Storage as a Service with Gluster

- Vijay Bellur
GlusterFS co-maintainer
Red Hat

Linux Foundation Vault 2016

Credits

Some slides/content borrowed & stolen from:

Atin Mukherjee

Jeff Darcy

Kaleb Keithley

Luis Pabon

Prasanna Kalever

Agenda

- Overview
  - Storage as a Service (StaaS)
  - Gluster
- Challenges
- Gluster for StaaS
- Use Cases
- Q&A

Storage as a Service

“Bi-modal IT”

Two modes of IT operations:

- Mode 1 is traditional - slow, emphasises safety and accuracy.
- Mode 2 is exploratory - fast moving, emphasizes agility and speed.

Storage in Mode1

- Traditional way of doing storage
- Administrator driven provisioning, management, tuning & monitoring
- Automation - helpful but not essential
- Mid-long term retention of provisioned storage

Storage in mode2

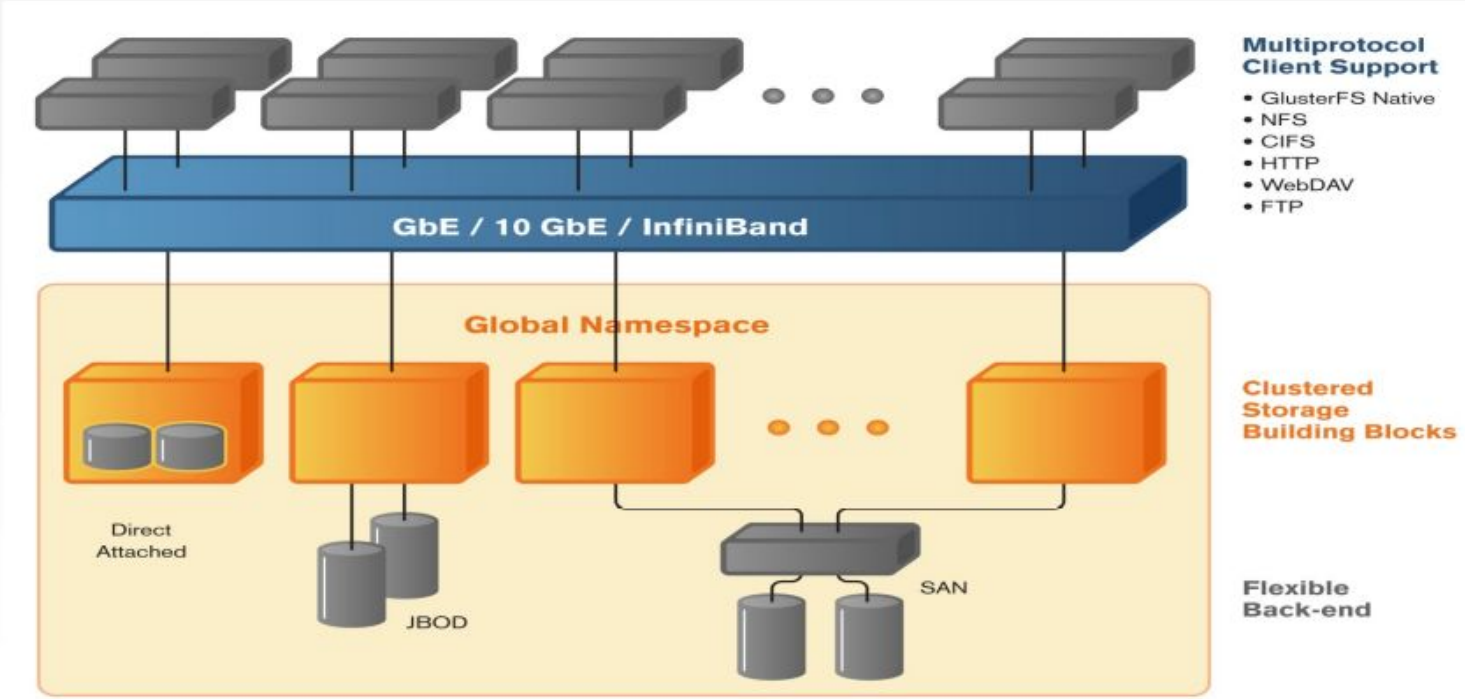
- New way of doing storage
- Devops driven provisioning, management & monitoring
- Automate or perish
- Typically short term retention of provisioned storage
- Storage in mode2 = “Storage as a Service”

Gluster

Gluster

- Scale-out distributed storage system
- Modular and extensible architecture
- **File**, Object and Block interfaces
- Layered on disk file systems that support extended attributes

