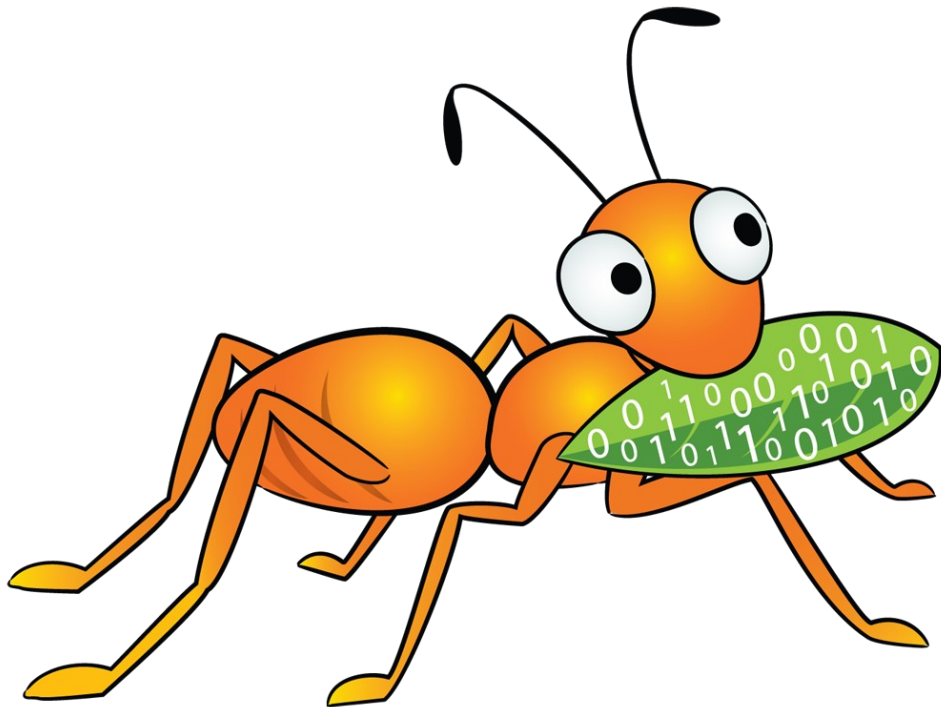


Data Distribution in Gluster



Niels de Vos
ndevos@redhat.com
Software Maintenance Engineer
Red Hat UK, Ltd.
February, 2012

Agenda

- Terminology
- Striped Volumes
- Replicated Volumes
- Distributed Volumes
- Distributed Replicated Volumes
- Elastic Hashing



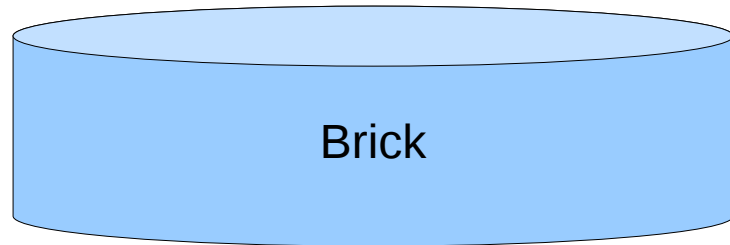
Terminology

- Brick:
 - Mountpoint used for data storage
- Volume:
 - Multiple bricks combined
- Node:
 - Server running the Gluster Daemon
 - Contains the Brick(s)
 - Provides access to the Volume(s)
- Trusted Pool



