Dm-writecache: a New Type Low Latency Writecache Based on NVDIMM

Huaisheng Ye | 叶怀胜

2019.10.19

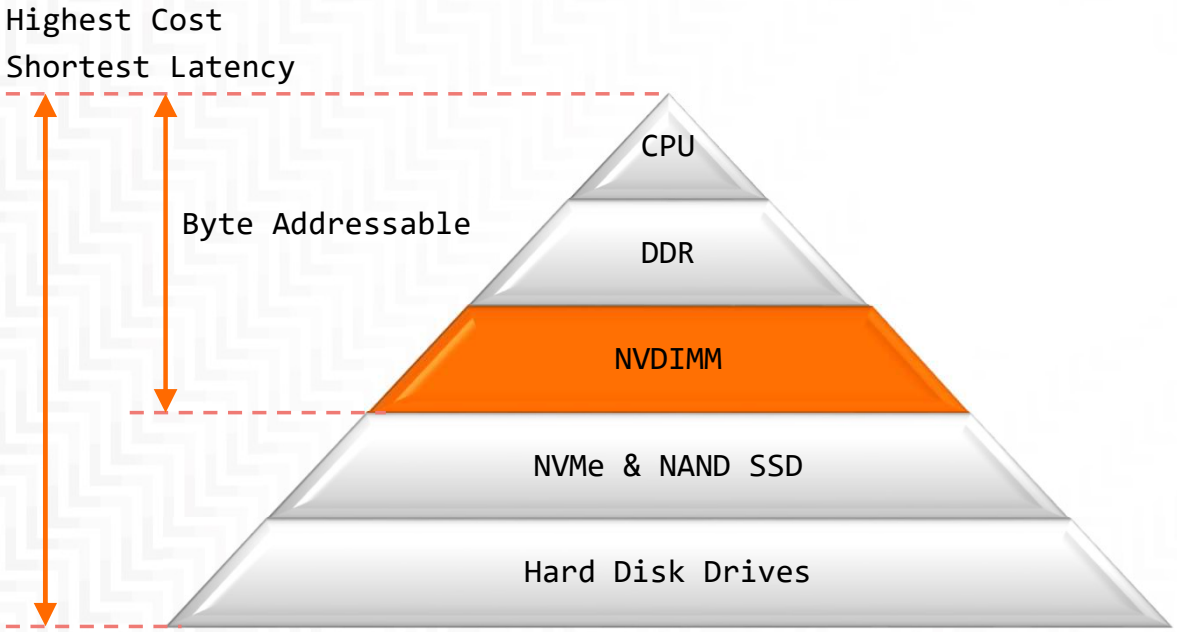
v4

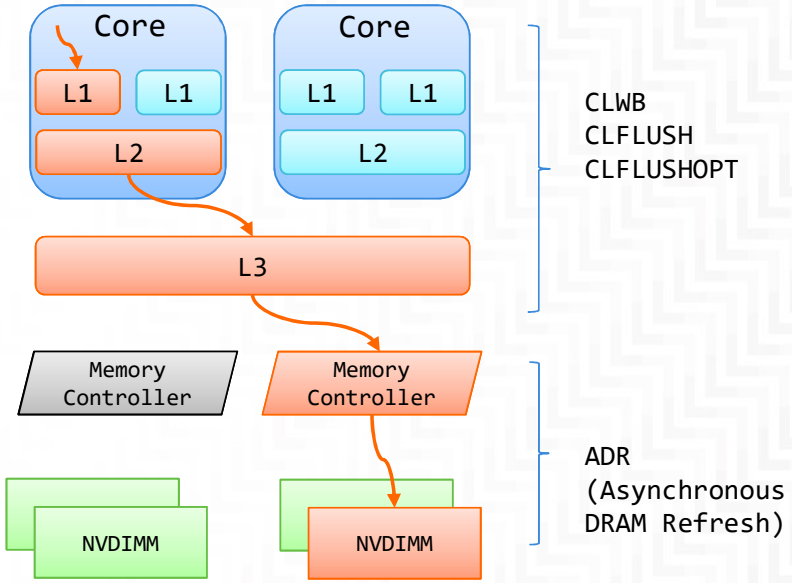
Agenda

1. NVDIMM
2. Device Mapper
3. The structure of dm-writecache
4. Performance comparison with dm-cache
5. Customized solution – Ceph OSD bluestore
6. Patchset from Lenovo
7. Q&A

NVDIMM

- NVDIMM-N
- NVDIMM-F
- AEP (Intel 3D XPoint)

