ZFS Improvements for HPC

Johann Lombardi

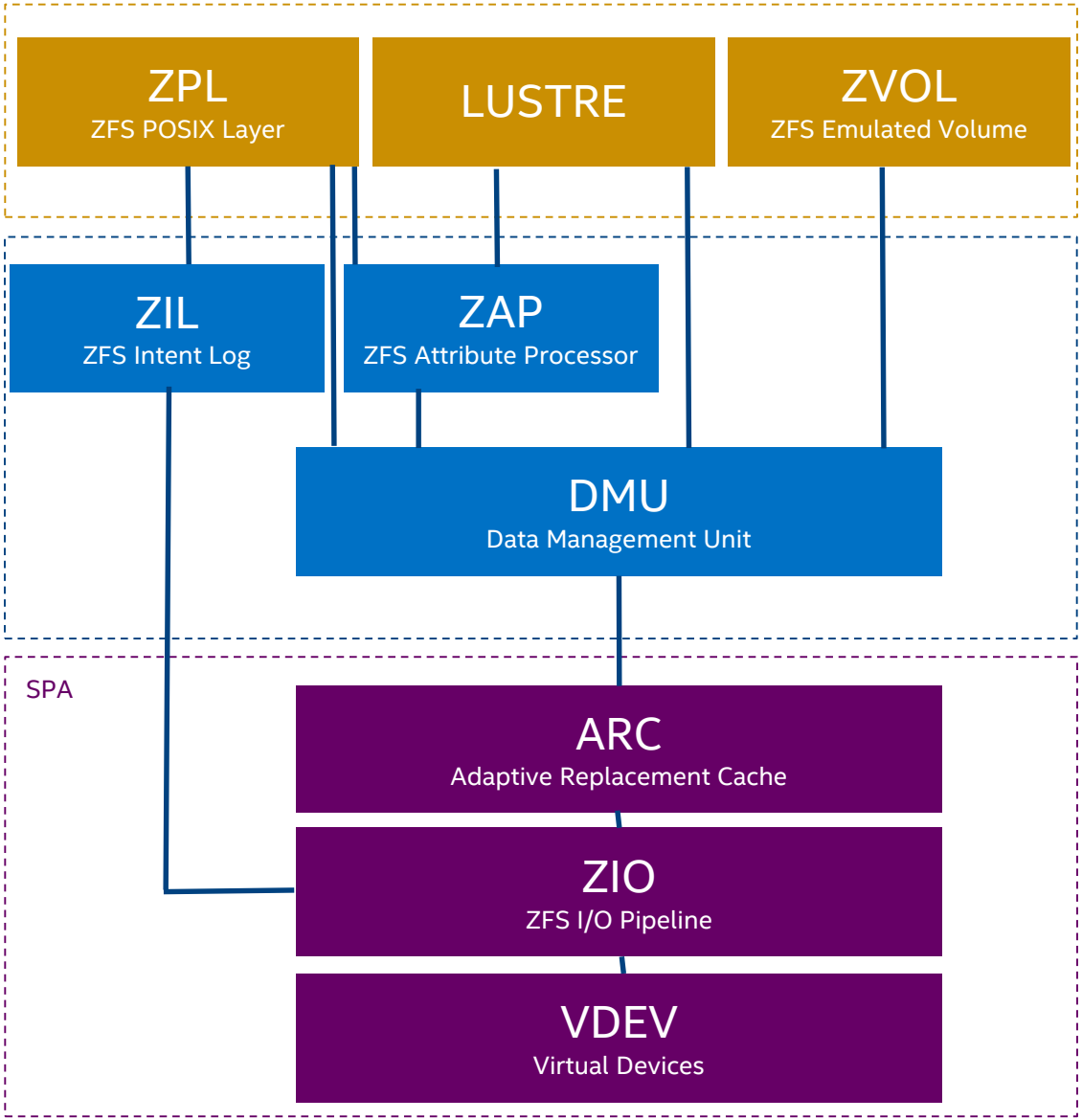
High Performance Data Division

Lustre*: ZFS Support

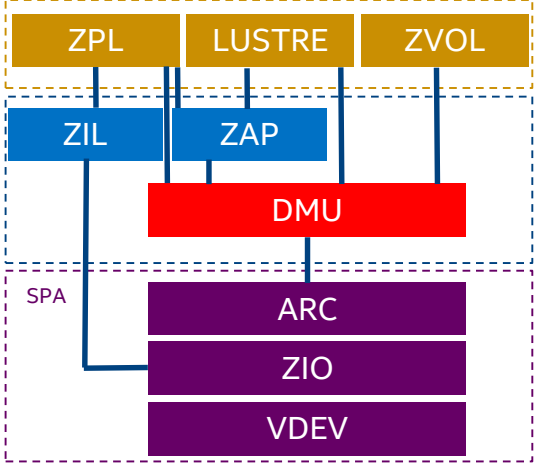
ZFS backend fully supported since 2.4.0

- Basic support for ZFS-based OST introduced in 2.3.0
- ORION project funded by LLNL
  - Network protocol independent of backing file system
  - New Object Storage Device (OSD) module integrated with ZFS Data Management Unit (DMU)
- lfsck support introduced in 2.6.0

