GlusterFS Implementation on OpenStack at CSIR Private Cloud

[1]Mukesh Pund,[2]Prachi Vats,[3]Salman Sheikh

[1]Senior Principal Scientist & Principal Investigator CSIR-KNOWGATE, CSIR-NISCAIR, Vigyan Suchna Bhawan, New Delhi, India
[2]System Engineer, TCS, Noida, India
[3]Former Trainee, CSIR-NISCAIR, Vigyan Suchna Bhawan, New Delhi, India

Abstract : GlutserFS is an economical, scalable, and reliable storage solution that can be used for storage requirement of Cloud Computing and Big Data as it efficiently solves the problem of managing huge size of data. The CSIR Private Cloud is developed for various Knowledge Resource Centers of CSIR laboratories in India. The GlusterFS is implemented in order to provide scalable distributed file system to CSIR Private Cloud and its services like nova, glance and cinder. The GlusterFS storage server pool is created of storage nodes to create single global namespace. The step by step implementation of GlusterFS on OpenStack at CSIR Private Cloud is presented in this paper.

Keywords - Affordable and Scalable Storage, Cloud Storage, Cloud Computing, Infrastructure-as-a-Service, Storage for Big Data, OpenStack, Open Source, GlusterFS, Cinder-Block Storage, Glance-Compute Image Repository, Nova Openstack component, CSIR Cloud, Cluster Storage.

I. Introduction

CSIR-Council of Scientific and Industrial Research has approved an activity for deploying a private cloud infrastructure for its network of Knowledge Resource Centers. This activity was approved under project entitled "CSIR Knowledge Gateway & Open Source Private Cloud Infrastructure (KNOWGATE)". One of the Objectives of the project was to provide cloud infrastructure that can be accessed as a service from any CSIR laboratories and it can also be used by individual Knowledge Resource Centers (KRCs) of CSIR laboratories for their computational needs.[11] The CSIR Private Cloud is setup at CSIR-National Institute of Science Communication and Information Resources (CSIR-NISCAIR) - the nodal laboratory for the project. The security/ privacy of the data was one of the major concern while deploying the cloud using OpenStack- an open source software.

Data storage becomes a challenge when size of the data becomes large. One of the major requirements of CSIR Private Cloud was to provide information communication technology solution using economical and robust cloud computing technology. Further, data would have to store within CSIR premises. After exploring many available technologies for cloud computing and data storage, it was decided to implement OpenStack with GlusterFS. In this paper, the process required to implement GlusterFS in centos as a part of IaaS cloud that is carried out to store multiple petabytes of data at CSIR-NISCAIR premises. Multifarious advantages make GlusterFS as an attractive option to use such as it scales to several petabyte, handles thousands of clients, uses commodity hardware and can be used on most of the disk file systems that support extended attributes etc. With the help of GlusterFS, capacity can be scaled, performance can be improved and it can be used as per demand. Gluster is a versatile Open Source storage technology that can be beneficial for developing public, private and hybrid cloud. It is being used by various enterprises at production stage.[2]

II. Overview and implementation of GlusterFS

GlusterFS is an open source cluster file system, capable of scaling to several petabytes of storage. GlusterFS package comes with two components, a server and a client. To mount GlusterFS file systems, the client computers need FUSE supported in the kernel. By using FUSE, no kernel patches are required, and therefore the software can be installed without any server downtime. GlusterFS server operates on Linux, FreeBSD and Solaris, and client runs only on Linux machines.[1]
GlusterFS basically cluster storage by clubbing disk storage and memory resources in order to manage data into a single namespace, implying it put different servers into a one single storage pool or one large network file system.
As need to optimize storage comes with large data which tends to increase each day. OpenStack cloud is an open source software, it can be easily adjoined with GlusterFS, which is again open source software. The need for GlusterFS comes in cloud with multi-tenant environment wherein resources grow or shrink on demand.[3]

II.1 Storage concepts in GlusterFS

- **Brick –**Bricks are storage units that consists of a directory path and a server on which it is created. Bricks have similar size and there can be n number of bricks on each node. They are the basic building blocks of a GlusterFS volume, which can be mounted using the GlusterFS Native Client.[4]

- **Trusted Storage Pool**
It is a trusted network of servers that will host storage resources. Storage pool consist of various servers that would host GlusterFS volumes and which is made from the bricks of the storage nodes.[4]

- **Block Storage –**Devices that enable us to move data, in form of blocks, between various systems.

- **Cluster –**It is Collection of storage servers.

- **Distributed File System –**A file system in which data is present on different nodes and allows user to make use of files as if they were present on own computer, without being aware of its actual location.

- **FUSE –**(File system in user space) - It enables to implement a functional file system in a user space program and even allows file creation over kernel. Some of its advantages are secure implementation, easily installed, stable etc.[7]

- **Translator –** A translator is a piece of code that performs actions that are invoked by user from mount point. It connects one or more sub volumes, and facilitates flow of data. Translator configuration can be managed via command line interface(CLI).

- **Volume –**A volume is a logical collection of bricks. Almost all the management operations of gluster takes place at volume, moreover one can host more than one volume on the same node. Volume can be of various types such as distributed, replicated etc.[3]

II.2 GlusterFS advantages

- **Innovation –** It helps in improving performance by unifying data and objects and eliminates metadata.
- **Elasticity –** GlusterFS adapts to growth and reduction of size of data.
- **Simplicity –**GlusterFS is easy to manage and independent from kernel while running in user space.
- **Scalable –** It provides a faster file system in absence of metadata .
- **Affordable –**It is OpenSource and deploys on commodity hardware.
- It has **Self-healing** ability and even can **re-balance data**.