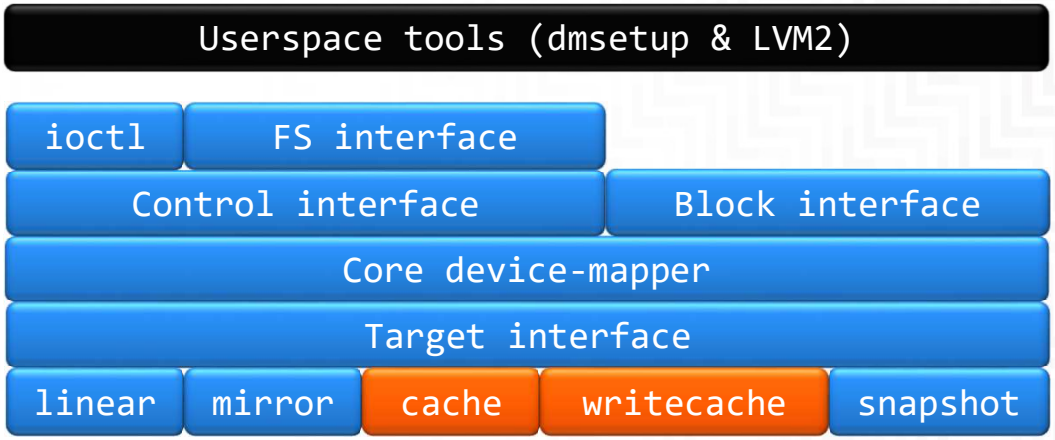




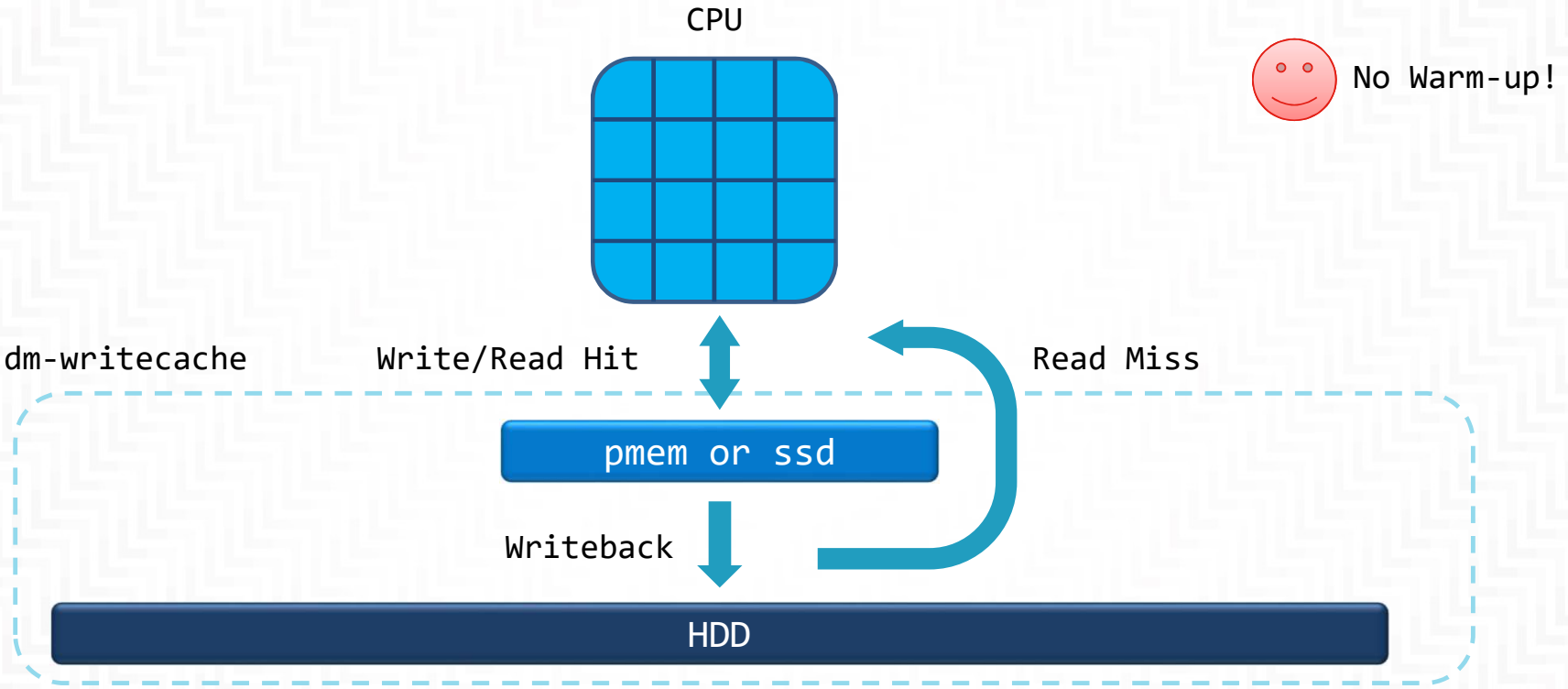
Device Mapper

- dm-linear
- dm-multipath
- dm-mirror
- dm-snapshot
- dm-cache v3.9
- dm-writecache v4.18

