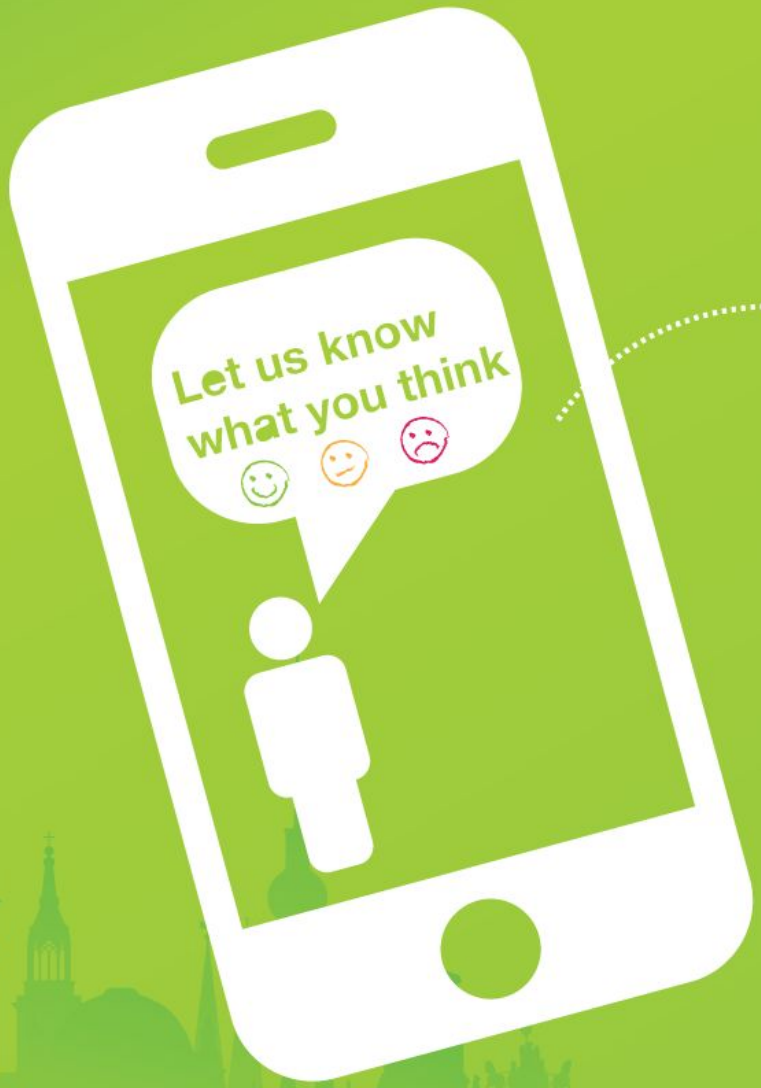
goto; conference



Click ‘engage’ to rate sessions and ask questions

Follow us on Twitter @GOTOber

www.gotober.com

Fighting Zombies: With Containers Towards Fault-Tolerant Infrastructure

FELIX HUPFELD, CTO QUOBYTE INC. @DRHU

GOTO BERLIN 2015

Preview

What is fault-tolerance?

Why fault-tolerance?

