ZBC/ZAC Support in Linux

Damien Le Moal
Western Digital

2016 Storage Developer Conference. © 2016 Western Digital Corporation. All Rights Reserved.

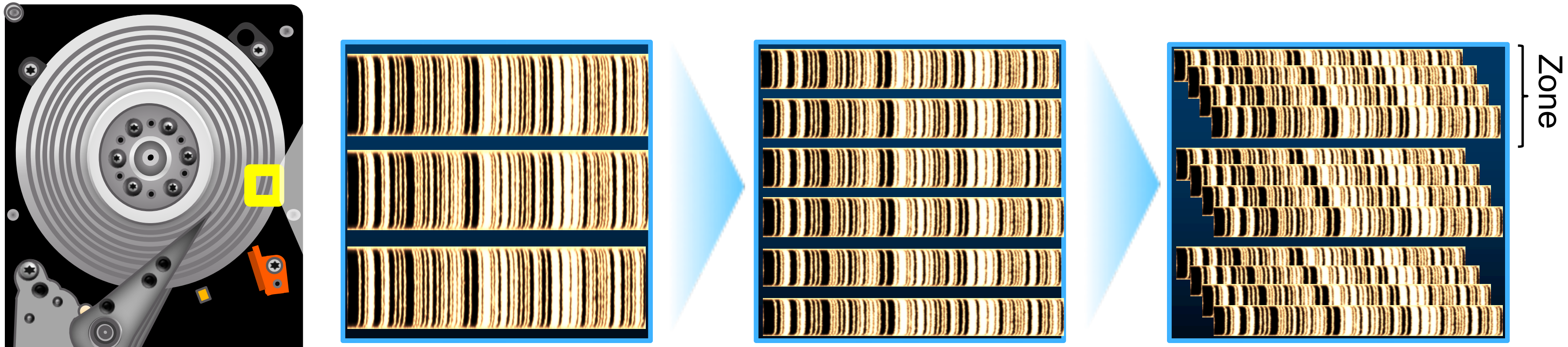
Outline

- Background: Shingled Magnetic Recording (SMR)
  - Device interface, standard and constraints on host software
- Linux kernel support
  - SCSI stack, block I/O stack, API
- Some evaluation results
  - File systems and device mapper
- Conclusion

Foreword and Acknowledgement

- ZBC/ZAC support in Linux is an ongoing effort
  - Mechanisms and API presented here may change in the final release
- This development is a community effort with many contributors
  - Dr Hannes Reinecke, Christoph Hellwig, Shaun Tancheff, Damien Le Moal
  - And many others

Shingled Magnetic Recording (SMR)



Conventional PMR HDD

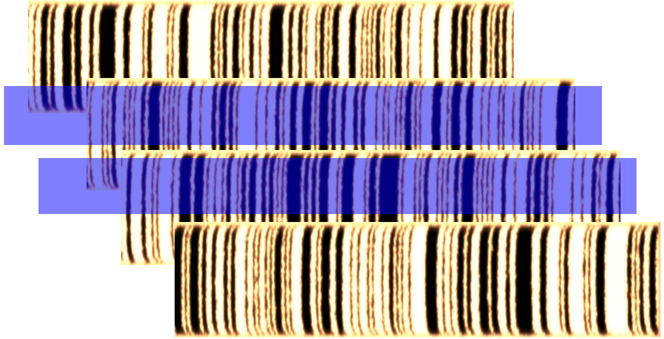
Data in Discrete Tracks. Capacity increase achieved with narrower tracks

SMR HDD

Data in Zones of Overlapped wider tracks

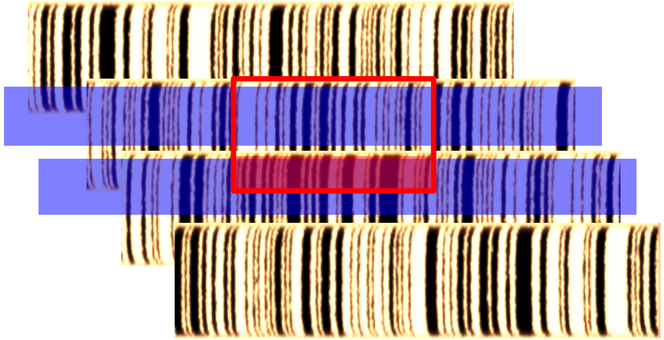
Higher Disk Capacity, And More !