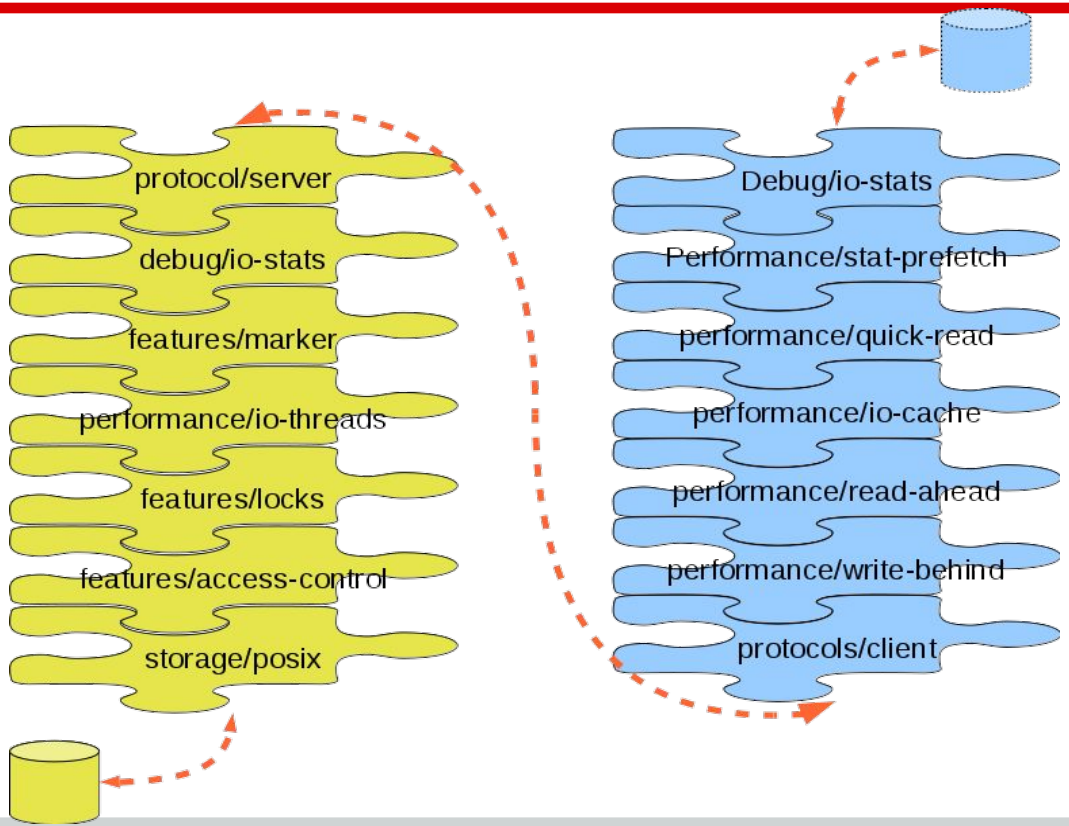
Typical Gluster Deployment



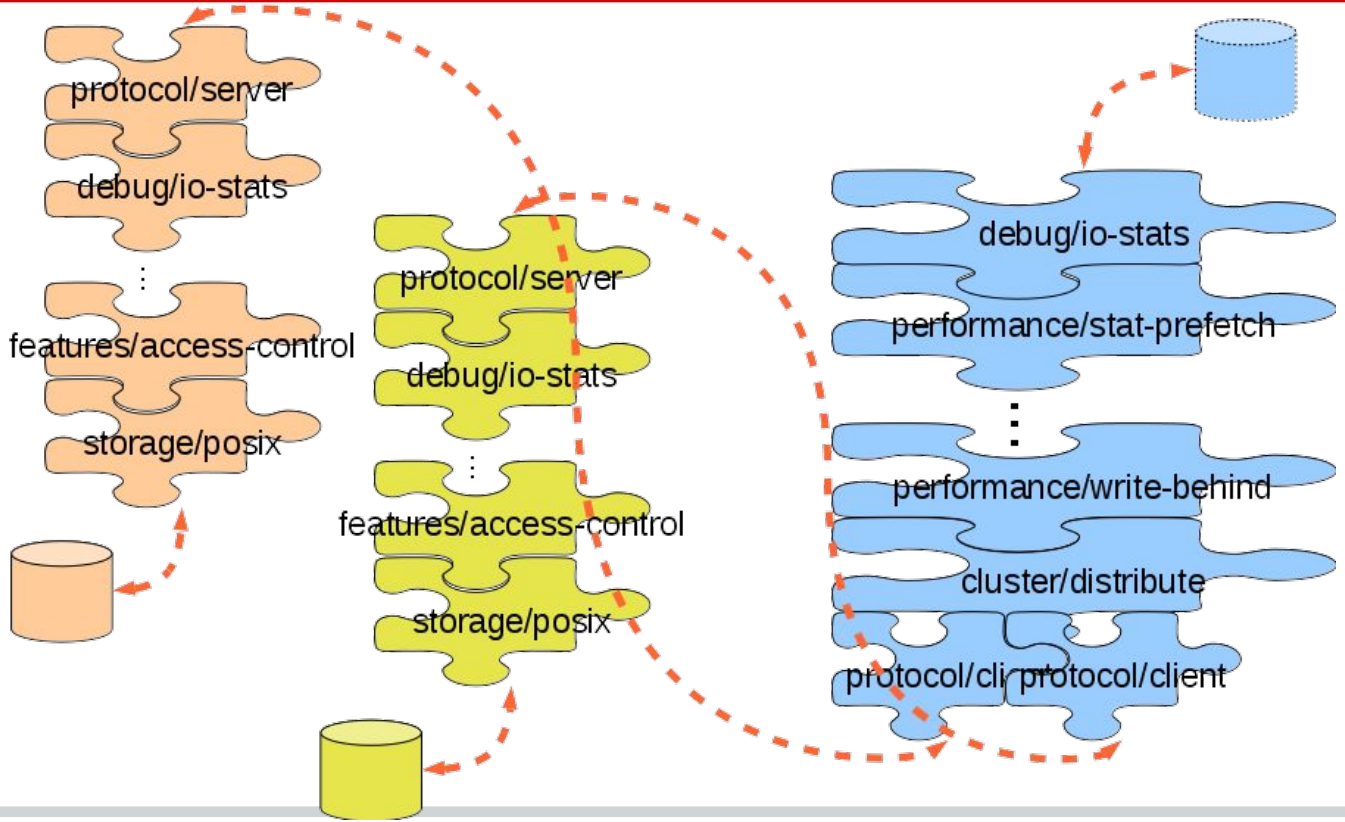
Gluster - Terminology

- Trusted Storage Pool - Collection of Storage Servers
- Brick - An export directory on a server
- Volume - Logical collection of bricks
- Sub-volume - part of a volume
- Translator - Stackable shared library performing one function

Gluster Translator Stack - Simple



Gluster Translator Stack - not so simple!



Challenges with StaaS

- Provisioning
  - Cannot wait for hours or days to service requests
- Scale
  - Economies of Scale
- Diverse workloads
- Noisy neighbor

Gluster - StaaS: Getting Ready

- gdeploy
- Heketi
- Containerized Gluster
- Gluster.Next

Gluster - StaaS: gdeploy

- Deploys Gluster using Ansible
- Prepares nodes to be added to Trusted Storage Pool
- More administrative procedures planned to be automated
    - Server lifecycle management