Fault-tolerance And Containers

Scheduling

State, Zombies, Naming

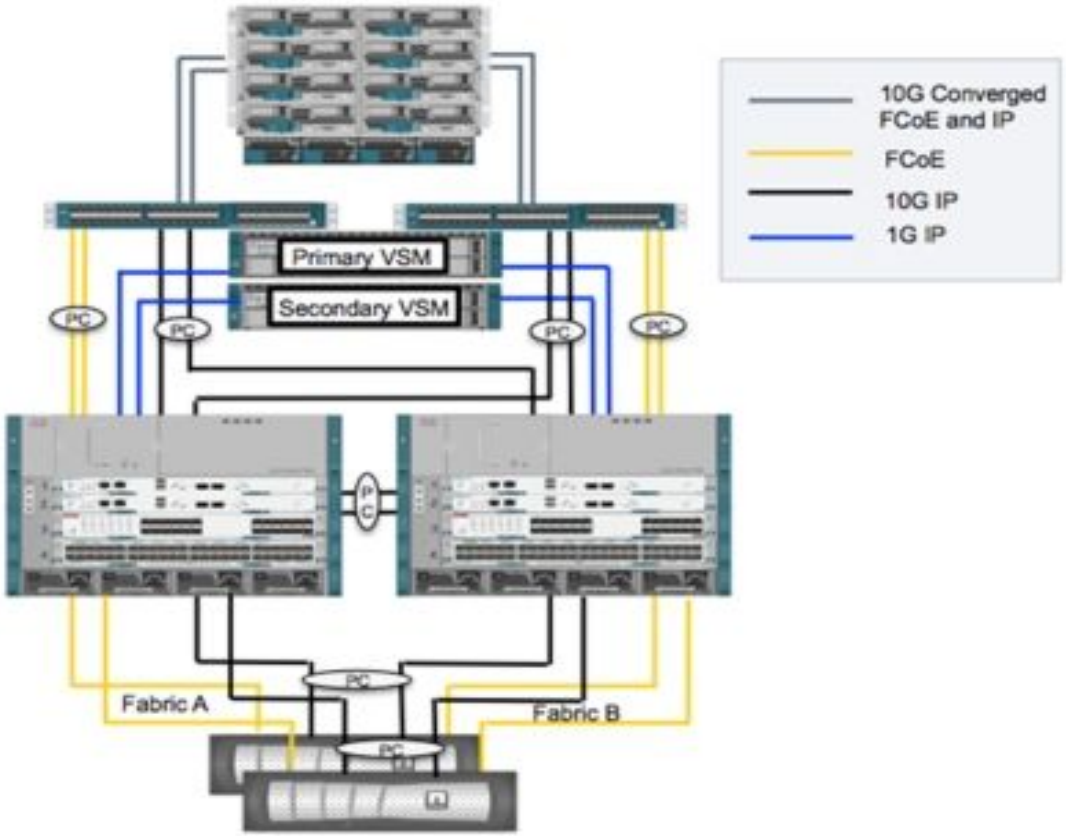
UCS B-Series Servers

UCS 6200 Fabric Interconnects

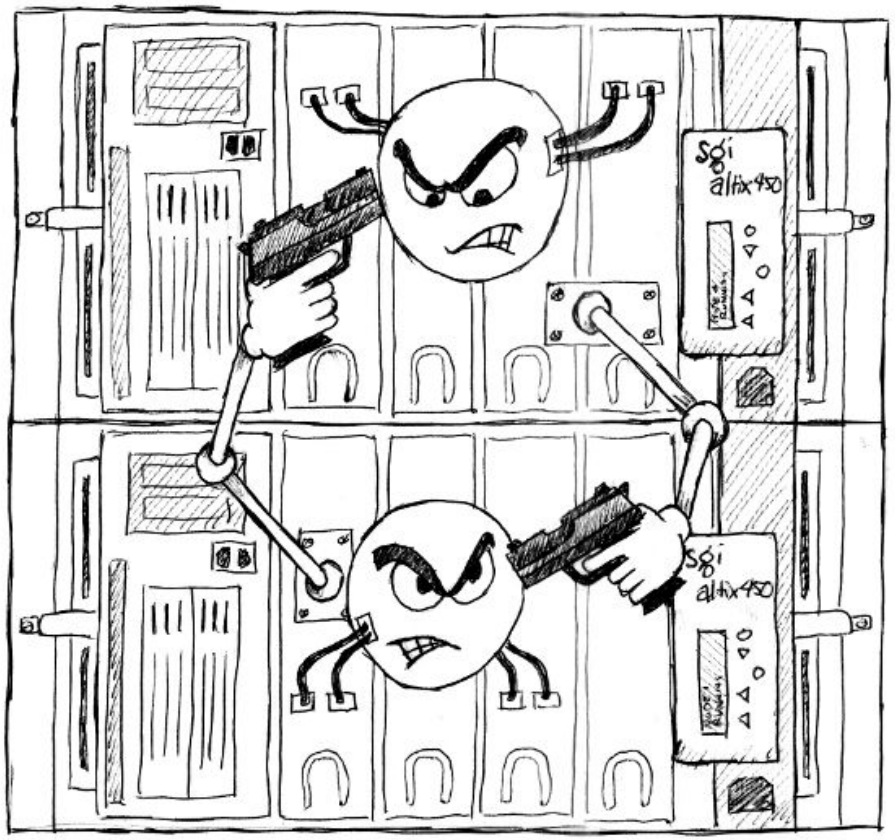
Nexus 1110 hosting
Nexus 1000v

Nexus 7000 Chassis
Supervisor 2E
10G M2 and F2/F2e Line Cards

NetApp FAS 22xx
NetApp FAS 32xx
NetApp FAS 62xx



STONITH - Shoot The Other Node In The Head



DON'T ANYBODY MOVE . . .

FAULT

- any hardware error
- most importantly split-brain
- no byzantine

TOLERANCE

- never affect correctness / safety / consistency
- should not affect availability
- handle automatically

Why fault-tolerance? High availability!

Why fault-tolerance? Operations!

Robustness

guards against operator mistakes

make system simpler to operate

at any scale

Why fault-tolerance? Operations!

Rolling Updates, Zero-downtime updates

Why fault-tolerance? Operations!

Fault **isolation**. Degrade individual hardware to an anonymous resource:

failure: page, replace asap

vs.

fix eventually

Why fault-tolerance? Operations!

Decouple hardware processes from service processes. No maintenance window for isolated tasks.

