GlusterFS – a Scale-Out Data Platform

John Mark Walker
Gluster Community Guy
Topics

- What is GlusterFS
  - Humble beginnings
- Evolution
- Community Process
- GlusterFS 3.3
- The Future is Gluster
Simple Economics

- Simplicity, scalability, less cost

Virtualized
Multi-Tenant
Automated
Commoditized
Scale on Demand
In the Cloud
Scale Out
Open Source
Simplicity Bias

- FC, FCoE, iSCSI → HTTP, Sockets
- Modified BSD OS → Linux / User Space / C, Python & Java
- Appliance based → Application based
Scale-out Open Source is the winner
All work and no play?

Conference Room

Meeting Round 1

US Head Office

Bedroom

Bengaluru Office
Community Deployments
Not a Storage Company

• At first a cluster-building company
• Engineering team excelled at building open source HPC systems
Necessity:
The Mother of Invention
The big idea:
Storage should be simple
What is Simple Storage?

- Low-risk, easy to deploy and administer, data consistency, open source, software-only, user space
What is GlusterFS, Really?

Gluster is a unified, distributed storage system

• User space, global namespace, stackable, POSIX-y, scale-out NAS platform, inspired by GNU Hurd
Some Features

• No single point of failure
• DHT
• Synchronous and asynchronous replication
• Proactive self-healing
What Can You Do With It?

- **Media** – Docs, Photos, Video
- **Shared storage** – multi-tenant environments
- **Big Data** – Log Files, RFID Data
- **Objects** – Long Tail Data
Standard Deployment

- Distributed over multiple servers
- Replicate volumes
- On top of disk FS (XFS, Ext4, ie. Xattrs)
- Multi-protocol access
Storage for Any Environment
Scale-out NAS for On-premises and Public Clouds

- Standardized NAS infrastructure
- On-premise and public cloud
- POSIX-ish
- Apps move easily between environments
- Replicate between both
First Versions

- Toolkit for building storage systems
- Very hacker-friendly
- Community integral part of development
  - Drove feature development
  - Repeatable use cases
Mid-2011 Snapshot

- Scale-out NAS
- Distributed and replicated
- NFS, CIFS and native GlusterFS
- User-space, stackable architecture
- Lots of users, not many devs

→ A good platform to build on
GlusterFS 3.3: Building on the Foundation

- Granular locking
- Proactive self-healing
- Improved rebalancing
- More access methods
Granular Locking

- Server fails, comes back
- Files evaluated
- Block-by-block until healed
Proactive Self-healing

- Performed server-to-server
- Recovered node queries peers

Server 1 - good

/  Symlink 1
Hidden |  Symlink 2
\  Symlink 3

Server 2 - recovered

File 1
File 2
File 3

Server 3 - good

Server 4 - good

File 1
File 2
File 3

Distributed

Replicated

Self-healing
Easier Rebalancing

- Now faster
  - Previously, created entire new hash set, moving data unnecessarily
  - Now recreates hash map and compares to old
- Easier to decommission server nodes
- Proof point for synchronous translator API
Unified File and Object (UFO)

- S3, Swift-style object storage
- Access via UFO or Gluster mount
Unified File and Object (UFO)

- Your gateway to the cloud
- Your data, accessed your way
HDFS Compatibility

- Run MapReduce jobs on GlusterFS
- Add unstructured data to Hadoop
4. Coming Attractions
API Check

- Ways to interface with GlusterFS
  - Translators
    - Stackable, async and sync
  - FUSE mount
    - GlusterFS client
  - Libgfapi
    - FUSE bypass
API Check

- Ways to interface with GlusterFS
  - Marker framework
  - Geo-replication, quickly ID changes
  - UFO RESTful API
  - HDFS library
  - Management API
- oVirt 3.1
Better VM Image Handling

- Better responsiveness for random I/O use cases
- Contribution: Block Device Translator
Enabling GlusterFS for Virtualization use

- QEMU-GlusterFS integration
  - Native integration, no FUSE mount
  - Gluster as QEMU block back end
  - QEMU talks to gluster and gluster hides different image formats and storage types underneath
  - Block device support in GlusterFS via Block Device translator
- Logical volumes as VM images
GlusterFS & QEMU
Libglusterfs Client API

– Previously abandoned
– Brought back to life
  • In part because of QEMU Fuse bypass contributions
Multi-Master Geo Rep

- Async rep previously only master-slave
- Multi-master gives admins greater flexibility
- Cascading, > 2-way
Split Brain

- Nodes cannot see each other, but can all still write
- Often due to network outages
- Sometimes results in conflicts
- Up to 3.2, GlusterFS had no concept of “quorum”
Quorum Enforcement

- Which node has valid data?
- If quorum, keep writing, else stop
  • Configurable option

Server 1
- No quorum
- Stops writing

Server 2
- Quorum
- Keeps writing

Server 3
- Quorum
- Keeps writing

Broken Connection
Quorum Enforcement

- After connection restored, self-heal kicks off

Replica 1

- No quorum
- Stops writing

Replica 2

- Quorum
- Keeps writing

Replica 3

- Quorum
- Keeps writing

Self-heal
Enhanced Quorum

- Quorum tracking on the servers
- Need quorum for any management changes
- 3rd party arbiters / observers so never N=2
Management UI & REST API

- Collaboration with oVirt project
- Management GUI for admins
- RESTful gateway for devs
- First community release... ?
### System

#### Clusters
- Default
  - Servers
  - Volumes
- data
- Servers
  - server1
  - server2
- Volumes
  - music
  - video

#### Volumes

<table>
<thead>
<tr>
<th>Name</th>
<th>Volume Type</th>
<th>Number of Bricks</th>
<th>Transport Type</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>music</td>
<td>Distribute</td>
<td>2</td>
<td>TCP</td>
<td>Up</td>
</tr>
<tr>
<td>video</td>
<td>Replicate</td>
<td>2</td>
<td>TCP</td>
<td>Up</td>
</tr>
</tbody>
</table>

#### Bricks

<table>
<thead>
<tr>
<th>Server</th>
<th>Brick Directory</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.16.159.159</td>
<td>/tmp/music-brick1</td>
<td>Up</td>
</tr>
<tr>
<td>10.16.159.161</td>
<td>/tmp/music-brick2</td>
<td>Up</td>
</tr>
</tbody>
</table>
Multi-tenancy & Encryption

- HekaFS created this for cloud deployments
- Being added to master branch
Down the Road

- Snapshots
- Versioning
- GeoRep Sparse Replicas
- File compression & de-dupe
Server-side Processing

- Implementing gfind, glocate
- Fast traversal of metadata in xattrs
  - Find and locate responsive
- Inotify-esque behavior: triggers based on i/o activity, ie. file close
  - Why rely on Hadoop batch-processing?
Goals

- Gluster is the standard platform for big data and cloud computing
  - Integration with every major big data, cloud and storage technology.
  - Signifies distributed data workloads
  - Encompasses storage and big data spheres
- In 2012, GlusterFS will be the “Foundation for Big Data”
Goal: Intelligent Storage

- Just storing and retrieving data is not enough
- Should be able to store, analyze, transform, mutilate, and retrieve
- Intelligent storage gives sysadmins and developers the ultimate data swiss army knife
Thank you!

John Mark Walker
Gluster Community Guy
johnmark@redhat.com