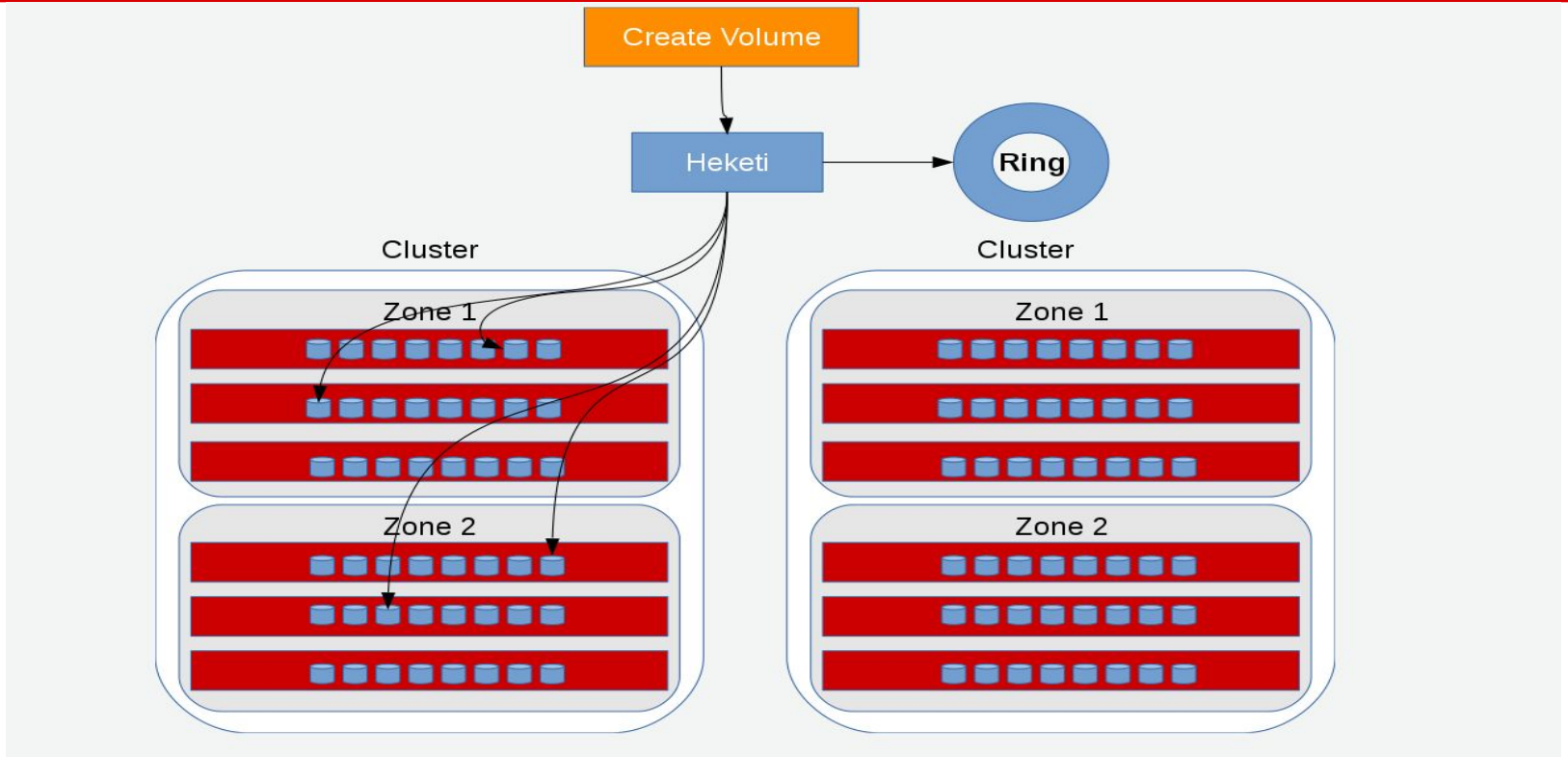
Gluster - StaaS: gdeploy

```
/ijay@deepthought:/tmp/gdeploy/playbooks$ ls
add-remote-file.yml              firewalld-ports-op.yml       georep-session-resume.yml      gluster-remove-brick.yml          lvconvert.yml                        setup-backend-and-deploy-gluster.yml
auto_lvcreate_for_gluster.yml    firewalld-service-op.yml     georep-session-start.yml       gluster-shared-volume-mount.yml   lvcreate.yml                         setup-backend.yml
backend-reset.yml                fscreate.yml                 georep-session-stop.yml        gluster-snapshot-activate.yml     lvremove.yml                         setup_ctdb.yml
bootstrap-nfs-ganesha.yml        ganesha-cluster-add.yml      georep-set-pemkeys.yml         gluster-snapshot-clone.yml        mount-in-samba-server.yml            setup.yml
cache_setup.yml                  ganesha-cluster-delete.yml   gluster-add-brick.yml          gluster-snapshot-config.yml       mount.yml                            shell_cmd.yml
check_package_installs_ganesha.yml  ganesha-conf-create.yml  gluster-client-cifs-mount.yml  gluster-snapshot-create.yml       move-file-from-local-to-remote.yml   sm_register.yml
chkconfig_service.yml            ganesha-ha.conf              gluster-client-fuse-mount.yml  gluster-snapshot-deactivate.yml   pcs-authentication.yml               snapshot-setup.yml
client_volume_umount.yml         ganesha-setup.yml            gluster-client-nfs-mount.yml   gluster-snapshot-delete.yml       pvcreate.yml                         subscription_manager.yml
configure-services.yml           ganesha-volume-configs.yml   gluster-peer-detach.yml        gluster-snapshot-restore.yml      pvremove.yml                         tear-down-ha-cluster.yml
copy-ssh-key.yml                 generate-public-key.yml      gluster-peer-probe.yml         gluster-volume-create.yml         pvresize.yml                         thin_lvcreate.yml
create-brick-dirs.yml            georep_common_public_key.yml  gluster-quota-disable.yml     gluster-volume-delete.yml         README.md                            tune-profile.yml
create-mount-points.yml          georep-fail-back.yml         gluster-quota-dsl.yml          gluster-volume-export-ganesha.yml  redhat_unregister.yml               umount.yml
disable-nfs-ganesha.yml          georep-secure-session.yml    gluster-quota-enable.yml       gluster-volume-rebalance.yml      replace_smb_conf_volname.yml         vgcreate.yml
disable-repos.yml                georep-session-config.yml    gluster-quota-limit-object.yml  gluster-volume-set.yml           run-script.yml                       vgextend.yml
edit-remote-file.yml             georep-session-create.yml    gluster-quota-limit-size.yml   gluster-volume-start.yml          service_management.yml               vgremove.yml
enable-nfs-ganesha.yml           georep-session-delete.yml    gluster-quota-ops.yml          gluster-volume-stop.yml           set-pcs-auth-passwd.yml              yum-operation.yml
enable-repos.yml                 georep-session-pause.yml     gluster-quota-remove.yml       lvchange.yml                      set-selinux-labels.yml               yum.repos.d
```