schedule maintenance window / drain

vs

pull out and repair

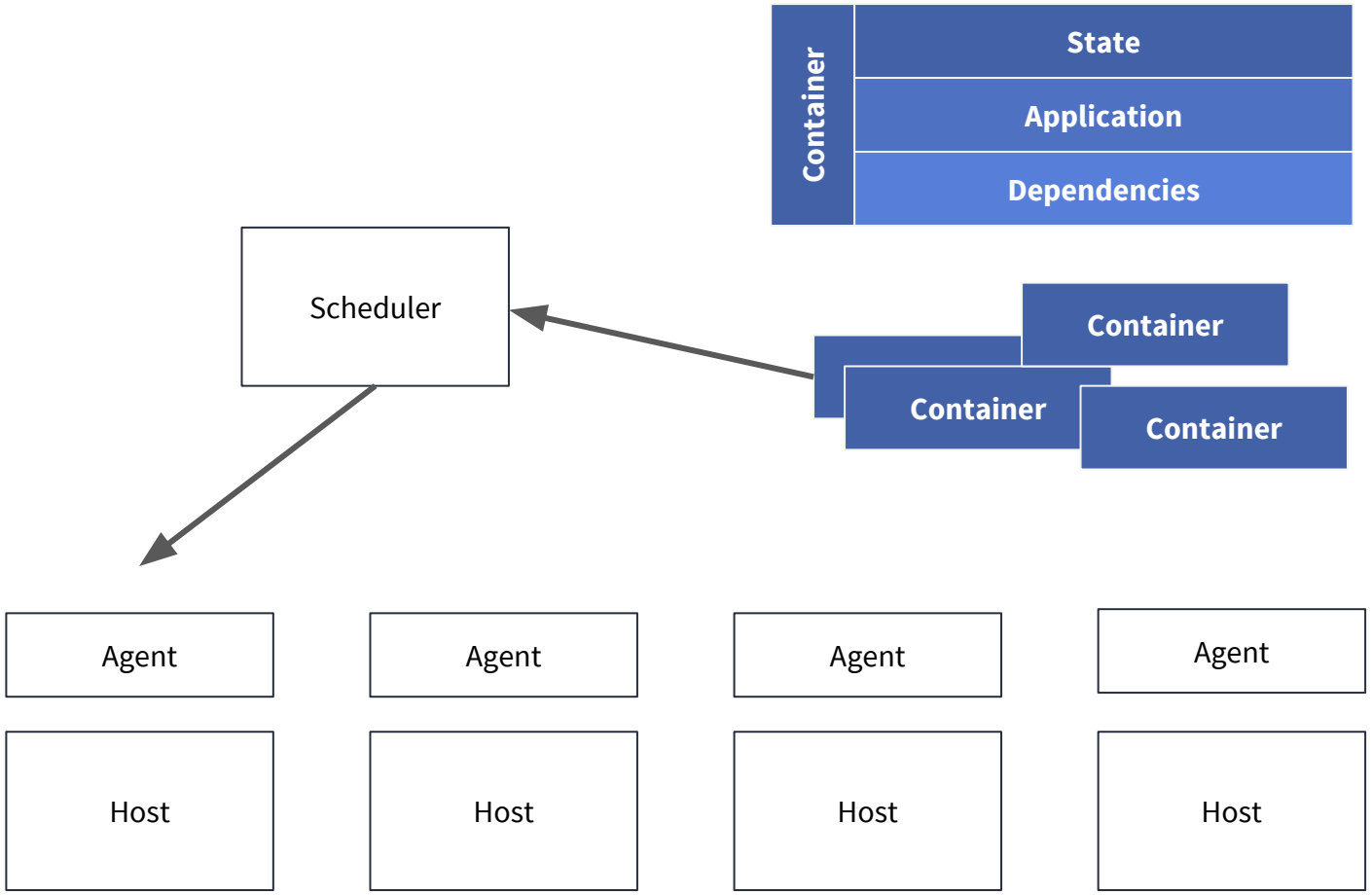
Scalable Operations. More servers / data, same headcount.
> 20k servers per DC OP at Facebook

Fault-tolerance and Containers

Containers decouple application from host (ship with dependencies).

Containers enable quick deployment of applications on any host - can react to failures quickly.

Reference Architecture



Problem: State

Move state away from container, from host, to *remote* and *redundant* storage.

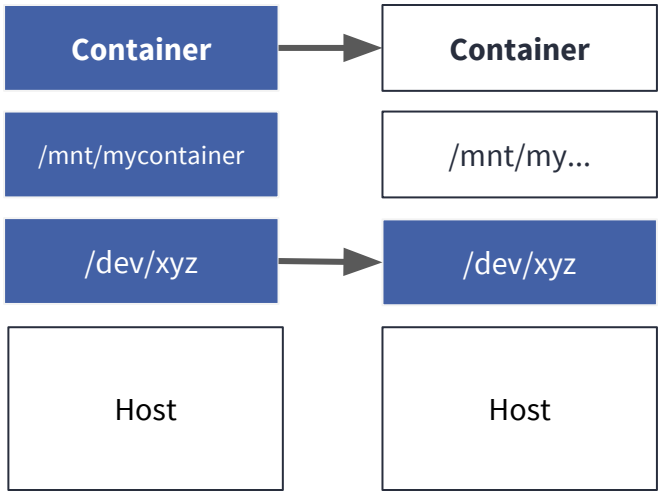
Options:

- NFS: doesn't scale, high-availability a problem in itself
- Probably want converged:
  - locality, homogeneous infrastructure
  - needs fault-tolerant storage though (Ceph, Quobyte)
- Object storage:
  - if you've designed your application accordingly
  - usually write once

Problem: State

Block storage (Ceph, EBS, ...):