II.3 Volumes

Types of Volumes:

- **Distributed Volume-** In this distribution of files takes place across bricks in volume. One can use it where storage scaling is required and redundancy has to be avoided. This can be achieved as one file can be stored only on one of the bricks. It is easy and cheap method to scale storage.[3]

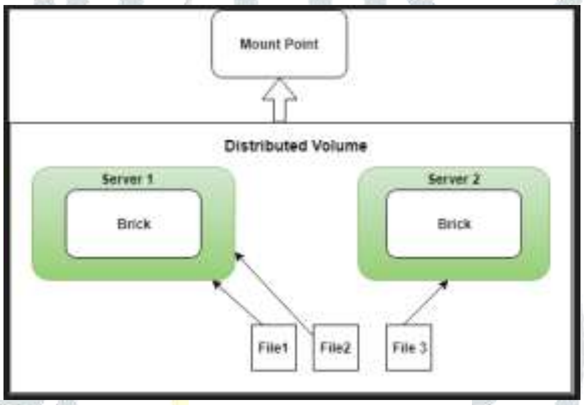


Fig -1: Distributed volume design

- **Replicated –**Replicated volumes replicate files across bricks in the volume. With replicated files one can avoid data loss problem as exact data copies are maintained on all bricks. In case of failure of one brick other can act as a backup. Thus making this volume type as more reliable and redundant. Like in above figure file1 is present on both brick1 and brick2.[3]

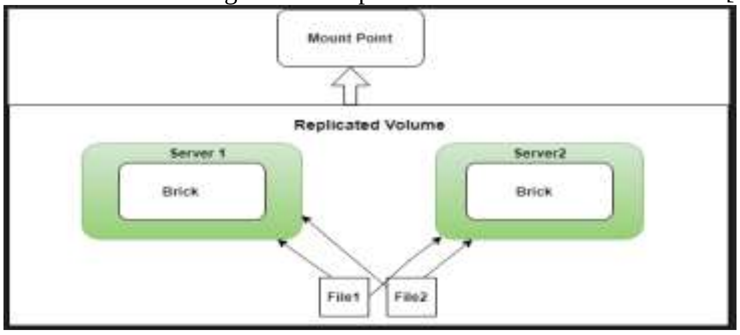


Fig -2: Replicated volume design

- **Distributed Replicated -** It basically distributes file across replicated bricks in a given volume. It is used when there is high demand of data and even scaling is in scenario. Consider an example of six bricks and replica count is 3, then first 3 bricks becomes replica of each other and then the next three.[3]

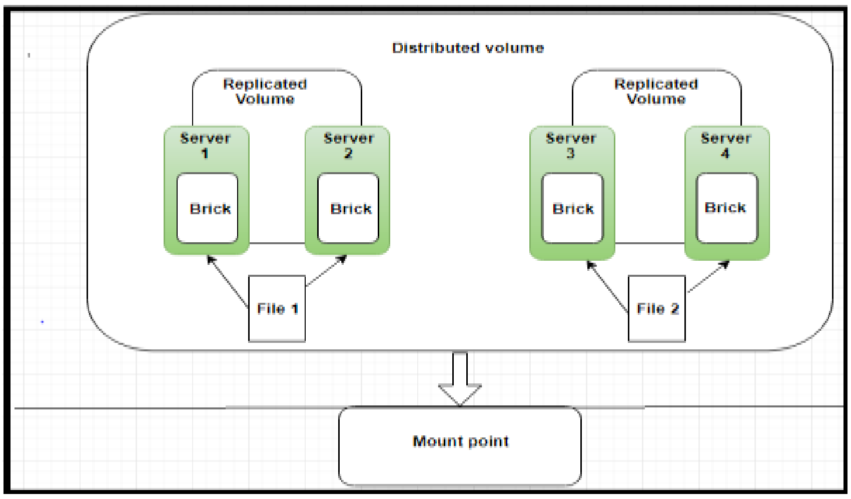


Fig -3: Distributed Replicated design

III. Need

In order to improve and save on count of infrastructure, plan was formulated to put forward and facilitate labs within Organization with a complete set up, thus came cloud IaaS implementation into picture, but there came a need to scale storage space for smooth and flawless functioning of available servers across organization. Hence, to facilitate OpenStack cloud services such as NOVA, CINDER, GLANCE with large storage space GlusterFS came into scenario. We considered GlusterFS as best storage solution because storage solution like SAN are very costly and were not feasible considering cost constraints. Therefore we searched various affordable storage solution and zeroed on GlusterFS i.e. available free of cost in community version. Further, GlusterFS meets all our requirements like scalability and robustness. One of the challenges was integrity on OpenStack Cloud.

IV. System Architecture

In our architecture, we used default GlusterFS volume i.e. distributed volume as we wanted to avoid redundancy. We have 3 servers and each has their own bricks and a specific file is stored on a particular brick. We have to avoid redundancy as we have different cloud services such as nova, cinder and glance, these will fetch their storage from these volumes. [4]

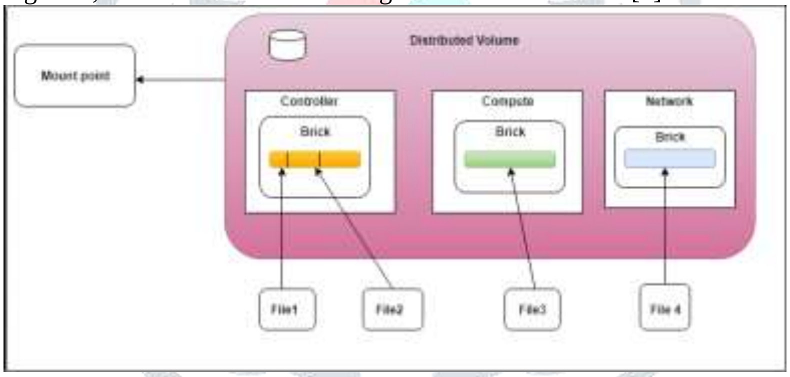


Fig -4: GlusterFS Architectural design

IV.1 Configuration Details

There are 3 nodes into consideration i.e. compute, controller and network node. We can even add nodes as per requirement to increase storage.