- Higher (read) track density increases disk capacity, and more…
  - Wider write head produces higher fields, enabling smaller grains and lower noise
  - Better sector erasure coding, reduced ATI exposure, and more powerful data detection and recovery



But…

- While track zones are independent, sectors cannot be modified independently within a zone
  - Random reads similar to PMR
  - But sequential writes within a zone
- Disk firmware can hide or expose zones and write constraint
  - Standardized disk interface



SMR Standards

- Command set
  - T10 (SCSI) Zoned Block Command (ZBC) and T13 (ATA) Zoned-device ATA command set (ZAC)
  - Both semantically identical
  - Latest drafts r05 forwarded to INCITS for processing towards publication
- SCSI to ATA translation (SAT) specifications updated
  - Draft in ballot review

2016 Storage Developer Conference. © 2016 Western Digital Corporation. All Rights Reserved.

Standardized Disk Models

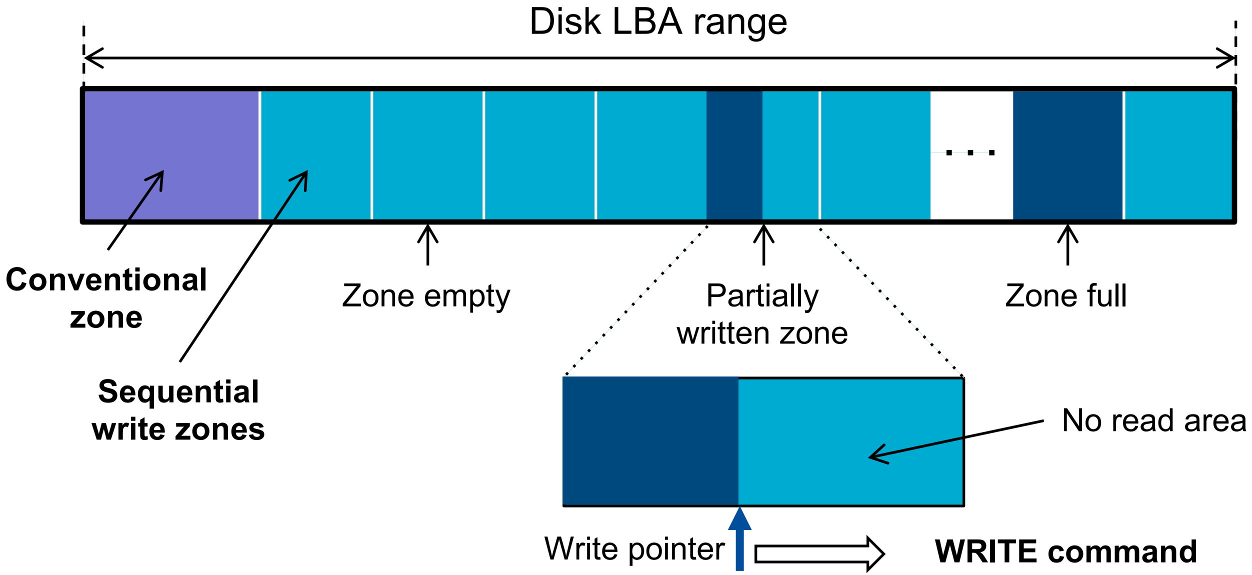
Model	Description	Impact on Host Software
Drive Managed (DM)	• Disk firmware handles random writes processing • Backward compatible (standard Device Type 0H) • Performance can be unpredictable in some workloads	**NONE**
Host Managed (HM)	• Host must use zone commands to handle write operations • Not backward compatible (Device type 14h) • Predictable Performance	HIGH Host must write sequentially into zones
Host Aware (HA)	• Disk firmware handles random writes processing • Backward compatible (standard Device type 0H) • Host can use zone commands to optimize write behavior • Performance can be unpredictable if the host sends a “sub-optimal” request	NONE ~ HIGH Depends on the amount of optimization

2016 Storage Developer Conference. © 2016 Western Digital Corporation. All Rights Reserved.

Standardized Zone Types

- Conventional zones
  - Unconstrained read & write operations
  - Optional for HA and HM
- Write pointer zones
  - HA: Sequential write preferred zones
    - Unconstrained read & write operations possible
  - HM Sequential write required zones
    - Write operations must be sequential
    - No read after write pointer position