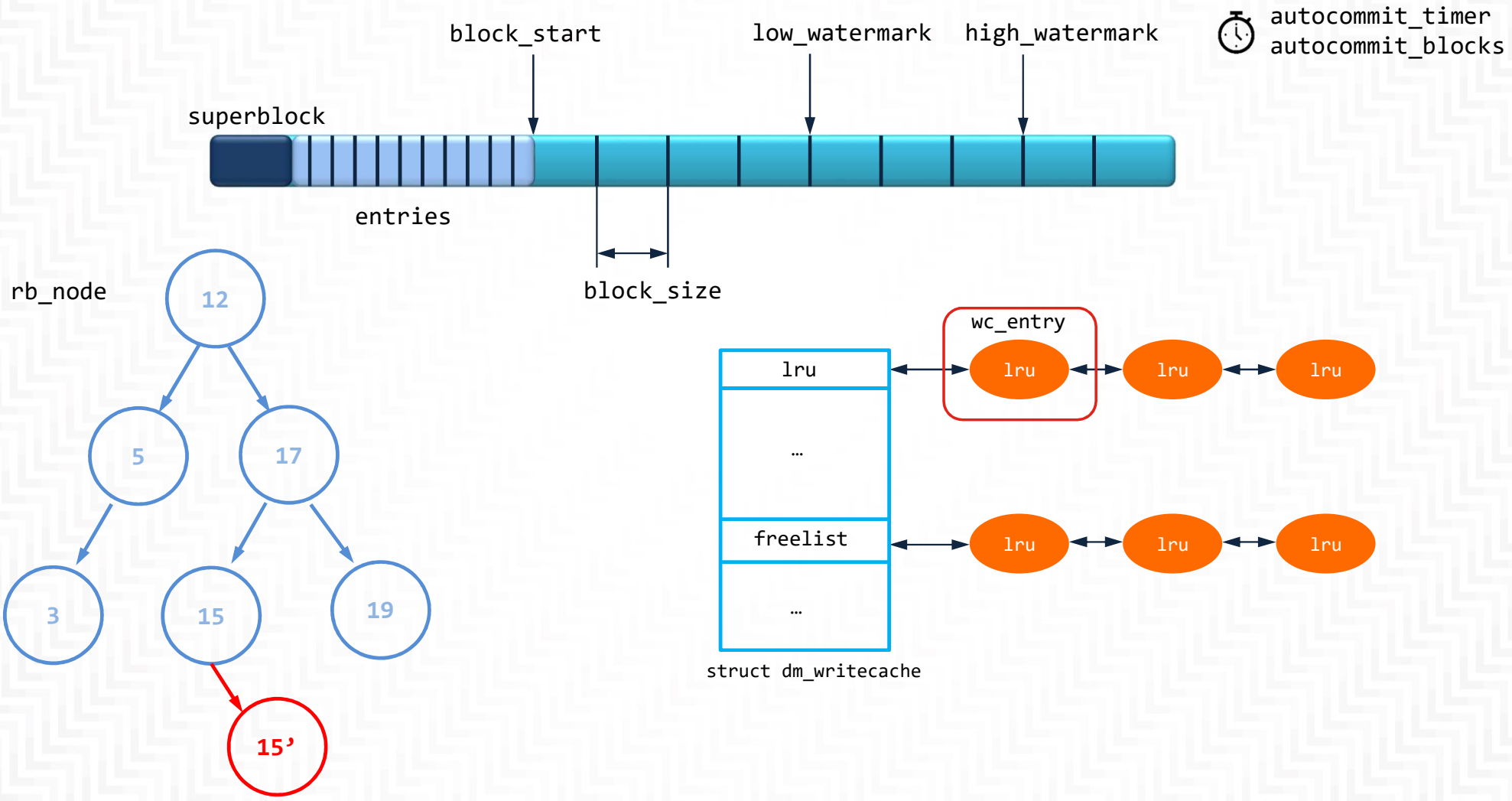
……



How dm-writecache works



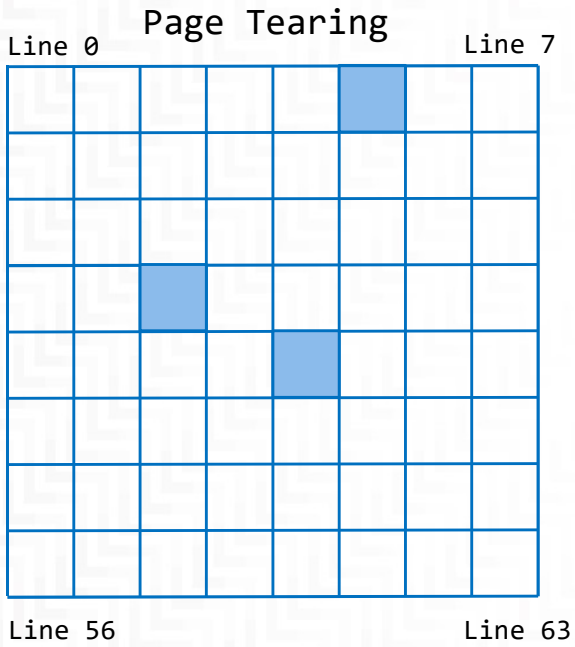
dm-writecache structure

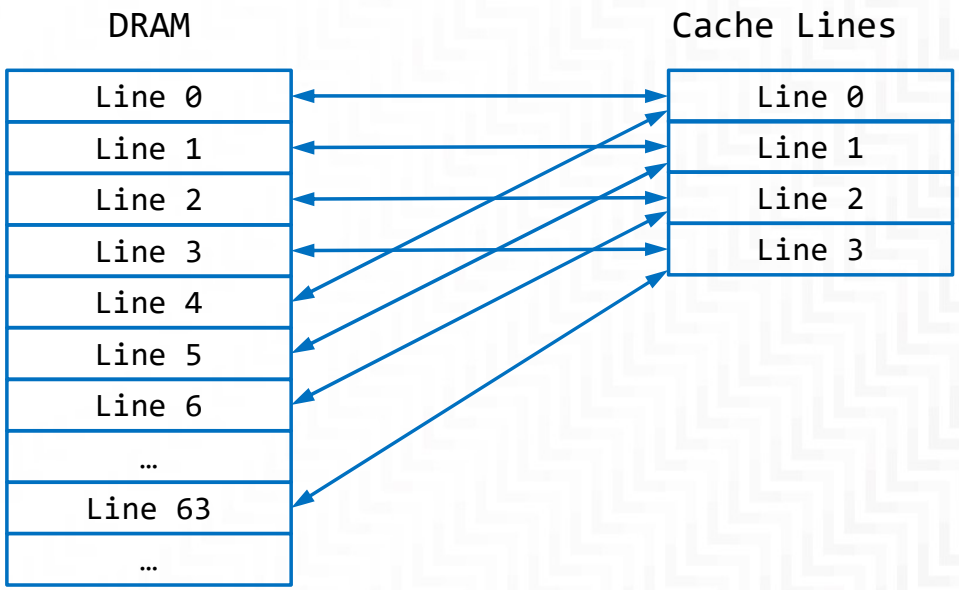


Cache lines

Direct mapped Cache fill: (a simple example of CPU cache lines)

- modern x84_64 has 64 bytes in each cache line, one 4K Page is divided into 64 lines





dm-writecache vs others

dm-writecache specialty:

- Dedicated to Persistent memory and fast SSD
- dm-cache, B-cache, Flushcache: no DAX, no byte addressable capability for NVDIMM.

dm-writecache benefits:

- No warm-up time for Writes
- Sorting blocks partially when writeback
- Compared with PMDK: totally transparent, no modification of Apps

Parameters of dm-writecache

No	Parameter Name	Explanation	Default
1	Cache type	Type of cache device – p or s	p
2	Original device	The underlying device that will be cached	(/dev/sd*)
3	Cache device	The cache device	(/dev/pmem*)
4	Block size	4096 is recommended and the maximum block size is the page size	4096
5	Number of optional	The number of optional parameters	0
6	Start sector	Offset from the start of cache device in 512-byte sectors	0
7	High watermark	Start writeback when the number of used blocks reach this watermark	50
8	Low watermark	Stop writeback when the number of used blocks drops below this watermark	45
9	Writeback jobs	Limit the number of blocks that are in flight during writeback. Setting this value reduces writeback throughput, but it may improve latency of read requests	Unlimited
10	Auto commit blocks	When the application writes this amount of blocks without issuing the FLUSH request, the blocks are automatically commited	Pmem: 64 SSD: 65536
11	Auto commit time	Autocommit time in milliseconds. The data is automatically commited if this time passes and no FLUSH request is received	1000
12	FUA	Applicable only to persistent memory - use the FUA flag when writing data from persistent memory back to the underlying device	On

HW Platform for testing

Thinksystem SR630

CPU	Intel(R) Xeon(R) Gold 6240 CPU @ 2.60GHz * 2
MEM	16G * 8
NVDIMM	Intel Optane DCPMM 128G * 4
HDD	300GB 15K 12Gbps SAS 2.5" HDD (JBOD)

Performance comparison with dm-cache

dm-writecache: (fsdax mode)

```
# lvcreate -n hdd_wc   -L 4G  vg /dev/sda1
# lvcreate -n cache_wc -L 1G  vg /dev/pmem0
# lvchange -a n vg/cache_wc
# lvconvert --type writecache --cachevol  cache_wc vg/hdd_wc
```

Namespace modes:

- fsdax
- sector
- raw
- devdax

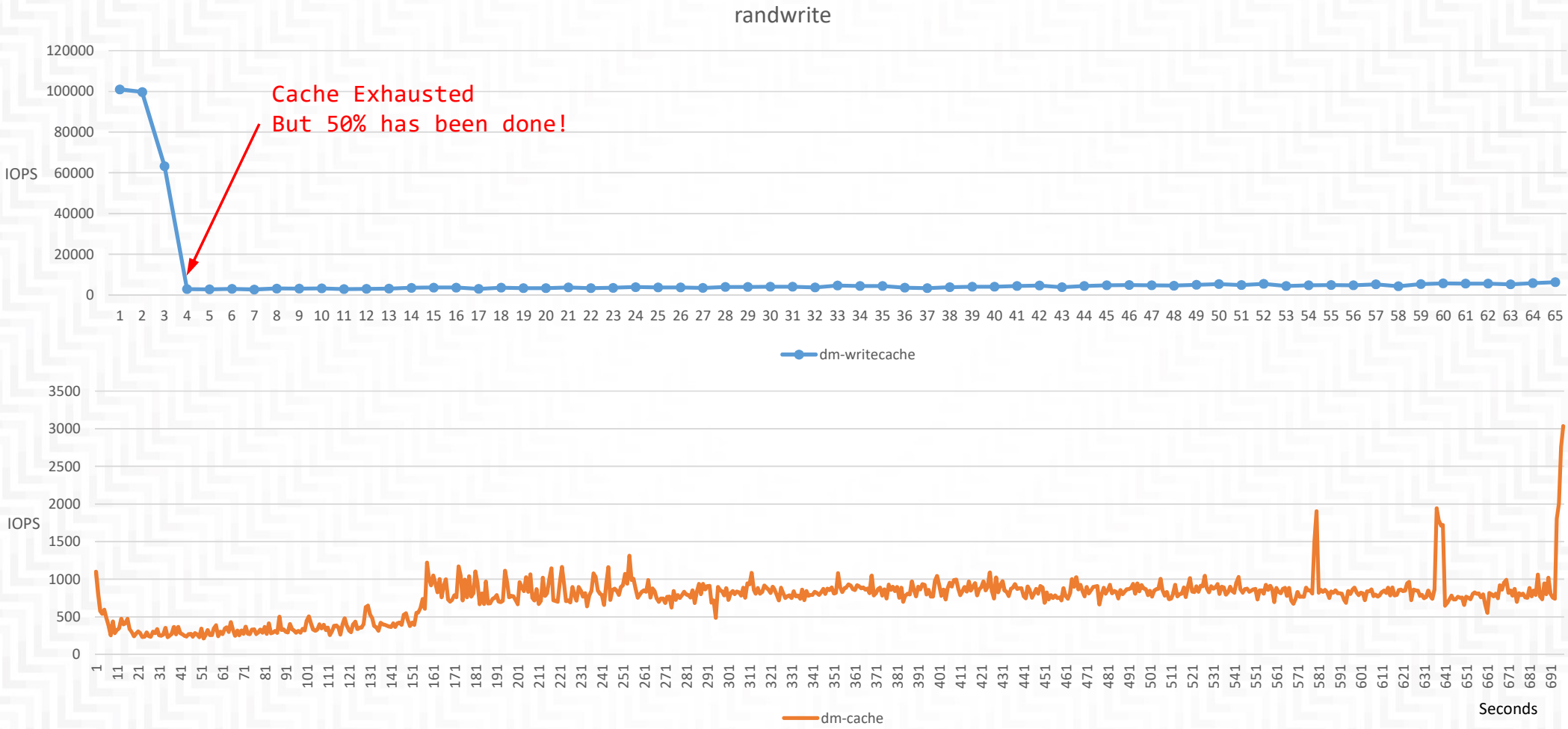
dm-cache: (sector mode)

```
# lvcreate -n hdd_s   -L 4G vg /dev/sda1
# lvcreate -n cache_s -L 1G vg /dev/pmem0.2s
# lvconvert --type cache --cachevol cache_s vg/hdd_s --chunksize=32 --cachemode writeback
```

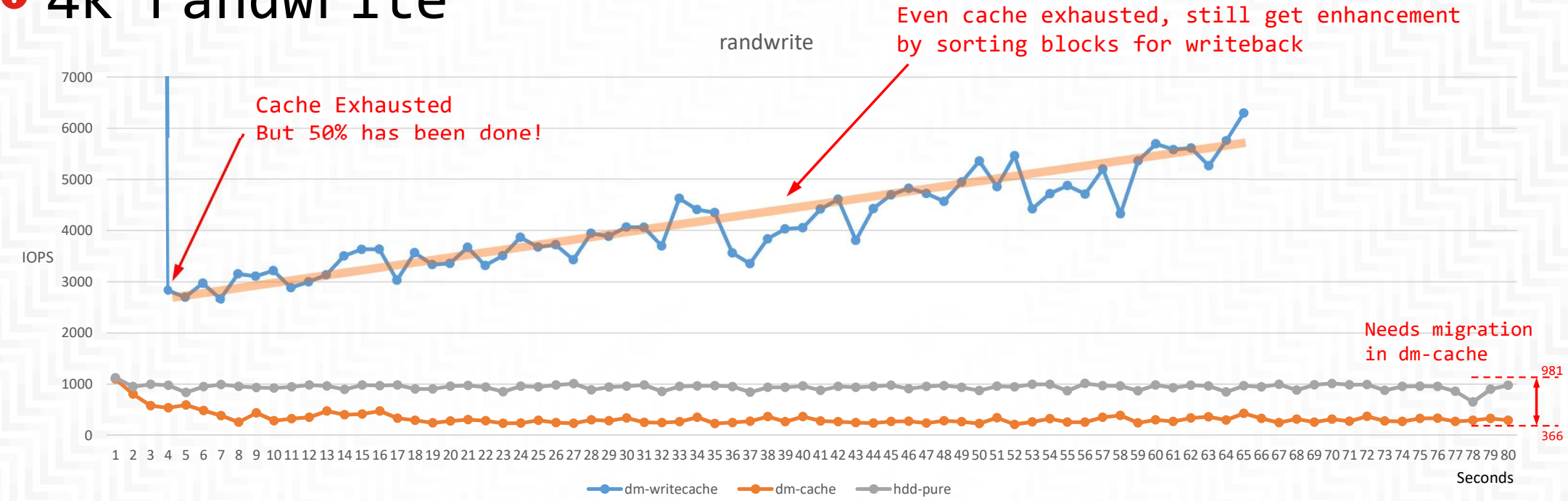
Eg.