- supply remote block device
- block device only accessible on one host

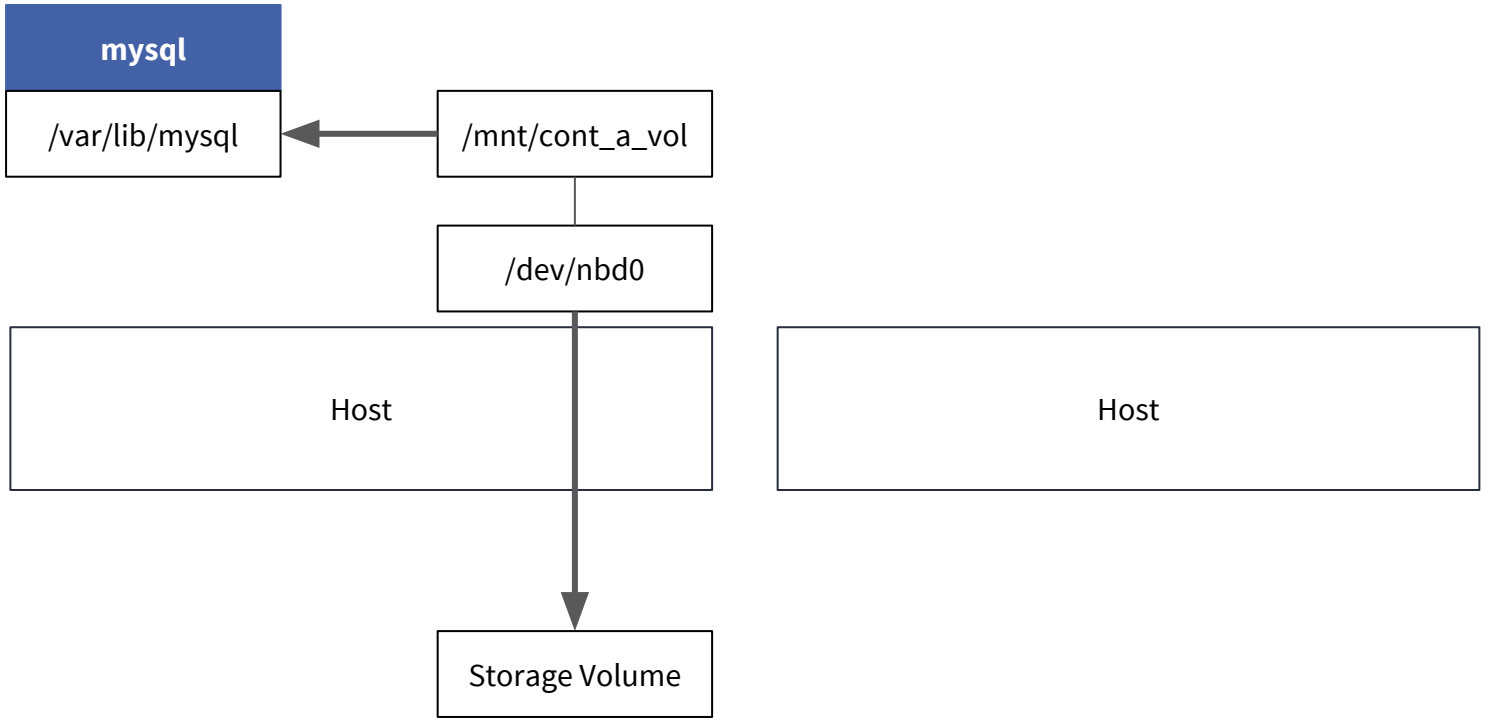


portworx

