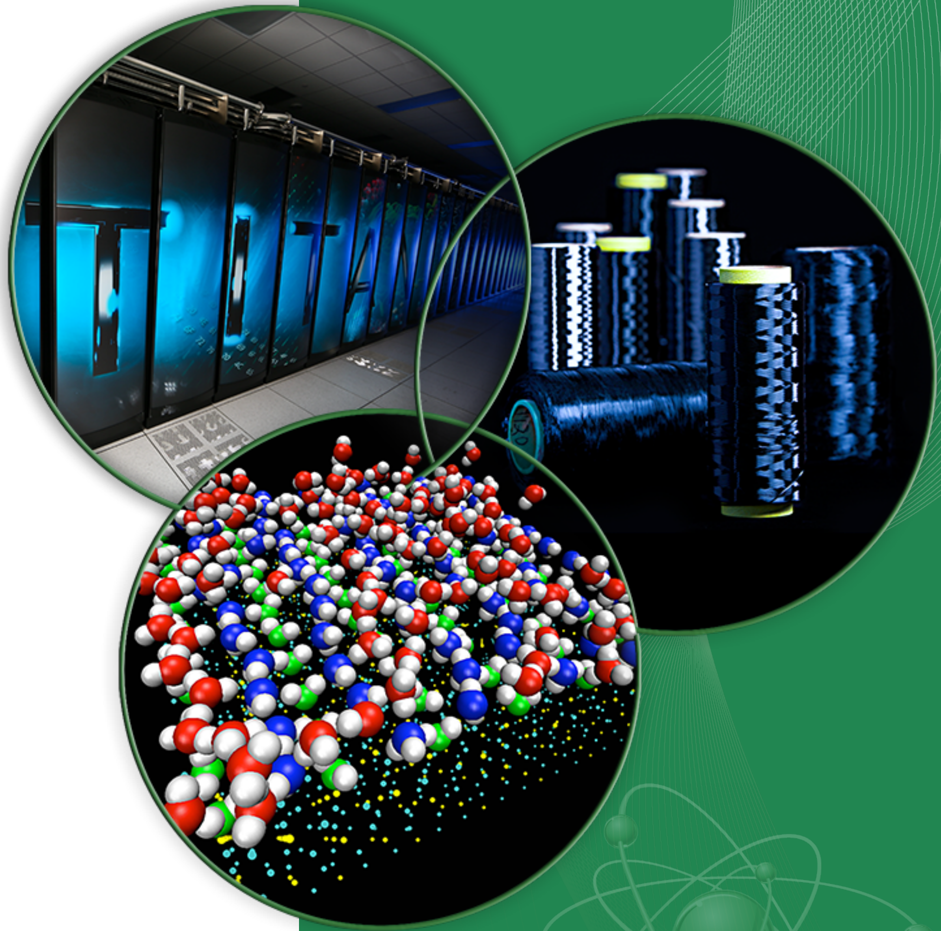
Oak Ridge National Laboratory
Computing and Computational Sciences Directorate

File System Administration and Monitoring

Jesse Hanley
Rick Mohr
Jeffrey Rossiter
Sarp Oral
Michael Brim
Jason Hill
Neena Imam
* Joshua Lothian (MELT)



Outline

- Starting/stopping a Lustre file system
- Mounting/unmounting clients
- Quotas and usage reports
- Purging
- Survey of monitoring tools

Starting a Lustre file system

- The file system should be mounted in the following order (for normal bringup):
  - MGT (Lustre will also mount the MDT automatically if the file system has a combined MGT/MDT)
  - All OSTs
  - All MDTs
  - Run any server-side tunings
- After this, the file system is up and clients can begin mounting
  - Mount clients and run any client-side tunings
- The commands for mounting share a similar syntax
  - mount -t lustre $DEVICE
- No need to start a service or perform a modprobe

Mount by label / path

Information about a target is encoded into the label

```
[root@testfs-oss3 ~]# dumpe2fs -h /dev/mapper/testfs-l28 | grep "^Filesystem
volume name"
Filesystem volume name:   testfs-OST0002
```

- These labels also appear under /dev/disk/by-label/

- If not using multipathing, this label can be used to mount by label:
  - `testfs-mds1# mount -t lustre -L testfs-MDT0000 /mnt/mdt`
  - Also avoid using this method when using snapshots
- If using multipathing, instead use the entry in /dev/mapper/. This can be set up in bindings to provide a meaningful name.
  - `testfs-mds1# mount -t lustre -L /dev/mapper/testfs-lun0`

Mounting Strategies

- These mounts can be stored in fstab.
  - Include noauto param – file system will not automatically mount at boot
  - Include _netdev param – file system will not mount if network layer has not started
  - These targets could then be mounted using:

```
mount -t lustre -a
```

