Using Gluster for your Storage Workloads

Niels de Vos
Software Engineer at Red Hat
Gluster Developer and Maintainer

Agenda

- Gluster Introduction
- Traditional Workload
- Containers, Kubernetes, DevOps!!

Gluster Basics

- Software Defined Storage
- Scale-out, distributed and high-available
- Designed as a filesystem
  - Block Storage as an add-on
  - Object Storage as an add-on
- Easy to install, configure and maintain
  - Packages available for several Linux distributions

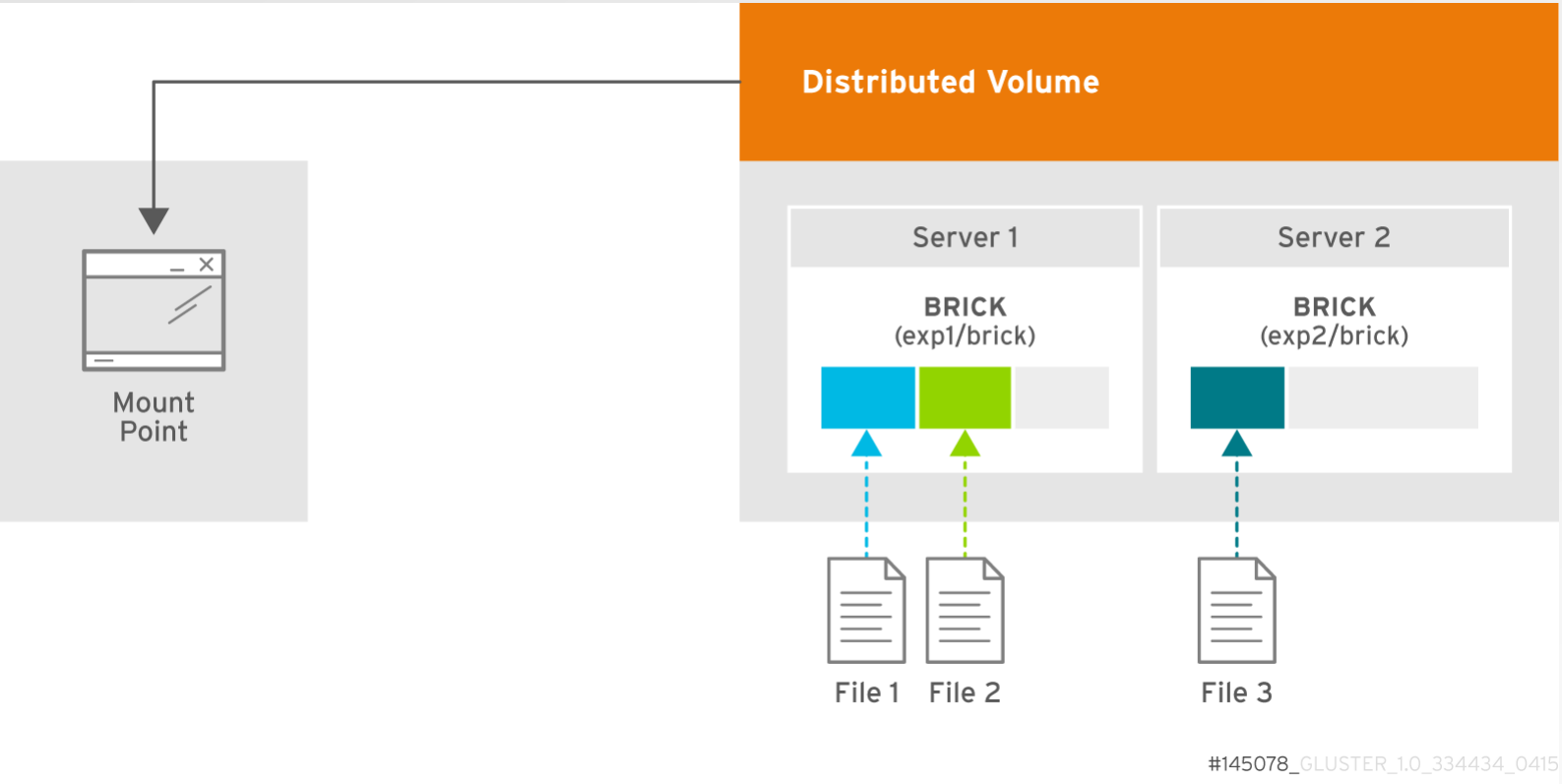
Commonly Used Stable Features

- Meta-data caching
- Geo-replication
- Volume snapshots
- Policy based split-brain resolution
- Brick multiplexing