Compute Node- VM instances are installed on compute node. For deploying and running virtual machine compute node runs hypervisor. Whenever VM is launched all of its resources which are required for running a virtual machine are taken from compute node such as VCPUs, RAM etc. It can be called as physical machine that hosts VMS.[5]

Controller Node-This is the control centre or engine of the cloud, all the shared OpenStack services and tool run on this node. It provides API's, scheduler and various shared services for cloud functioning. Controller node has various cloud components hosted on it such as dashboard, image store, identity service. Moreover Nova compute and neutron server are configured on Controller itself.[5]

Network Node- All the network core is handled at network node, it gives virtual networking and related services to provide network layer architecture for our VMs. It make use of network 3 layer and DHCP network services. [6]

IV.2 Implementation

Step 1:**Install gluster repository on all Nodes**

To start with the configuration of GlusterFS, its repository must be installed on machines for which volumes have to be created.

GlusterFS :

[root@controller]#cd /etc/yum.repos.d/

[root@controller]#wget http://download.gluster.org/pub/gluster/GlusterFS/LATEST/RHEL/GlusterFS-epel.repo

Step 2:Install packages of GlusterFS on all machines[3]

[root@controller]#yum install -y GlusterFSGlusterFS-fuse GlusterFS-server

Step 3:**Installremi-release repository:**

[root@controller]# wget http://rpms.famillecollet.com/enterprise/remi-release-6.rpm

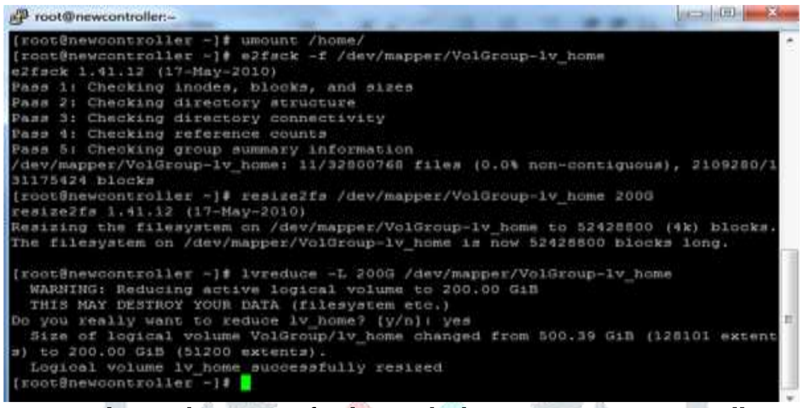
[root@controller]#rpm -ivh remi-release-6.rpm

Step 4:Install following packages
[root@controller]#yum install -y gcc kernel-headers kernel-devel
[root@controller]#yum install -y keepalived

Before starting GlusterFS configuration, there is a need to manage storage on 3 nodes as per need of organization, in 3 node cloud implementation every node will take its storage for e.g. 290 GB is taken from controller, 1800 GB from network node and from compute we have a total storage of 3.8 TB, all the above storage is taken for Gluster. Thus resize home for all the 3 nodes and then create certain bricks. As per architecture these bricks will be clubbed into volumes and the volume is mounted on directory which is used by the services to scale the storage.

Step 5: Reducing /home partition on Controller Node[9]

A new partition from the home volume was created after reducing its space to certain limits. This was to utilize the home storage in efficient way as per need.
[root@controller]#umount /home/
[root@controller]#e2fsck -f /dev/mapper/VolGroup-lv_home
[root@controller]# resize2fs /dev/mapper/VolGroup-lv_home 200G
[root@controller]#lvreduce -L 200G /dev/mapper/VolGroup-lv_home

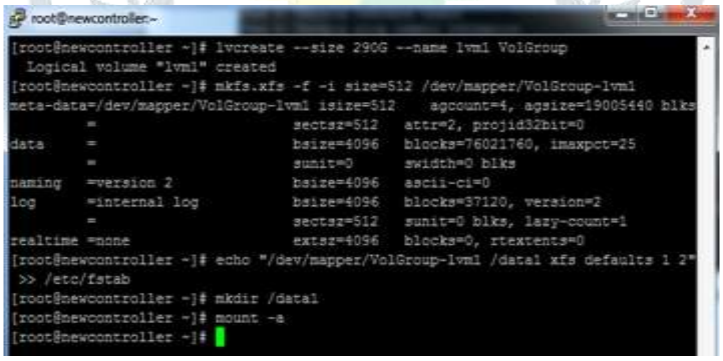


```
[root@newcontroller ~]# umount /home/
[root@newcontroller ~]# e2fsck -f /dev/mapper/VolGroup-lv_home
e2fsck 1.41.12 (17-May-2010)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/dev/mapper/VolGroup-lv_home: 11/32800768 files (0.0% non-contiguous), 2109280/1
31175424 blocks
[root@newcontroller ~]# resize2fs /dev/mapper/VolGroup-lv_home 200G
resize2fs 1.41.12 (17-May-2010)
Resizing the filesystem on /dev/mapper/VolGroup-lv_home to 52428800 (4k) blocks.
The filesystem on /dev/mapper/VolGroup-lv_home is now 52428800 blocks long.

[root@newcontroller ~]# lvreduce -L 200G /dev/mapper/VolGroup-lv_home
  WARNING: Reducing active logical volume to 200.00 GiB
  THIS MAY DESTROY YOUR DATA (filesystem etc.)
Do you really want to reduce lv_home? [y/n]: yes
  Size of logical volume VolGroup/lv_home changed from 500.39 GiB (128101 extent
s) to 200.00 GiB (51200 extents).
  Logical volume lv_home successfully resized
[root@newcontroller ~]#
```