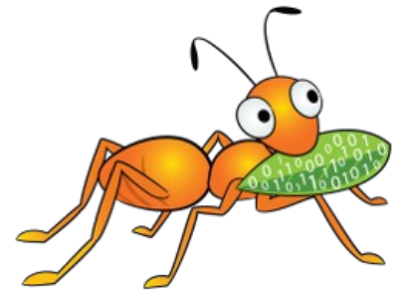
Terminology: Brick

- Mountpoint used for data storage
- Underlying physical storage for Volumes



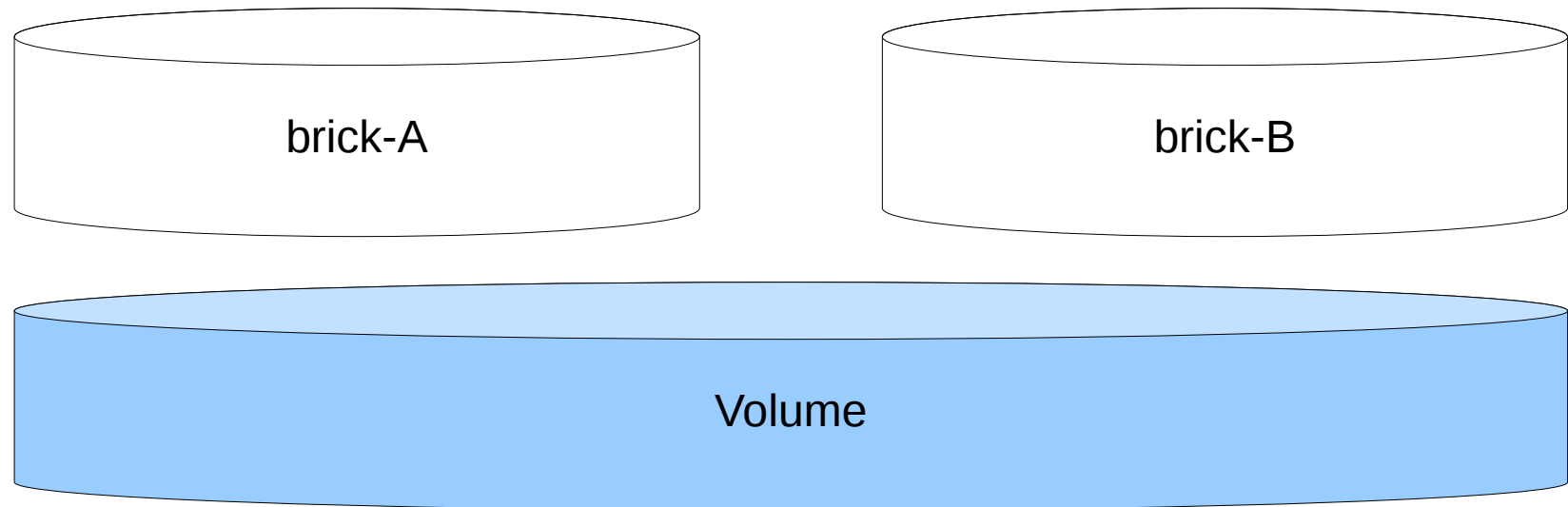
Creating a Brick

```
# mkfs -t xfs -i size=512 /dev/sdb1  
# mkdir -p /bricks/storage  
# vi /etc/fstab  
# mount /bricks/storage
```



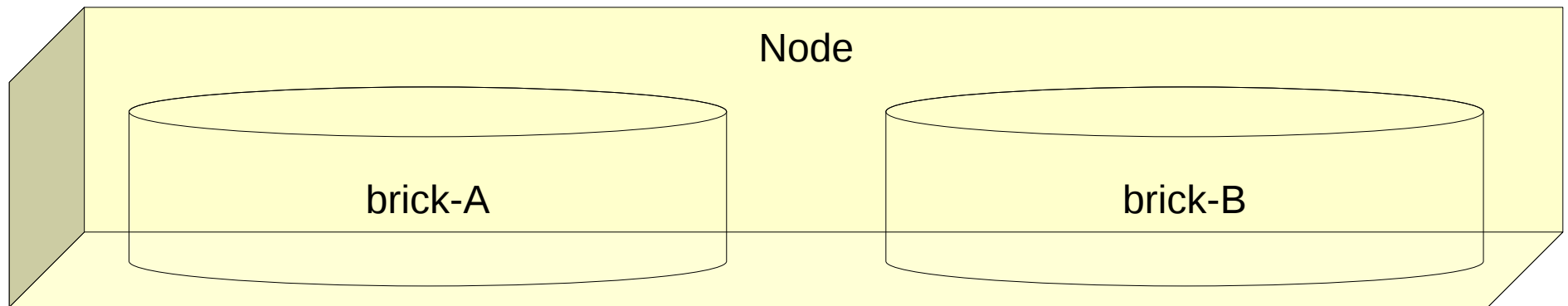
Terminology: Volume

- Multiple bricks combined
- Mountable by Gluster clients



Terminology: Node

- Server running the Gluster Daemon
- Contains the Brick(s)
- Provides access to the Volume(s)

