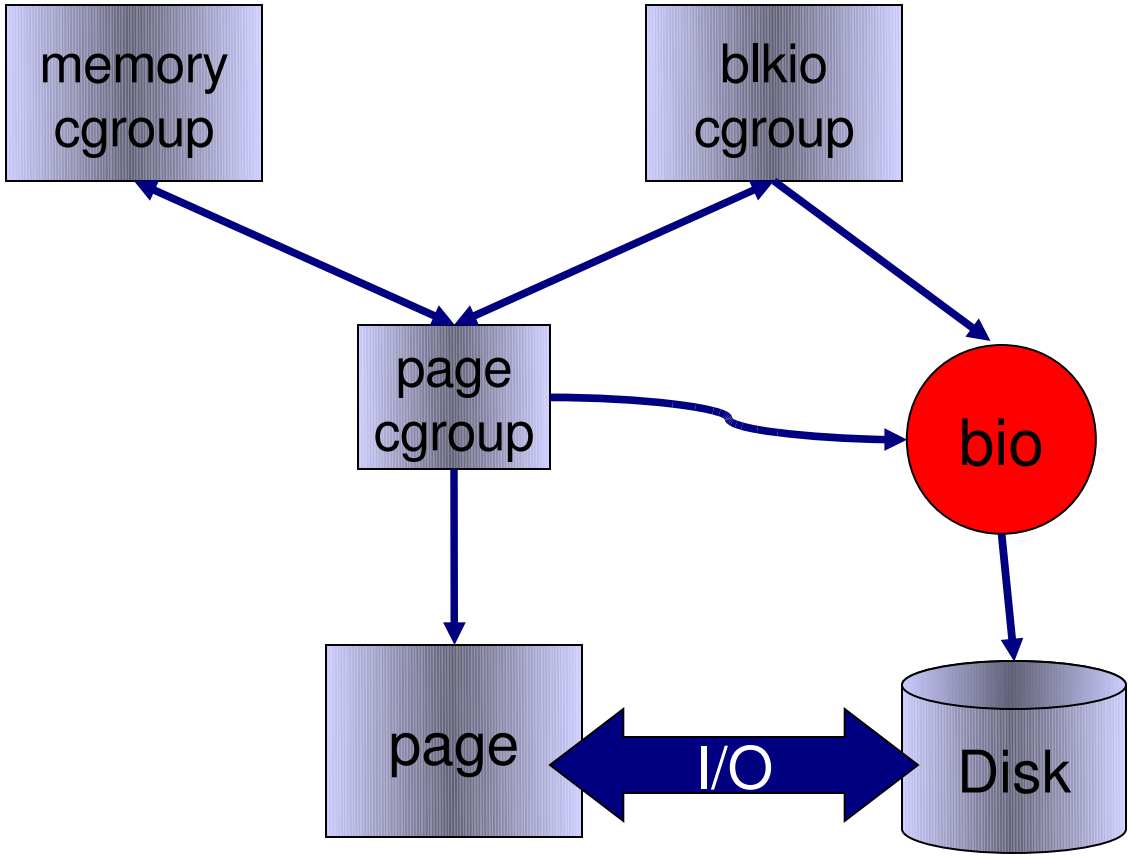
blkio-cgroup

When dm-ioband get an IO request., dm-ioband can determine to which cgroup a bio belongs and charge it to a correspondent group properly.



