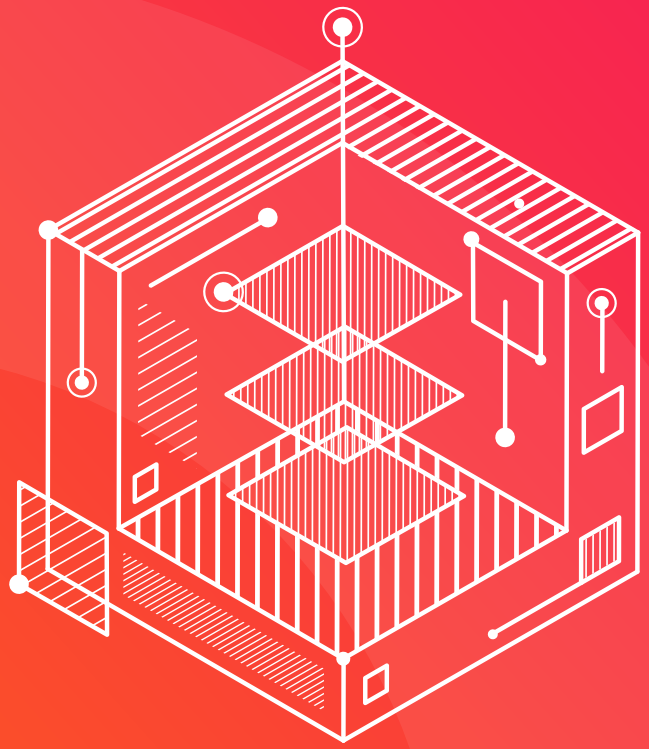
Checksum

TECHNICAL CAPABILITY SHEET

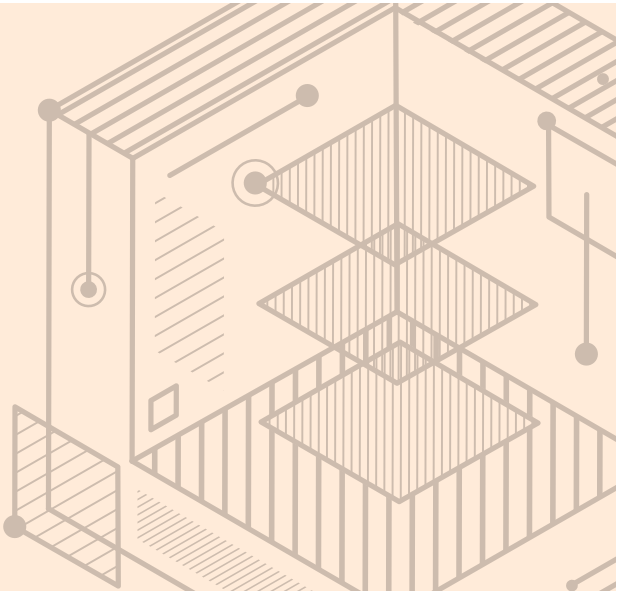


open DRIVES

Capability Definition

Atlas Core executes a checksum capability to ensure data integrity. The checksum is an algorithmically-derived small-sized block of data (also called a hash) calculated from a larger block of data. By comparing checksums to see if they precisely match, Atlas Core can determine whether data has been corrupted—either through a data transfer error or bit-rot degradation of the data on disk. Once Atlas Core detects a data integrity error through checksum comparison when reading data from disk, it then initiates a recovery process to regain the uncorrupted data.

Note: Many different algorithms are available within Atlas Core for calculating checksums. The default setting for Atlas Core is the Fletcher 4 algorithm, which is a lightweight approach to error detection without being resource-intensive to the point that it would affect storage performance.



Outcome & Value

The outcome of the checksum capability is a high level of data integrity. Though highly unlikely, data may be corrupted during transmission through the I/O pipeline, but it can also succumb to bit rot over time depending on operating conditions. Bit rot is a result of some sort of deterioration of the information stored on physical media such as HDDs, usually occurring over time as opposed to a catastrophic failure of the entire disk. By fully checksumming all data that enters our file system's I/O pipeline (ZIO pipeline) when written to or read from disk, Atlas Core keeps track of the level of data integrity and automatically initiates remediation efforts when it detects data corruptions through checksum mismatches.

How It Works

Atlas Core uses the Fletcher algorithm to compute checksums in order to verify the integrity of data written and stored on disk, read from disk, and/or transmitted across a network through an I/O pipeline. While other checksum algorithms are available as custom settings, the Fletcher algorithm is the default and recommended method, the reason being that the Fletcher method of computing checksums provides greatly improved performance especially with general-purpose processors.

When data enters the ZIO pipeline whether from SMB, NFS, or even the internal operating system, Atlas Core immediately performs a checksum operation to calculate a hash value. The same checksum process occurs when writing data to disk—with each transaction group being stored on disk, Atlas Core calculates a checksum for the transactional group to accompany the transaction group ID and the actual data itself. Upon reading data from disk, Atlas Core again derives a checksum and compares it to the checksum value previously associated with the data. An integrity error can be assumed if the checksum values do not perfectly match.

OpenDrives uses the phrase fully checksummed to indicate instances when Atlas Core calculates a checksum for other pieces of information already checksummed, indicating multiple recursive layers of preserving data integrity. Basically, everything associated with data that's written to storage media, including checksums and parity bits, has its own checksum value calculated within a tree-like structure. In essence, fully checksumming indicates that a secondary checksummed roll up of all checksums across all data takes place to enhance data integrity.