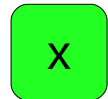
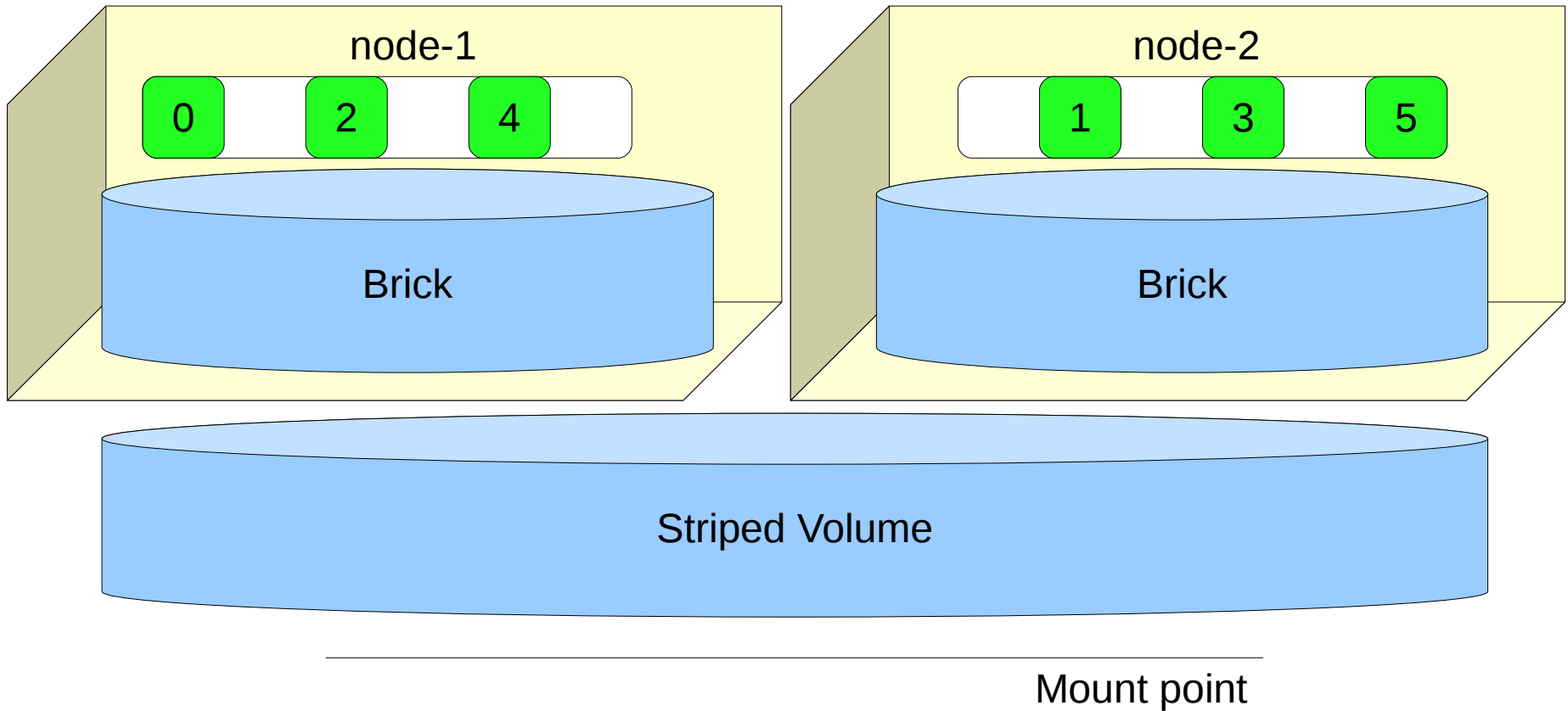


Striped Volumes

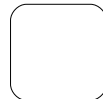
- Comparable with RAID-0
- Sparse files contain parts of the actual data
- Contents of the files are distributed over Bricks
- Losing one Brick results in loss of all the files
- No redundancy available (at the moment)