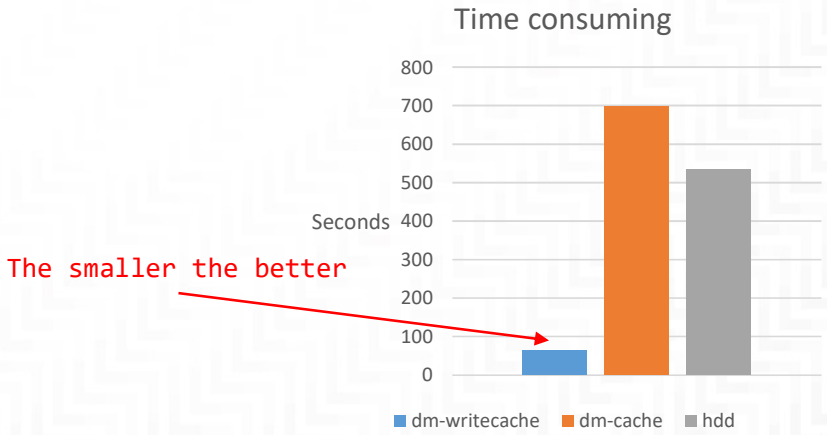
```
# fio -filename=/dev/vg/hdd_wc -direct=1 -iodepth=20 -rw=randwrite -ioengine=libaio -bs=4k -loops=1
-size=2g -group_reporting -name=mytest1 --write_iops_log=fio-randwrite-wc --log_avg_msec=1000
```

4k randwrite (Full View)

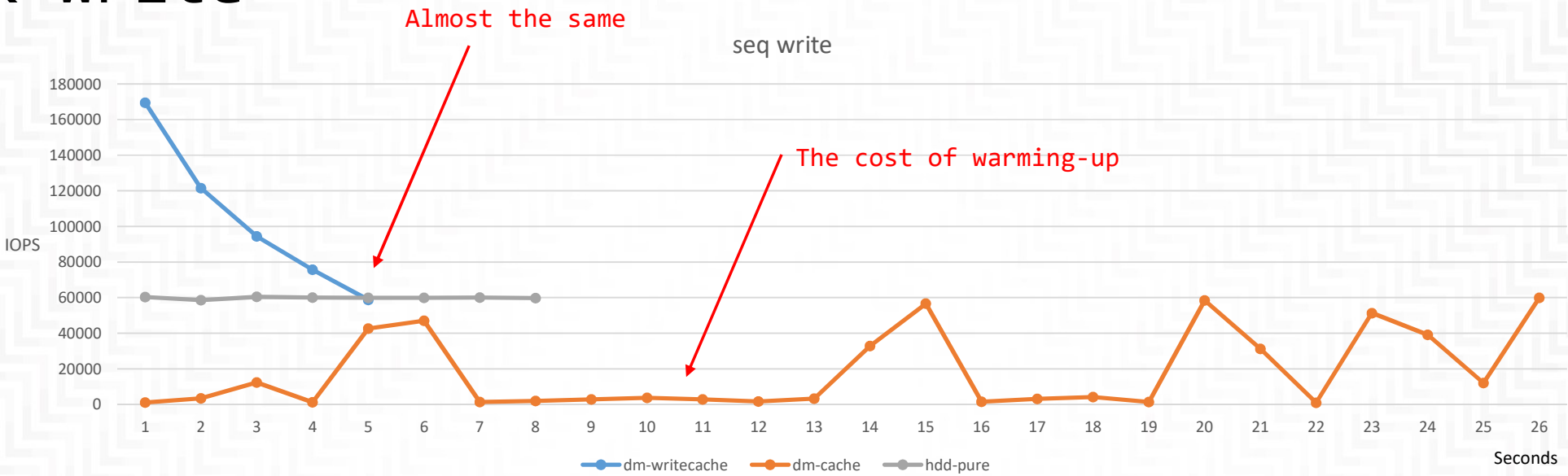


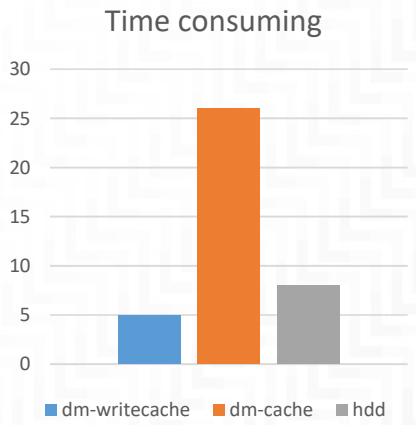
4k randwrite



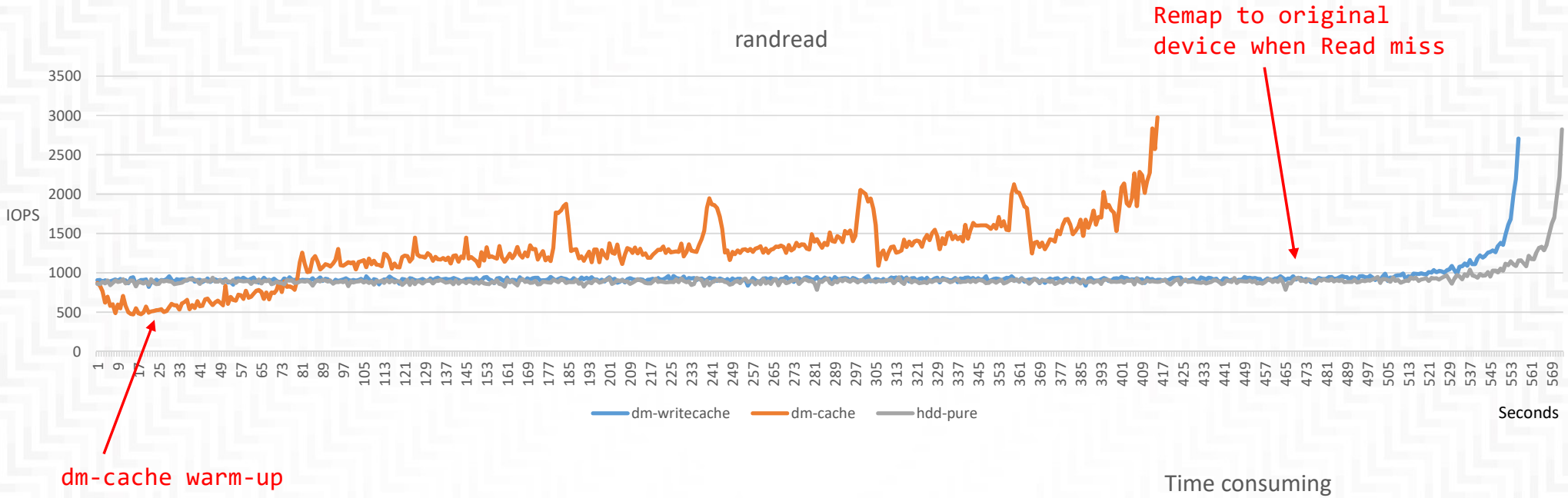


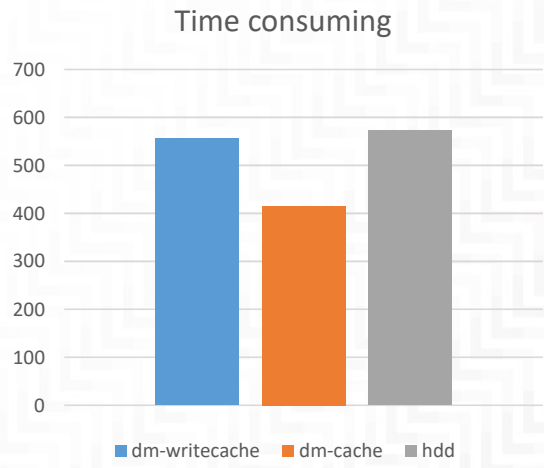
4k write


Almost the same
seq write
The cost of warming-up
IOPS
180000
160000
140000
120000
100000
80000
60000
40000
20000
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
Seconds
dm-writecache
dm-cache
hdd-pure

