Integrating GlusterFS with iSCSI Target

Dhananjaya G[1] Professor Anala M R[2]
[1]M.Tech Student [2]Associate professor
[1]Department of Computer Networking and Engineering [2]Department of Computer Science and Engineering
[1,2]RV college of engineering, Bangalore

Abstract— Server as the core of the information systems in today's business operations playing a pivotal role, data storage on server is becoming the lifeline to maintain the normal operation of the millions of businesses and organizations. Correct choice of independent control of the server or cluster systems and storage devices can improve system performance and operational capabilities. The user request rate and response time of business server data storage system are dependent on the selection of the storage devices. These devices can greatly affect system performance. To effectively protect data and give linear performance for iSCSI initiator, it is required to modify the iSCSI target driver to provide Gluster file system backend storage as distributed block device. This paper aims at discussing ways of modifying iSCSI Target (ISTGT) driver to support Gluster file system.

Keywords: Gluster, iSCSI, UserSpace (FUSE)

I. INTRODUCTION

GlusterFS is an open source, clustered file system capable of scaling to several peta-bytes and handling thousands of clients [1]. GlusterFS can be flexibly combined with commodity physical, virtual, and cloud resources to deliver highly available and performant enterprise storage at a fraction of the cost of traditional solutions.

GlusterFS clusters together storage building blocks over InfiniBand RDMA and/or TCP/IP inter-connect, aggregating disk and memory resources and managing data in a single global namespace. GlusterFS is based on a stackable user space design, delivering exceptional performance for diverse workloads.

Gluster storage platform and Gluster File system offer the most reliable and simple open source clustered file system running on commodity hardware. The solution features the following specific advantages:

- No metadata server—Gluster File system uses a fully distributed architecture (together with the Gluster Elastic Hash) removing the need for a metadata server and consequently, a potential bottleneck.
- High performance global namespace—Gluster File system offer scale out to hundreds of petabytes with linear performance using 10GigE and InfiniBand (native RDMA).
- High availability—Gluster File system uses replication to survive hardware failures, and automatically performs self-healing to restore performance. Data is stored using an NFS-like native format.
- Stackable user space design—Simple to install and portable, Gluster File system has no kernel dependencies. The stackable design also enables you to configure specific workload profiles, as required.

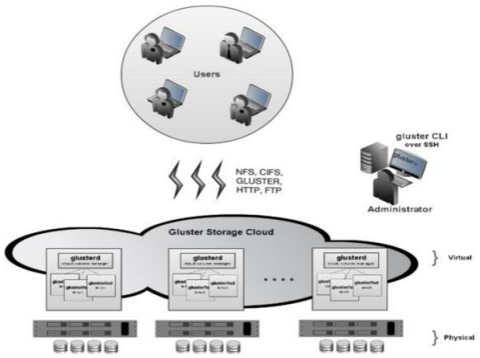


Fig. 1: Virtualized Cloud Environment

GlusterFS is designed for today's high-performance; Figure shows virtualized cloud environments. Unlike traditional data centers, cloud environments require multi-tenancy along with the ability to grow or shrink resources on demand. Enterprises can scale capacity, performance, and availability on demand, with no vendor lock-in, across on-premise, public cloud, and hybrid environments.

The internet Small Computer Systems Interface (iSCSI) is a way to share storage over a network. Unlike NFS, which works at the file system level [2], iSCSI works at the block device level. In iSCSI terminology, the system that shares the storage is known as the *target [3]*. The storage can be a physical disk, or an area representing multiple disks or a portion of a physical disk. The clients which access the iSCSI storage are called *initiators*. To initiators, the storage available through iSCSI appears as a raw, unformatted disk known as a LUN [4]. Device nodes for the disk appear in /dev/ and the device must be separately formatted and mounted.

II. WAYS OF INTEGRATION

To effectively protect data and give linear performance for iSCSI initiator, it is required to modify the FreeBSD ISTGT driver [5] to provide Gluster file system backend storage as distributed block device. iSCSI on Gluster can be set up using the Linux Target driver. This is a user space daemon that accepts iSCSI (as well as iSER and FCoE.) It interprets iSCSI CDBs and converts them into some other I/O operation, according to user configuration. In our case, we can convert the CDBs into file operations that run against a gluster file. The file represents the LUN and the offset in the file the LBA.

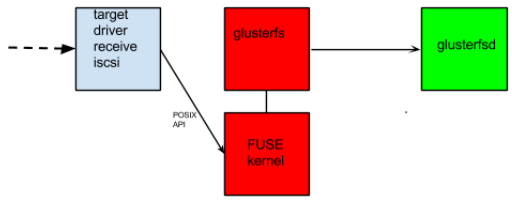
The block device is a file based image, which acts as the backend for the Linux SCSI target. The file resides in gluster, so enjoys gluster's feature set. But the client only sees a block device. The Linux SCSI target presents it over iSCSI.

The first method is to use POSIX API's to write the target driver. Since GlusterFS is user space file system it

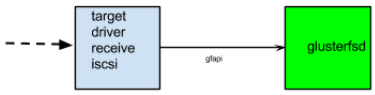
uses File System in UserSpace (FUSE) module to hook itself with the VFS and it redirects calls to glusterfs which is in user space. GlusterFS used FUSE incurred data copies and context switches. That FUSE penalty can be avoided using libgfapi. The libgfapi can be inserted into the Linux target driver rather easily, as it has a nice extensible framework. In this manner IO commands may be forwarded directly to the storage server.

The diagram below depicts gluster's datapath when FUSE is present and absent. Using gfapi, the red boxes are removed (a kernel trip and extra hop).

GLUSTER SERVER USING TGTD+FUSE



GLUSTER SERVER USING JUST TGTD



To make that happen modified the Linux target driver to have a "gluster" module. The implementation was mostly a one–to-one replacement of POSIX APIs with "gluster" APIs. For example read and writes were translated into gf_read and gf_writes. Performance results with target driver modified using gfapi shows 10-20% performance improvement.

III. CONCLUSION

Modified the iSCSI target driver to provide Gluster file system backend storage as distributed block device to effectively protects data and gives linear performance for the iSCSI initiator. ISTGT linux target driver written using libgfapi increased the performance by 10-20%.

REFERENCES

[1] www.gluster.org/
[2] RFC 1094, Network File System Protocol Specification, Network Working Group, March 1989.
[3] J. Satran, K. Meth, C. Sapuntzakis, M. Chadalapaka, and E. Zeidner, "Internet Small Computer System Interface (iSCSI)", RFC 3720, Apr 2004.
[4] U.Troppens, R.Erkens, W.Muller, "Storage Networks Explained: Basics and Application of Fibre Channel SAN, NAS, iSCSI and InfiniBand", John Wiley, 2nd edition, ISBN: 978-0-470-74143-6, pp. 1 – 159, 2009.
[5] http://www.peach.ne.jp/archives/istgt/