Host Disk View



ZBC & ZAC Command Set

- 2 main commands
  - **REPORT ZONES**: get disk zone layout and zone status
    - Sequential zone write pointer position
  - **RESET WRITE POINTER**: “rewind” a sequential zone
    - Set write pointer at the beginning of the zone
- 3 additional commands for software optimization
  - **OPEN ZONE**: keep a zone FW resources locked
  - **CLOSE ZONE**: release a zone FW resources
  - **FINISH ZONE**: fill a zone

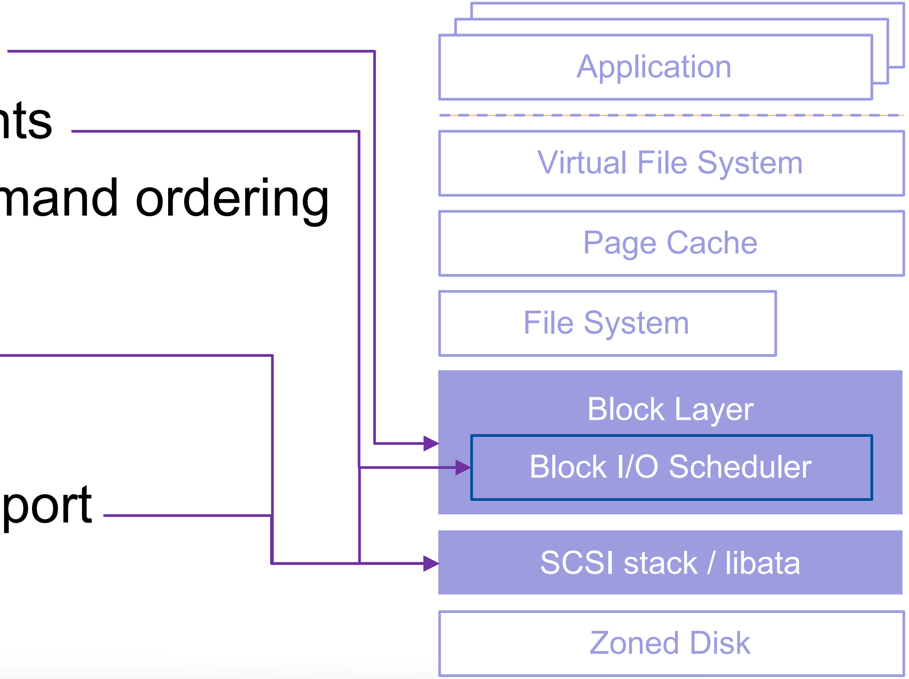
Linux Kernel: What Do We Have ?

- As of kernel v 4.7
  - ZAC command set and translation from ZBC implemented
    - But no ZBC support in the SCSI disk driver
  - SG_IO is the only interface available to issue ZBC commands
    - From applications only
  - Host aware drives are seen as regular block devices
    - No differentiation with regular disks
  - Host managed drives are exposed as SG node
    - No block device file

2016 Storage Developer Conference. © 2016 Western Digital Corporation. All Rights Reserved.

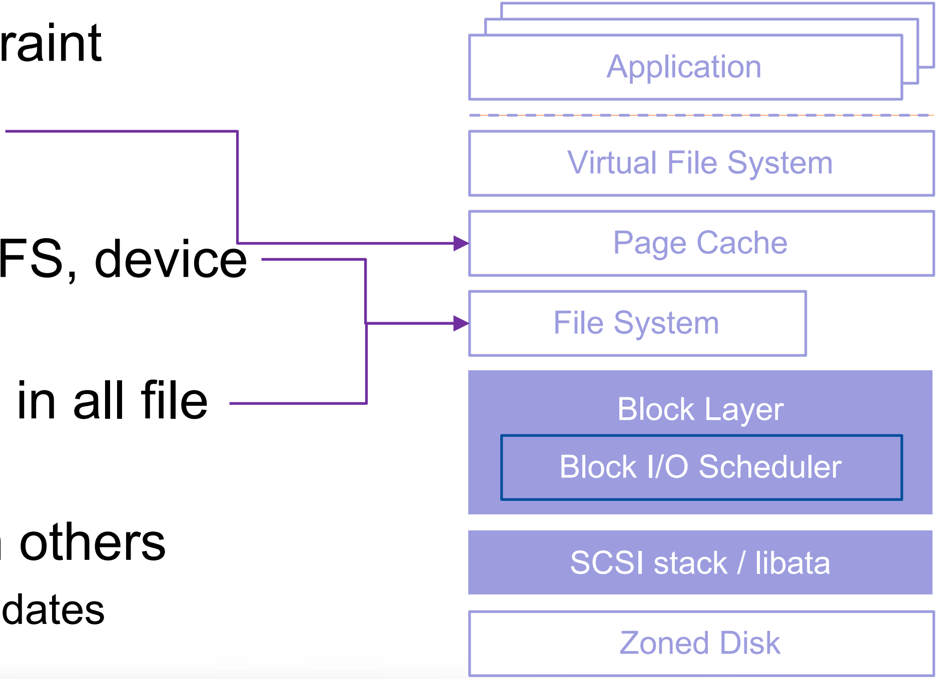
What Is Needed ?

