Lustre File System

PRATIK JAIN

School of Informatics and Computing, Bloomington, IN 47408, U.S.A.

Corresponding authors: jainps@iu.edu

Paper-1, March 21, 2017

We introduce the Lustre file system and gives a brief overview on its applications in the industry, its architecture, and the steps required for successfully installing and configuring the file system. Not only its applications in various fields like HPC and Big Data are explored, but also the areas in which this file system is not recommended are discussed.

Keywords: Lustre File System, Object based file system, Object based storage device

https://github.com/pratik11jain/sp17-i524/blob/master/paper1/S17-IR-2012/report.pdf

INTRODUCTION

Lustre is a type of parallel distributed file system. It was started as a research project in 1999 by Peter J. Braam. It is now generally used for large-scale cluster computing. The name Lustre is a combination of Linux and cluster [1]. It is often used in supercomputers due to its high-performance capabilities and open licensing. The Lustre file system is scalable and can be part of multiple computer clusters with tens of thousands of client nodes, tens of petabytes of storage on hundreds of servers, and more than a terabyte per second of aggregate I/O throughput. A Lustre file system was first installed for production use in March 2003, on one of the largest supercomputers at the time, the MCR Linux Cluster at Lawrence Livermore National Laboratory. Lustre file system software is available under the GNU General Public License and can be utilized for computer clusters ranging in size from small workgroup clusters to large-scale, multi-site clusters. This makes this file system a popular choice for businesses with large data centers, including in various industries such as simulation, life science, meteorology, rich media, oil and gas, and finance.

ARCHITECTURE

The Lustre architecture is a storage architecture for clusters. Its central component, the Lustre file system, is supported on the Linux operating system and provides a POSIX standard-compliant UNIX file system interface. The architecture is used for many different kinds of clusters and it is best known for powering many of the largest HPC clusters worldwide [2]. Lustre file system is used by many HPC sites as a site-wide global file system, serving dozens of clusters. Its ability to scale capacity and performance for any need reduces the need to deploy many separate file systems, such as one for each compute cluster and avoiding the need to copy data between compute clusters simplifies storage management. In addition to aggregating storage capacity of many servers, the I/O throughput is also aggregated and scales with additional servers. Also, throughput and/or capacity can be easily increased by adding servers dynamically. Lustre's scalable architecture has three main components, first, the Metadata Server that provides metadata services for a file system and manages a Metadata Target that stores the file metadata, second, the Object Storage Servers that manage the Object Storage Targets that store the file data objects and third, the clients that access and use the data. Lustre presents all clients with a unified namespace for all of the files and data in the file system and allows concurrent and coherent read and write access to the files in the filesystem [3].
Followings are the principal foundations of Lustre:

Object-based storage devices

Unlike conventional storage devices, an Object Based Disk (OBD) or Object-Based Storage Device (OBSD) is one that works at the level of files rather than at the level of individual blocks. The OBD internally keeps track of allocated objects, which blocks belong to each object, free space, etc. rather than exposing these details to the operating system. This architecture looks at devices that can manipulate file objects. Typical commands executed as a part of the object interface are create, destroy, read/write block X in object N and read/write attributes of objects [4].

A stackable object driver model

In addition to direct drivers which control storage, there are logical object drivers, client object drivers and associated target drivers. For example, RAID can be implemented by having a logical object driver that speaks with multiple direct drivers [5]. Other interesting logical drivers can perform HSM, parallel I/O and cryptographic operations. Lustre's logical object driver

manages snapshots of file systems. Client drivers are responsible for packing up object requests and shipping them to targets. This can exploit SAN's such as Fibre Channel, InfiniBand and Gigabit Ethernet. Since the interface is uniform, logical drivers can be stacked on top of direct drivers or clients.

Object-based file systems

There are at least three types of file systems that can be imagined in the object storage model. OBDFS is an object-based file system that is meant for the use with non-shared storage devices. An inode file system provides direct access to objects named by an object id. Third are cluster file systems. The traditional cluster file system can become significantly simpler than those implemented with shared block storage devices.

LUSTRE FILE SYSTEM AND STRIPING

The ability to stripe data across multiple OSTs in a round-robin fashion is one of the main factors leading to the high performance of Lustre file systems. Users can optionally configure each file the number of stripes, stripe size, and OSTs that are used. Striping can be used to improve performance when the aggregate the bandwidth to a single file exceeds the bandwidth of a single OST. The ability to stripe is also useful when a single OST does not have enough free space to hold an entire file.

IMPLEMENTATION

In a typical Lustre installation on a Linux client the filesystem driver module is loaded into the kernel and the filesystem is mounted like any other local or network filesystem. Even though it may be composed of tens to thousands of individual servers and filesystems, client applications see a single, unified filesystem. On some massively parallel processor (MPP) installations, computational processors can access a Lustre file system by redirecting their I/O requests to a dedicated I/O node configured as a Lustre client [6]. Another approach used in the early years of Lustre is the user-level liblustre library. This provided userspace applications with direct filesystem access. Liblustre allows computational processors to mount and use the Lustre file system as a client. Using liblustre, the computational processors could access a Lustre file system even if the service node on which the job was launched is not a Linux client. Liblustre allowed direct data movement between application space and the Lustre OSSs. It did not require an intervening data copy through the kernel, thus providing access from computational processors to the Lustre file system directly in a constrained operating environment.