blkio-cgroup

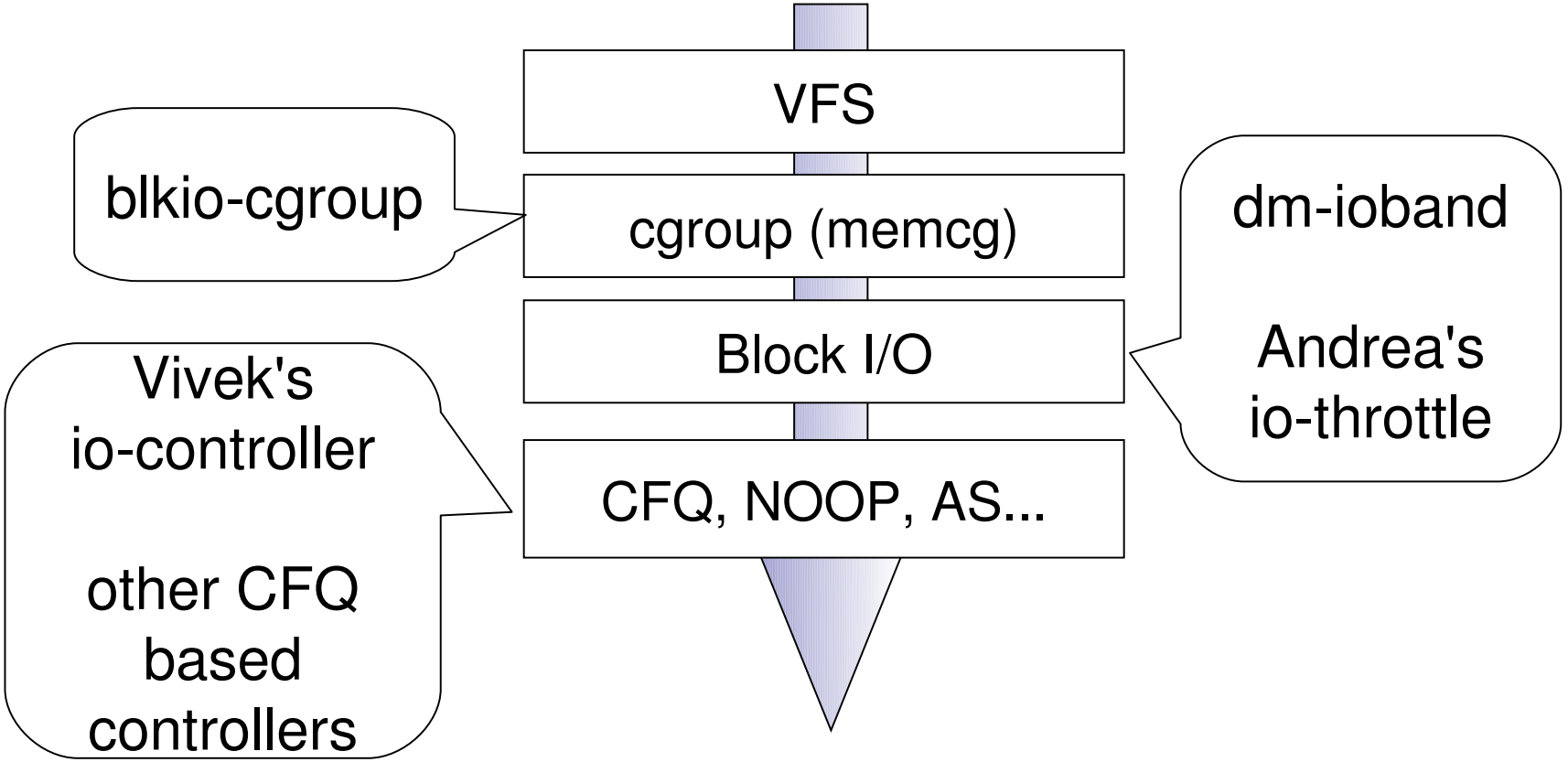
The data structure of blkio-cgroup.



IO controller Mini-summit

IO controller Mini-summit

There were so many proposals of IO controller and had not reached a consensus.



IO controller Mini-summit

The mini-summit is held on Oct 17th (The day before the kernl summit) to discuss a future prospect/direction of development of IO controller.