- API integrated in block I/O stack
- Respect read and write constraints
  - Ensure sequential write command ordering
  - No read after write pointer
- New device type support
  - Host managed
- ZBC and ZAC command set support
  - Zone information and control



What Is Not Being Considered

- Hide HM sequential write constraint
  - No changes to page cache
    - Too complex
  - Responsibility of disk user (FS, device mapper or application)
- Natively support zoned devices in all file systems
  - Some are better suited than others
    - f2fs, nilfs, btrfs are good candidates
    - Device mapper for others



2016 Storage Developer Conference. © 2016 Western Digital Corporation. All Rights Reserved.

Upper Block Layer

- I/O constraints require differentiation from regular block devices
  - Block device request queue is flagged as “zoned” with the device type (HA or HM)
  - A zone information cache is attached to the device request queue
    - On-the-fly I/O checks possible without needing a disk access for a zone report
    - Implemented as a RB-tree for efficiency

```
struct blk_zone {
        struct rb_node   node;
        unsigned long    flags;
        sector_t         len;
        sector_t         start;
        sector_t         wp;
        unsigned int     type : 4;
        unsigned int     cond : 4;
        unsigned int     non_seq : 1;
        unsigned int     reset : 1;
};

unsigned int blk_queue_zoned(struct
request_queue *q)
```

Zoned Block Device API

- Zone information access
  - Cache only or with update from disk
- Zone manipulation
  - Reset write pointer, open zone, close zone, finish zone
- Upper block I/O layer communicate operations down to lower layers in the usual manner
  - Block I/O operation codes

blk_lookup_zone
blkdev_report_zone

blkdev_reset_zone
blkdev_open_zone
blkdev_close_zone
blkdev_finish_zone

```
REQ_OP_ZONE_REPORT
REQ_OP_ZONE_RESET
REQ_OP_ZONE_OPEN
REQ_OP_ZONE_CLOSE
REQ_OP_ZONE_FINISH
```

Lower Layers: SCSI Disk Driver

- Modified to create a zoned block device for HA and HM drives
  - Initializes zone cache
    - Fills zone information for entire LBA range
  - Zone report is outside of critical I/O path
    - Single threaded work queue
    - Avoid deadlocks and simplify error processing
- Request order is not modified
  - Ensure single threaded HBA request submission from dispatch queue to maintain user submission order
    - Unaligned write or read errors can be tracked to HBA problems

