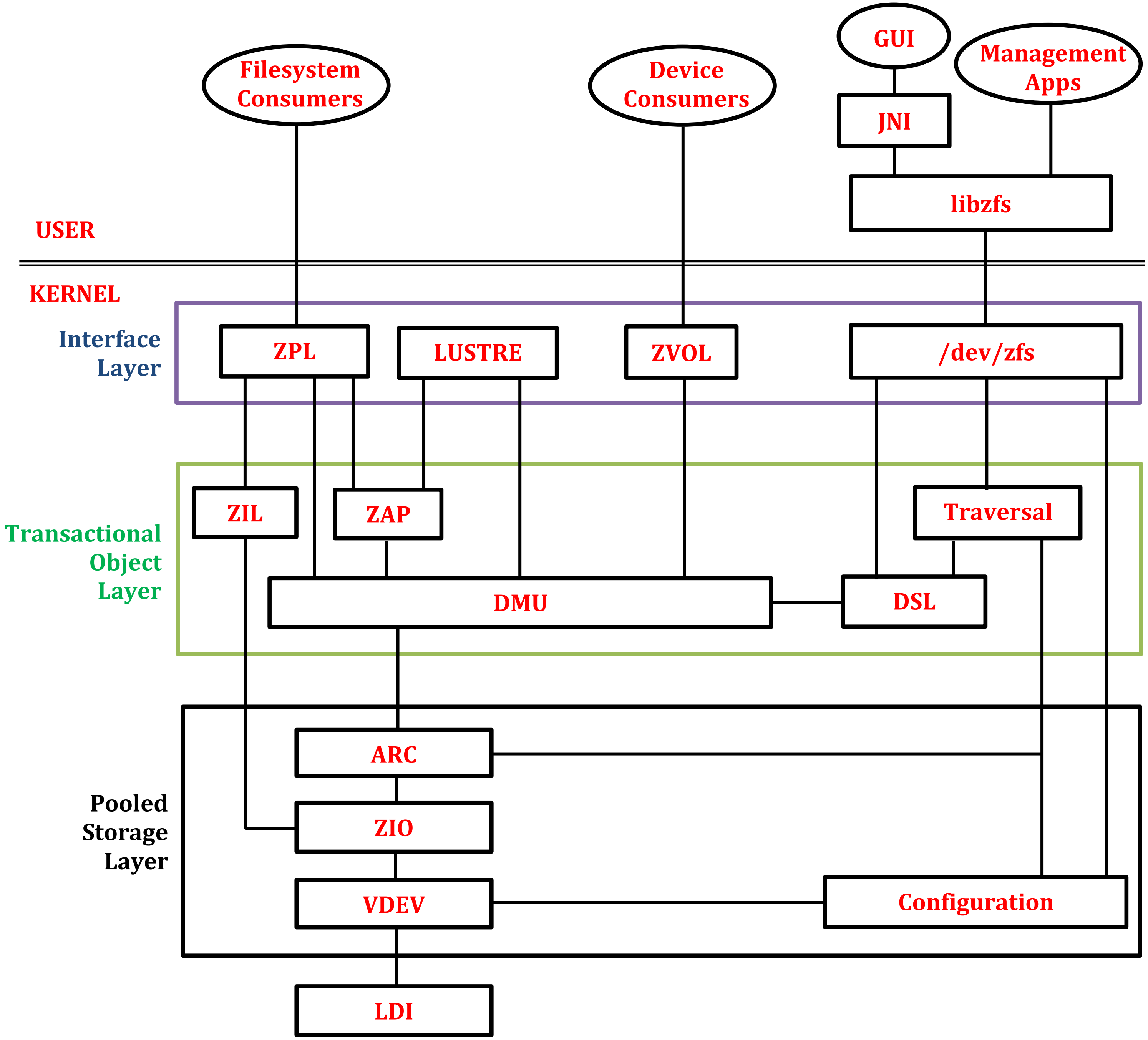
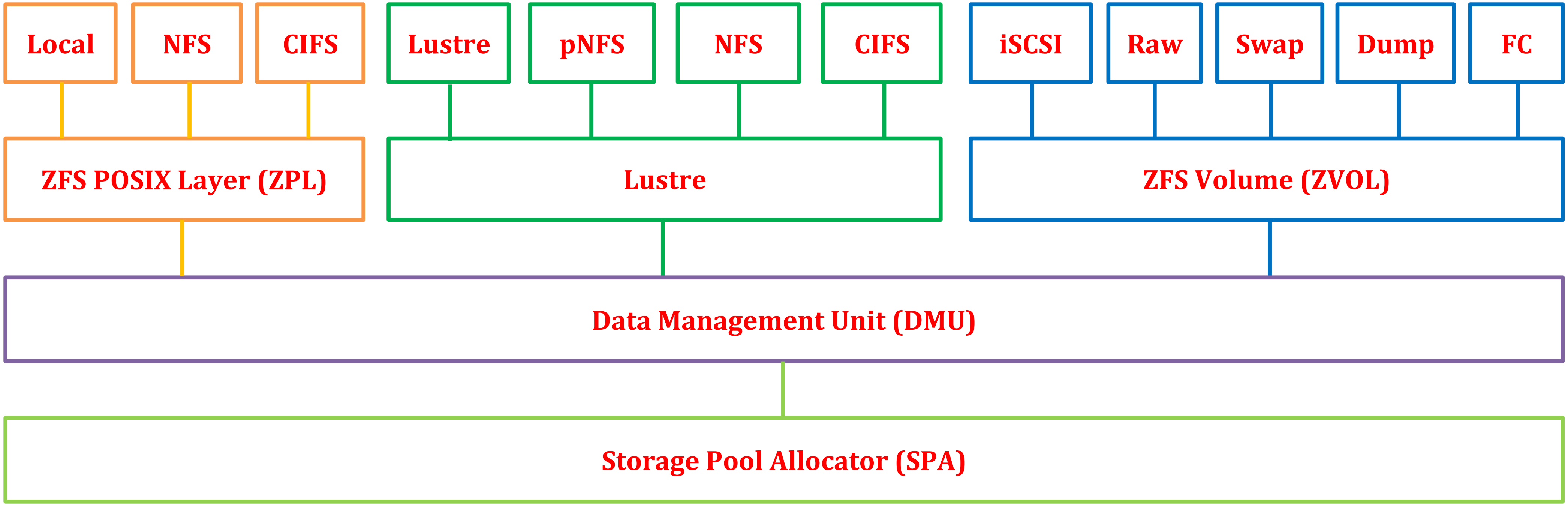
DSL	– Dataset and Snapshot Layer
DMU	– Data Management Unit
VDEV	– ZFS Virtual Device
ZAP	– ZFS Attribute Processor
ZIL	– ZFS Intent Log
ZIO	– ZFS I/O Pipeline
ZPL	– ZFS POSIX Layer
ZVOL	– ZFS Volume
LDI	– Layered Driver Interface
ARC	– Adaptive Replacement Cache
JNI	– Java Native Interface
libzfs	– Primary interface for management apps to interact with the ZFS kernel module
/dev/zfs	– Primary point of control for **libzfs**
LUSTRE	- Lustre Object Storage Device (OSD) Level.
Traversal	– Traversal provides a safe, efficient, restartable method of walking all data with in a live pool.
Configuration	– Public interface to configuring **zfs**