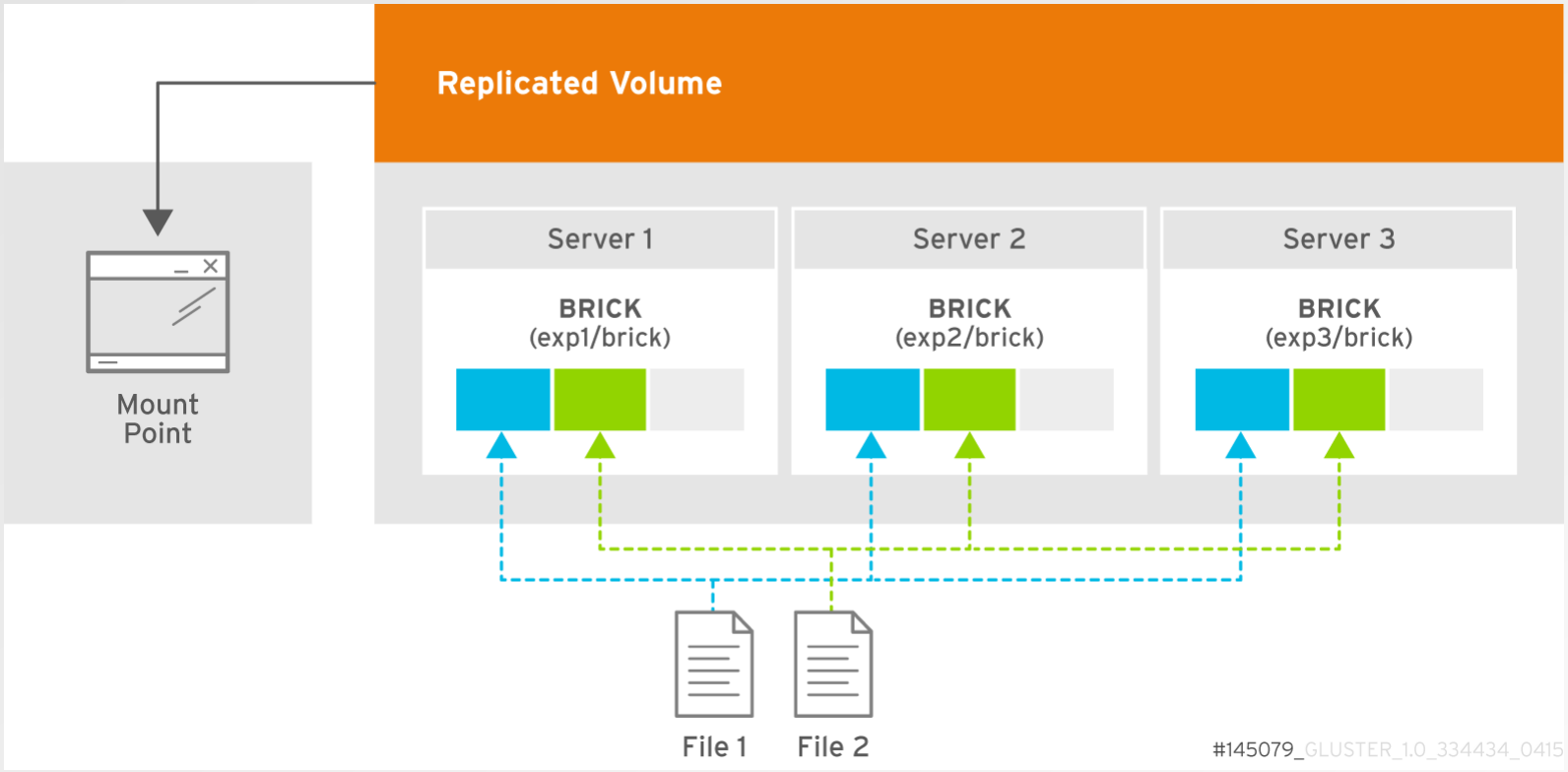
Gluster Basics

- Scalable Storage
  - Distributed Volumes
- High-Availability
  - 3-way Replication
  - 2-way Replication + Arbiter
  - Dispersed Volumes
- Flexible Storage
  - Distributed + HA