- This process lends itself to automation

Client Mounting

- To mount the file system on a client, run the following command:
  - mount -t lustre MGS_node:/fsname /mount_point, e.g., mount -t lustre 10.0.0.10@o2ib:/testfs /mnt/test_filesystem
- As seen above, the mount point does not have to map to the file system name.
- After the client is mounted, run any tunings

Stopping a Lustre file system

- Shutting down a Lustre file system involves reversing the previous procedure. Unmounting all block devices on a host stops the Lustre software.
  - First, unmount the clients
    - On each client, run:
      - umount -a -t lustre #This unmounts all Lustre file systems
      - umount /mount/point #This unmounts a specific file system
  - Then, unmount all MDT(s)
    - On the MDS, run:
      - umount /mdt/mount_point (e.g., /mnt/mdt from the previous example)
  - Finally, unmount all OST(s)
    - On each OSS, run:
      - umount -t lustre –a

Quotas

- For persistent storage, Lustre supports user and group quotas. Quota support includes soft and hard limits.
  - As of Lustre 2.4, usage accounting information is always available, even when quotas are not enforced.
  - The Quota Master Target (QMT) runs on the same node as the MDT0 and allocates/releases quota space. Due to how quota space is managed, and that the smallest allocable chuck is 1MB (for OSTs) or 1024 inodes (for MDTs), a quota exceeded error can be returned even when OSTs/MDTs have space/inodes.

Usage Reports

- As previously mentioned, accounting information is always available (unless explicitly disabled).
  - This information can provide a quick overview of user/ group usage:
    - Non root users can only view the usage for their user and group(s)
      - # lfs quota -u myuser /lustre/mntpoint
    - For more detailed usage, the file system monitoring software Robinhood provides a database that can be directly queried for metadata information. Robinhood also includes special du and find commands that use this database.

Purging

- A common use case for Lustre is as a scratch file system, where files are not intended for long term storage. In this case, purging older files makes sense.
- Policies will vary per site, but for example, a site may want to remove files that have not been accessed nor modified in the past 30 days.

Purging Tools

- An administrator could use a variety of methods in order to purge data.
  - The simplest version includes a find (or lfs find) to list files older than x days, then remove them.
    - Ex: lfs find /lustre/mountpoint -mtime +30 -type f
      This would find files that have a modification time stamp older than 30 days
  - A more advanced technique is to use software like Lester to read data directly from a MDT.
    - https://github.com/ORNL-TechInt/lester

Handling Full OSTs

One of the most common issues with a Lustre file system is an OST that is close to, or is, full.