Striped Volumes



allocated data block

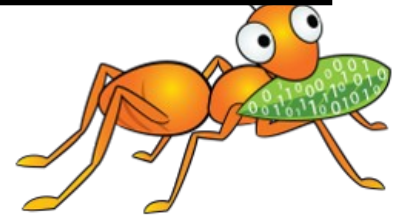


unallocated/sparse data block



Creating a Striped Volume

```
# gluster volume create my-striped-vol \  
    stripe 2 \  
    node1:/bricks/stripe node2:/bricks/stripe  
# gluster volume start my-striped-vol  
# gluster volume info my-striped-vol  
Volume Name: my-striped-vol  
Type: Stripe  
Status: Started  
Number of Bricks: 2  
Transport-type: tcp  
Bricks:  
Brick1: node1:/bricks/stripe  
Brick2: node2:/bricks/stripe
```

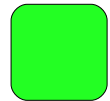
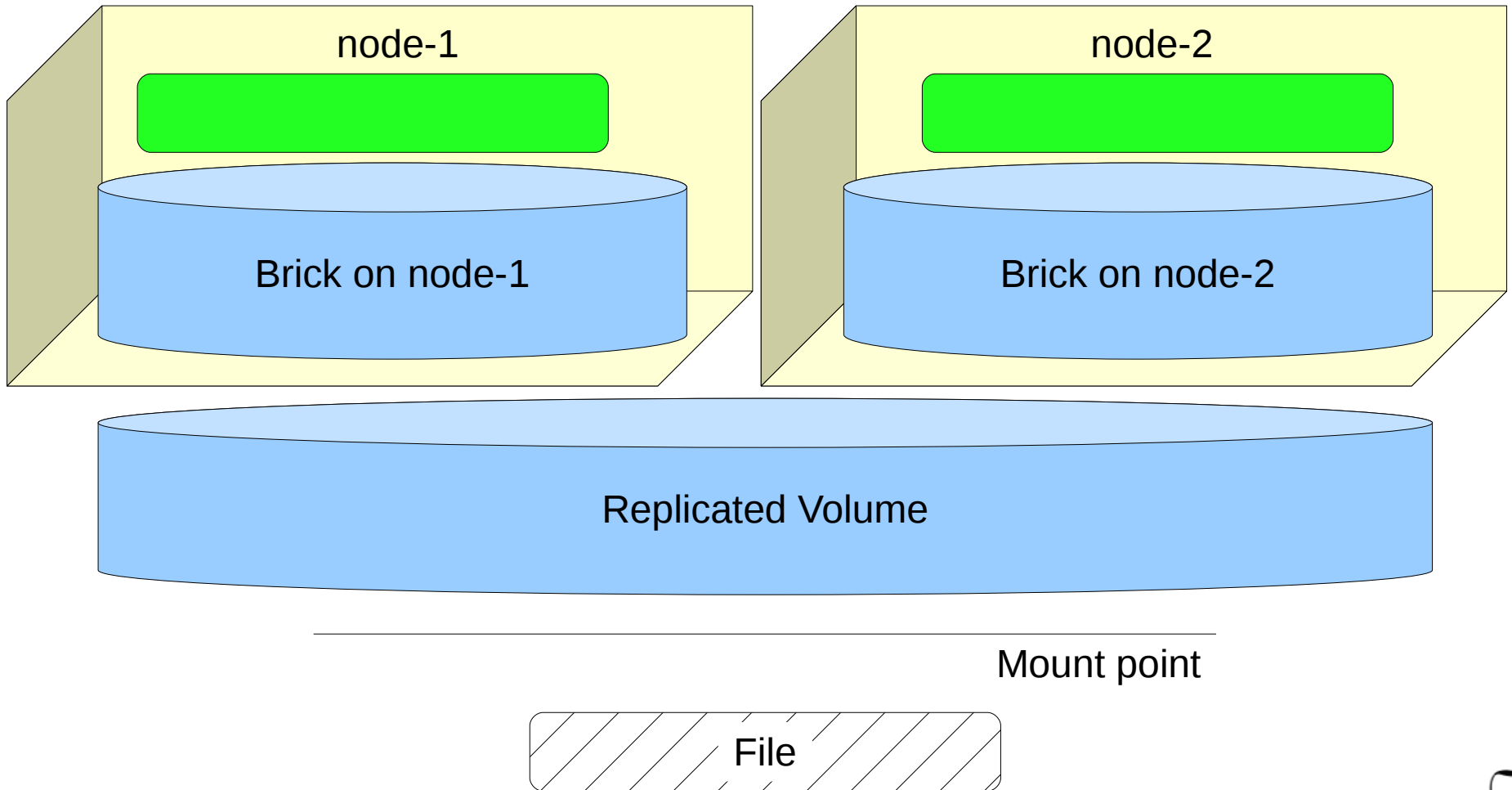