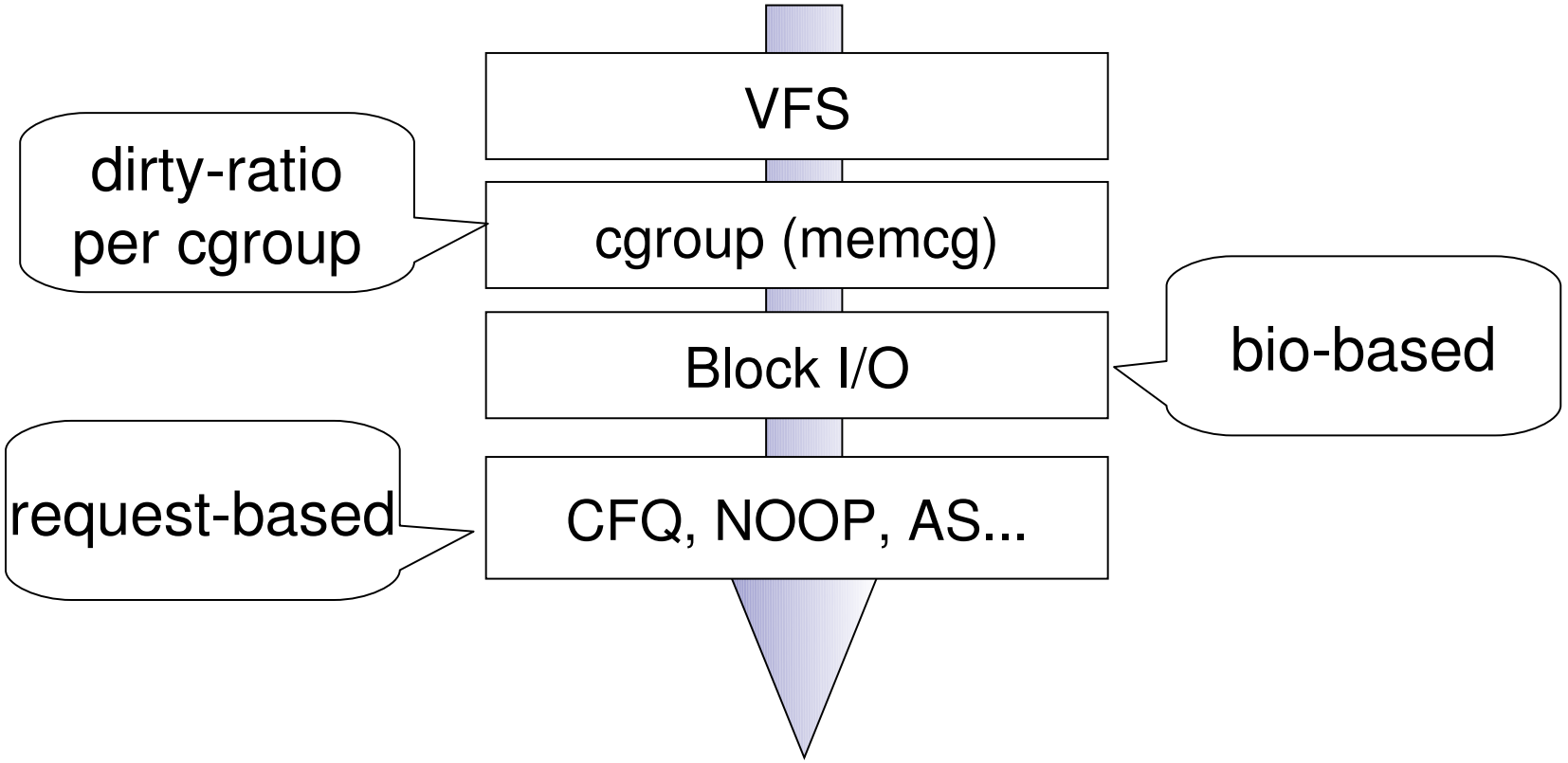
Some kernel maintainers and IO controller developers were gathered to discuss.

Andrea was absent, but he emailed to all attendees prior to the mini-summit. His opinion is hard-limited IO control is necessary for pay-per-use-services.