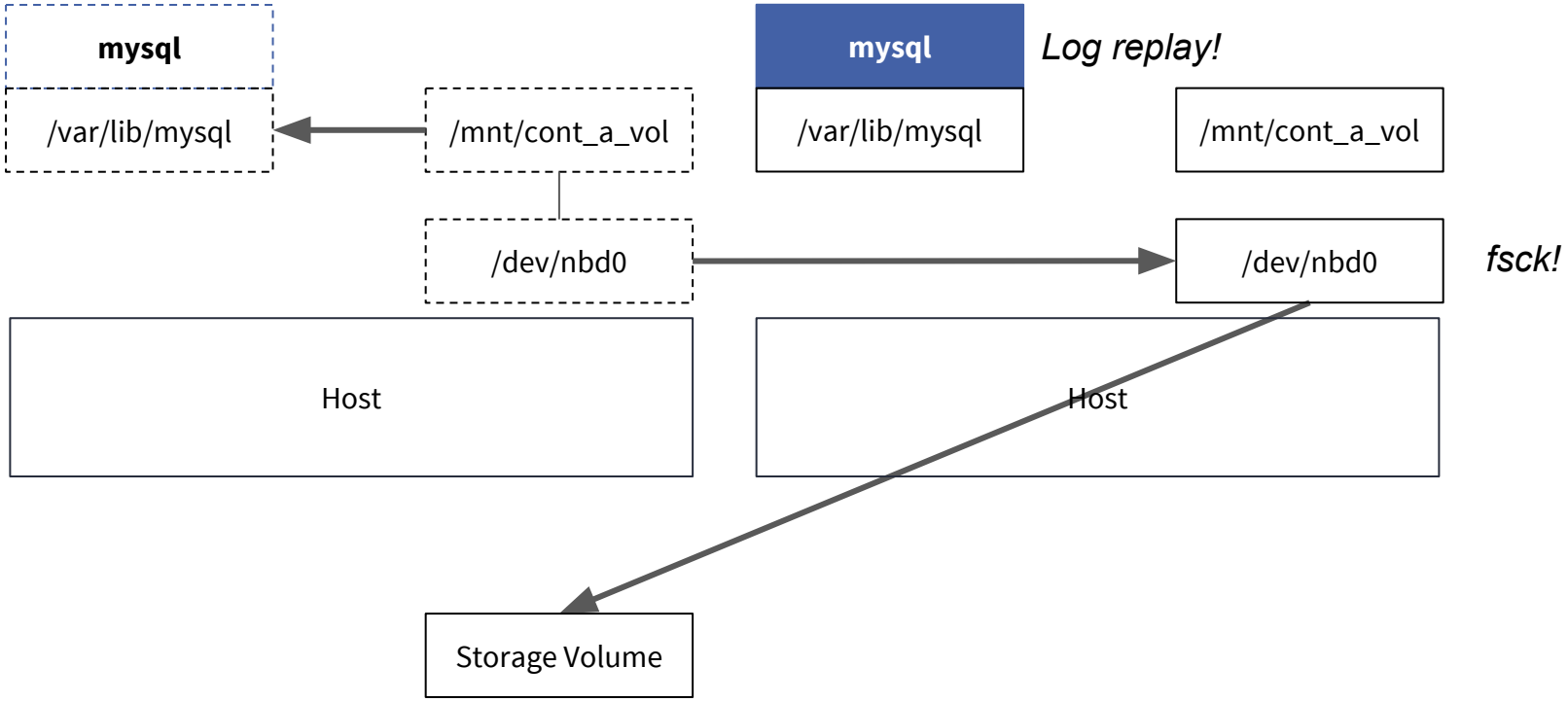
ceph

amazon web services EBS

Problem: State



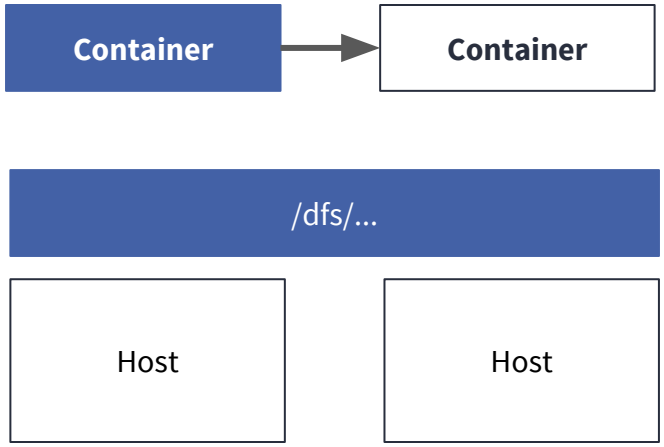