Checksums are spread across volumes, much like the data itself, preventing the data and checksums for that data from residing on the same physical disk. This drives lower latencies due to the data and checksums transacting in parallel from different disks rather than serially from a single disk.

Automated data correction occurs on read operations only. Atlas Core reconstitutes data to match the calculated data checksum either from the parity bits (of which two exist for each written transaction) or from a recursive process if the parity option is not available or functional. For the latter, we perform an automated bi-weekly scrub to verify and then roll up all checksums from disk to file system.

Characteristics

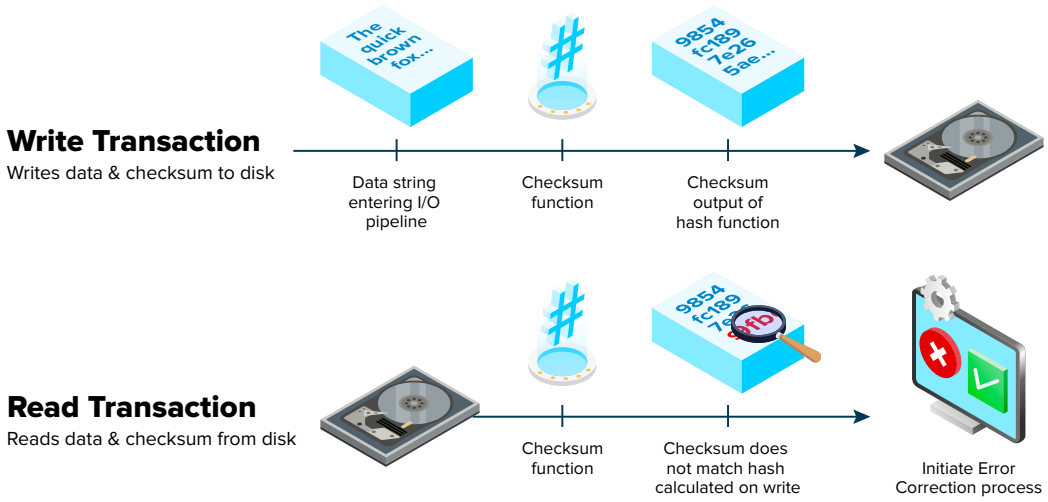
The checksumming capability has the following characteristics:

- Ensures data integrity for data traversing the ZIO pipeline so that data stored on and read from disk have a high level of data integrity.
- Initiates in-flight data recovery (at the moment data is being accessed) when checksum mismatches occur, indicating data errors caused either by transmission or bit-rot.
- Leverages a default Fletcher algorithm which carries a high level of data integrity without the potentially significant performance issues that might arise by using other algorithms such as SHA-256.
- Utilizes 2 parity bits (per OpenDrives pool best practices) for each transaction also to preserve data integrity, though this can technically be 1 or 3 depending on how the build of the pool.
- Automated scrubs occur every other week to verify and roll up all checksum.
- For multiple layers of redundancy, Atlas Core even checksums the checksum values as well as parity bits.

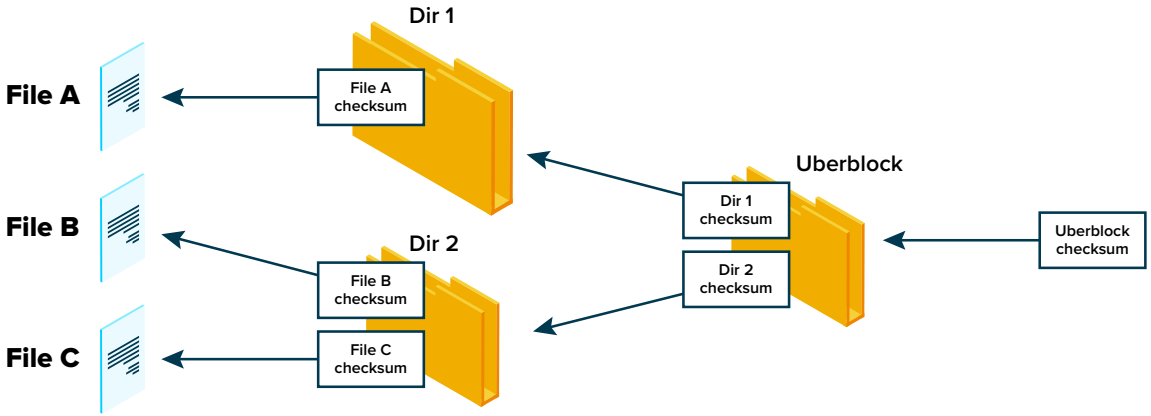
Further Reading

Checksum and snapshots capabilities are closely related. Please refer to the Snapshot Capability Sheet which explains in more detail the mechanics behind the way in which Atlas Core creates system snapshots for data integrity and recovery.

The following diagram illustrates the basic process of creating checksum values for data transactions that can then be compared for data integrity—data errors can be assumed when the checksum of data upon write and then read are not precisely the same:



To ensure data integrity further, Atlas Core creates a tree structure in which checksums themselves are checksummed, what is called a merkle tree allowing for very rapid validation of data integrity across large quantities of data. The following diagram illustrates a simplified version of this tree structure in which multiple recursive checksums are calculated along the entirety of the tree branch:



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