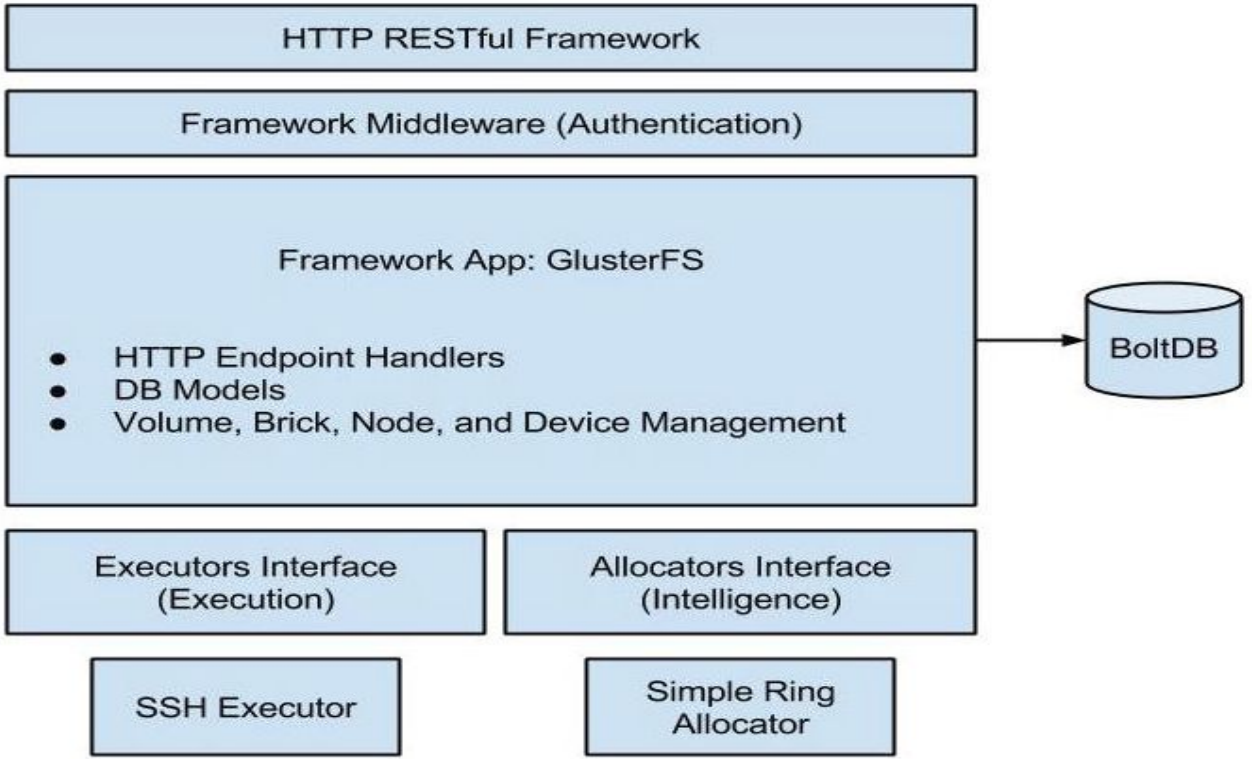
Gluster - StaaS: Heketi

- Dynamic Share Provisioning with Gluster volumes
- Eases brick provisioning - LVs, VGs, filesystem etc.
  - Overlaps with gdeploy
- Automatically determines brick locations for fault tolerance
  - Rack/Availability Zone Aware
- Exposes high level ReST interfaces for management
  - create share, expand share, delete share etc.

Heketi Illustrated



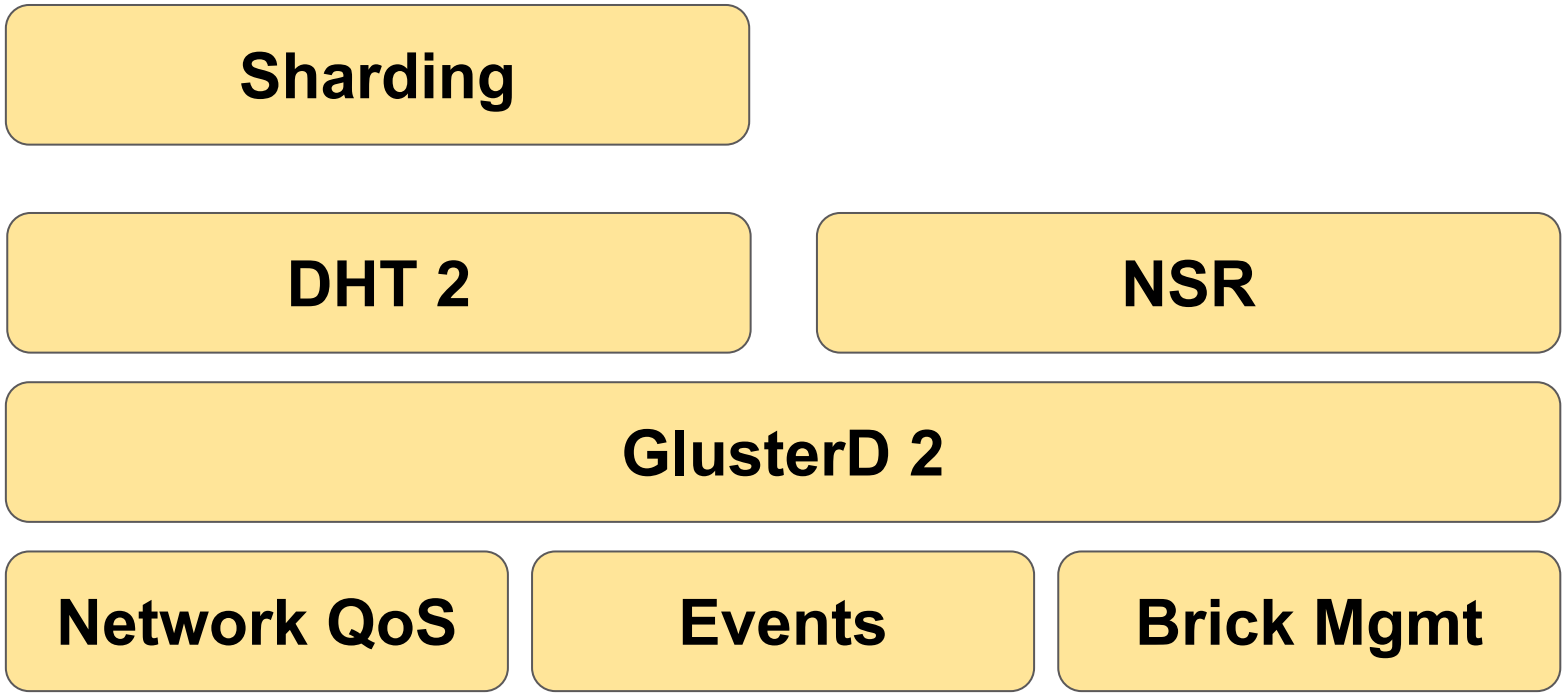
Heketi - Architecture



Containerized Gluster

- All Gluster processes containerized
- Uses bind mounts from host for storage
- Eases Gluster lifecycle operations
- Eases hyperconvergence with Gluster

