NexentaStor: An Introduction to ZFS's Hybrid Storage Pool

A common whiteboard session captured on paper

Chris Nelson, Sales Engineer, Nexenta Systems

June 2012

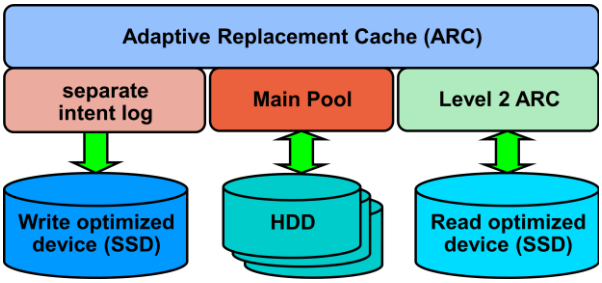
NexentaStor: An Introduction to ZFS's Hybrid Storage Pool

Introduction

For more than a decade, storage system performance has remained rather stagnant while drive capacities and application performance demands steadily have increased. The result of this trend is an expensive problem: Storage users are forced into buying expensive hard disk drives (HDDs) to get a moderate performance boost (by reducing I/O latency) and/or forced into over-buying capacity in order to meet performance requirements. With the advent and decreasing price of flash, storage vendors are integrating it into their products to solve this problem. Nexenta's ZFS technology is leading the industry in its ability to automatically and intelligently use flash in a storage system that offers the appropriate capacity and performance capabilities at a total cost that is dramatically lower than most legacy storage systems (and even still lower than legacy vendors who have begun to use flash as well).

The Parts of a ZFS Hybrid Storage Pool

In a ZFS Hybrid Storage Pool (HSP), there typically are three varieties of hardware—DRAM, flash, and spinning HDDs, with flash being used in two distinct ways. The following sections explore each of these.



Adjustable Replacement Cache (ARC): The ARC (sometimes known as the "Adaptive" Replacement Cache) lives in DRAM. It is the first destination for all data written to a ZFS pool, and it is the fastest (i.e., lowest-latency) source for data read from a ZFS pool. When data is requested from ZFS, it looks first to the ARC; if it is there, it can be retrieved extremely fast (typically in nanoseconds) and provided back to the application. The contents of the ARC are balanced between the most recently used (MRU) and most frequently used (MFU) data.

Level-Two ARC (L2ARC): The L2ARC lives in flash. In concept, it is an extension of the ARC. Without an L2ARC, data that could not fit in the ARC would have to be retrieved from HDDs when requested. That is when drive speed makes a difference, but the performance difference between "fast" (e.g., 15k-RPM) and "slow" (e.g., 7,200-RPM) is in terms of latencies measured as a few milliseconds or several milliseconds; both are *dramatically* slower than ARC accesses measured in nanoseconds. L2ARC, in flash, fits nicely between the two—both in terms of price and performance. Buying hundreds of gigabytes of flash is cheaper than the same capacity of DRAM (though still more expensive today than HDDs), and flash's I/O latencies typically are measured in microseconds—slower than DRAM but still *far faster* than even "high-performance" HDDs. The L2ARC is populated by data first placed in the ARC as it becomes apparent that the data might get squeezed out of the ARC, and not every piece of data that existed in ARC will make it to the L2ARC (those that do not would be retrieved from HDDs instead, if requested); the algorithms that manage L2ARC population are automatic and intelligent (and tuned by Nexenta when appropriate).

ZFS Intent Log (ZIL) and Separate Intent Log (SLOG): The ZIL is used to handle "synchronous" writes—write operations that are required by protocol (e.g., NFS, SMB/CIFS) to be stored in a non-volatile location on the storage device before they can be acknowledged to the application. Application threads typically stop and wait for

synchronous write operations to complete, so reducing the latency of synchronous writes has a direct impact on application performance. ZFS can do this by using a separate intent log (SLOG) for the ZIL—typically on a flash device. All writes (whether synchronous or asynchronous) are written into the ARC in DRAM, and synchronous writes also are written to the ZIL before being acknowledged. Under normal conditions, ZFS regularly bundles up all of the recent writes in the ARC and flushes them to the spinning drives—at which point the data in the ZIL is no longer relevant (because it now exists on its long-term, non-volatile destination) and can be replaced by new writes. The ZIL only is read from when synchronous writes in the ARC are unable to be written to spinning disk—like after a power failure or controller failover—at which point ZFS reads the ZIL and places that data onto the spinning drives as intended. One might compare this concept to non-volatile RAM (NVRAM) from storage vendors like NetApp, but where NVRAM uses batteries that can wear out and have other issues, write-optimized SLC flash devices do not need batteries. And while NVRAM scalability is limited to available slots, adding SLOGs is as easy as adding HDDs. (There is a major price difference too!) Like L2ARC, the ZIL/SLOG is managed automatically and intelligently by ZFS: Writes that need it are accelerated without any additional effort by the administrator.