IO controller Mini-summit

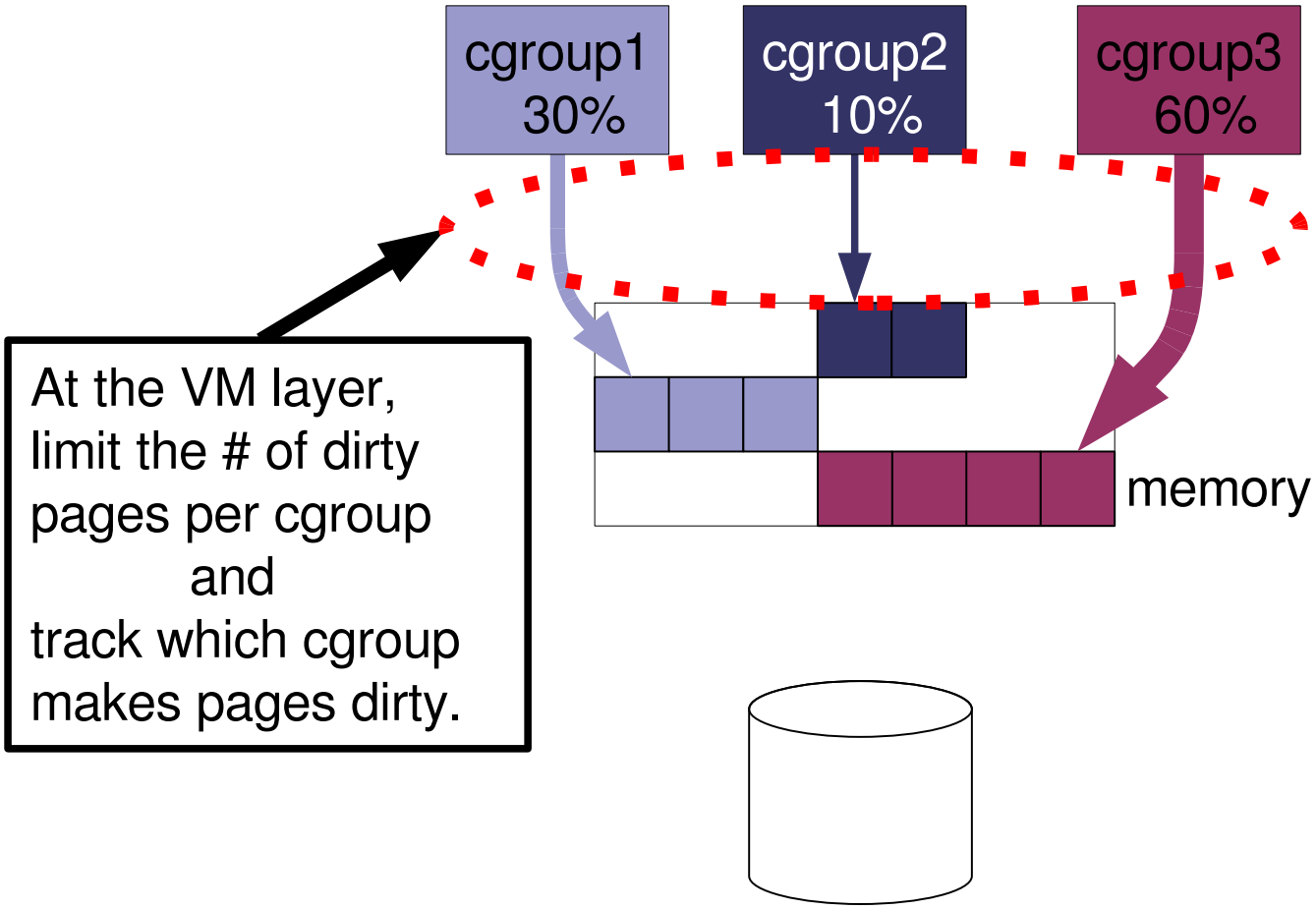
We have reached a consensus on:

- Implement both request-based controller and bio-based controller to meet all IO controlling needs. And dirty-ratio implement at cgroup.



