NFS-Ganesha and Clustered NAS on Distributed Storage System, GlusterFS

Soumya Koduri
Meghana Madhusudhan
Red Hat
AGENDA

- NFS(-Ganesha)
- Distributed storage system - GlusterFS
- Integration
- Clustered NFS
- Future Directions
- Step-by-step guide
- Q&A
NFS
NFS

- Widely used network protocol
- Many enterprises still heavily depend on NFS to access their data from different operating systems and applications.

Versions:

- Stateless NFSv2 [RFC 1094] & NFSv3 [RFC 1813]
  - Side-band protocols (NLM/NSM, RQUOTA, MOUNT)
- Stateful NFSv4.0 [RFC 3530] & NFSv4.1/pNFS [RFC 5661]
  - NFSv4.2 protocol being developed
NFS-Ganesha
NFS-Ganesha

➢ A user-space, protocol-complaint NFS file server
➢ Supports NFS v3, 4.0, 4.1, pNFS and 9P from the Plan9 operating system.
➢ Provides a FUSE-compatible File System Abstraction Layer (FSAL) to plug in to any own storage mechanism
➢ Can provide simultaneous access to multiple file systems.

Active participants:
➢ CEA, Panasas, Red Hat, IBM, LinuxBox
Benefits of NFS-Ganesha

➢ Dynamically export/unexport entries using D-Bus mechanism.
➢ Can manage huge meta-data and data caches
➢ Can act as proxy server for NFSv4
➢ Provides better security and authentication mechanism for enterprise use
➢ Portable to any Unix-like file-systems
➢ Easy access to the services operating in the user-space (like Kerberos, NIS, LDAP)
Modular Architecture

- **RPC Layer**: implements ONC/RPCv2 and RPCSEC_GSS (based on libntirpc)

- **FSAL**: File System Abstraction Layer, provides an API to generically address the exported namespace

- **Cache Inode**: manages the metadata cache for FSAL. It is designed to scale to millions of entries

- **FSAL UP**: provides the daemon with a way to be notified by the FSAL that changes have been made to the underlying FS outside Ganesha. These information is used to invalidate or update the Cache Inode.
NFS-Ganesha Architecture

Network Forechannel
- RPC Dispatcher
- Dup Req
- RPC Sec GSS
- NFSv3, NFSv4.x/pNFS, RQUOTA, 9P

Cache Inode

SAL

FSAL

FSAL_UP

Backend (POSIX, VFS, ZFS, GLUSTER, GPFS, LUSTRE)

Network Backchannel

Admin DBUS
Distributed storage - GlusterFS
GlusterFS

➢ An open source, scale-out distributed file system
➢ Software Only and operates in user-space
➢ Aggregates Storage into a single unified namespace
➢ No metadata server architecture
➢ Provides a modular, stackable design
➢ Runs on commodity hardware
Architecture

- Data is stored on disk using native formats (e.g. ext4, XFS)
- Has client and server components
  - Servers, known as storage bricks (glusterfsd daemon), export local filesystem as volume
  - Clients (glusterfs process), creates composite virtual volumes from multiple remote servers using stackable translators
  - Management service (glusterd daemon) manages volumes and cluster membership
Terminologies

➢ **Trusted Storage Pool**: A storage pool is a trusted network of storage servers.

➢ **Brick**: Brick is the basic unit of storage, represented by an export directory on a server in the trusted storage pool.

➢ **Volume**: A volume is a logical collection of bricks. Most of the gluster management operations happen on the volume.
Best Fit and Optimal Workloads:

- Large File & Object Store (using either NFS, SMB or FUSE client)
- Enterprise NAS dropbox & object Store / Cloud Storage for service providers
- Cold Storage for Splunk Analytics Workloads
- Hadoop Compatible File System for running Hadoop Analytics
- Live virtual machine image store for Red Hat Enterprise Virtualization
- Disaster Recovery using Geo-replication
- ownCloud File Sync n' Share

Not recommended

- Highly transactional like a database
- Workloads that involve a lot of directory based operations
GlusterFS Deployment

Multiprotocol Client Support
- GlusterFS Native
- NFS
- CIFS
- HTTP
- WebDAV
- FTP

Flexible Back-end

Clustered Storage Building Blocks
Integration with GlusterFS
libgfapi

➢ A user-space library with APIs for accessing Gluster volumes.

➢ Reduces context switches.

➢ Many applications integrated with libgfapi (qemu, samba, NFS Ganesha).

➢ Both sync and async interfaces available.

➢ C and python bindings.