Replicated Volumes

- Comparable with RAID-1
- File duplication for redundancy
- Complete files available



Replicated Volumes

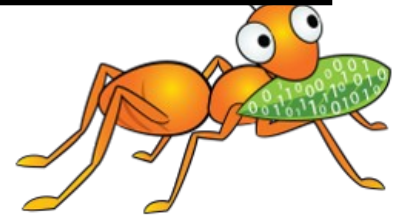


allocated physical file



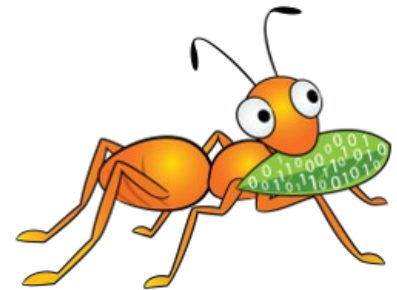
Creating a Replicated Volume

```
# gluster volume create my-replicated-vol \  
  replica 2 \  
  node1:/bricks/rep1 node2:/bricks/rep1  
# gluster volume start my-replicated-vol  
# gluster volume info my-replicated-vol  
Volume Name: my-replicated-vol  
Type: Replicate  
Status: Started  
Number of Bricks: 2  
Transport-type: tcp  
Bricks:  
Brick1: node1:/bricks/rep1  
Brick2: node2:/bricks/rep1
```