Fig -5: Displaying the output of reducing the home partition on Controller node

Creating new partition with 290G space on Controller (LVM)

A new partition was created on controller node from the respective home directory after reduction. The partition was named as **lvm1**.
[root@controller]#lvcreate --size 290G --name lvm1 VolGroup
[root@controller]#mkfs.xfs -f -i size=512 /dev/mapper/VolGroup-lvm1
[root@controller]#echo "/dev/mapper/VolGroup-lvm1 /data1 xfs defaults 1 2">> /etc/fstab
[root@controller]#mkdir /data1
[root@controller]#mount -a

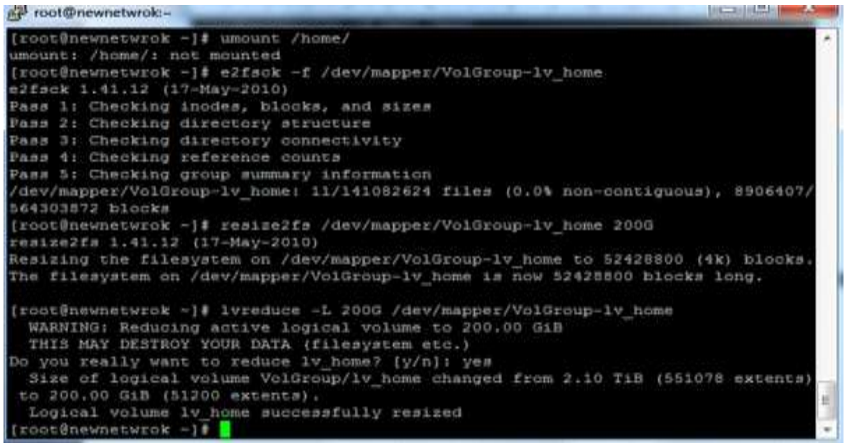


```
[root@newcontroller ~]# lvcreate --size 290G --name lvm1 VolGroup
  Logical volume "lvm1" created
[root@newcontroller ~]# mkfs.xfs -f -i size=512 /dev/mapper/VolGroup-lvm1
meta-data=/dev/mapper/VolGroup-lvm1 isize=512    agcount=4, agsize=19005440 blks
         =                       sectsz=512   attr=2, projid32bit=0
data     =                       bsize=4096   blocks=76021760, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0
log      =internal log           bsize=4096   blocks=37120, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
[root@newcontroller ~]# echo "/dev/mapper/VolGroup-lvm1 /data1 xfs defaults 1 2"
 >> /etc/fstab
[root@newcontroller ~]# mkdir /data1
[root@newcontroller ~]# mount -a
[root@newcontroller ~]#
```

Fig -6: Displaying output of creation of new partition on Controller

After creating a partition lvm1, a mount point is created in order to bring storage in use, this is achieved via xfs(Extended File system).

Step 6: Reducing /home partition on Network Node[9]
[root@network]#umount /home/
[root@network]#e2fsck -f /dev/mapper/VolGroup-lv_home
[root@network]#resize2fs /dev/mapper/VolGroup-lv_home 200G
[root@network]#lvreduce -L 200G /dev/mapper/VolGroup-lv_home

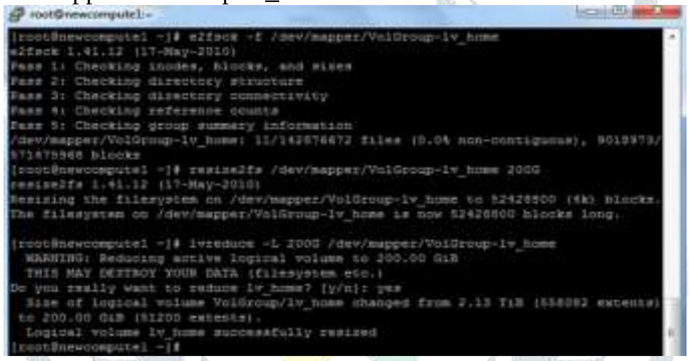


```
[root@newnetwrok ~]# umount /home/
umount: /home/: not mounted
[root@newnetwrok ~]# e2fsck -f /dev/mapper/VolGroup-lv_home
e2fsck 1.41.12 (17-May-2010)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/dev/mapper/VolGroup-lv_home: 11/141082624 files (0.0% non-contiguous), 8906407/
564303872 blocks
[root@newnetwrok ~]# resize2fs /dev/mapper/VolGroup-lv_home 200G
resize2fs 1.41.12 (17-May-2010)
Resizing the filesystem on /dev/mapper/VolGroup-lv_home to 52428800 (4k) blocks.
The filesystem on /dev/mapper/VolGroup-lv_home is now 52428800 blocks long.

[root@newnetwrok ~]# lvreduce -L 200G /dev/mapper/VolGroup-lv_home
  WARNING: Reducing active logical volume to 200.00 GiB
  THIS MAY DESTROY YOUR DATA (filesystem etc.)
Do you really want to reduce lv_home? [y/n]: yes
  Size of logical volume VolGroup/lv_home changed from 2.10 TiB (551078 extents)
 to 200.00 GiB (51200 extents).
  Logical volume lv_home successfully resized
[root@newnetwrok ~]#
```

Fig -7: Displaying the output of reducing the home partition on Network node

Creating new partition with 1800G space on network (LVM)
[root@network]#lvcreate --size 1800G --name lvm1 VolGroup
[root@network]#mkfs.xfs -f -i size=512 /dev/mapper/VolGroup-lvm1
[root@network]#echo "/dev/mapper/VolGroup-lvm1 /data1 xfs defaults 1 2">> /etc/fstab
[root@network]#mkdir /data1
[root@network]#mount -a