➢ Available via 'glusterfs-api*' packages.
NFS-Ganesha + GlusterFS

NFS-Ganesha

<table>
<thead>
<tr>
<th>Cache Inode</th>
<th>SAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>FSAL_GLUSTER</td>
<td></td>
</tr>
</tbody>
</table>

libgfapi

GlusterFS Volume

GlusterFS Brick

GlusterFS Brick

NFS-Ganesha

GlusterFS

NFS-Ganesha + GlusterFS
Integration with GlusterFS

➢ Integrated with GlusterFS using 'libgfapi' library

That means,

• Additional protocol support w.r.t. NFSv4, pNFS
• Better security and authentication mechanisms for enterprise use.
• Performance improvement with additional caching
Clustered NFS
Clustered NFS

➢ Stand-alone systems:
  • are always bottleneck.
  • cannot scale along with the back-end storage system.
  • not suitable for mission-critical services

➢ Clustering:
  • High availability
  • Load balancing
  • Different configurations:
    • Active-Active
    • Active-Passive
Server Reboot/Grace-period

➢ NFSv3:
  ➢ Stateless. Client retries requests till TCP retransmission timeout.
  ➢ NLM/NSM:
    ➢ NSM notifies the clients which reclaim lock requests during server's grace period.

➢ NFSv4.x:
  ➢ Stateful. Stores information about clients persistently.
  ➢ Reject client request with the errors NFS4ERR_STALE_STATEID / NFS4ERR_STALE_CLIENTID
  ➢ Client re-establishes identification and reclaims OPEN/LOCK state during grace period.
Challenges Involved

➢ Cluster wide change notifications for cache invalidations
➢ IP Failover in case of Node/service failure
➢ Coordinate Grace period across nodes in the cluster
➢ Provide “high availability” to stateful parts of NFS
  • Share state across the cluster
  • Allow state recovery post failover
Active-Active HA solution on GlusterFS

Primary Components

- Pacemaker
- Corosync
- PCS
- Resource agents
- HA setup scipt ('ganesha-ha.sh')
- Shared Storage Volume
- UPCALL infrastructure
Clustering Infrastructure

➢ Using Open-source services

➢ **Pacemaker**: Cluster resource manager that can start and stop resources

➢ **Corosync**: Messaging component which is responsible for communication and membership among the machines

➢ **PCS**: Cluster manager to easily manage the cluster settings on all nodes
Cluster Infrastructure

- **Resource-agents**: Scripts that know how to control various services.
  
  - New resource-agent scripts added to
    
    - **ganesha_mon**: Monitor NFS service on each node & failover the Virtual IP
    - **ganesha_grace**: Puts entire cluster to Grace using d-bus signal
  
  - If NFS service down on any of the nodes
    
    - Entire cluster is put into grace via D-bus signal
    - Virtual IP fails over to a different node (within the cluster).
HA setup script

- Located at `/usr/libexec/ganesha/ganesha-ha.sh`.
- Sets up, tears down and modifies the entire cluster.
- Creates resource-agents required to monitor NFS service and IP failover.
- Integrated with new Gluster CLI introduced to configure NFS-Ganesha.

**Primary Input:** `ganesha-ha.conf` file with the information about the servers to be added to the cluster along with Virtual IPs assigned, usually located at `/etc/ganesha`. 
Upcall infrastructure

➢ A generic and extensible framework.
  • used to maintain states in the glusterfsd process for each of the files accessed
  • sends notifications to the respective glusterfs clients in case of any change in that state.

➢ Cache-Invalidation: Needed by NFS-Ganesha to serve as Multi-Head

Config options:

#gluster vol set <volname> features.cache-invalidation on/off
#gluster vol set <volname> features.cache-invalidation-timeout <value>
Shared Storage Volume

➢ Provides storage to share the cluster state across the NFS servers in the cluster

➢ This state is used during failover for Lock recovery

➢ Can be created and mounted on all the nodes using the following gluster CLI command -

```bash
#gluster volume set all cluster.enable-shared-storage enable
```
Limitations

➢ Current maximum limit of nodes forming cluster is 16
➢ Heuristics for IP failover
➢ Clustered DRC is not yet supported
Clustered NFS-Ganesha

Shared Storage Volume

Node A
Node B
Node C
Node D

Clustering Infrastructure (Pacemaker/Corosync)

NFS-Ganesha service
Virtual IP
ganesha_mon
ganesha_grace
Clustered NFS-Ganesha

Shared Storage Volume

Clustering Infrastructure (Pacemaker/Corosync)

Node A  Node B  Node C  Node D

NFS-Ganesha service  Virtual IP  ganesha_mon  ganesha_grace

NFS Client
Clustered NFS-Ganesha

Shared Storage Volume

Clustering Infrastructure (Pacemaker/Corosync)

Node A
Node B
Node C
Node D

NFS-Ganesha service
Virtual IP
ganesha_mon
ganesha_grace

NFS Client
Clustered NFS-Ganesha

Shared Storage Volume

Clustering Infrastructure (Pacmaner/Corosync)

Node A

Node B

Node C

Node D

NFS-Ganesha service
Virtual IP

ganesha_mon

ganesha_grace

NFS Client

09/30/15
Clustered NFS-Ganesha

Shared Storage Volume

Clustering Infrastructure (Pacemaker/Corosync)

NFS-Ganesha service
Virtual IP
\texttt{ganesha\_mon}
\texttt{ganesha\_grace}

NFS Client

Node A
Node B
Node C
Node D

In Grace

Clustered NFS-Ganesha
Next
pNFS (Parallel Network File System)