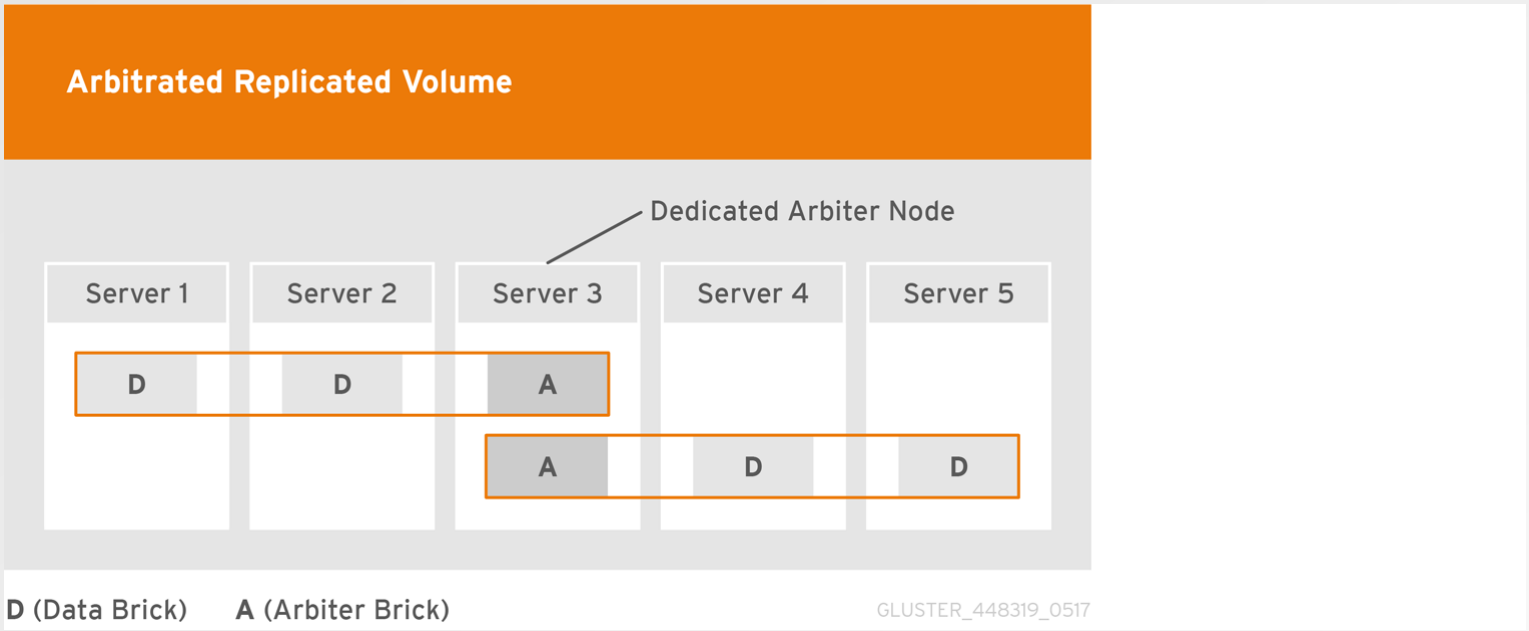
Distributed Volumes



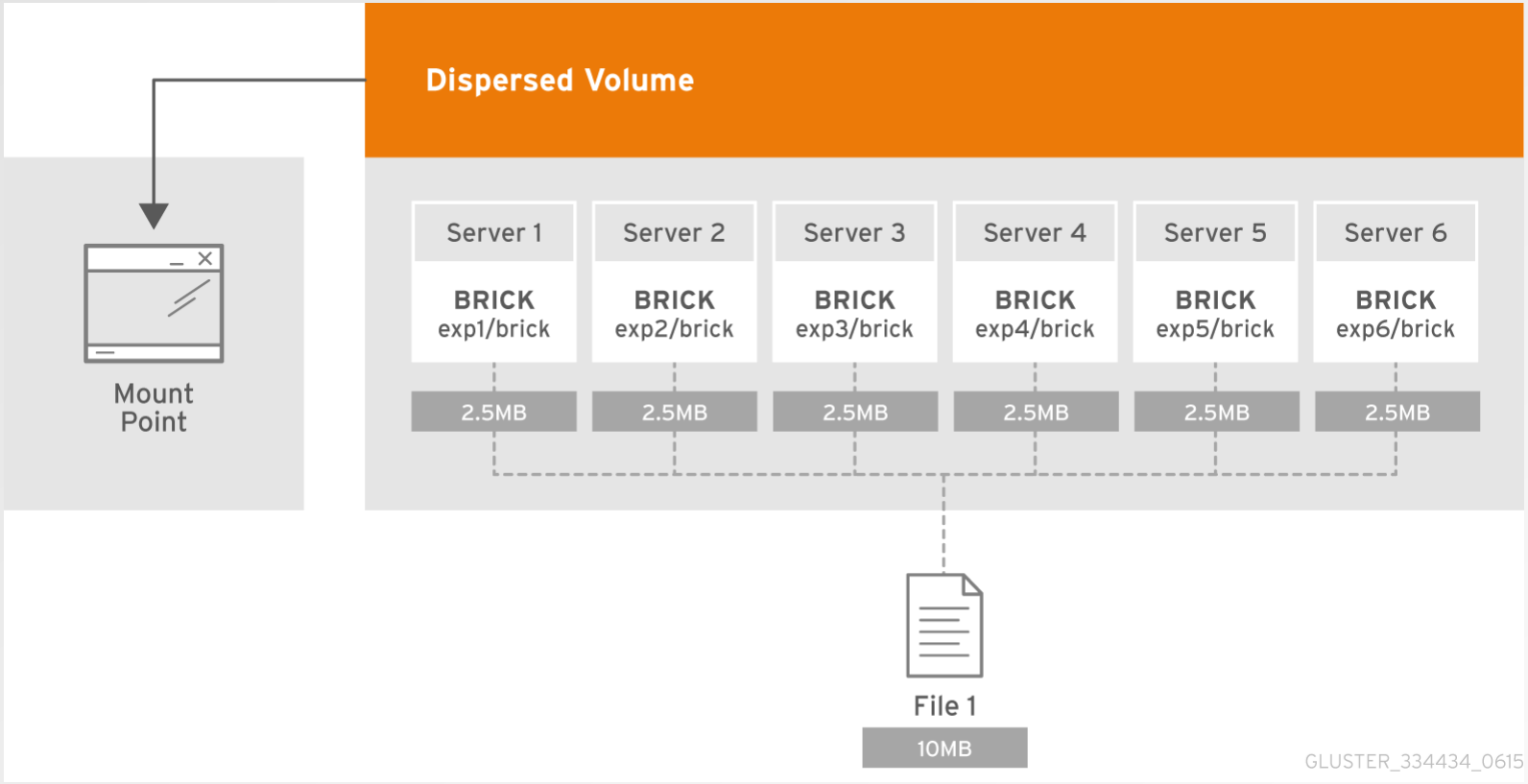
3-way Replication



2-way Replication + Arbiter



Dispersed (Erasure Coded) Volumes



NAS use-case

- NFS access with NFS-Ganesha
- Windows and Mac support through Samba

- Direct Attached Guster to file services
  - HA configured for the file service
- Disk-image file mounted inside HA-VM
  - HA taken care of by the hypervisors

NAS Examples

- Home directories, Windows Profiles
- Backup target
- Media archives
- Websites
- Image/video processing
- … well, just like most network filesystems

Block Storage

- Native QEMU integration with libgfapi
- iSCSI through gluster-block and tcmu-runner

- Large performance advantage for single-system access
- Little network overhead compared to filesystem operations

Block Storage Examples

- Virtual machines, available with oVirt
- Loopback mounted disk-images
  - Mount the Gluster Volume over FUSE/NFS/…
  - Loopback mount the disk-image
- Multipath-iSCSI
  - Out-of-the-box support for many Operating Systems

Block Storage use-cases

- Small file workloads, ‘git clone’ like Jenkins
- Applications with their own replication and clustering
  - ElasticSearch
  - Cassandra
  - MySQL

Containers with Kubernetes

- Pre-provisioned PersistentVolumes
- Dynamic provisioned PersistentVolumeClaims

- Standardized Gluster Volume creation by defining a StorageClass, or more than one

Dynamic PVC workflow

1. Create a PersistentVolumeClaim
2. K8s passes the PVC request to the provisioner