Step 7: Reducing /home partition on Compute Node[9]
[root@compute]#umount /home/
[root@compute]# e2fsck -f /dev/mapper/VolGroup-lv_home
[root@compute]# resize2fs /dev/mapper/VolGroup-lv_home 200G
[root@compute]#lvreduce -L 200G /dev/mapper/VolGroup-lv_home



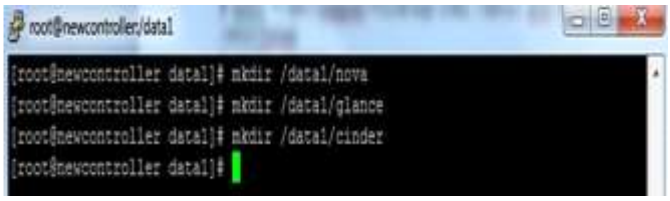
```
[root@newcompute1 ~]# e2fsck -f /dev/mapper/VolGroup-lv_home
e2fsck 1.41.12 (17-May-2010)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/dev/mapper/VolGroup-lv_home: 11/142876672 files (0.0% non-contiguous), 9018973/
571475968 blocks
[root@newcompute1 ~]# resize2fs /dev/mapper/VolGroup-lv_home 200G
resize2fs 1.41.12 (17-May-2010)
Resizing the filesystem on /dev/mapper/VolGroup-lv_home to 52428800 (4k) blocks.
The filesystem on /dev/mapper/VolGroup-lv_home is now 52428800 blocks long.

[root@newcompute1 ~]# lvreduce -L 200G /dev/mapper/VolGroup-lv_home
  WARNING: Reducing active logical volume to 200.00 GiB
  THIS MAY DESTROY YOUR DATA (filesystem etc.)
Do you really want to reduce lv_home? [y/n]: yes
  Size of logical volume VolGroup/lv_home changed from 2.13 TiB (558082 extents)
 to 200.00 GiB (51200 extents).
  Logical volume lv_home successfully resized
[root@newcompute1 ~]#
```

Fig -8: Displaying the output of reducing the home partition on Compute node

Creating new partition with 1800G space on Compute (LVM)
[root@compute]#lvcreate --size 1800G --name lvm1 VolGroup
[root@compute]#mkfs.xfs -f -i size=512 /dev/mapper/VolGroup-lvm1
[root@compute]#echo "/dev/mapper/VolGroup-lvm1 /data1 xfs defaults 1 2">> /etc/fstab
[root@compute]#mkdir /data1
[root@compute]#mount -a

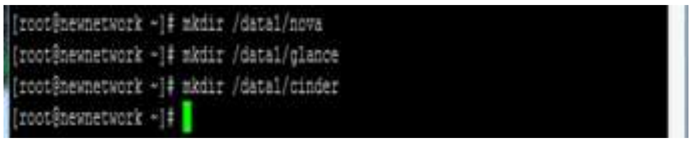
Step 8: On controller node add the peers
In order to reflect changes of GlusterFS restart the gluster service.
To add a server to a trusted storage pool, run gluster peer probe followed by the hostname or IP of the server.
To add any new server to existing Gluster, peer the new server from original server(here original server is controller node)
[root@controller]# service glusterd restart
[root@controller]# gluster peer probe <Ip Address of node>[10]
[root@controller]# gluster peer probe 192.168.10.4

Step 9: Create bricks on controller, network and compute
Bricks are created on controller i.e. data1/nova etc.
[root@controller]#mkdir /data1/nova
[root@controller]#mkdir /data1/glance
[root@controller]#mkdir /data1/cinder



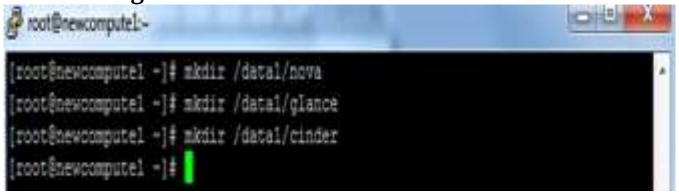
```
[root@newcontroller data1]# mkdir /data1/nova
[root@newcontroller data1]# mkdir /data1/glance
[root@newcontroller data1]# mkdir /data1/cinder
[root@newcontroller data1]#
```

Fig -9: creation of bricks on Controller node



```
[root@newnetwork ~]# mkdir /data1/nova
[root@newnetwork ~]# mkdir /data1/glance
[root@newnetwork ~]# mkdir /data1/cinder
[root@newnetwork ~]#
```

Fig -10: creation of bricks on Network node



root@newcompute1:~

```
[root@newcompute1 ~]# mkdir /data1/nova
[root@newcompute1 ~]# mkdir /data1/glance
[root@newcompute1 ~]# mkdir /data1/cinder
[root@newcompute1 ~]#
```

Fig -11: creation of bricks on Compute node

Step 10: Configure your GlusterFS volume on Controller Node
While creating a consolidated volume for OpenStack services, one particular brick from each node was selected and combined to that of other nodes in order to prepare storage for a particular OpenStack service. For example data1/nova from controller, network and compute was taken together under one volume for nova OpenStack service.
[root@controller]# gluster volume create NOVA controller:/data1/nova compute:/data1/nova network:/data1/nova [10]
[root@controller]# gluster volume create CINDER controller:/data1/cinder compute:/data1/cinder network:/data1/cinder [10]
[root@controller]# gluster volume create GLANCE controller:/data1/glance compute:/data1/glance network:/data1/glance [10]
To add new bricks to the volume
# gluster volume add-brick <volume name> <new brick name>