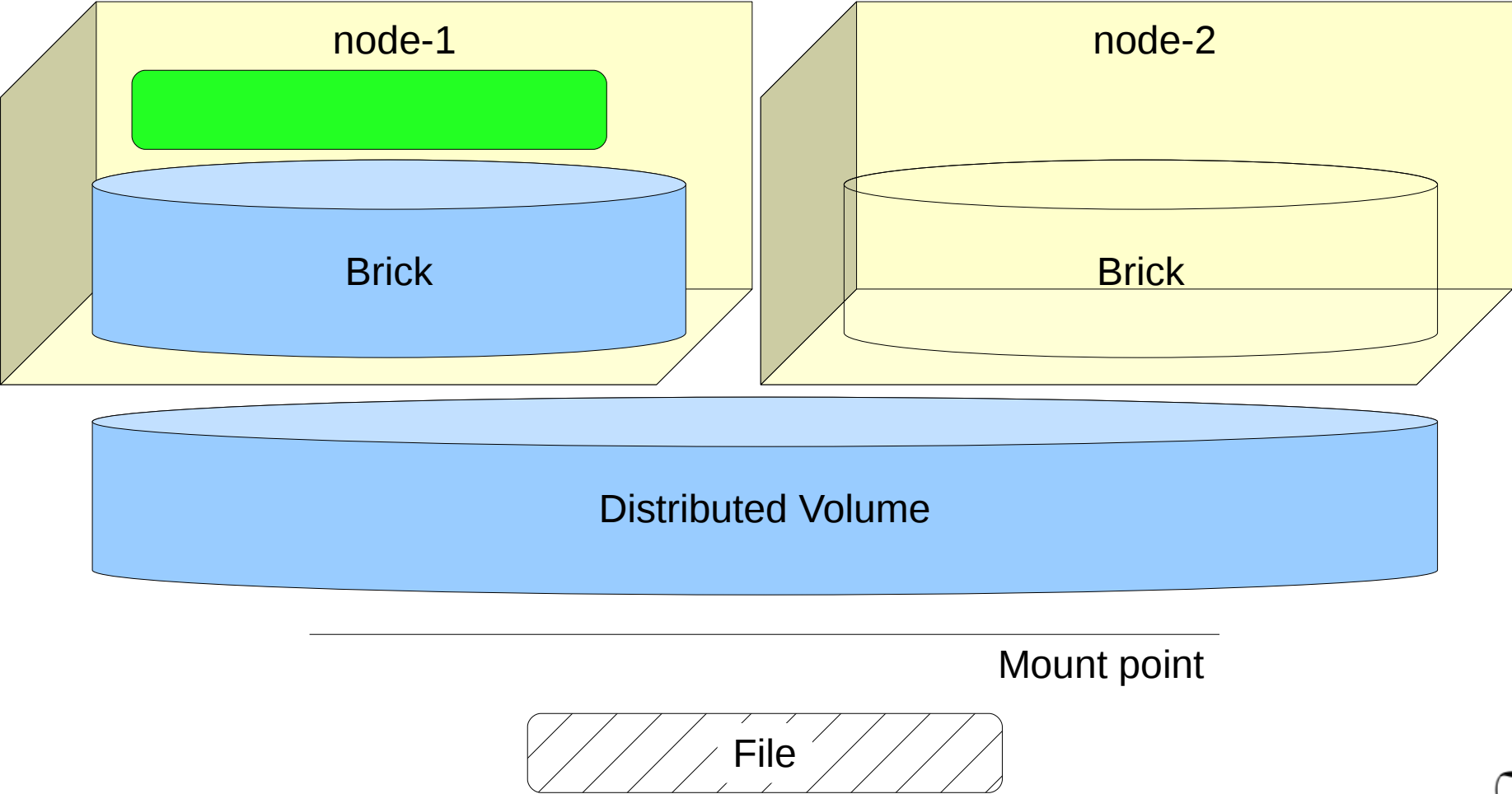



Distributed Volumes

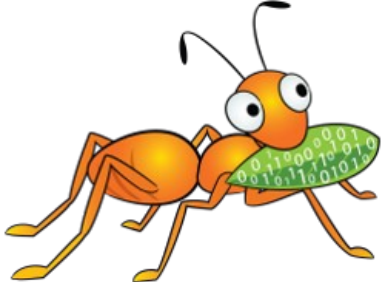
- Just a Bunch Of Bricks (JBOB)
- Comparable with JBOD (Just a Bunch Of Disks)
- No duplicating
- Complete files available



Distributed Volumes

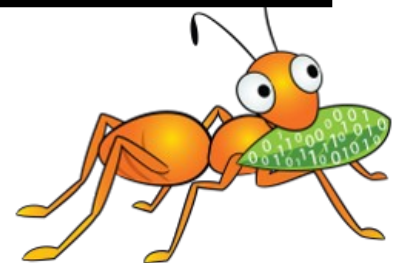


 Allocated physical file



Creating a Distributed Volume

```
# gluster volume create my-distributed-vol \  
    node1:/bricks/dist node2:/bricks/dist  
# gluster volume start my-distributed-vol  
# gluster volume info my-distributed-vol  
Volume Name: my-distributed-vol  
Type: Distribute  
Status: Started  
Number of Bricks: 2  
Transport-type: tcp  
Bricks:  
Brick1: node1:/bricks/dist  
Brick2: node2:/bricks/dist
```