2016 Storage Developer Conference. © 2016 Western Digital Corporation. All Rights Reserved.

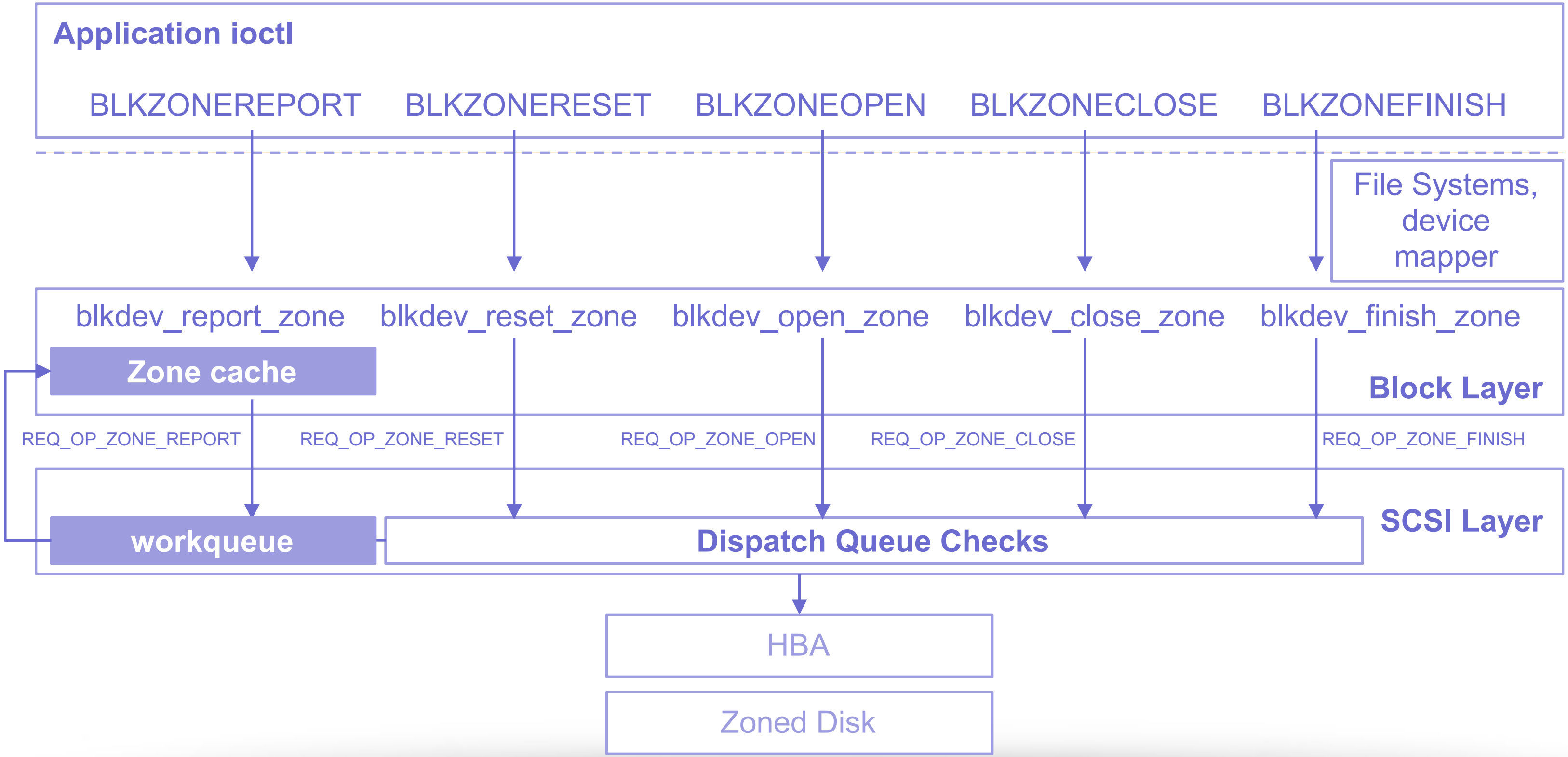
Lower Layers: Read & Write Processing

- All read and write requests in sequential zones are checked at dispatch time
  - Read after write pointer are not sent to the disk
    - Zero-out request buffer and return success
    - Avoids boot-time errors for HM disks (partition table read)
  - Write not at write pointer are failed without being sent to the disk
    - Write pointer position advanced in zone information cache for successfully checked write requests
- Request completing with error trigger a zone report execution
  - Update zone cache information with current disk state

Lower Layers: Zone Commands

- A minimal zone state machine is maintained with the zone cache
  - Zone condition: empty, open, closed, full
- Upper layer initiated zone operation requests trigger an update of the zone cache information at dispatch time
  - Before command completion
  - Consistent with command queueing and read/write checks
- Similarly to read & write errors, zone commands failure trigger a zone report
  - Except for zone report itself, for obvious reasons

Block I/O Stack Final Overview



File Systems

- Work to natively support zoned block devices in file systems also on-going
  - f2fs and btrfs
- Basic problem to solve is common to both candidates
  - Block allocation on write + block I/O issuing is not atomic
    - Sequential block allocation does not necessarily result in sequential writes
  - Some optimizations doing “update-in-place” must be disabled
    - Maintain sequential write pattern
  - Integration of zone reset on block reclaim