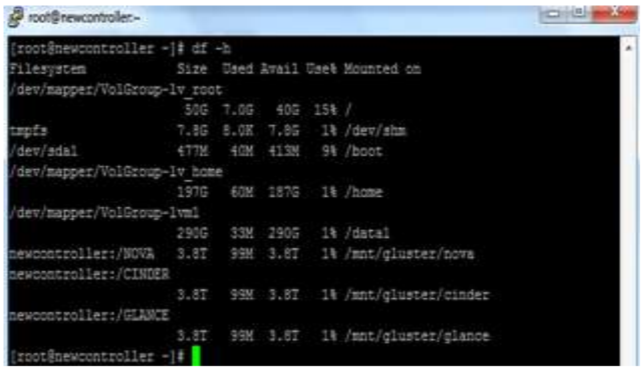
Step 11: Giving IP access on each volume
This is to define accessibility.
[root@controller]#gluster volume set NOVA auth.allow 192.168.*,10.0.0.*
[root@controller]#gluster volume set CINDER auth.allow 192.168.*,10.0.0.*
[root@controller]# gluster volume set GLANCE auth.allow 192.168.*,10.0.0.*

Step 12: Start Volume
Start the volume, in order to make it usable in cloud.
[root@controller]#gluster volume start CINDER
[root@controller]#gluster volume start NOVA
[root@controller]#gluster volume start GLANCE

Step 13: Creating gluster mount points On controller Node
Mount these consolidated volumes on controller for particular OpenStack services.
[root@controller]#mount -t GlusterFScontroller:/NOVA /mnt/gluster/nova/
[root@controller]#mount -t GlusterFScontroller:/GLANCE /mnt/gluster/glance/
[root@controller]#mount -t GlusterFScontroller:/CINDER /mnt/gluster/cinder/

Step 14: Creating gluster mount points On compute Node
[root@compute]#mkdir -p /mnt/gluster/nova
[root@compute]# mount -t GlusterFS compute:/NOVA /mnt/gluster/nova/

Step 15: Checking the mount points
We have checked whether gluster has been properly created
[root@controller]#df –h

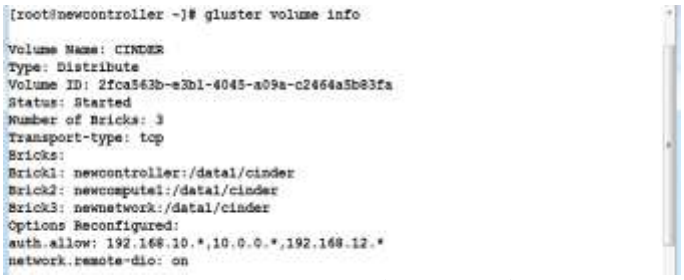


root@newcontroller:~

```
[root@newcontroller ~]# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/mapper/VolGroup-lv_root
                       50G  7.0G   40G  15% /
tmpfs                 7.8G  8.0K  7.8G   1% /dev/shm
/dev/sda1             477M   40M  413M   9% /boot
/dev/mapper/VolGroup-lv_home
                      197G   60M  187G   1% /home
/dev/mapper/VolGroup-lvm1
                      290G   33M  290G   1% /data1
newcontroller:/NOVA   3.8T   99M  3.8T   1% /mnt/gluster/nova
newcontroller:/CINDER
                      3.8T   99M  3.8T   1% /mnt/gluster/cinder
newcontroller:/GLANCE
                      3.8T   99M  3.8T   1% /mnt/gluster/glance
[root@newcontroller ~]#
```

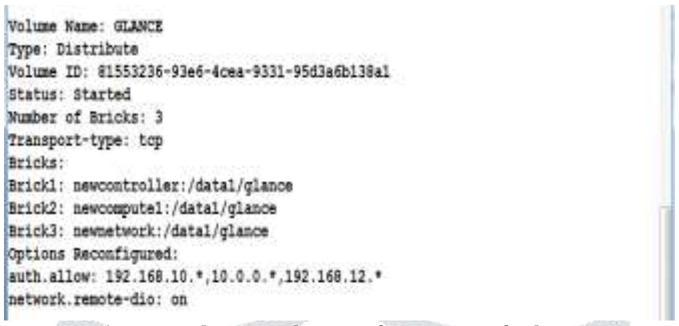
Fig -12: Display the mount points of all Volumes

To know the gluster information, run 'gluster volumes info' command, which gives all the information about all volumes which were created. It includes Volume name, status of volume, number of bricks mounted etc. The following image describes the output of the gluster volume info command. It shows all volumes which we have created above.



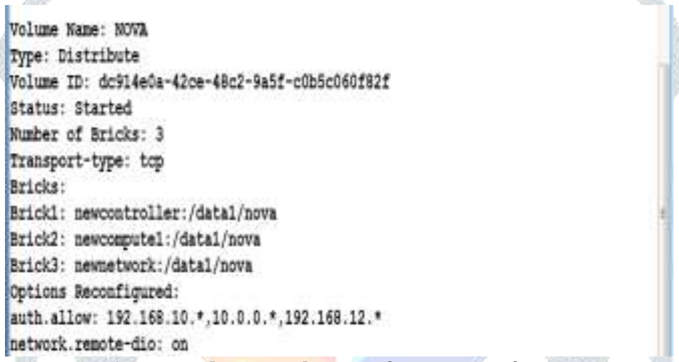
```
[root@newcontroller ~]# gluster volume info

Volume Name: CINDER
Type: Distribute
Volume ID: 2fca563b-e3b1-4045-a09a-c2464a5b83fa
Status: Started
Number of Bricks: 3
Transport-type: tcp
Bricks:
Brick1: newcontroller:/data1/cinder
Brick2: newcompute1:/data1/cinder
Brick3: newnetwork:/data1/cinder
Options Reconfigured:
auth.allow: 192.168.10.*,10.0.0.*,192.168.12.*
network.remote-dio: on
```