Universal storage: File, block, object, network with Lustre.



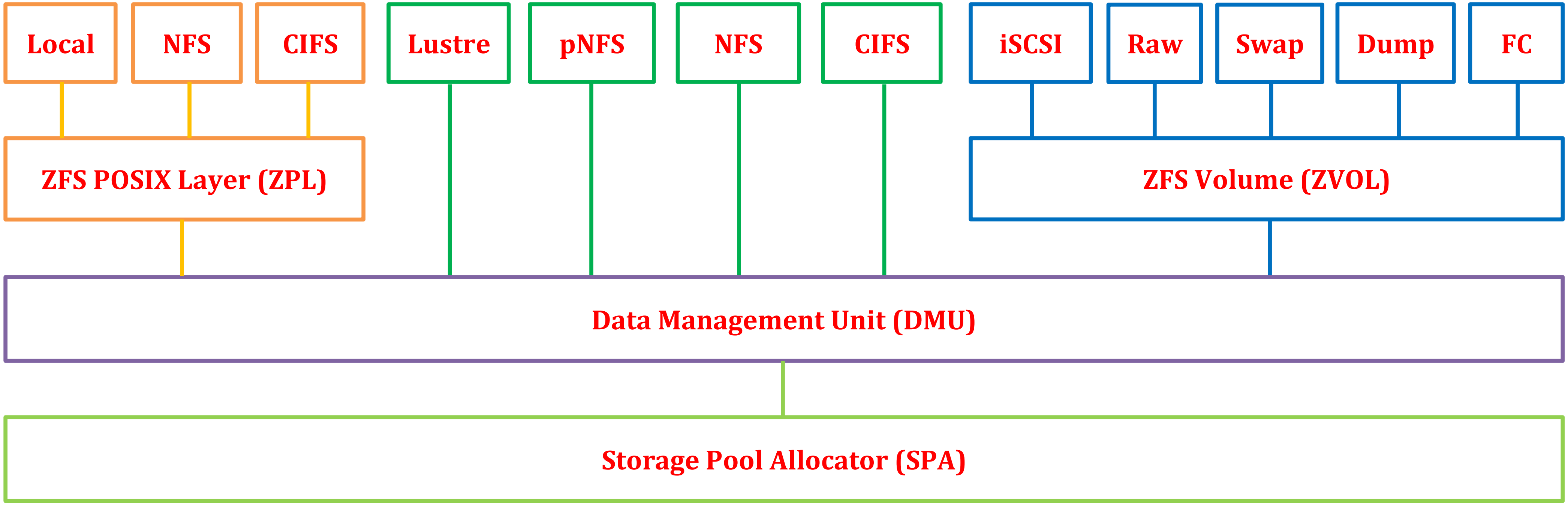
Local NFS CIFS Lustre pNFS NFS CIFS iSCSI Raw Swap Dump FC

ZFS POSIX Layer (ZPL) Lustre ZFS Volume (ZVOL)

