Project Aplo: GlusterFS Container Converged with OpenShift

Luis Pabón
Red Hat Summit / DevNation
2016
What does Aplo mean?
Solution

Provide a solution that will run GlusterFS as containers in OpenShift pods.

Integrate GlusterFS deployment and management with OpenShift services.
Goal Workflow with Dynamic Provisioning

Persistent Volume Claim is submitted
Goal Workflow with Dynamic Provisioning

OpenShift request volume to be created
Persistent volume is created by storage system and registered with OpenShift
Goal Workflow with Dynamic Provisioning

OpenShift binds persistent volume to persistent volume claim request.
Goal Workflow with Dynamic Provisioning

Volume can now be used by Pod to provide persistent storage
Since dynamic provisioning is not currently available, the Administrator creates many volumes and registers them with OpenShift
Workflow for Release 1.0

Persistent volume claim is submitted
Workflow for Release 1.0

An available persistent volume is picked out of the pool and bound to the persistent volume claim.
Workflow for Release 1.0

Volume can now be used by Pod to provide persistent storage
Paradigm Shift

Traditional Storage Management

Converged Storage Management
Aplo Architecture

OpenShift Cluster

Client

OpenShift Masters

OpenShift Nodes with storage

Gluster Cluster 0 ... Gluster Cluster N

OpenShift Nodes
Aplo Technologies

- OpenShift
- Heketi
- GlusterFS
OpenShift provides platform as a service infrastructure based on Kubernetes container management.
GlusterFS provides containerized distributed storage.
Heketi provides dynamic GlusterFS volume life cycle management.
OpenShift
OpenShift

- Based on Kubernetes
- Many improvements over Kubernetes like multitenancy
- Can easily create, deploy, and manage containerized applications
- Composed of a number of master systems, and a set of nodes.
- Can be run on RHEL or RH Atomic Host
Definitions

- Container - Docker container
- Node - Runs containers
- Master - Kubernetes Master node
- Host net - Container can use ports on the host network
- Cluster net - OVS based SDN
- Privileged Mode - Root privileges
- Labels - Key-value pairs that can be attached to objects
- oc - OpenShift Client
OpenShift Object Definitions

- *Pod - A collection of containers
- Service - Abstraction access point to a Pod
- Route - Allows external access to a service
- Endpoints - Describe access points
- Persistent Volume - Cluster wide definition of a volume
- Persistent Volume Claim - User volume request
Pod Example

```yaml
apiVersion: v1
type: Pod
metadata:
  name: busybox
type: Spec
containers:
  - type: Image
    name: busybox
    command:
      - type:睡
      - "3600"
    volumeMounts:
      - type: MountPath
        mountPath: /glusterfs
        name: mypvc
volumes:
  - name: mypvc
    persistentVolumeClaim:
      claimName: glusterfs-claim
```
OpenShift Object Definitions

- Pod - A collection of containers
- Service - Abstraction access point to a Pod
- *Route - Allows external access to a service
- Endpoints - Describe access points
- Persistent Volume - Cluster wide definition of a volume
- Persistent Volume Claim - User volume request
Routing using HAProxy
OpenShift Object Definitions

- Pod - A collection of containers
- Service - Abstraction access point to a Pod
- Route - Allows external access to a service
- Endpoints - Describe access points
- *Persistent Volume - Cluster wide definition of a volume
- *Persistent Volume Claim - User volume request
Kubernetes GlusterFS Mount Plugin

Node A

- Application Pod
- Bind Mount Volume
- Mounted Volume

Node B

- GlusterFS Pod
OpenShift Object Definitions

- Job - Container definition that runs to completion
- Secret - Store opaque data, up to 16MB
- Deployment Config - Deploys container replicas
- List - Collection of objects
- Template - A List with variables
GlusterFS
GlusterFS Overview
GlusterFS Volume

- Brick is the combination of a node and an export directory.
- Volume is a collection of bricks
Creating a volume

# gluster volume create myvolume
   replica 2
   transport tcp
   server1:/export/brick/myvolume1
   server2:/export/brick/myvolume2
   server3:/export/brick/myvolume3
   server4:/export/brick/myvolume4
   server5:/export/brick/myvolume5
   server6:/export/brick/myvolume6
# gluster volume start myvolume
# mount -t glusterfs server1:myvolume
   /mnt/gluster/myvolume
Now do the same for...
Wait... what?
Heketi
Heketi

- Creates GlusterFS volumes dynamically
- Intelligent brick allocator
- REST application with authentication
- Supports multiple GlusterFS clusters
Architecture

HTTP RESTful Framework

Framework Middleware (Authentication)

Framework App: GlusterFS
- HTTP Endpoint Handlers
- DB Models
- Volume, Brick, Node, and Device Management

BoltDB

Executors Interface (Execution)

Allocators Interface (Intelligence)

SSH Executor

Simple Ring Allocator
Heketi Cluster
Heketi Cluster

Cluster

Zone 1
Heketi Cluster
Heketi Cluster
Heketi Cluster
Heketi Cluster

Create Volume

Heketi

Cluster

Zone 1

Zone 2

Cluster

Zone 1

Zone 2
Heketi Cluster

Create Volume → Heketi

Cluster
Zone 1
Zone 2

Cluster
Zone 1
Zone 2
Heketi Cluster
Demo
Demo

tmpfs 100M 0 100M 0% /run/user/
1000
192.168.10.100:vol_8710a0473034be3ff088955073d4d414 100G 67M 100G 1% /mnt
$ sudo bash
# cd /mnt
# ls
# echo 'I love GlusterFS!' > index.html
# ls -al
total 13
drw-r-xr-x. 4 root root 4096 Jun 28 00:52 .
dr-xr-xr-x. 18 root root 4096 Jun  8 10:45 ..
-rw-r--r--. 1 root root 18 Jun 28 00:52 index.html
drw-r-xr-x. 3 root root 4096 Jun 28 00:52 trashcan
# exit
$ sudo umount /mnt
ls
glusterfs-topology.json heketi-storage.json nginx.yml
$ less nginx.yml
$ oc create -f nginx.yml
service "my-nginx-svc" created
route "my-nginx-router" created
persistentvolumeclaim "nginx-claim" created
pod "nginx" created
$ oc get pvc
NAME STATUS VOLUME CAPACITY ACCESSMODES AGE
nginx-claim Bound glusterfs-8710a047 100Gi RWX 4s
$ curl http://my-nginx-router-default.cloudapps.example.com/
I love GlusterFS!
$
More information

https://github.com/heketi/heketi
Q&A