Fig -13: Gluster volume information of Cinder



```
Volume Name: GLANCE
Type: Distribute
Volume ID: 81553236-93e6-4cea-9331-95d3a6b138a1
Status: Started
Number of Bricks: 3
Transport-type: tcp
Bricks:
Brick1: newcontroller:/data1/glance
Brick2: newcompute1:/data1/glance
Brick3: newnetwork:/data1/glance
Options Reconfigured:
auth.allow: 192.168.10.*,10.0.0.*,192.168.12.*
network.remote-dio: on
```

Fig -14: Gluster volume information of Glance



```
Volume Name: NOVA
Type: Distribute
Volume ID: dc914e0a-42ce-48c2-9a5f-c0b5c060f82f
Status: Started
Number of Bricks: 3
Transport-type: tcp
Bricks:
Brick1: newcontroller:/data1/nova
Brick2: newcompute1:/data1/nova
Brick3: newnetwork:/data1/nova
Options Reconfigured:
auth.allow: 192.168.10.*,10.0.0.*,192.168.12.*
network.remote-dio: on
```

Fig -15: Gluster volume information of Nova

Configuring cloud Services with GlusterFS Storage

While implementing nova or any OpenStack service it is required to inform from where that service will take its storage. For example, we need to tell cinder from where it will fetch its storage. This would render use of GlusterFS volumes created above for IAAS cloud. Now instead of hitting one single storage each service would refer to its own.

Step 16:Configure Glance storage directory in /etc/glance/glance-api.conf on Controller Node[6]
[root@controller]#vim /etc/glance/glance-api.conf
filesystem_store_datadir = /mnt/gluster/glance/images [Directory that the Filesystem backend store]

[root@controller]#mkdir -p /mnt/gluster/glance/images
[root@controller]#chown -R glance.glance /mnt/gluster/glance/
[root@controller]#serviceOpenStack-glance-api restart
Whatever images are created in OpenStack will be stored in /mnt/gluster/images directory.

Step 17:Configure Cinder storage on Controller Node[6]

[root@controller]#vim /etc/cinder/cinder.conf

GlusterFS_shares_config=/etc/cinder/shares.conf
volume_driver=cinder.volume.drivers.GlusterFS.GlusterFSDriver
GlusterFS_mount_point_base=/mnt/gluster/cinder/volumes

Base directory contains mount points for gluster sharing and volumes that we have created are stored in /mnt/gluster/cinder directory.

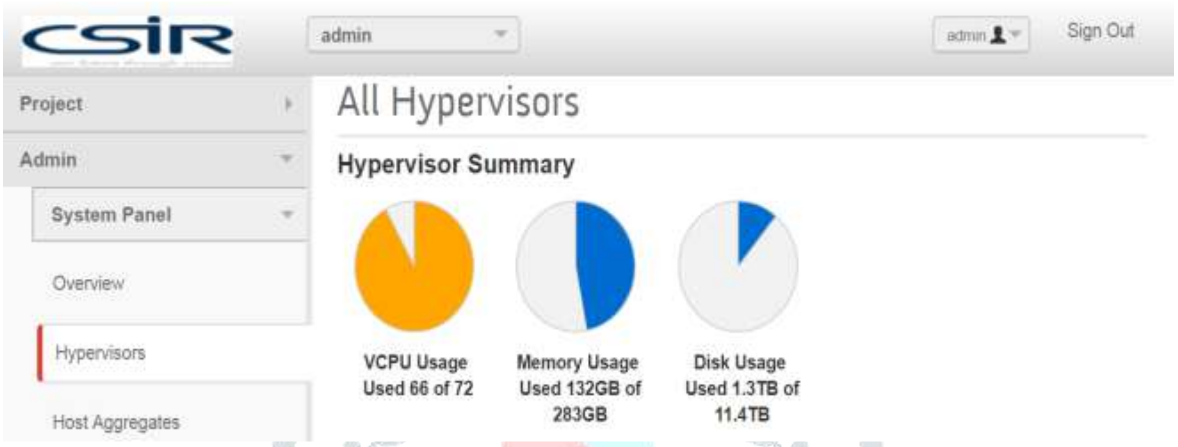
Edit /etc/cinder/shares.conf file
[root@controller]#vim /etc/cinder/shares.conf
192.168.10.3:/CINDER
[root@controller]#mkdir /mnt/gluster/cinder/volumes
[root@controller]#chown -R cinder.cinder /mnt/gluster/cinder
[root@controller]#serviceOpenStack-cinder-api restart

[root@controller]#serviceOpenStack-cinder-backup restart
[root@controller]#serviceOpenStack-cinder-scheduler restart
[root@controller]#serviceOpenStack-cinder-volume restart
Step 18:Configure Nova directory in /etc/nova/nova.conf on controller and Compute[6]
[root@controller]#vim /etc/nova/nova.conf
state_path = /mnt/gluster/nova [Top-level directory for maintaining nova's state]
instances_path = $state_path/instances
[root@controller]#mkdir /mnt/gluster/nova/instances/
[root@controller]#chown -R nova.nova /mnt/gluster/nova

All the instances of cloud are stored in /mnt/gluster/nova/instances

V. Storage Configuration of GlusterFS in OpenStack Cloud

Below is a snapshot of cloud dashboard displaying storage configuration details. It depicts aggregate storage which is available for cloud instances which is obtained from various distributed compute nodes configured under GlusterFS. The storage is expendable by deploying additional compute nodes.