Gluster.Next - Main Components



DHT 2

- Problem: directories on all subvolumes
  - directory ops can take O(n) messages
  - not just mkdir or traversals - also create, rename, and so on
- Solution: each directory on one subvolume
  - can still be replicated etc.
  - each brick can hold data, metadata, or both
  - by default, each is both just like current Gluster

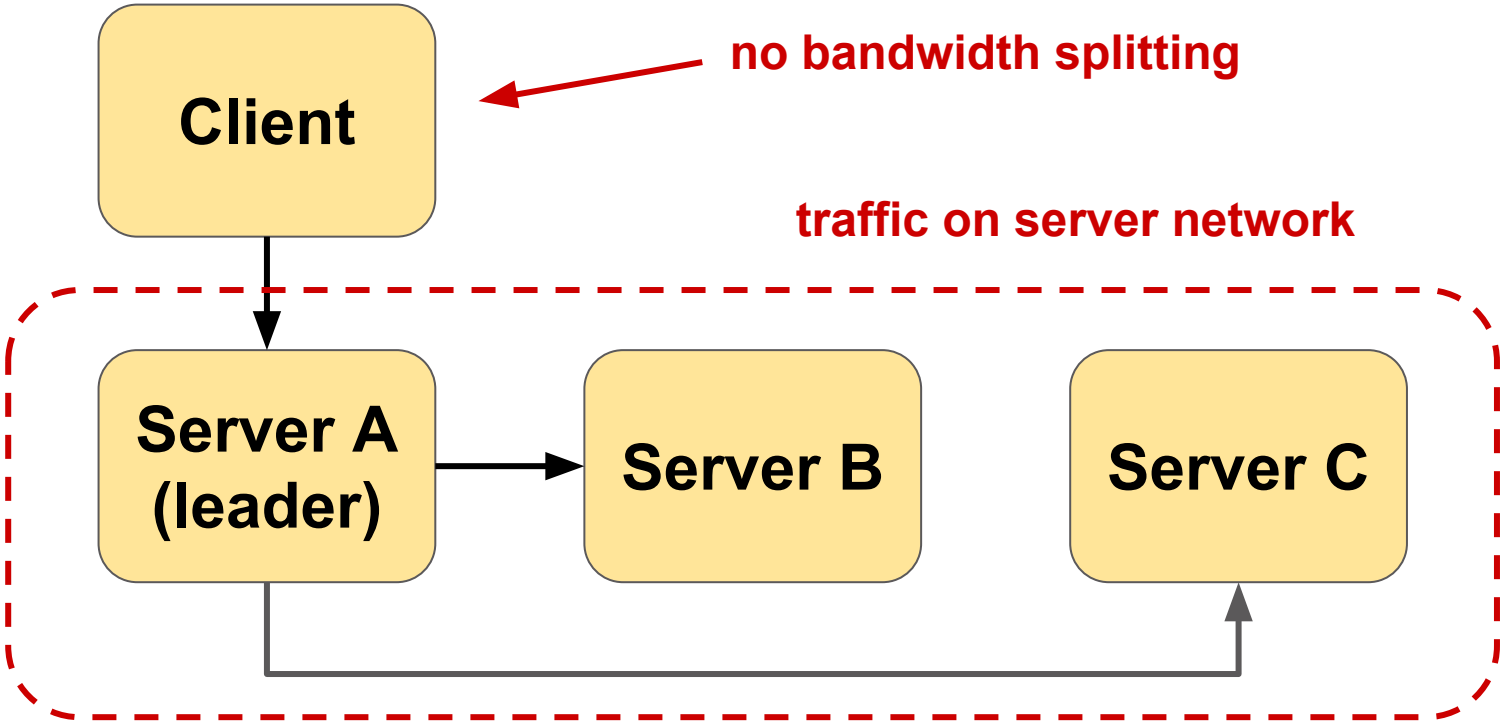
DHT 2 (continued)

- Improved layout handling
  - central (replicated) instead of per-brick
  - less space, instantaneous “fix-layout” step
  - layout generations help with lookup efficiency
- Flatter back-end structure
  - makes GFID-based lookups more efficient
  - good for NFS, SMB