➢ Introduced as part of NFSv4.1 standard protocol
➢ Needs a cluster consisting of M.D.S. (meta data server) and D.S. (Data server)
➢ Any filesystem can provide pNFS access via NFS-Ganesha by means of the FSAL easy plugin architecture
➢ Support for pNFS protocol ops added to FSAL_GLUSTER
➢ Currently supports only FILE LAYOUT
Future Directions

• NFSv4 paves the way forward for interesting stuff
• Adding NFSv4.x feature support for GlusterFS
  – Directory Delegations
  – Sessions
  – Server-side copy
  – Application I/O Advise (like posix_fadvise)
  – Sparse file support/Space reservation
  – ADB support
  – Security labels
  – Flex File Layouts in pNFS
Contact

Mailing lists:
- nfs-ganesha-devel@lists.sourceforge.net
- gluster-users@gluster.org
- gluster-devel@nongnu.org

IRC:
- #ganesha on freenode
- #gluster and #gluster-dev on freenode

Team: Apeksha, ansubram, jiffin, kkeithley, meghanam, ndevos, saurabh, skoduri
References & Links

Links (Home Page):

https://github.com/nfs-ganesha/nfs-ganesha/wiki
http://www.gluster.org

References:

http://gluster.readthedocs.org
http://blog.gluster.org/
http://www.snia.org/sites/default/files/Poornima_NFS_GaneshaForClustered NAS.pdf
http://clusterlabs.org/doc/
Q & A
BACKUP-
Step-by-step guide
Required Packages

Gluster RPMs (>= 3.7)
• glusterfs-server
• glusterfs-ganesha

Ganesha RPMs (>= 2.2)
• nfs-ganesha
• nfs-ganesha-gluster

Pacemaker & pcs RPMs
Pre-requisites

- Ensure all machines are DNS resolvable
- Disable and stop NetworkManager service, enable and start network service on all machines
- Enable IPv6 on all the cluster nodes.
- Install pacemaker pcs ccs resource-agents corosync
  - `#yum -y install pacemaker pcs ccs resource-agents corosync` on all machines
- Enable and start pcsd on all machines
  - `#chkconfig --add pcsd; chkconfig pcsd on; service pcsd start`
- Populate `/etc/ganesha/ganesha-ha.conf` on all the nodes.
Pre-requisites

- Create and mount the Gluster shared volume on all the machines
- Set cluster auth password on all machines
  
  ```
  #echo redhat | passwd --stdin hacluster
  #pcs cluster auth on all the nodes
  ```
- Passwordless ssh needs to be enabled on all the HA nodes.
  
  - On one (primary) node in the cluster, run:
    ```
    #ssh-keygen -f /var/lib/glusterd/nfs/secret.pem
    ```
  - Deploy the pubkey ~root/.ssh/authorized keys on _all_ nodes, run:
    ```
    #ssh-copy-id -i /var/lib/glusterd/nfs/secret.pem.pub root@$node
    ```
Sample 'ganesha-ha.conf'

# Name of the HA cluster created. must be unique within the subnet
HA_NAME="ganesha-ha-360"

# The gluster server from which to mount the shared data volume.
HA_VOL_SERVER="server1"

# The subset of nodes of the Gluster Trusted Pool that form the ganesha HA cluster.
# Hostname is specified.
HA_CLUSTER_NODES="server1,server2,..."

# Virtual IPs for each of the nodes specified above.
VIP_server1="10.0.2.1"
VIP_server2="10.0.2.2"
Setting up the Cluster

New CLIs introduced to configure and manage NFS-Ganesha cluster & Exports

#gluster nfs-ganesha <enable/disable>

- Disable Gluster-NFS
- Start/stop NFS-Ganesha services on the cluster nodes.
- Setup/teardown the NFS-Ganesha cluster.

#gluster vol set <volname> ganesha.enable on/off

- Creates export config file with default parameters
- Dynamically export/unexport volumes.
Modifying the Cluster

➢ Using HA script `ganesha-ha.sh` located at `/usr/libexec/ganesha`.

➢ Execute the following commands on any of the nodes in the existing NFS-Ganesha cluster

➢ To add a node to the cluster:

```bash
#./ganesha-ha.sh --add <HA_CONF_DIR> <HOSTNAME> <NODE-VIP>
```

➢ To delete a node from the cluster:

```bash
#./ganesha-ha.sh --delete <HA_CONF_DIR> <HOSTNAME>
```

Where,

- `HA_CONF_DIR`: The directory path containing the ganesha-ha.conf file.
- `HOSTNAME`: Hostname of the new node to be added
- `NODE-VIP`: Virtual IP of the new node to be added.
Modifying Export parameters

On any of the nodes in the existing ganesha cluster:

- Edit/add the required fields in the corresponding export file located at `/etc/ganesha/exports`.

- Execute the following command:

  ```
  ./ganesha-ha.sh --refresh-config <HA_CONFDIR> <Volname>
  ```

  Where,

  - **HA_CONFDIR**: The directory path containing the `ganesha-ha.conf` file
  - **Volname**: The name of the volume whose export configuration has to be changed.
Thank you!

Soumya Koduri
Meghana Madhusudhan