OAK RIDGE National Laboratory

Orion Lustre and Best Practices

Frontier Training Workshop

Jesse Hanley

Feb 16th, 2023

U.S. DEPARTMENT OF ENERGY


EXIT
WARNING

Orion – Scratch File System

- Based on Lustre and HPE ClusterStor
- 679 PB usable namespace
- Available from Frontier, DTNs, Andes, some other smaller resources
- Configured with some of the latest Lustre features to provide a better user experience
- 90-day purge policy

Lustre – A Parallel and Distributed File System

- Orion is a cluster of servers (~650 nodes)
- Each node plays a role in providing a POSIX namespace for users (`/lustre/orion/`)
- A file on Lustre consists of one or more components that may hit one or more servers
- Lustre has a distributed lock management process for concurrent access to files or regions within files

Lustre - DNE

- Distributed Namespaces (DNE) provides a way to scale metadata workloads
- Orion namespace is sharded across 40 metadata targets (MDT)
- Each project is tied to a specific MDT, with file metadata in that project residing on the same MDT

Lustre Striping - DoM

- Orion uses a feature called Data-on-MDT (DoM) where a portion of the file is stored along with the file's metadata (e.g., ownership, permissions, …)
- Currently, directories are configured to store up to the first 256KB with DoM
- Reduces contention on OSTs and aims to provide better performance for small file I/O

Lustre Striping – Tiers and PFL

- Each Orion Object Storage Server (OSS) serves one flash-based Object Storage Target (OST) and two HDD based OSTs
- Orion uses a feature called Progressive File Layout (PFL) to change the striping of a file as it grows
- For example, a file smaller than 8MB will be striped to a single OST, which larger files will stripe across multiple OSTs, taking advantage of more hardware resources

Best Practices – Default Striping

- The default striping set on Orion is targeted to work well for a variety of workloads
- In most cases, users should use this default striping. Though possible, manual striping should only occur after careful consideration and under collaboration with OLCF staff
- The default striping policy may change due to findings in production

Best Practices – Avoiding excessive attribute access

- Some file attributes such as file size require communication with several Lustre servers.
- Example: file `data.out` is striped across 8 OSTs; to retrieve the size, a remote procedure call (RPC) must be sent to each of the corresponding servers
- These attributes are retrieved when using system calls such as `stat()`. Commands like `` `ls --color` `` and `` `ls -l` `` perform these calls
- Depending on the directory content, a single `` `ls -l` `` command could perform thousands of RPCs

Best Practices – Lustre Optimized Tooling

- The Lustre client contains some optimized alternatives to some standard Linux utilities
  - `lfs find` provides a Lustre-specific `find` alternative
  - `lfs df` provides a Lustre-specific `df` alternative

Best Practices – Software Builds

- Building software on Lustre typically means a large quantity of small file I/O
- Compilation typically consists of a number of small file operations such as open/read/write/close/deletion; each operation may suffer overhead from the coordination efforts

Best Practices – File Access Options

- When possible, access files as read-only (`O_RDONLY` flag)
- If file access time updates are not needed, include the `O_NOATIME` flag
- Rather than opening/closing/appending a file repeatedly in a loop, open the file once before the loop
- Since Lustre has native byte-range locking capabilities, reevaluate if `flock()` is needed
- Since Lustre natively has data consistency features, reevaluate if `fsync()` is needed; this system call can drastically reduce performance

Lustre - Locking

- Lustre uses a distributed lock management (LDLM) system for consistency and access
- Concurrent operations on files/directories flow through LDLM
- Locks are generally managed on a per-client level
- There are limits to the number of concurrent locks each client can have on each storage target (MDT/OST)

Best Practices – Minimizing Lock Impact

Some common anti-patterns:

- Multiple clients opening the same byte range of a file for writing
- Multiple clients appending to the same file (subset of previous)
- Multiple clients concurrently creating numerous files in the same directory
- Concurrent non-overlapping writes to the same file without lockahead

Best Practices – Data Lifecycle Management

- Orion is a scratch file system
- Though we strive to provide a stable platform, there is no substitute for regular backups of data
- Planning and preparation for directory structure can both lead to performance improvements and easier management of data sets

Questions

- And thank you!