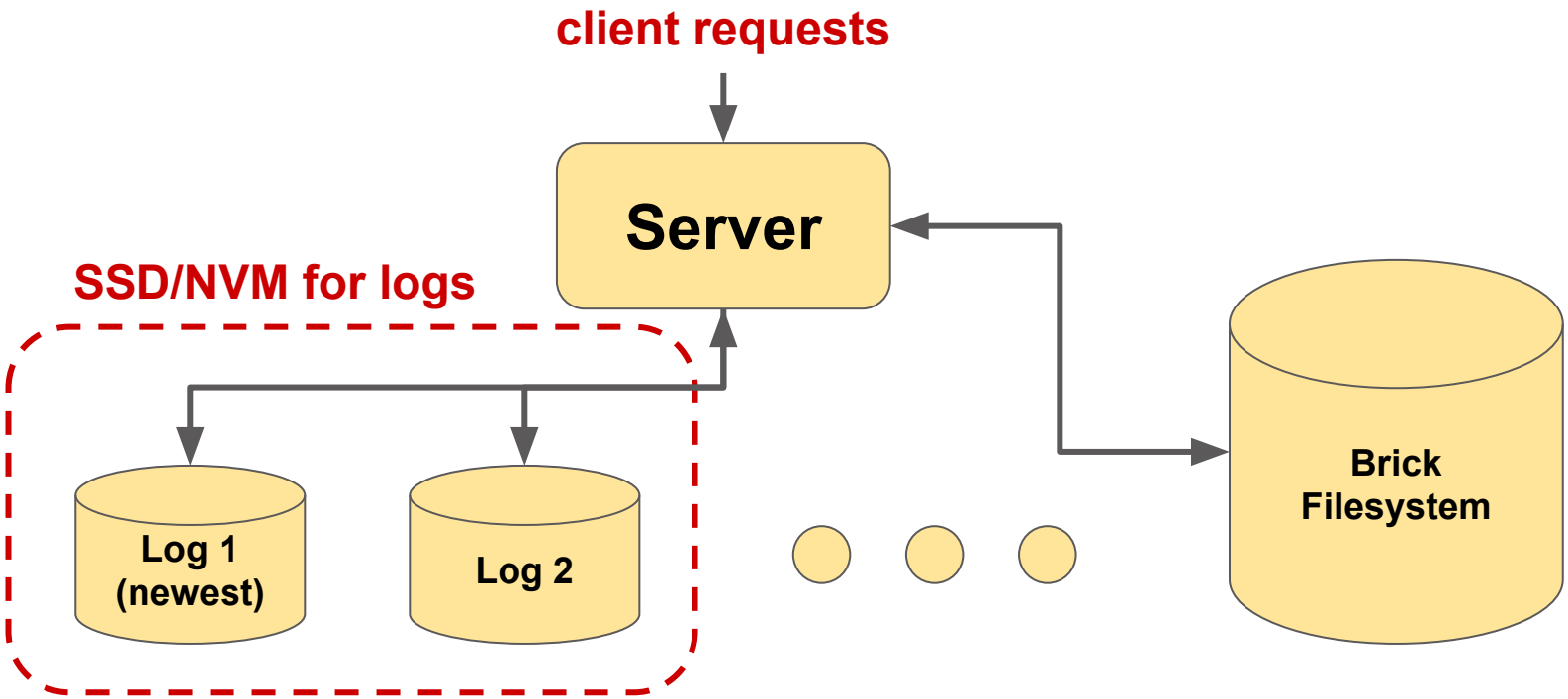
JBR - Journal Based Replication

- Server-side with temporary leader
  - vs. client-side, client-driven
  - can exploit faster/separate server network
- Log/journal based
  - can exploit flash/NVM (“poor man’s tiering”)
- More flexible consistency options
  - fully sync, ordered async, hybrids

JBR Illustrated (data flow)



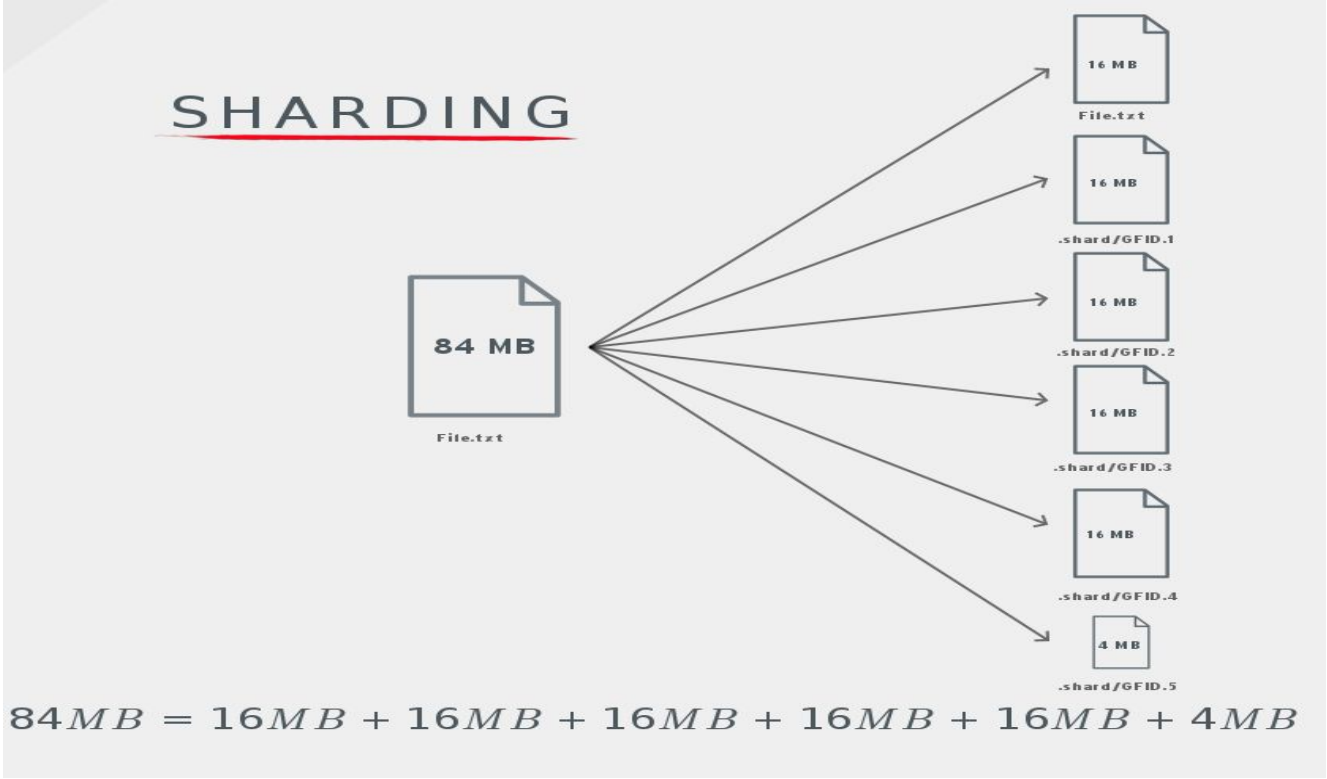
JBR Illustrated (log structure)



Sharding

- Spreads data blocks across a gluster volume
- Primarily targeted for VM image storage right now
- File sizes not bound by brick or disk limits
- More efficient healing, rebalance, tiering and geo-replication
- Yet another translator in Gluster