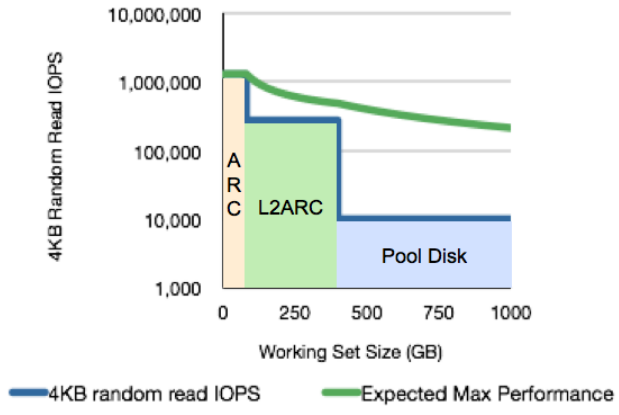
Note: Fibre Channel (FC) and iSCSI applications rarely understand the concept of a volatile write cache, so it is appropriate to treat every FC and iSCSI write as a synchronous operation. Thus, SLOG devices are especially important to FC and iSCSI deployments on ZFS.

Hard Disk Drives (HDD): With the ARC, L2ARC, and ZIL/SLOG providing the bulk of the performance from a ZFS Hybrid Storage Pool, spinning drives are relegated to the job they do well—providing lower-performance, higher-density, low-cost storage capacity. Until the day that flash competes with HDDs on a dollar-per-gigabyte front, the right balance of DRAM and flash for performance, and HDDs for capacity, results in a total cost of ownership (TCO) that is less—both initially and over the long-term—than solving both requirements using all flash or all HDDs.

Note: While it is no longer the primary purpose of HDDs in ZFS to provide performance, RAID layout and drive speed still can impact overall performance—sometimes significantly. This will be discussed more fully in the follow-on paper describing how to size a ZFS storage system.

A New Storage Parameter: Working Set Size

For legacy storage systems, sizing means determining necessary capacity, IOPS, and throughput and then doing some simple math to determine the number of spindles that could provide those numbers (with some thought given to parity overhead, controller limitations, etc.). As the industry moves toward more sophisticated caching methodologies in storage systems, such as NexentaStor, a new parameter for expressing storage needs has become evident: The “Working Set Size” (WSS) can be described as the subset of total data that is actively worked upon (e.g., 500GB of this quarter’s sales data out of a total database of 20TB). Knowing the WSS makes it possible to size ARC, L2ARC, and even HDDs more accurately, but few applications today have an awareness of WSS. Nexenta is working with its partners in the industry to develop tools to make this measurement easier and to provide guidance based on its experience with thousands of deployments.



A Few Frequently Asked Questions

Doesn't flash wear out quickly when used for writes?

Yes, flash does wear out, but today's enterprise flash vendors have done a lot to improve wear-leveling algorithms, reserve spare flash cells, and otherwise extend the longevity of flash in enterprise devices. So, don't buy a USB thumb drive for your ZIL; know that not all SSDs are created equal; consider Nexenta's Hardware Supported List (HSL); or contact a sales representative or sales engineer for specific guidance.

What happens if a flash drive fails?

First of all, data isn't lost; losing a ZIL/SLOG or L2ARC device is simply an impact to performance. Any data in the L2ARC also is in the spinning HDDs, so losing an L2ARC device simply means that any requests for the data that were on the failed flash drive now must be serviced by reads from the spinning HDDs. If multiple flash drives are used for L2ARC, they are used in a striped fashion, and the remaining devices will continue to be used if one fails. Any data in the ZIL/SLOG also is in the ARC until it is flushed to the spinning HDDs, which NexentaStor does every five seconds by default. Thus, only if the SLOG device fails *and* controller power is lost within five seconds would data be lost. This is probabilistically extremely rare, but many customers prefer the comfort of mirrored SLOG devices nonetheless, so the options to stripe or mirror data across SLOG devices both exist.

Can I put L2ARC in the head nodes in HA Clusters?

Yes, but there is a tradeoff to consider. Most NexentaStor HA Cluster deployments consist of two server "head nodes" (containing CPUs, DRAM for ARC, and PCIe cards for client-side Ethernet/Fibre connectivity and back-end storage connectivity) and JBODs with shared storage that can be physically accessed by either head node if one fails. Placing the L2ARC devices in the JBODs means that they can be used by either head node, regardless of the state of the cluster. But, if squeezed for space, it might make sense to utilize drive slots in the head nodes for L2ARC and accept the temporary performance impact (of some SSDs being unusable if a head node is down) until the cluster is back to its optimal condition.

Can I use PCIe-based flash devices?

Yes, but the tradeoff here is much like that described in the preceding answer. PCIe-based flash devices obviously must reside in the head nodes, so if a head fails or is powered off, the other node cannot utilize that flash. (There is no cross-node cache mirroring in ZFS today.) While that is the downside, the upside is that PCIe-based flash devices offer lower I/O latencies than SSD flash devices so, where performance is most critical, some customers prefer to have PCIe-based flash devices in the head nodes and then plan to restore the HA Clusters to the optimal state as quickly as possible if something should happen.

What about an all-SSD pool?

Many Nexenta customers use all-SSD pools. Where the budget allows, this offers a very low-latency solution and negates the benefit of the L2ARC and SLOG (unless SSDs with differing performance characteristics are used).

Read Next: *An Introduction to Sizing a ZFS Storage Solution*