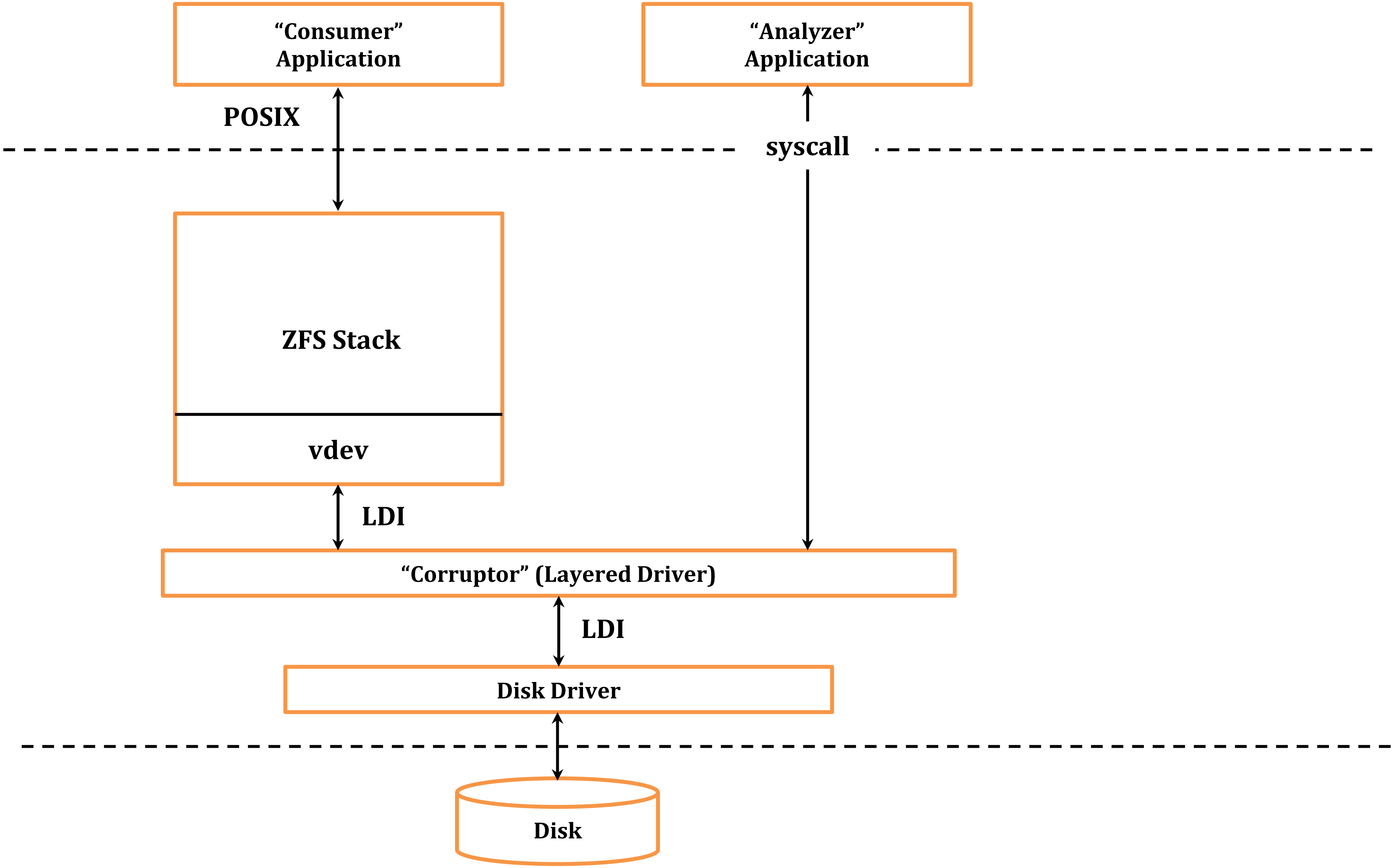
Data Management Unit (DMU)

Storage Pool Allocator (SPA)

Universal storage: File, block, object, network without Lustre.