2016 Storage Developer Conference. © 2016 Western Digital Corporation. All Rights Reserved.

Device Mapper: dm-zoned

- Expose a zoned block device as a regular block device
  - Allows using any file system
- Uses conventional zones as “write buffer”
  - Aligned writes go straight to sequential zone
  - Random/unaligned writes are first staged to write buffer zones
    - Configurable number of buffer zones
    - Buffer zones must be reclaimed (rewritten to sequential zones)
  - Zone indirection table used to track write locations
    - Used for read processing

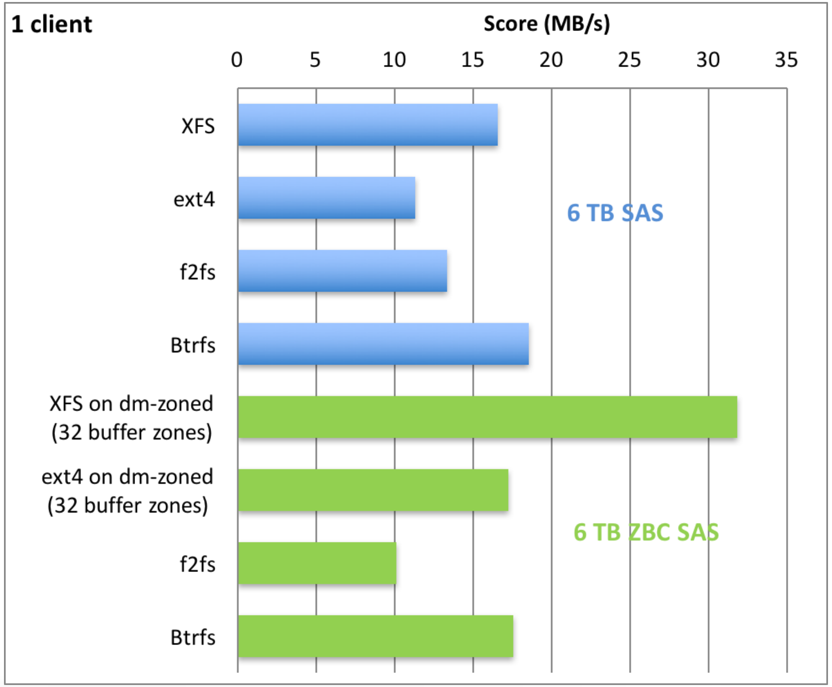
Performance Evaluation

- Patched 4.7 kernel base
- Focus on file systems
  - Native support file systems: f2fs, btrfs
  - Unmodified file systems + dm-zoned: ext4, XFS
- Comparison of ZBC enabled solutions with regular disk use
  - Same physical disk for all experiments
    - SAS 6 TB disk with regular firmware or “hacked” ZBC enabled firmware (256 MB zones with 1% of LBA space as conventional zones)
- dbench scores are used as a performance metric

2016 Storage Developer Conference. © 2016 Western Digital Corporation. All Rights Reserved.

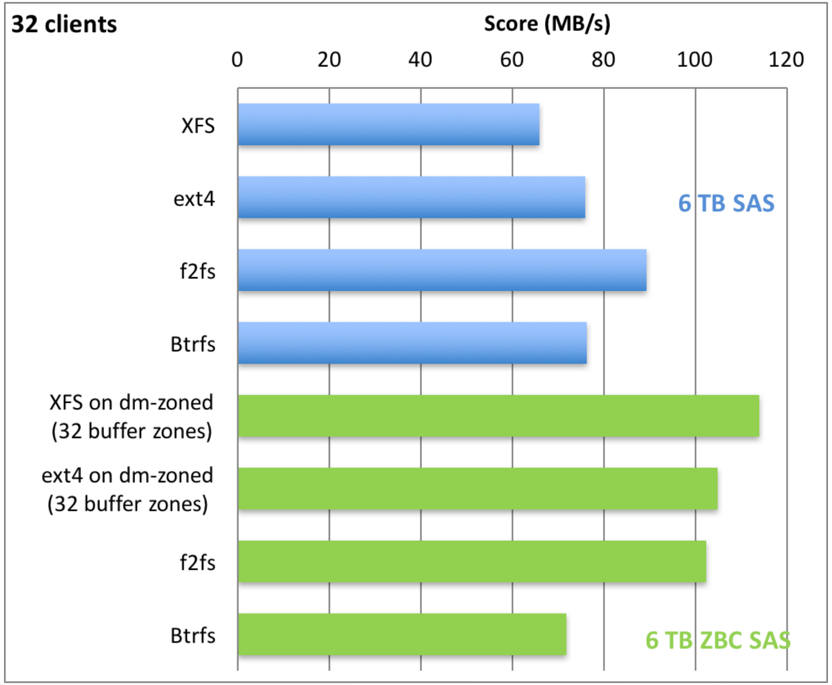
Dbench Results (1 client)

