Lustre* File Striping
Legal Disclaimer

- THIS DOCUMENT AND RELATED MATERIALS AND INFORMATION ARE PROVIDED "AS IS" WITH NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT OF INTELLECTUAL PROPERTY RIGHTS, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION, OR SAMPLE. INTEL ASSUMES NO RESPONSIBILITY FOR ANY ERRORS CONTAINED IN THIS DOCUMENT AND HAS NO LIABILITIES OR OBLIGATIONS FOR ANY DAMAGES ARISING FROM OR IN CONNECTION WITH THE USE OF THIS DOCUMENT.
- All products, product descriptions, plans, dates, and figures are preliminary based on current expectations and subject to change without notice. Availability in different channels may vary.
- Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation in the United States and other countries.
- *Other names and brands may be claimed as the property of others.
- Copyright 2016 © Intel Corporation. All rights reserved.
Module Overview
Module Overview

- File striping - What is it?
- Advantages and Disadvantages
- Striping new files
- Object allocation within a file
- File system policies
- Manually overriding a striping policy
File Striping - What Is It?
Lustre* File Striping - Definition

What exactly is file striping?
- Reading from and writing to files across multiple OSTs, in parallel
- Stripe count - total number of objects that make up a Lustre* file
- Stripe size - amount of sequential data that is written to an object before moving on to write to the next object
- Stripe offset - term used when selecting the OST that will contain the first object
- The speed gained from parallel I/O (striping) is the reason Lustre* exists
Advantages and Disadvantages
Lustre* File Striping - Advantages

Advantages

- Each OST can deliver only so much bandwidth and space
- Reading/writing with multiple OSTs increases aggregate bandwidth linearly
- It also allows the creation of large files
- File size is not limited by the constraints of a single OST file system
- The fastest file system at scale, and open source. What more could you want?
Lustre* File Striping - Disadvantages

Disadvantages

- Increased metadata overhead with many stripes
- Extra network connections needed
- Unnecessary server contention when many files use many stripes at once
- Increased risk that [part of] a file may become unavailable
- Larger striping patterns use more OSTs and increase the odds...
- File layout needs to be decided when the file is created
Lustre* File Striping - Recommendation

General use case

- Stripe only over as many OSTs as required for:
  - Desired performance
  - Desired file size
- This minimizes the risk of:
  - Degraded OST
  - Unavailable OST
Striping New Files
Striping New Files

New files might not be striped over all OSTs
- Striping policy can be applied to a directory
  - Files created within that directory will inherit the default striping policy of that directory
- Changing the striping of an existing file?
  - Need to create a new file with the new policy, then copy old file to the new

Lustre* has a default striping policy
- Stripe size of 1MB (amount of data per stripe)
- Stripe count of 1 (number of OSTs to stripe over)
- Stripe offset of “-1” (minus one means use “best” OST)

The default storage policy can be modified
- Per file, per directory, or for the entire tree
- For OST pools
- Changes to striping policies do not affect existing files
Lustre* File Striping Diagram

File A

File B

File C

Object

OST00

OST01

OST02

1
2
3

4
5

7

6

1

1
Object Allocation within a New File
File Allocation Algorithms – RR vs. QoS

Round Robin (RR)

- Is the faster algorithm of the two
- Allocates objects sequentially across all the available OSTs
- Example with 8 OSTs available (using fictitious numbers):
  File 1: OST0, OST1, OST2
  File 2: OST3, OST4, OST5, OST6
  File 3: OST7, OST0, OST1, OST2, OST3, OST4, OST5
  File 4: OST6, OST7, OST0, OST1, OST2

RR always used when OSTs are "equally full"

- "Equally full" is defined by the value in:
  /proc/fs/lustre/lov/*/qos_threshold_rr (default value is 17%)
- Meaning, if OST % available space differs by less than 17%, RR is used
File Allocation Algorithms – RR vs. QoS (cont.)

QOS file allocation

- Always used when OSTs are not “equally full”
  - OST % available space differs by `qos_threshold_rr` or more
- The algorithm selects OSTs with more free space over those with less free space
- May or may not allocate objects equally across OSTs
  - Uses a weighted free space algorithm (% utilization, as well as other factors)
- Allocation of OSTs is controlled by
  `/proc/fs/lustre/lov/*/qos_prio_free` (default value is 91%)
  - 0 means each OST is allocated once (priority for balance)
  - 100 means OSTs are selected proportional to % utilization
    - Less full more likely to be selected more than once
File System Policies
Examples of File System Policies

“Single policy that stripes across all OSTs”
- Typical example for a checkpointing file system

“Three policies for three subdirectories”
- One with a large stripe count for large files
- One with a medium stripe count for medium files
- One with a small stripe count for small files
- Applicable to sites running applications with known optimal striping patterns

File system root directory (e.g. /mnt/Lustre01) retains default policy
- Subdirectories within contain user-defined policies
- Applicable to sites with experienced HPC developers
Determine Striping Policy

Both files and directories contain striping patterns

Check pattern with `lfs getstripe`

Example: Check striping on the `a` directory

```bash
# lfs getstripe -v /mnt/lustre/striped-files/
OBDS:
0: lustre-OST0000_UUID ACTIVE
1: lustre-OST0001_UUID ACTIVE
2: lustre-OST0002_UUID ACTIVE
3: lustre-OST0003_UUID ACTIVE
/mnt/lustre/striped-files
Stripe_count: -1 stripe_size: 1048576 stripe_offset: -1
```

Note that the default stripe count for this directory is -1 (stripe new files over all OSTs)
This directory is NOT set to the default stripe count of 1 stripe per file
Manually Overriding a Striping Policy
Setting the File Striping

The `lfs setstripe` command parameters:

```
  lfs setstripe [-d] | [--size | -s stripe-size] [--count | -c stripe-cnt] \ 
                   [--offset | -o start-ost] [--pool | -p <pool>] \ 
                   <filename|dirname>
```

Size is stripe size per OST

- Can be set in multiples of 64KB
- Using "-s 0" will use the file system (not directory). The default for the file system is 1024KB

Count is the number of OSTs to stripe over

- Using "-c 0" will use the file system (not directory) default
- Using "-c -1" will use all available OSTs

Offset is the starting OST to write to

- Using "-o -1" will allow Lustre* to choose the best OST to start with
Module Summary
Module Summary

- File striping - What is it?
- Advantages and Disadvantages
- Striping new files
- Object allocation within a file
- File system policies
- Manually overriding a striping policy
Congratulations! You have completed:
Lustre* File Striping