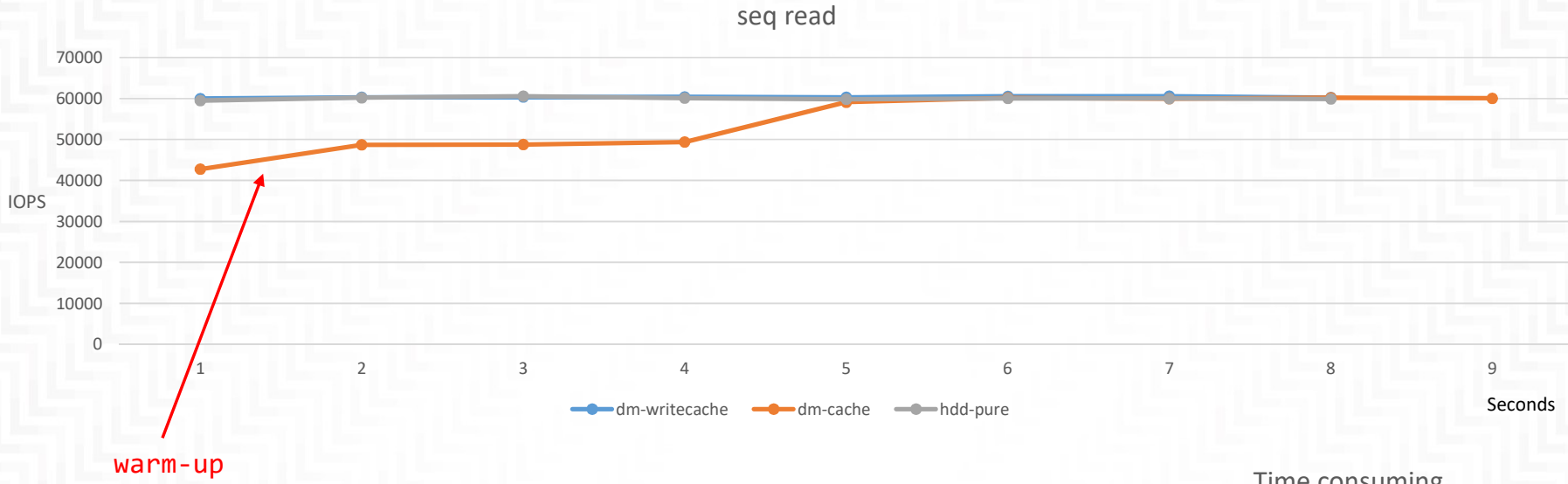

Time consuming
30
25
20
15
10
5
0
dm-writecache
dm-cache
hdd

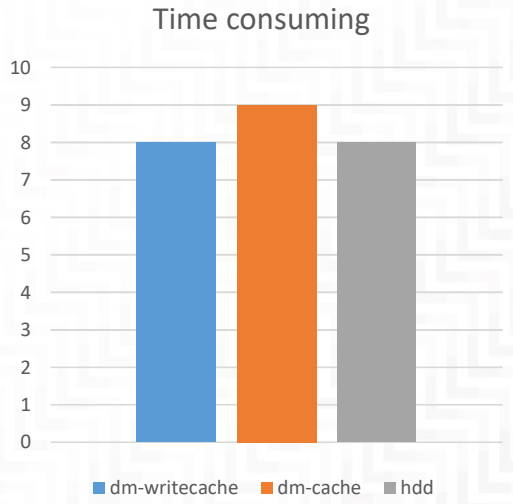
4k randread


randread
Remap to original device when Read miss
IOPS
3500
3000
2500
2000
1500
1000
500
0
dm-writecache
dm-cache
hdd-pure
Seconds
dm-cache warm-up

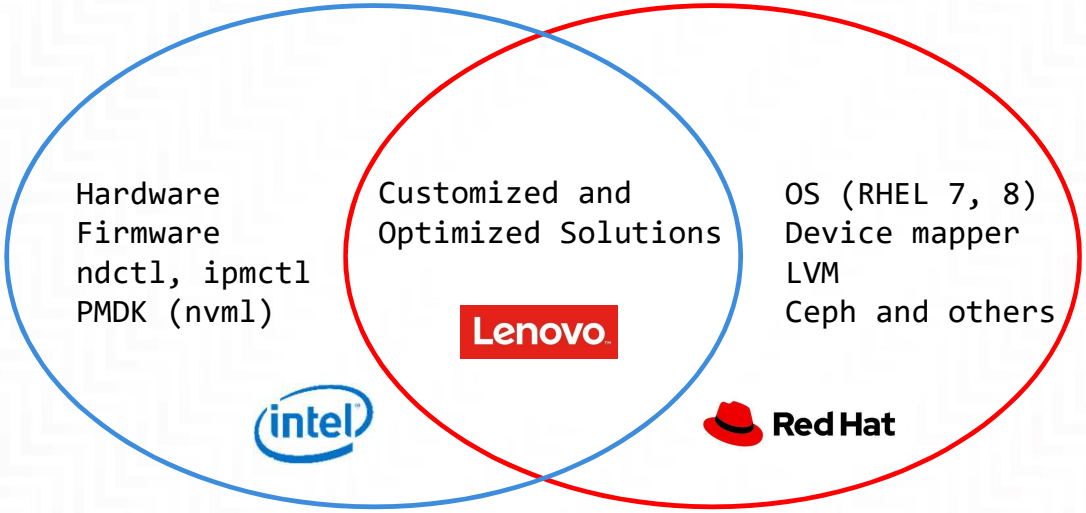

Time consuming
700
600
500
400
300
200
100
0
dm-writecache
dm-cache
hdd

4k read


seq read
70000
60000
50000
40000
30000
20000
10000
0
IOPS
1
2
3
4
5
6
7
8
9
Seconds
dm-writecache
dm-cache
hdd-pure
warm-up