- Small score drop for native f2fs and btrfs
  - Loss of some optimizations leading to random writes
- dm-zoned cases show significantly higher scores
  - Short term benefits of pure sequential write pattern (reduced seek overhead)



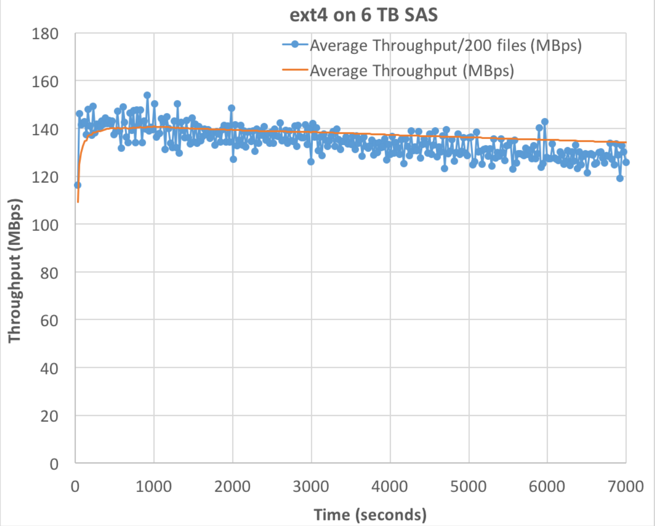
Dbench Results (32 clients)

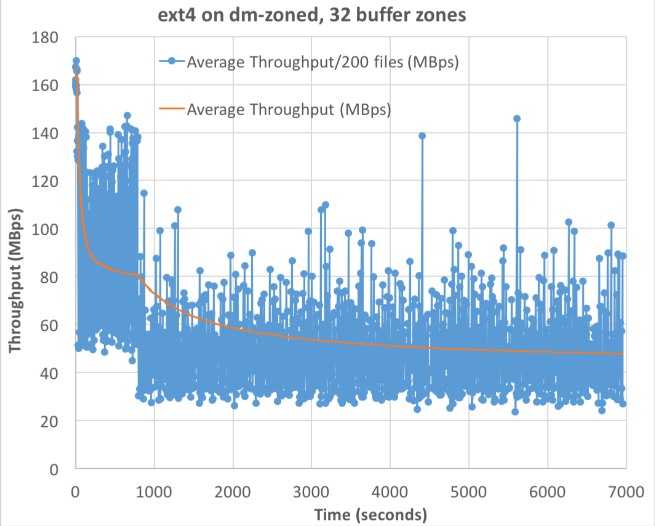
- Same small score drop observed for btrfs
  - f2fs improves
- dm-zoned cases advantage still present
  - Write pattern not changing with higher number of clients



dm-zoned High Duty-Cycle Performance

- Buffer zone reclaim has a cost under sustained write access
  - Incoming write operations must wait for buffer zones reclaim





Release Schedule

- Aiming for inclusion of block I/O stack changes into kernel 4.9
  - Stable release likely in December
  - May be delayed to 4.10 (February 2017)
    - 4.9 merge window rapidly approaching
- Following releases will likely see inclusion of support for file systems and ideally a device mapper
  - F2fs, btrfs, …
  - Dm-zoned, zdm, ...

Conclusion

- ZBC support plan is a compromise between simplicity and usability
  - Changes limited to the block I/O stack
    - Most within the SCSI disk driver
    - Critical areas such as the page cache are untouched
- Early work on file systems validated the overall architecture and API
  - Changes for native support mostly limited to ensuring sequential write submission
- Device mapper enables all that cannot easily be natively supported
  - Performance will depend on application

2016 Storage Developer Conference. © 2016 Western Digital Corporation. All Rights Reserved.

Thank you !

Questions ?