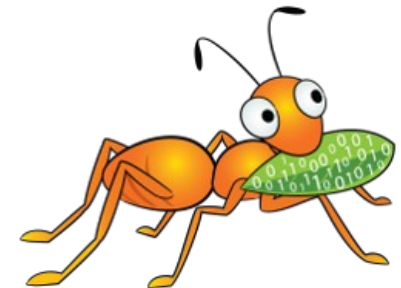
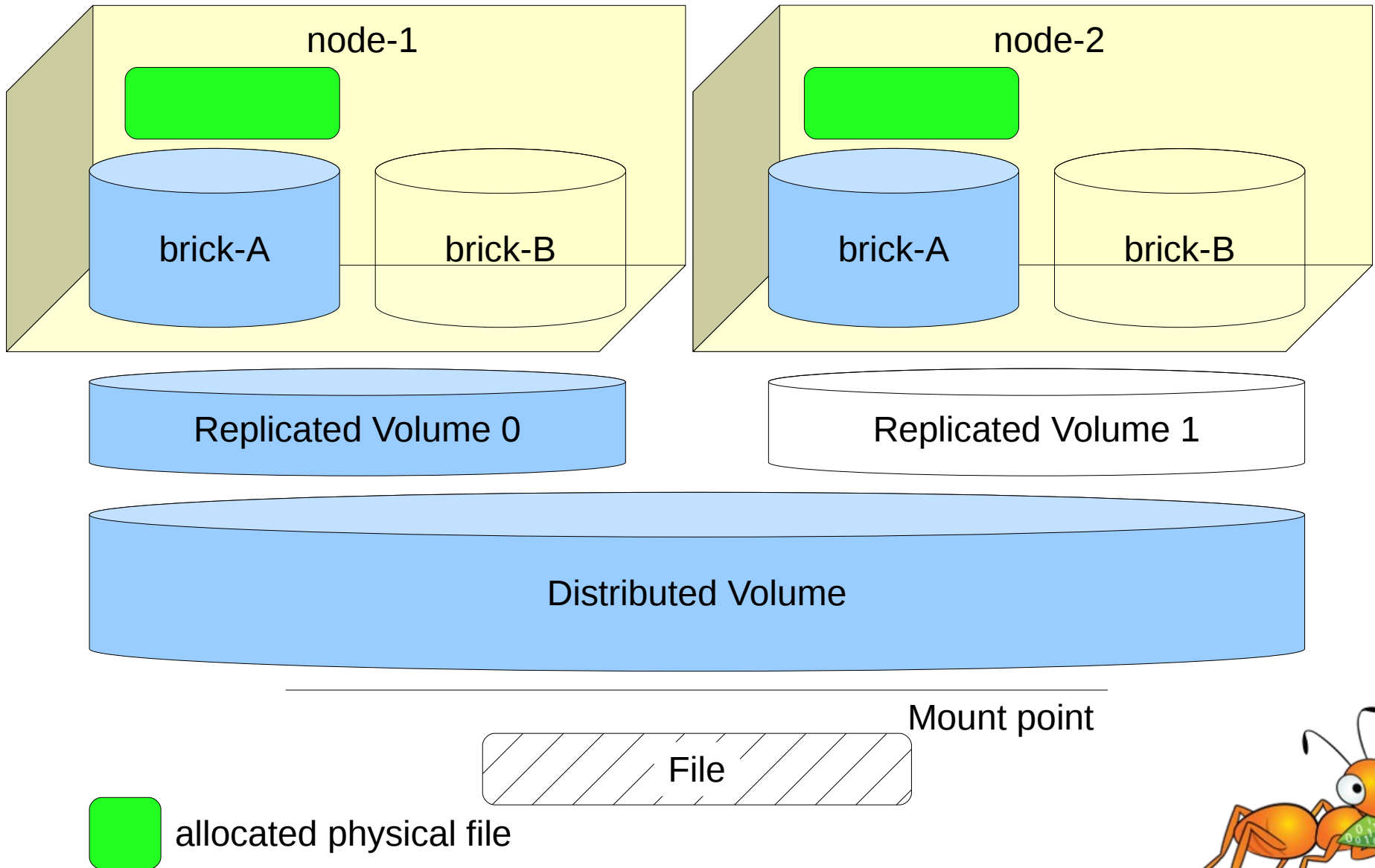


Distributed Replicated Volumes

- Just a Bunch Of Bricks (JBOD)
- Comparable with JBOD (Just a Bunch Of Disks) and RAID-1
- File duplication for redundancy
- Complete files available



Distributed Replicated Volumes



Creating a Distributed Replicated Volume

```
# gluster volume create my-dist-repl-vol \  
    replica 2 \  
    node{1,2}:/bricks/dist-repl-A \  
    node{1,2}:/bricks/dist-repl-B  
# gluster volume start my-dist-repl-vol  
# gluster volume info my-dist-repl-vol  
Volume Name: my-distributed-vol  
Type: Distributed-Replicate  
Status: Started  
Number of Bricks: 2 x 2 = 4  
Transport-type: tcp  
Bricks:  
Brick1: node1:/bricks/dist-repl-A  
Brick2: node2:/bricks/dist-repl-A  
Brick3: node1:/bricks/dist-repl-B  
Brick4: node2:/bricks/dist-repl-B
```