ZFS I/O Stack


ZPL
ZFS POSIX Layer
LUSTRE
ZVOL
ZFS Emulated Volume
ZIL
ZFS Intent Log
ZAP
ZFS Attribute Processor
DMU
Data Management Unit
SPA
ARC
Adaptive Replacement Cache
ZIO
ZFS I/O Pipeline
VDEV
Virtual Devices

Data Management Unit



DMU – Data Management Unit

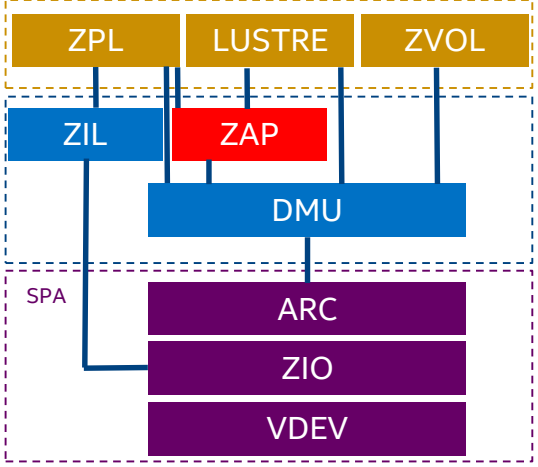
- General-purpose transactional object store
  - dnode defines objects
  - object set is a collection of objects
  - transaction is a series of operations that must be committed to disk as a group, all or nothing.
- Built atop flat address space (SPA)

Lustre* & DMU

All objects of a Lustre target are stored in a single DMU objset

- Each Lustre FID is assigned a dnode in the objset
- On-disk format compatible with ZFS Posix Layer (ZPL)
  - Lustre targets can be mounted as ZFS file systems on Linux
- User/group dnode accounting not supported by ZFS
  - Currently done in zfs-osd (improvements in LU-2600)
  - Patch provided to ZFS community to handle user/group dnode quota in ZFS (LU-2435)
- System Attributes
  - Lustre relies heavily on extended attributes (xattrs)
  - store Lustre xattrs with dnode for better performance

ZFS Attribute Processor



ZAP – ZFS Attribute Processor

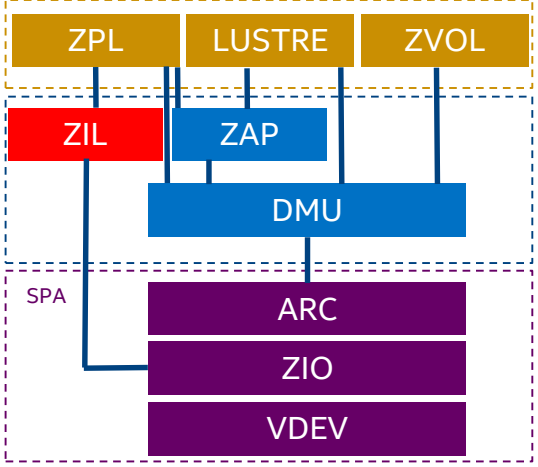
- Hash table storing {key, value} associations within an object
- microZAP: single block, simple attributes (64-bit number) and limited key length (50 bytes)
- fatZAP: large number of entries and long keys/values
- Used by ZFS Posix Layer (ZPL) for:
  - Directories
  - ZFS attributes
  - User/Group space accounting & quota

Lustre* & ZAP

ZAP used for all Lustre index files

- Lustre directories
- Quota index files
- Object index (OI)
  - Lustre FID / dnode association
  - Fast OI insert/lookup/delete is required for good metadata performance
- Hash function to distribute values uniformly across buckets
- Increase indirect and leaf block size from 4K to 16K
  - Improved create/destroy rate with mds-survey by ~10% (LU-5391)

ZFS Intent Log



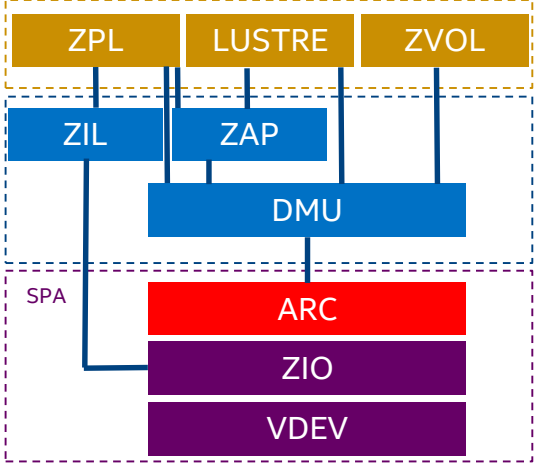
ZIL – ZFS Intent Log

- Copy-on-write is efficient for long running transactions, but not for short transactions
  - Metadata overhead too high
- ZIL is a per-dataset transaction log for synchronous data
  - Synchronous data quickly flushed to this special log w/o expensive transaction group commit
  - ZIL log replayed only in case of crash
  - No data copy for writes larger than 64KB
  - ZIL can be stored on a separate device

