BitRot detection in GlusterFS

Venky Shankar
Gaurav Garg

We are, Gluster developers at Red Hat

... participate in meetups, open source events
... hang out on #freenode: gluster, gluster-dev
nick: overclk, ggarg
... interact with community: gluster-devel@gluster.org
gluster-users@gluster.org

OK, enough. Let’s get started…

GlusterFS Quick Tour

Where’s my data?

- Distributed
- Local filesystem (*brick*)
  - XFS
  - EXT3, EXT4
  - BTRFS
- Prerequisite
  - POSIX compatible
  - Xattr support

Understanding data corruption

Corruption?

How ?

- Direct “brick” manipulation
  - Script bug
  - Admin
  - Malicious

SELINUX

Corruption?

How ?

(cont..)

- Silent corruption
  - Disk itself
    - Firmware bug
    - Mechanical wear
    - Ageing

Illustration


0010110011011010
00101100101010
TODAY
20XX

Solution: Integrity checks

Integrity Check

Consistency

- Track data modifications
  - Checksum (signature)
  - Persistent
- Verify during access
  - Recompute and check
- Repair if corrupted

Enough of theory, show me how it’s done.

Implementation

Constraints on choices

- Big fat-file story
- Deployments
  - **Distribute + Replicate**
  - Stripe, now [3.7+] *sharding*
  - Erasure coded

Implementation

Constraints on choices (cotd..)

- In-band data signing
  - Costly
  - RMW cycle
  - Degraded I/O performance
- Verification
  - “*Ditto*”

Implementation

Details

- Out-of-band data signing
  - Daemon
  - Asynchronous
    - Policy
    - Strong hash (reason ?)
- Verification
  - Daemon (scrubber)
  - On-demand
  - Pre-scrubbed

Implementation

Details (cotd..)

- Object versioning
  - Versioned upon modification
  - Versioning xattr (64 bit)
  - Reflect “object state”
- Signature
  - xattr
  - Attached to a “version”

Implementation

Details (cotd..)

- Integrity checking
  - Periodic
    - daily, weekly, etc..
  - Filesystem scan
    - Signature mismates
    - Matching version
  - QoS
    - Controlled crunching
- Corrupted objects
  - Denies access (EIO)
  - Repairable
    - Replica, Codes

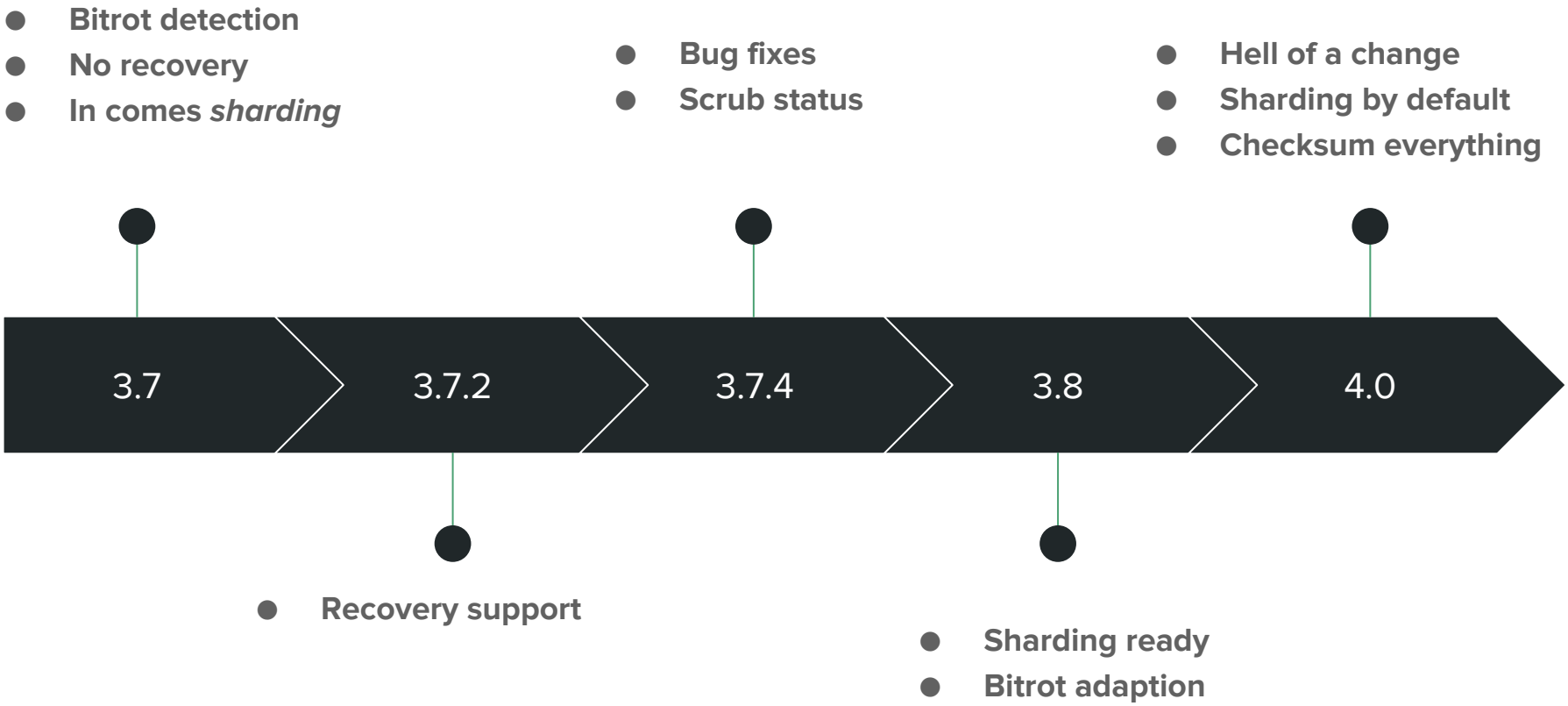
Use cases

Use Cases

- Small files
- Long lived data
  - Archival storage
  - WORM workload

Future

- Replica consistency
- Metadata checksumming
- Offloading
  - BTRFS
- Sharding adaption
- GlusterFS 4.0 [Interesting!]
  - In-band (*weaker hash*)
  - Checksum everything
  - Default
  - Lost (phantom) writes


Bitrot detection
No recovery
In comes *sharding*
Bug fixes
Scrub status
Hell of a change
Sharding by default
Checksum everything
3.7
3.7.2
3.7.4
3.8
4.0
Recovery support
Sharding ready
Bitrot adaption

Q & A