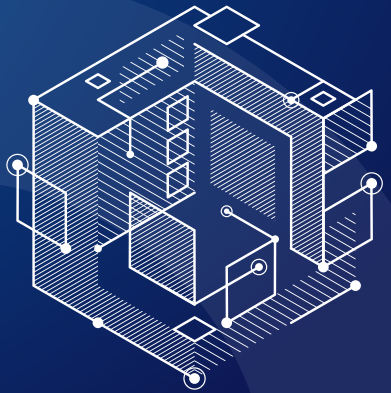
# CACHING

TECHNICHAL CAPABILITY SHEET

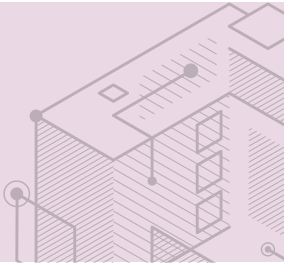


open DRIVES

## Capability Definition

Caching is the process of leveraging both RAM memory (which is incredibly fast) and NVMe storage (also fast, but not like RAM) to write user requested data. Providing data to end users and applications from cache, rather than accessing storage media like HDDs, is many times faster. Atlas leverages ZFS' file and volume manager capabilities to oversee the different layers of cache memory, determine what needs to be written to each layer, and which data should remain (or be "pinned"). Atlast runs smart logic to determine what type of data from disk needs to be cached, in which layer of cache the information should be stored, and how long it should remain in cache based on the capacity of cache space at any given time.

**Note:** Caching is not the same for every OpenDrives storage system. The Ultimate series, for example, already uses NVMe as general data storage media, and so it only uses flash memory for its cache. Therefore, be cognizant of the fact that caching will behave slightly differently — and with different results — depending on the series and configuration of the OpenDrives storage system in question.



## Outcome & Value

Atlas' caching (also referred to as 'inline caching' or 'intelligent caching') enhances performance and responsiveness by retrieving data more quickly from faster cache layers. At all times, Atlas is also using intelligent algorithms to analyze usage patterns and anticipate data requests before they are made. Working in tandem, these capabilities significantly speed up workflows and data requests, as well as drive a much better experience for end users. Atlas minimizes or eliminates the sluggishness users feel accessing data. Faster performance and quicker workflow execution equates to higher user adoption – a direct impact to your organization's productivity and bottom line.

## How It Works

Atlas creates layers of cache, potentially several contingent on the type of memory and storage media available in the system. All OpenDrives storage systems have RAM memory that acts as the top layer of read cache, known as Adaptive Replacement Cache (ARC). Atlas takes advantage of as much RAM capacity as is available to maximize requested data in the ARC, more memory equaling higher performance gains across more users. Keep in mind that RAM memory provides the best performance (on the order of nanoseconds), so reads from ARC are incredibly fast and contribute most to smooth execution of data reads. Of course, Atlas uses RAM memory for other processes as well, and RAM is a finite, valuable resource. Therefore, Atlas enacts intelligent assessment of the data which is placed into ARC and how long it remains there. To illustrate, data that is being accessed by multiple team members or at a greater frequency will stay pinned into cache, as opposed to "cooler" data that will be removed to make space for new requests.

NVMe storage provides the next layer of read cache, known as L2ARC. While not quite as fast as RAM, NVMe storage operates in the microsecond range, which is still significantly faster than spinning disks. L2ARC serves as a type of overflow for the top level RAM-based ARC cache. However, because of intelligent logic built into Atlas, it can also prefetch a large amount of information from the storage pools, depending on the amount of space allocated to the L2ARC. Using intelligent algorithms, Atlas uses usage patterns to determine data it can prefetch, before being explicitly requested, and stage in L2ARC, anticipating data the user will require next in order to make it available from a faster layer instead of waiting to read it from slower storage.

To make this determination, Atlas must distinguish between different types of data because all data simply isn't equal in terms of immediate usability, by the user or application, and a subsequent uplift in performance. Data that is Most Frequently Used (MFU) has a much higher probability of being requested than data that has been Most Recently Used (MRU). Therefore, Atlas gives a higher priority to MFU data than to MRU data. Likewise, data that's all been written at approximately the same time has a higher probability of being requested if a portion has already been read.

The treatment of these different types of data and built-in intelligence to fetch and optimally place data into ARC or L2ARC comprise the real value of Atlas' caching capability, but there's more. Filling these two layers of cache is just half of the process; determining which data then needs to be evicted to make way for higher-priority information efficiently is equally critical. Atlas performs eviction determination even before the actual data needs to be deprecated (from ARC to L2ARC) or removed from cache entirely. Data deprecated from ARC may still exist in L2ARC until the intelligent logic determines that higher-priority MFU or MRU data needs to replace it. All of this is completely automatic and transparent to the user. The user simply feels the effect through better performance, system responsiveness, and the elimination of lag on data request.

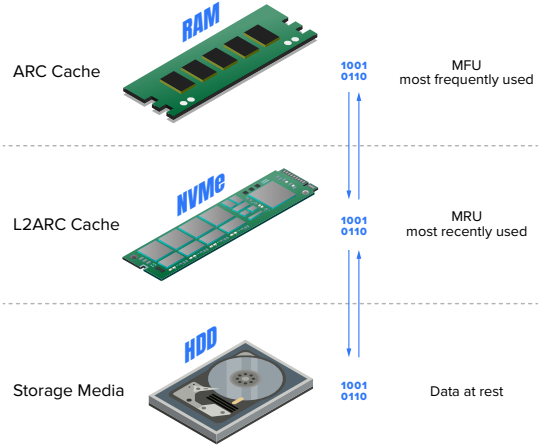
**Note:** Prefetch is a type of data caching and retrieval based on the same concept of staging information in cache before it is requested by a user or application. Intelligent caching is the mechanism of managing data added to, held in, and evicted from cache layers. In contrast, prefetching is based on specific data requests — especially in linear streams — and caching, then reading additional data prior to the actual request.

## Characteristics

The caching capability has the following characteristics:

- Multiple layers of cache (ARC, L2ARC) depending on the system configuration. Similarly, Atlas systems with L2ARC have more ways to tune caching operations.
- Intelligent operation that distinguishes between different types of data (MFU, MRU) and manages the location, deprecation, and ultimate eviction of data
- Persistent caching means that Atlas has a much better understanding of MFU and MRU data — upon reboot or failure, it accesses transaction logs to try to restore as much in cache as possible
- L2ARC provides the prefetching capability, and if considerable free space is available the system will use intelligent logic to prefetch based on current data use or MFU/MRU determinations
- Media with high frame rates really benefits from our caching and prefetching capabilities
- Cache often maintains multiple copies of data, both in ARC and L2ARC, until delisting/deprecation from ARC occurs. All of these operations are maintained by Atlas via a master cache file.
- To see caching at work, play some media for a given duration and then stop. You can see in the Atlas Dashboard the bandwidth continues to increase, which indicates a freshly booted system attempting to continue to cache MFU data.

**The following diagram illustrates the caching layers and deprecation/eviction processes. The diagram displays both ARC and L2ARC cache, the storage media, and movement of data between these layers.**




### Further Reading

**As previously stated, caching drives the prefetch operation. For more information about prefetching, please refer to the Prefetch Capability Sheet which explains in more detail the mechanics behind ZFS' prefetch capabilities which we've implemented within Atlas Core.**



sales@opendrives.com | +1 888-778-5491