Sharding Illustrated



Profiling Workloads

```
Data Written: 810970688 bytes

Interval 0 Stats:
   Block Size:                1b+                2b+                4b+
 No. of Reads:                 0                  0                  0
No. of Writes:                64                 54                 11

   Block Size:             4096b+             8192b+            16384b+
 No. of Reads:                 0                  0                  0
No. of Writes:                 4                 72                211

   Block Size:            32768b+            65536b+           131072b+
 No. of Reads:                 0                  0                  0
No. of Writes:               115                197               5995

 %-latency   Avg-latency   Min-Latency   Max-Latency   No. of calls         Fop
 ---------   -----------   -----------   -----------   ------------        ----
      0.00       0.00 us       0.00 us       0.00 us            171      FORGET
      0.00       0.00 us       0.00 us       0.00 us             85     RELEASE
      0.00       0.00 us       0.00 us       0.00 us           3289  RELEASEDIR
      0.01      39.00 us      23.00 us      54.00 us              4     STATFS
      0.01      73.50 us      15.00 us     129.00 us              4     READDIR
      0.01      77.75 us      15.00 us     135.00 us              4    GETXATTR
      0.10      38.94 us      19.00 us     320.00 us             64       FLUSH
      0.19      36.74 us       2.00 us     460.00 us            127     OPENDIR
      0.19      74.38 us      45.00 us     196.00 us             64       WRITE
      0.21      81.41 us      43.00 us     241.00 us             64        LINK
      0.35      84.97 us      47.00 us     280.00 us            104       RMDIR
      0.40      59.38 us      29.00 us     231.00 us            168     SETATTR
      0.43      84.20 us      31.00 us     368.00 us            128      UNLINK
      0.70      68.97 us      14.00 us    7878.00 us            256    FINODELK
      1.26     495.95 us     139.00 us   12476.00 us             64      CREATE
      1.38      30.56 us       8.00 us    6871.00 us           1136     ENTRYLK
      1.50     362.44 us     108.00 us    5291.00 us            104       MKDIR
      1.59      53.07 us      13.00 us    5497.00 us            752     INODELK
      2.77     671.31 us      57.00 us   56365.00 us            104     SETXATTR
      3.36      97.27 us      18.00 us    5315.00 us            869      LOOKUP
     41.67   16389.59 us    4496.00 us   26715.00 us             64       FSYNC
     43.89    5753.58 us      28.00 us   44188.00 us            192     FXATTROP

    Duration: 1082627 seconds
   Data Read: 0 bytes
Data Written: 810970688 bytes
```

Profiling Workloads - more to come

- JSON Statistics dump - 3.8, thank you Facebook!
- Per translator statistics aggregation
- Workload aware share/cluster allocation

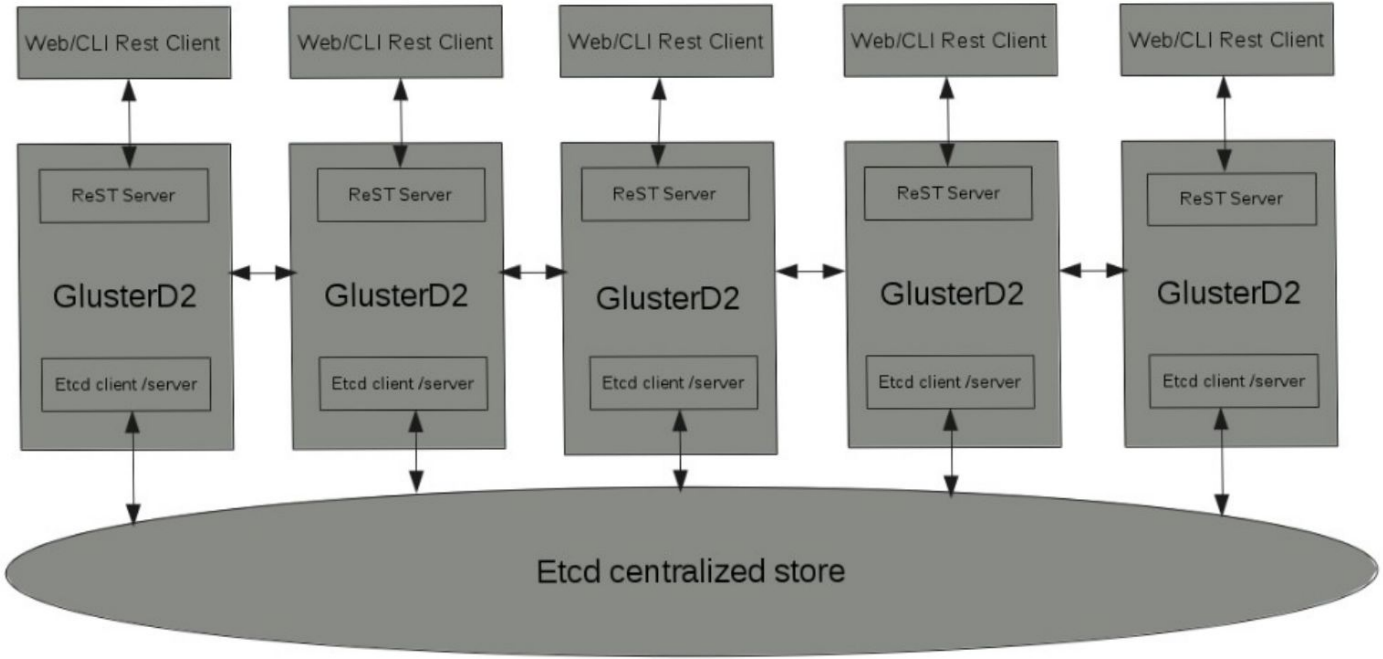
Storage + Network QoS

- Necessary to avoid hot spots at high scale
  - avoid starvation, cascading effects
- A single activity or type of traffic (e.g. self-heal or rebalance) can be:
  - directed toward a separate network
  - throttled on a shared network
- User gets to control front-end impact vs. recovery time
- Policies for tenant based queuing

GlusterD2

- More efficient/stable membership
  - especially at high scale
- Stronger configuration consistency
- Modularity and plugins
- Exposes ReST interfaces for management
- Core implementation in Go

GlusterD2 - Architecture



Event Framework

- Export node and volume events in a more consumable way
- Support external monitoring and management
- Currently: via storaged (DBus)