3. The Gluster provisioner requests Heketi to create a new Gluster Volume
4. Heketi decides where to create bricks
5. Heketi creates the Gluster Volume with the standard `gluster` command

StorageClass

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: replica-3-on-ssd-for-big-files
provisioner: kubernetes.io/glusterfs
parameters:
  resturl: "http://127.0.0.1:8081"
  restuser: "admin"
  secretNamespace: "default"
  secretName: "heketi-secret"
  volumetype: "replicate:3"
  volumeoptions: "features.shard on"
```


PersistentVolumeClaim

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: iso-images
spec:
  storageClassName: replica-3-on-ssd-for-big-files
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 100Gi
```


Pod

```
kind: Pod
apiVersion: v1
metadata:
  name: task-pv-pod
spec:
  volumes:
    - name: public-iso-images
      persistentVolumeClaim:
       claimName: iso-images
  containers:
    - name: public-downloads
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/usr/share/nginx/html"
          name: public-iso-images
```


Upcoming Kubernetes Features

- Cloning of PVCs
- Reduced storage requirements with arbiter
- Snapshot functionality
- ...

Thank you for your attention!

References:

- Gluster Homepage (https://gluster.org)
- Red Hat Gluster Storage 3.3 Administration Guide (http://red.ht/2tqOaqB)
- Gluster Docs (http://docs.gluster.org)
- Gluster Community (https://www.gluster.org/community/)

This presentation can be found at
https://people.redhat.com/ndevos/talks/2018-04-LOADays