Lustre* & ZIL

- Lustre does not take advantage of the ZIL yet
  - Synchronous I/O is a performance killer
    - O_DIRECT, O_DSYNC, fsync(3C)
  - Some Lustre internal mechanisms might also trigger fsync()
    - e.g. Commit On Share (COS), sync_on_lock_cancel
- ZIL replay after crash could break Lustre transno ordering
- Work to support ZIL in ZFS-OSD under discussion in LU-4009

Adaptive Replacement Cache



ARC – Adaptive Replacement Cache

- Central point of memory management caching data from all active storage pools
- Replacement for the Linux page cache
- Self-tuning cache adjusted based on I/O workload
- Active OI ZAP blocks should be cached as much as possible to avoid repeated I/Os
  - LU cache interaction (LU-5164)
  - FID prefetching (LU-5041)

Increase Block Size to 1MB+

- Supported maximum block size of 128KB
- Impact traditional HPC workload of large I/Os
  - Random 1MB reads from rotating disks has higher bandwidth than random 128KB reads
  - read-modify-write required when running over a RAID array with a stripe width larger than 128KB
- Resilvering/scrubbing would benefit from a larger block size
- Draft patch to support 1MB block size available for testing
  - https://github.com/zfsonlinux/zfs/issues/354

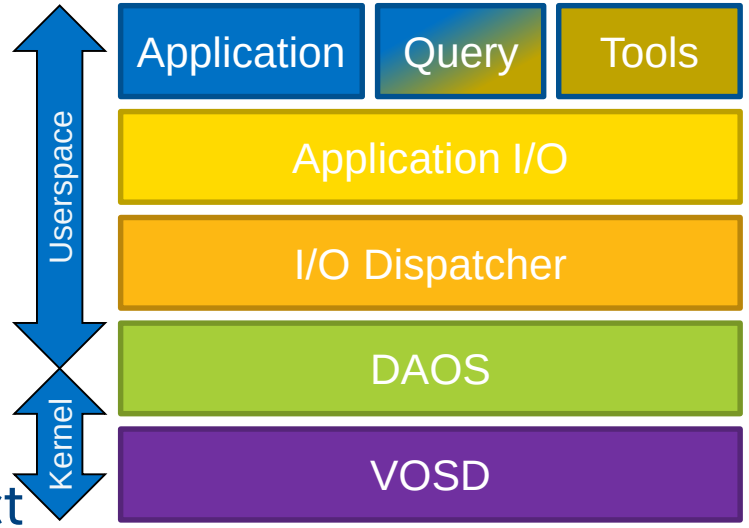
DKMS Packaging

- ZFSonLinux (ZoL) comes with DKMS packages
  - Recompile ZFS modules automatically against current kernel and further upgrade
- Lustre* now also supports DKMS
  - Lustre server modules automatically rebuilt when ZFS modules or kernel are upgraded
  - LU-1032

Other Future Enhancements

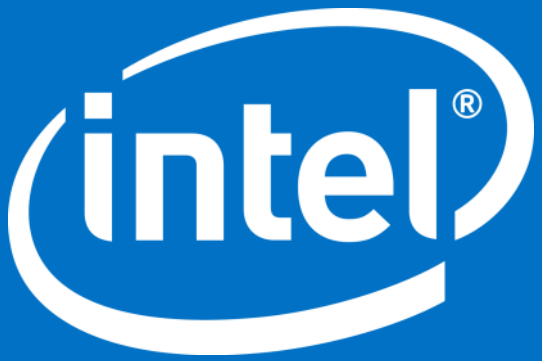
- Multiple Mount Protection (MMP)
- Large dnode support
- Persistent L2ARC across reboot
- RAIDZ/RAIDZ2 improvements

Versioning OSD



VOSD – Versioning OSD

- Developed for the Fast Forward project
- Lowest layer of the exascale storage stack
- Extensions to the ZFS OSD
  - Multiple active datasets to be accessible through an OST
  - Use ZFS snapshot capability to freeze a dataset “version”
  - Implement Version Intent Logging (VIL) to log I/O operations for future versions
    - Versions are integrations and snapshotted in order
    - VIL was developed as a ZIL extension
    - VIL patch available here: http://git.whamcloud.com/ff/daos_lustre.git/blob/HEAD:/contrib/patches/daos_zfs/zfs-0.6.1_daos.patch


intel®