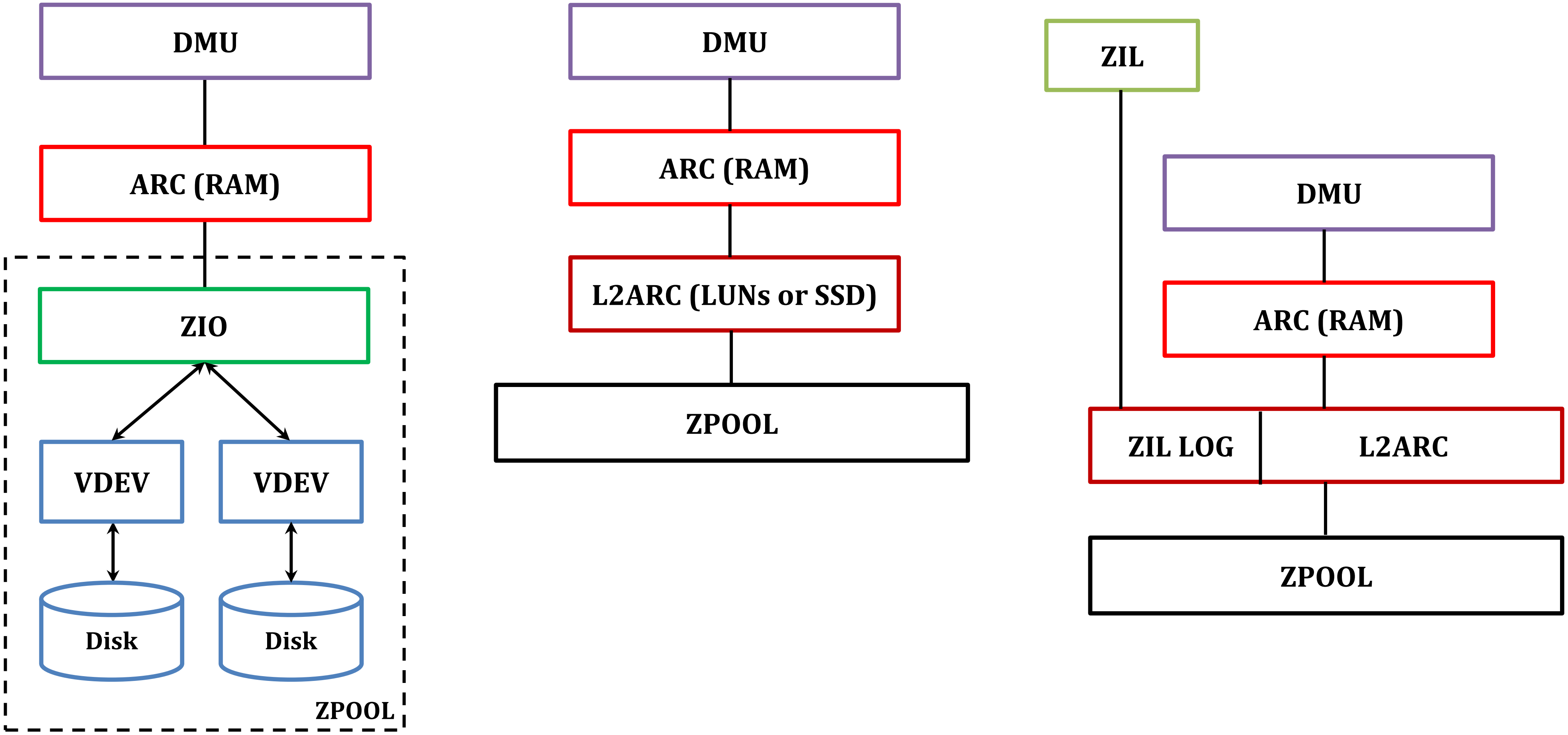
Corruption analysis framework.



ZFS – ZPOOL Cache ARC and L2ARC.

ZFS ARC: ZFS Adjustable Replacement Cache (ARC), based on ARC cache algorithm. ZFS ARC can use up to 88 percent of available physical memory, and adjust the memory usage according to the kernel needs.

ZFS L2ARC: ZFS L2ARC is level two adjustable replacement caches, and provide an additional layer of caching between main memory and disk. Usually for L2ARC-cache can to be use the fastest devices on LUNs or SSD. It increases the great performance of random-read workloads of static content.



Compression

Internally, ZFS allocates data using multiples of the device's sector size, typically either 512 bytes or 4KB (see above). When compression is enabled, a smaller number of sectors can be allocated for each block. The uncompressed block size is set by the recordsize (defaults to 128KB) or volblocksize (defaults to 8KB) property (for filesystems vs volumes).

The following compression algorithms are available:

- LZ4 - New algorithm added after feature flags were created. It is significantly superior to LZJB in all metrics tested. It is new default compression algorithm (compression=on) in OpenZFS, but not all platforms have adopted the commit changing it yet.
- LZJB - Original default compression algorithm (compression=on) for ZFS. It was created to satisfy the desire for a compression algorithm suitable for use in filesystems. Specifically, that it provides fair compression, has a high compression speed, has a high decompression speed and detects incompressible data detection quickly.
- GZIP (1 through 9) - Classic Lempel-Ziv implementation. It provides high compression, but it often makes IO CPU-bound.
- ZLE (Zero Length Encoding) - A very simple algorithm that only compresses zeroes.

If you want to use compression and are uncertain which to use, use LZ4. It averages a 2.1:1 compression ratio while gzip-1 averages 2.7:1, but gzip is much slower. Both figures are obtained from testing by the LZ4 project on the Silesia corpus. The greater compression ratio of gzip is usually only worthwhile for rarely accessed data.