INSTALLATION

Following is the overview of steps needed for installing Lustre. The first step is to setup Lustre Filesystem Hardware. Lustre runs on most commodity hardware with any kind of block storage device including single disks, software and hardware RAID and logical volume manager. For servers, 64-bit architectures are recommended. Lustre allows for multiple MDSes for high availability. The size of the MDT's backing file system should be chosen based on the total number of files planned to be stored in the Lustre file system, and the aggregate OST space should be chosen based on the total amount of data planned to be stored in the file system. Estimating space requirements early can dictate hardware requirements. After this, the Lustre software is installed. Lustre runs on a variety of Linux kernels from Linux distributions including RHEL, CentOS, and SLES. When using the Lustre ldiskfs OSD only, it will be necessary to patch the kernel before building Lustre. The required Lustre RPMs or source can be downloaded here [7]. Metadata and Object Storage Server require the Lustre patched Linux kernel, Lustre modules, Lustre utilities and e2fsprogs installed. The clients require the Lustre client modules, client utilities and, optionally, the Lustre patched kernel. For configuring Lustre, the Lustre Networking (LNET) kernel modules have a variety of module parameters that can be set in the /etc/modprobe.d/lustre.conf file. The type of network used and globally-available networks can be specified along with routes in a Lustre configuration. To set up and tune the filesystem, Lustre provides a variety of configuration utilities that include mkfs.lustre to format a disk for a Lustre service, tunefs.lustre to modify configuration information on a Lustre target disk, lctl to directly control Lustre via an ioctl interface and mount.lustre to start a Lustre client or target service.

WHERE NOT TO USE IT?

Although a Lustre file system can function in many work environments, it is not necessarily the best choice for all applications [2]. Although, it is best suited for uses that exceed the capacity that a single server can provide, in some use cases, a Lustre file system can perform better with a single server than other file systems due to its strong locking and data coherency. A Lustre file system is not particularly well suited for "peer-to-peer" usage models where clients and servers are running on the same node, each sharing a small amount of storage, due to the lack of data replication at the Lustre software level. In such uses, if one client or server fails, then the data stored on that node will not be accessible until the node is restarted.

CONCLUSION

The Lustre file system is an open-source, parallel file system that supports many requirements of leadership class HPC simulation environments and enterprise environments worldwide. Because Lustre file systems have high performance capabilities and open licensing, it is often used in supercomputers. Lustre file systems are scalable and can be part of multiple computer clusters with tens of thousands of client nodes, tens of petabytes of storage on hundreds of servers, and more than a terabyte per second of aggregate I/O throughput. Lustre file systems is a popular choice for businesses with large data centers, including those in industries such as meteorology, simulation, oil and gas, life science, rich media, and finance. Lustre provides a POSIX compliant interface and many of the largest and most powerful supercomputers on Earth today are powered by the Lustre file system. Its architecture contains 3 main components - the Metadata Server, the Object Storage Servers and the clients. There are a few fields in which use of Lustre file system is not recommended. Here, we have covered the basic components of Lustre File system, the overview about the installation steps of lustre file system and also a few scenarios in which the use of lustre file system is not recommended. Thus the paradigm of lustre file system is briefly introduced.

REFERENCES

[1] Aviso Legal, "Ungrid status report 2010," Web Page, Nov. 2010, accessed 2017-02-25. [Online]. Available: http://www.ungrid.unal.edu.co/cluster/status.htm

[2] Intel, “The lustre*software release 2.x operations manual,” Web Page, Dec. 2016, accessed 2017-02-25. [Online]. Available: http://doc.lustre.org/lustre_manual.xhtml
[3] OpenSFS, EOFS, “Getting started with lustre,” Web Page, Dec. 2016, accessed 2017-02-25. [Online]. Available: http://lustre.org/getting-started-with-lustre/
[4] Wikipedia, “Object storage,” Web Page, Feb. 2017, accessed 2017-02-13. [Online]. Available: https://en.wikipedia.org/wiki/Object_storage#Object-based_storage_devices
[5] OpenSFS, EOFS, “About the lustre file system,” Web Page, Dec. 2016, accessed 2017-02-25. [Online]. Available: http://lustre.org/about/
[6] Wikipedia, “Blue gene,” Web Page, Jan. 2017, accessed 2017-02-13. [Online]. Available: https://en.wikipedia.org/wiki/Blue_Gene
[7] OpenSFS, EOFS, “Download lustre,” Web Page, Dec. 2016, accessed 2017-02-25. [Online]. Available: http://lustre.org/download-lustre/