All Hypervisors

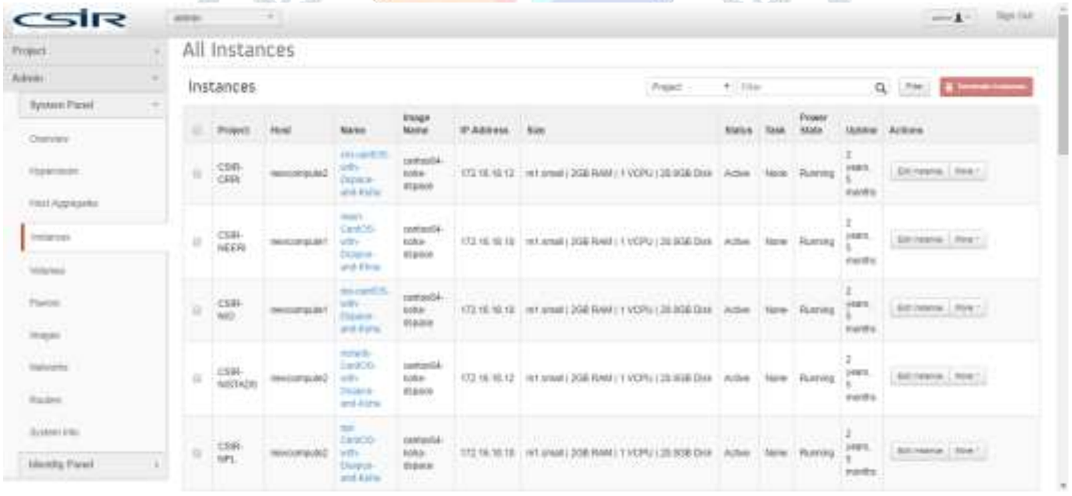
Hypervisor Summary

VCPU Usage
Used 66 of 72

Memory Usage
Used 132GB of 283GB

Disk Usage
Used 1.3TB of 11.4TB

Fig -16: The sample storage available for CSIR Private Cloud



All Instances

Instances

	Project	Host	Name	Image Name	IP Address	Size	Status	Task	Power State	Uptime	Actions
	CSIR-CRRI	newcompute2	[illegible]	centos64-koha-dspace	172.16.10.12	m1.small | 2GB RAM | 1 VCPU | 20.0GB Disk	Active	None	Running	2 years, 5 months	Edit Instance More
	CSIR-NEERI	newcompute1	[illegible] CentOS-with-Dspace-and-Koha	centos64-koha-dspace	172.16.10.18	m1.small | 2GB RAM | 1 VCPU | 20.0GB Disk	Active	None	Running	2 years, 5 months	Edit Instance More
	CSIR-NIO	newcompute1	[illegible]	centos64-koha-dspace	172.16.10.18	m1.small | 2GB RAM | 1 VCPU | 20.0GB Disk	Active	None	Running	2 years, 5 months	Edit Instance More
	CSIR-NISTADS	newcompute2	[illegible] CentOS-with-Dspace-and-Koha	centos64-koha-dspace	172.16.10.12	m1.small | 2GB RAM | 1 VCPU | 20.0GB Disk	Active	None	Running	2 years, 5 months	Edit Instance More
	CSIR-NPL	newcompute2	[illegible] CentOS-with-Dspace-and-Koha	centos64-koha-dspace	172.16.10.18	m1.small | 2GB RAM | 1 VCPU | 20.0GB Disk	Active	None	Running	2 years, 5 months	Edit Instance More

Fig -17: The sample list of Virtual Machines/ instances running under CSIR Private Cloud

Fig. 17 shows how the aggregate storage has been utilized by various virtual machines(instances). The configuration of virtual machines include no. of CPUs, RAM & Storage etc.

VI. Conclusion

Though cloud has emerged as a forceful paradigm for managing and delivering services over Internet, but key component that cloud provides is large storage, thus to manage this large storage GlusterFS turns out to be of key importance. The Storage requirement for CSIR private cloud has been successfully met with GlusterFS and is running for more than two years.

In this paper we have gone through all the steps which are required to get from bare hardware to a fully functional GlusterFS, in order to scale storage and use it as a one single volume for entire IaaS implementation. Apart from cloud, GlusterFS can even be used for content delivery, media streaming and for basic purpose i.e. to scale storage for general use by creating a pool from those systems whose storage is not in use within an organization. Thus optimizing storage utilization.

References

[1] Junghan Kim, Inhyuk Kim, Taehyoung Kim,Young Ik Eom, Hong-Yeon kim, Youngkyun Kim "Design and Implementation of Networking Virtualization for Cluster File Systems", 2009 International Conference on Computational Science and Its Applications. IEEE, 2009.

[2] Pillai. Sarath (Published, July 21, 2013). Retrieved from https://www.slashroot.in/gfs-gluster-file-system-complete-tutorial-guide-for-an-administrator [Accessed May 2018]

[3] https://docs.gluster.org/

[4] Alex Davies, Alessandro Orsaria. "Scale Out with GlusterFS", Linux Journal Volume 2013 Issue 253 November 2013, ACM 2013.

[5] GitHub, Retrieved from https://gluster.readthedocs.io/en/latest/Administrator%20Guide/Setting%20Up%20Volumes/[Accessed July 2018]

[6] OpenStack Foundation. https://docs.OpenStack.org [Accessed June 2018]

[7] Oracle and its affiliates. Retrieved from, https://docs.oracle.com/cd/E36784_01/html/E54155/archover.html [Accessed May 2018]

[8] GitHub, Retrieved from https://github.com/libfuse/libfuse [Accessed July 2018]

[9] Kumar.Pradeep. Linux Tutorials and Guides. Retrieved from https:// https://www.linuxtechi.com/reduce-size-lvm-partition/ [Accessed June 2018]

[10] Divya Muntimadugu, Bhavana Mohanraj, Anjana Suparna Sriram. Configuring and Managing Red Hat Storage Server. Retrieved from https://access.redhat.com/documentation/en-US/Red_Hat_Storage/2.1/html/Administration_Guide/ [Accessed July 2018]

[11] CSIR-NISCAIR. Retrieved from, http://knowgate.niscair.res.in/[Accessed June 2018]