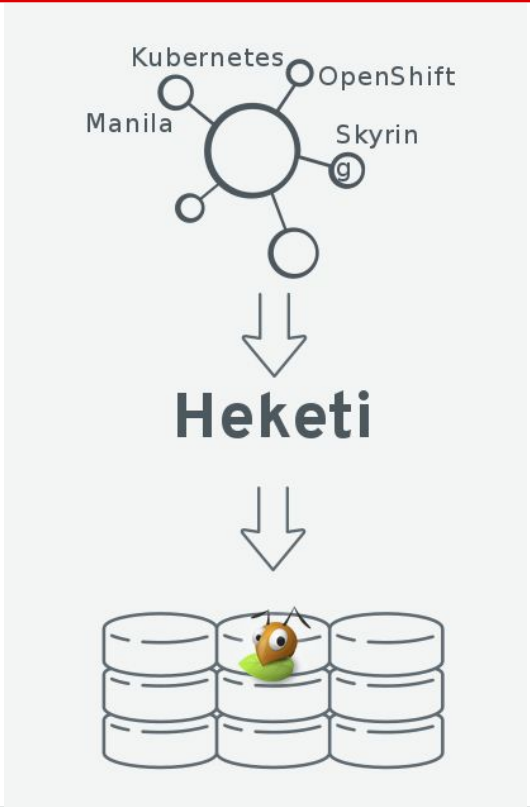
Brick Management

- Multi-tenancy, snapshots, etc. mean more bricks to manage
  - possibly exhaust cores/memory
- Heketi can help
  - can also make things worse (e.g. splitting bricks)
- One daemon/process must handle multiple bricks to avoid contention/thrashing
  - core infrastructure change, many moving parts

Gluster-StaaS:

Usecases

StaaS with Gluster.Next



- User/Tenant driven provisioning of shares.
- Talk to as many Gluster clusters and nodes
- Tagging of nodes for differentiated classes of service
- QoS for preventing noisy neighbors

Containers with Gluster StaaS

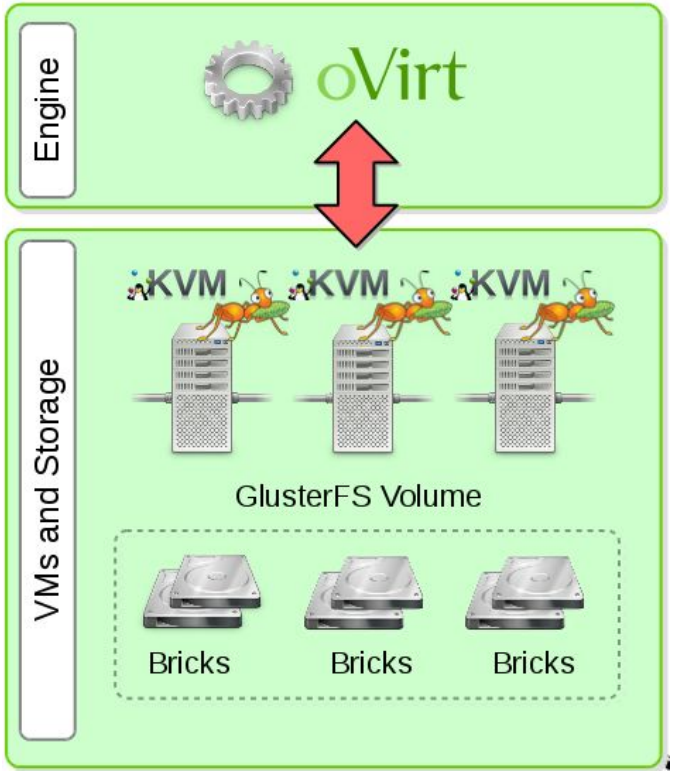
- Persistent storage for stateless Containers
  - Non-shared/Block : Gluster backed file through iSCSI
  - Shared/File: Multi-tenant Gluster Shares / Volumes

- Heketi to ease provisioning
  - “Give me a non-shared 5 GB share”
  - “Give me a shared 1 TB share”

- Shared Storage use cases being integrated with Docker, Kubernetes & OpenShift

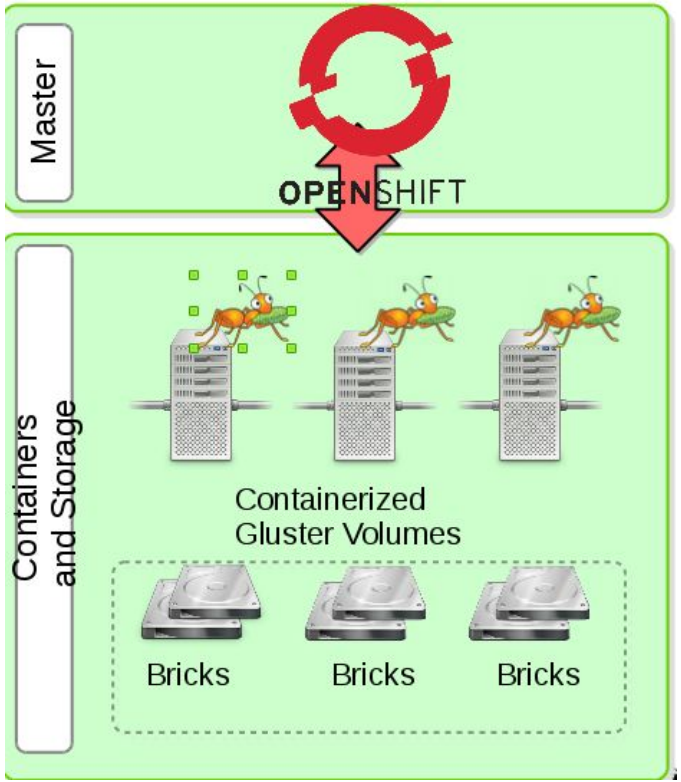
Hyperconvergence with VMs

- Gluster Processes are lightweight
- Benign self-healing and rebalance with sharding
- oVirt & Gluster - already integrated management controller
- geo-replication of shards possible!



Integration with OpenShift

- Server nodes are used both for containers and storage
- Containerized Gluster exports bind mounted directories from hosts
- Tenants consume volumes or sub-directories of volumes exported through FUSE



Gluster.Next

When?

Gluster.Next Phase 1

Gluster 3.8

- May/June 2016
- Experimental features from Gluster.Next
  - dht2, NSR, glusterd2, Eventing
- Sub-directory export support for FUSE
- UNIX-domain sockets for I/O
  - slight boost in hyperconverged setups

Gluster.Next Phase 2

Gluster 4.0

- December 2016
- Everything that we've discussed so far :-)
- And more...

Challenges with StaaS - Addressed with Gluster!

- Provisioning
  - gdeploy & Heketi
- Scale
  - Gluster.Next
- Diverse workloads
- Multi-tenant challenges
  - QoS, Isolation, Heketi, sub-directory exports

Thank You!
@vbellur

IRC: #gluster, #gluster-dev on freenode