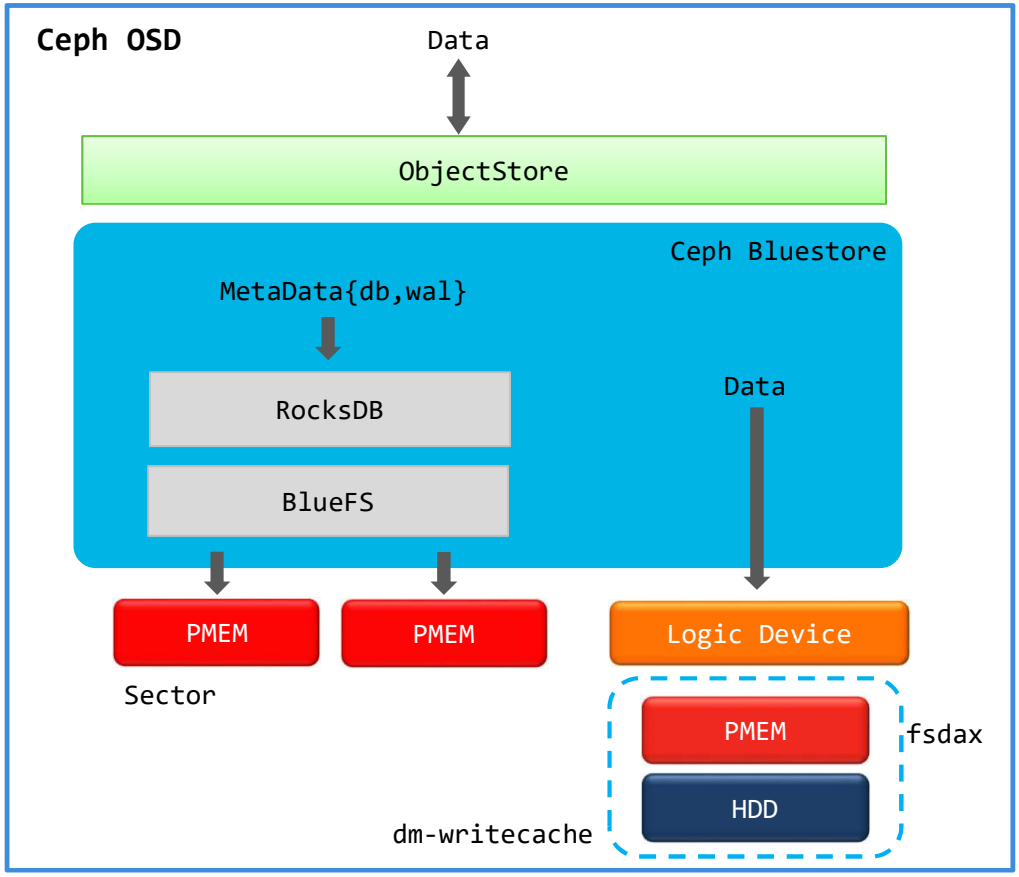

Time consuming
10
9
8
7
6
5
4
3
2
1
0
dm-writecache
dm-cache
hdd

How does Lenovo play over NVDIMM long term



Customized Ceph OSD

- NVDIMM serves as pmem writecache for HDD device
- Performance boost for random write
- No code change in Ceph
- POC is ready



Prepare Logical volume for Ceph OSD

dm-writecache:

```
# lvcreate -n osd0.wal        -L 4G    vg-ceph /dev/pmem1.1s
# lvcreate -n osd0.db         -L 4G    vg-ceph /dev/pmem1.1s
# lvcreate -n osd0            -L 40G   vg-ceph /dev/sdb1

# lvcreate -n osd0.cache-wc   -L 10G   vg-ceph /dev/pmem1
# lvchange -a n vg-ceph/osd0.cache-wc
# lvconvert --type writecache --cachevol  osd0.cache-wc vg-ceph/osd0

# ceph-deploy osd create --data vg-ceph/osd0 --block-db vg-ceph/osd0.db --block-wal vg-ceph/osd0.wal --bluestore
target01
```

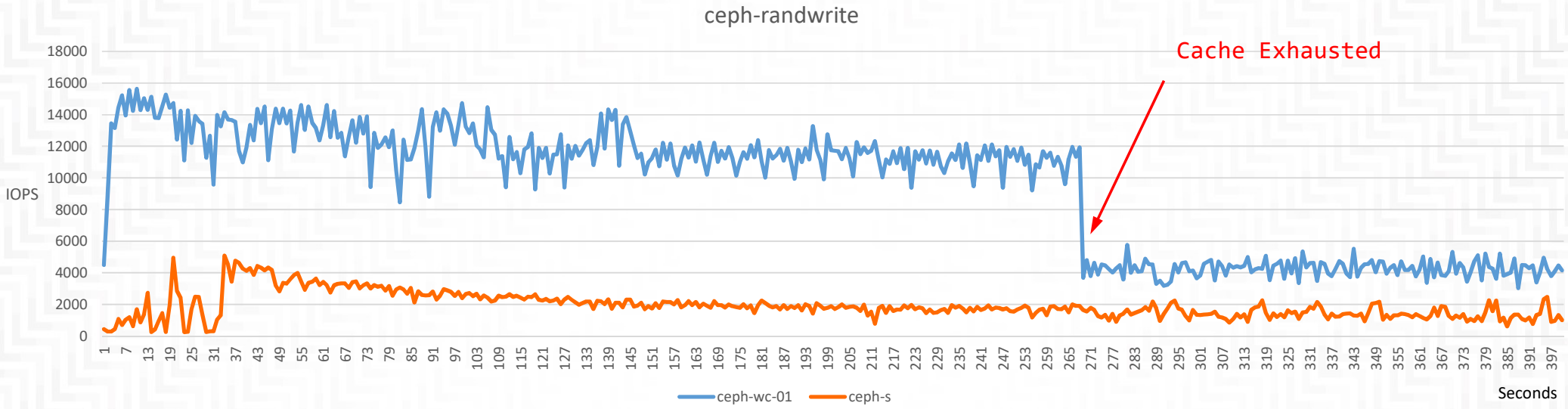
---

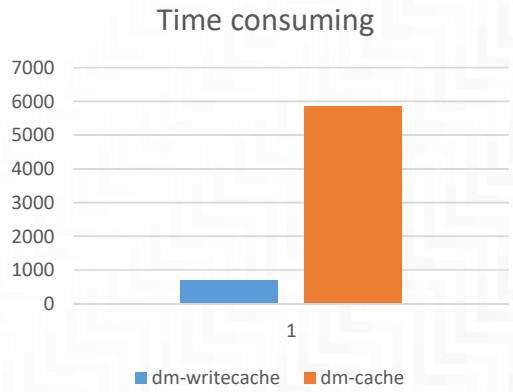
dm-cache:

```
# lvcreate -n osd1.wal        -L 4G    vg-ceph /dev/pmem1.1s
# lvcreate -n osd1.db         -L 4G    vg-ceph /dev/pmem1.1s
# lvcreate -n osd1            -L 40G   vg-ceph /dev/sdb1

# lvcreate -n osd1.cache-s    -L 10G   vg-ceph /dev/pmem1.1s
# lvconvert --type cache      --cachevol osd1.cache-s vg-ceph/osd1 --chunksize=32 --cachemode writeback

# ceph-deploy osd create --data vg-ceph/osd1 --block-db vg-ceph/osd1.db --block-wal vg-ceph/osd1.wal --bluestore
target01
```

```
# fio -filename=/dev/rbd0 -direct=1 -iodepth=20 -rw=randwrite -ioengine=libaio -bs=4k -loops=1  -size=20g
-group_reporting -name=mytest1 --write_iops_log=fio-randwrite-s-ceph-01 --log_avg_msec=1000
```