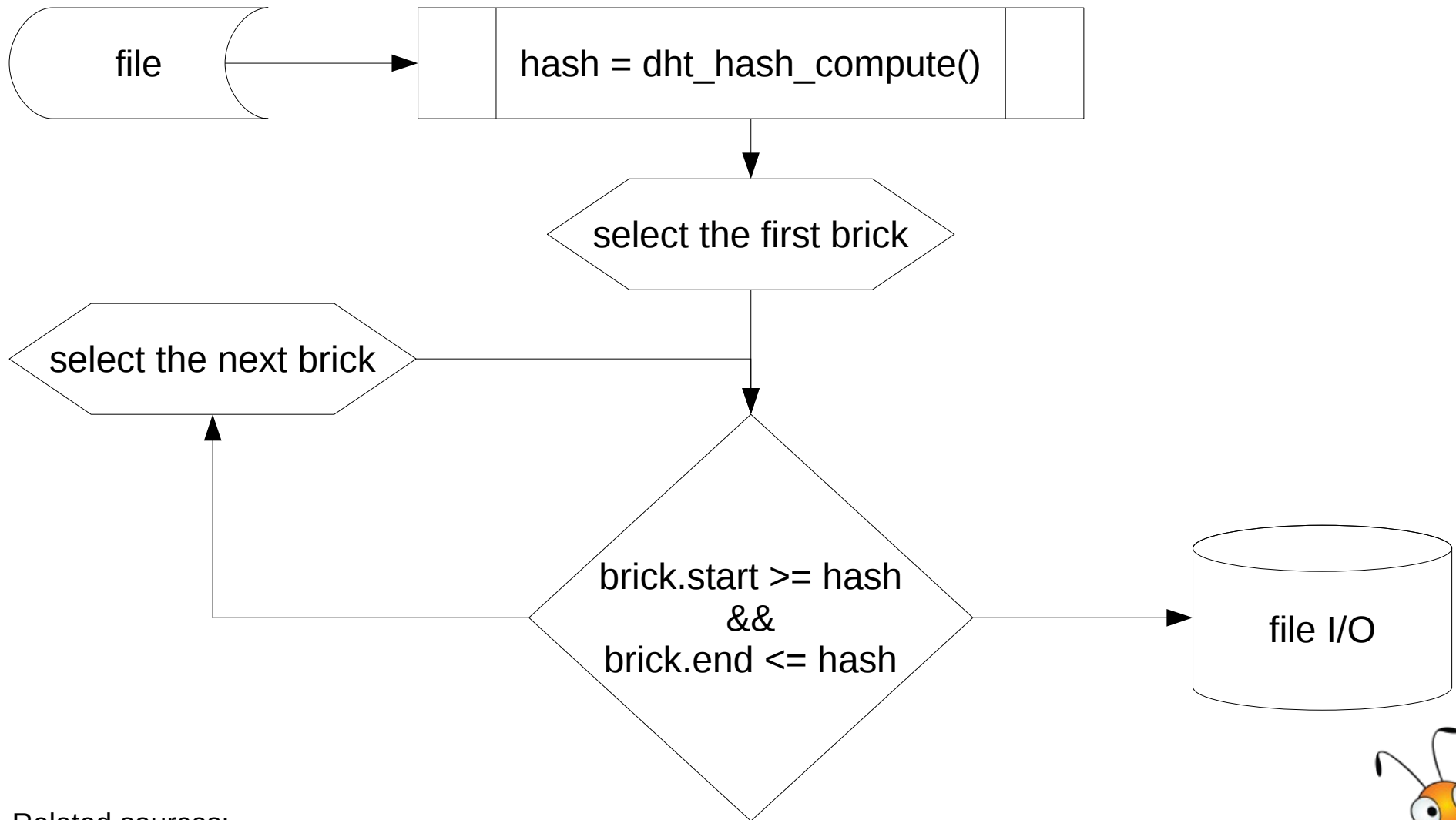


Elastic Hashing & File Distribution

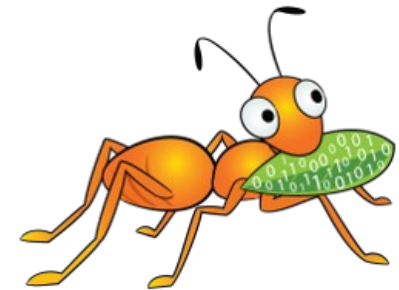
- Only known attribute is the path
- A hash based on the filename gets calculated
 - Davies-Meyer hashing function
- A hash has a finite number of results $\{0..N\}$
- A brick has a range of hash results
- Loop through the bricks and check if the hash result is in the hash range



Distributed Hash Table Xlator



Related sources:
xlators/cluster/dht/src/dht-layout.c:dht_layout_search()
libglusterfs/src/hashfn.c



Keep in Touch

- `gluster-devel@nongnu.org`
 - <http://lists.nongnu.org/archive/html/gluster-devel/>
- #gluster on Freenode
- twitter.com/RedHatStorage



Questions?