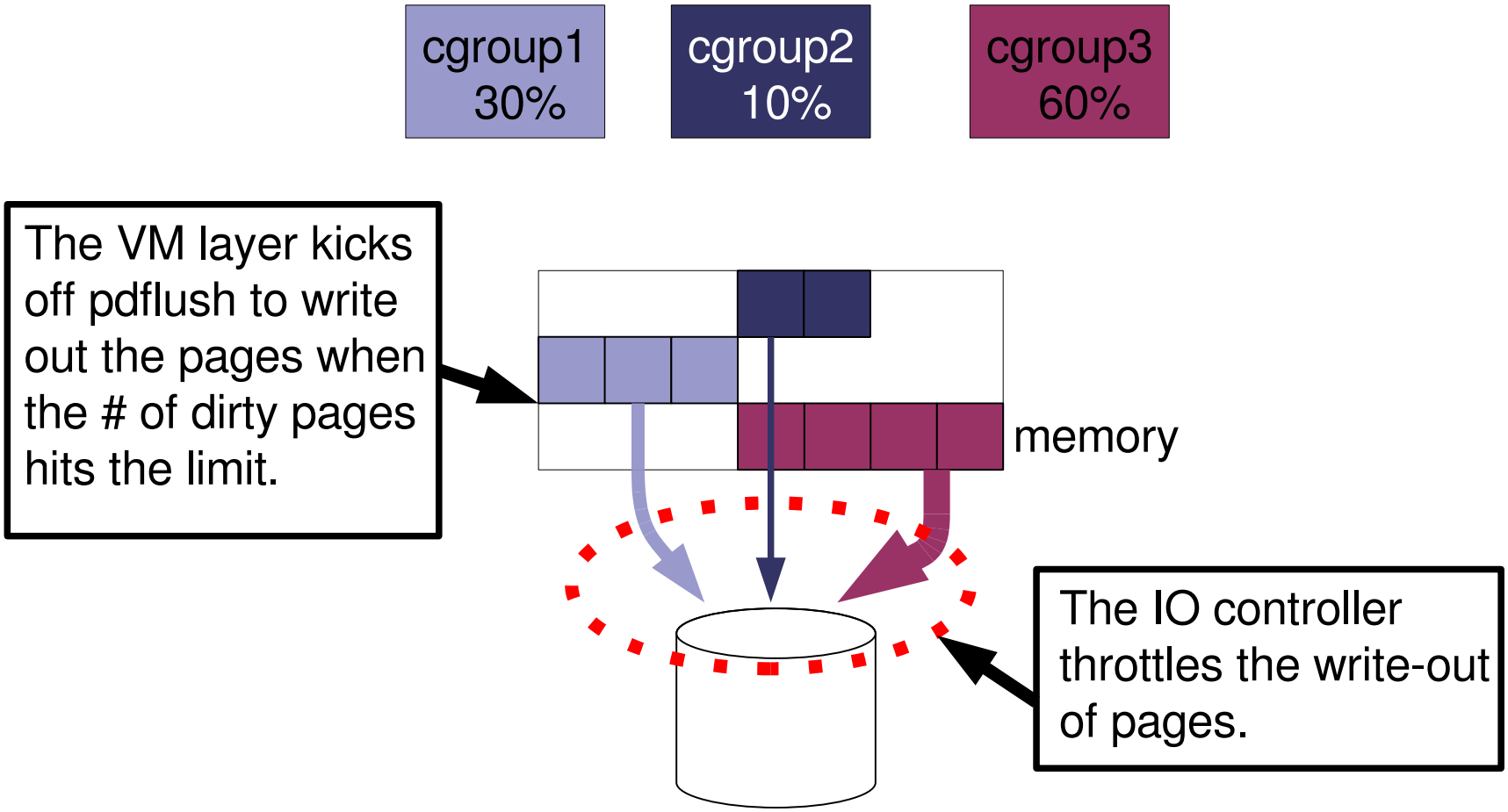
IO controller Mini-summit

- Cooperate with VM layer to control buffered(delayed) writes



IO controller Mini-summit

- Cooperate with VM layer to control buffered(delayed) writes



IO controller Mini-summit

Both bio-based and request-based controller will be implemented, but users don't need to care the difference of two controllers.

- Even when the two controllers are enabled, the new IO controllers avoid starvation, priority inversion issues, and so on.
- Provide a single configuration interface.
- Selectable scheduling policy: proportional weight, bandwidth gurantee and limit, and anything users want.

IO controller Mini-summit

Roadmap

- Make CFQ cgroup aware scheduler and add basic cgroup infrastructures (target 2.6.33 or 2.6.34)
- Once the new CFQ based scheduler is merged to mainline, then start work on bio-based controller and buffered write control. (can be done in parallel)

IO controller Mini-summit

The topics and the conclusions of this summit can be referred at the IO controller Mini-summit web site.

http://sourceforge.net/apps/trac/ioband/wiki/iosummit

Finally, We could get on the same boat of developing IO controller!!

Future work

Future work

- Re-implement dm-ioband's algorithm into the block later. It makes dm-tools no longer required.
- Try to fix the known issues, especially cooperation with the IO scheduler.
- Improve the performance for SAN storages, SSD and upcoming storage devices.

Linux Block IO Bandwidth Controller Project.

ETRI and VA Linux have launched a project for the development of block IO bandwidth controller on Linux at SourceForge.net. More information of dm-ioband and blkio-cgroup is available at:

http://sourceforge.net/apps/trac/ioband/

Please join the project if you are interested in!

This work was partly funded Ministry of Economy, Trade and Industry (METI) of Japan as the Secure Platform Project of Association of Super-Advanced Electronics Technologies (ASET).

Thank you!