4k randwrite for Ceph rbd



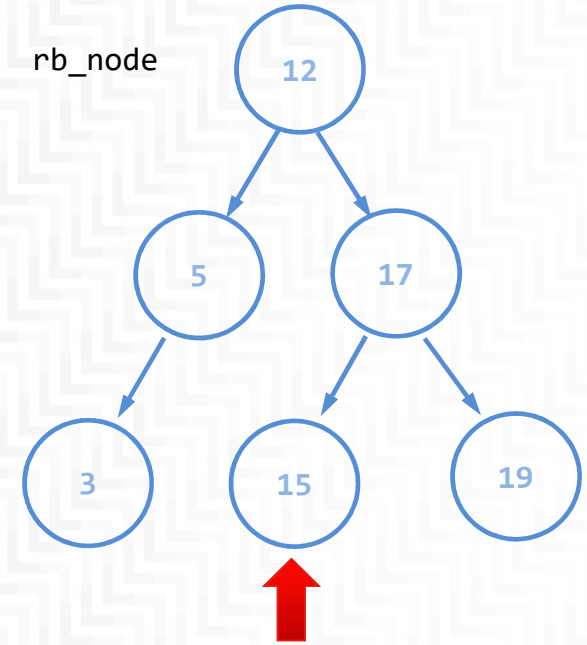


Patchset from Lenovo OS team

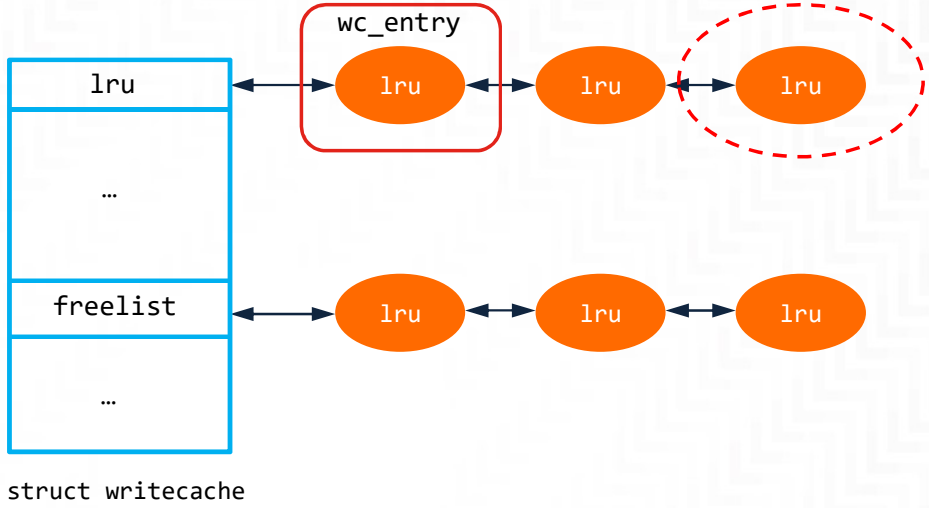
1. Lenovo has contributed 26 patches to kernel community
2. Most of them belong to NVDIMM and Device mapper subsystem
3. Continues to submit features to community
4. Latest patchset is performance optimization, saves 50+% time on writeback_all

https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=5229b4896e8f32bda4bfe29ff91e594ae7aa8a75
https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=f8011d334426cee77276a1038b627b5cb0470258
https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=09f2d6563055b8ff0948cefb8911a4de0d559963
https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=6d1959138c8bdaf69f1116c86c77e6733db6ab34

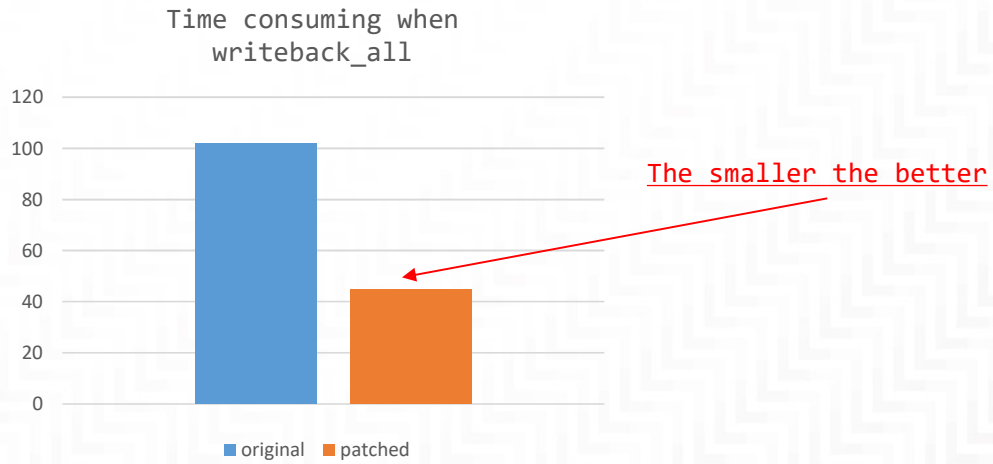
Patchset for writeback_all



struct writeback_list wbl;



struct writecache



Reference

https://www.kernel.org/doc/html/latest/admin-guide/device-mapper/writecache.html
https://ceph.com/resources/
https://pmem.io/documents/NVDIMM_DriverWritersGuide-July-2016.pdf
https://docs.pmem.io/ndctl-users-guide
https://docs.pmem.io/ipmctl-user-guide
https://github.com/pmem/ndctl

Q&A

WeChat

Any suggestion and comment is welcome!

THANK YOU

DAKUJEM DANK BEDANKT MERCI TAKK 谢谢
ありがとう СПАСИБО GRACIAS DZIĘKUJĘ DANKE
OBRIGADO БЛАГОДАРЯ GRAZIE תודה GRACIAS

Lenovo™