- To view OST usage, run the “lfs df” command. An example of viewing a high usage OST
  - `[root@mgmt ~]# lfs df /lustre/testfs | sort -rnk5  | head -n 5`

  -
    ```
    testfs-OST00dd_UUID  15015657888 12073507580  2183504616
    85% /lustre/testfs[OST:221]
    ```
- Once the index of the OST is found, running “lfs quota” with the –I argument will provide the usage on that OST.
  - for user in $(users); do lfs quota -u $user -I 221 /lustre/testfs; done

Handling Full OSTs (cont.)

- Typically, an OST imbalance that results in a filled OST is due to a single user with improperly striped files.
- The user can be contacted and asked to remove/ restripe the file, or the file can be removed by an administrator in order to regain use of the OST.
- It is often useful to check for running processes (tar commands, etc) that might be creating these files.
- When trying to locate the file causing the issue, it's often useful to look at recently modified files

Monitoring

- There are some things that are important to monitoring on Lustre servers. These include things like high load and memory usage.

Monitoring – General Software

- Nagios
  - Nagios is a general purpose monitoring solution.
  - A system can be set up with host and service checks. There is native support for host-down checks and various service checks, including file system utilization.
  - Nagios is highly extensible, allowing for custom checks
    - This could include checking the contents of the /proc/fs/lustre/health_check file.
  - It's an industry standard and has proven to scale to hundreds of checks
  - Open source (GPL)
  - Supports paging on alerts and reports. Includes a multi-user web interface
  - https://www.nagios.org

Monitoring – General Software

- Ganglia
  - Gathers system metrics (load, memory, disk utilization, …) and stores the values in RRD files.
  - Benefits to RRD (fixed size) vs downsides (data rolloff)
  - Provides a web interface for these metrics over time (past 2hr, 4hr, day, week, …)
  - Ability to group hosts together
  - In combination with collectl, can provide usage metrics for Infiniband traffic and Lustre metrics
  - http://ganglia.sourceforge.net/
  - http://collectl.sourceforge.net/Tutorial-Lustre.html

Monitoring – General Software

- Splunk
  - "Operational Intelligence"
  - Aggregates machine data, logs, and other user-defined sources
  - From this data, users can run queries. These queries can be scheduled or turned into reports, alerts, or dashboards for Splunk's web interface
  - Tiered licensing based on indexed data, including a free version.
  - Useful for generating alerts on Lustre bugs within syslog
  - There are open source alternatives such as ELK stack.
  - https://www.splunk.com/

Monitoring – General Software

- Robinhood Policy Engine
  - File system management software that keeps a copy of metadata in a database
  - Provides find and du clones that query this database to return information faster.
  - Designed to support millions of files and petabytes of data
  - Policy based purging support
  - Customizable alerts
  - Additional functionality added for Lustre file systems
  - https://github.com/cea-hpc/robinhood/wiki

Monitoring – Lustre tools

- Lustre provides information on a low level about the state of the file system
- This information lives under /proc/
- For example, to check if any OSTs on an OSS are degraded, check the contents of the files located at *Iproc/fs/lustre/obdfilter/*/degraded*
- Another example would be to check if checksums are enabled. On a client, run:
  - *cat /proc/fs/lustre/osc/*/checksums*
- More details can be found in the Lustre manual

Monitoring – Lustre tools

- Lustre also provides a set of tools
  - The lctl {get,set}_param functions display the contents or set the contents of files under /proc

    ```
    lctl get_param osc.*.checksums
    lctl set_param osc.*.checksums=0
    ```

    - This command allows for fuzzy matches
  - The lfs command can check the health of the servers within the file system:

    ```
    [root@mgmt ~]# lfs check servers
    testfs-MDT0000-mdc-ffff880e0088d000: active
    testfs-OST0000-osc-ffff880e0088d000: active
    ```

    - The lfs command has several other possible parameters

Monitoring – Lustre tools

– The llstat and llobdstat commands provide a watch-like interface for the various stats files

- llobdstat: /proc/fs/lustre/obdfilter/<ostname>/stats
- llstat: /proc/fs/lustre/mds/MDS/mdt/stats, etc. Appropriate files are listed in the llstat man page
- Example:

```
[root@sultan-mds1 lustre]# llstat -i 2 lwp/sultan-MDT0000-lwp-MDT0000/stats
/usr/bin/llstat: STATS on 06/08/15 lwp/sultan-MDT0000-lwp-MDT0000/stats on 10.37.248.68@o2ib1
snapshot_time             1433768403.74762
req_waittime              1520
req_active                1520
mds_connect               2
obd_ping                  1518
lwp/sultan-MDT0000-lwp-MDT0000/stats @ 1433768405.75033
Name          Cur.Count  Cur.Rate  #Events  Unit          last       min       avg       max    stddev
req_waittime  0          0         1520     [usec]           0        46    144.53     14808    380.72
req_active    0          0         1520     [reqs]           0         1      1.00         1      0.00
mds_connect   0          0         2        [usec]           0        76   7442.00     14808  10417.10
obd_ping      0          0         1518     [usec]           0        46    134.91       426     57.51
^C
```

Monitoring – Lustre Specific Software

- LMT
  - "The Lustre Monitoring Tool (LMT) is a Python-based, distributed system that provides a top-like display of activity on server-side nodes"
  - LMT uses cerebro (software similar to Ganglia) to pull statistics from the /proc/ file system into a MySQL database
- lltop/xltop
  - A former TACC staff member, John Hammond, created several monitoring tools for Lustre file systems
  - lltop - Lustre load monitor with batch scheduler integration
  - xltop - continuous Lustre load monitor

Monitoring – Lustre Specific Software

- Sandia National Laboratories created OVIS to monitor and analyze how applications use resources
  - This software aims to monitor more than just the Lustre layer of an application
  - https://cug.org/proceedings/cug2014_proceedings/includes/files/pap156.pdf
  - OVIS must run client-side, where most of the other monitoring tools presented here are run from the Lustre servers
- Michael Brim and Joshua Lothian from ORNL created Monitoring Extreme-scale Lustre Toolkit (MELT)
  - This software uses a tree-based infrastructure to scale out
  - Aimed at being less resource intensive than solutions like collectl
    - http://lustre.ornl.gov/ecosystem/documents/LustreEco2015-Brim.pdf

Summary

- How to start and stop a Lustre file system
- Steps to automate these procedures
- Software, both general and specialized, to monitor a file system

Acknowledgements

This work was supported by the United States Department of Defense (DoD) and used resources of the DoD-HPC Program at Oak Ridge National Laboratory.