Problem: State



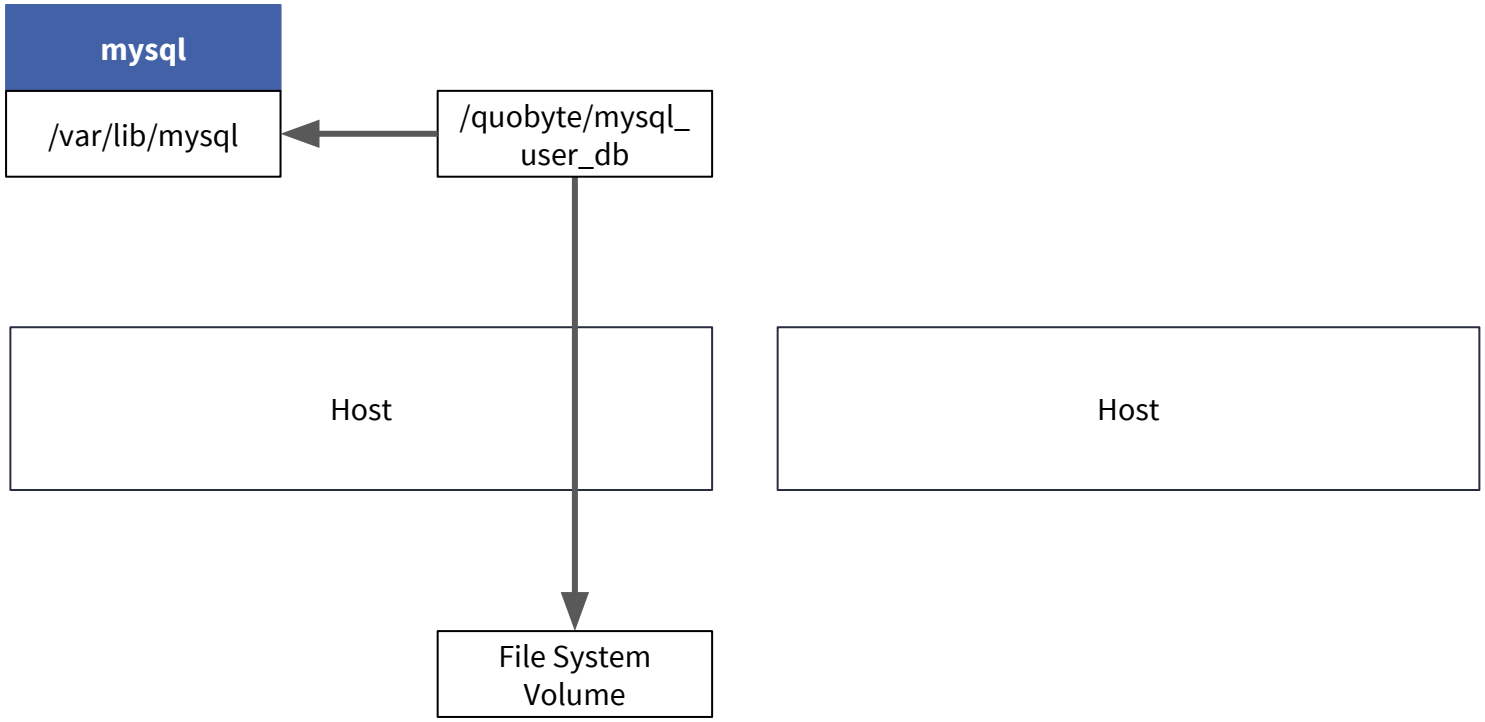
Problem: State

File systems (Quobyte, GlusterFS, HDFS, ...)

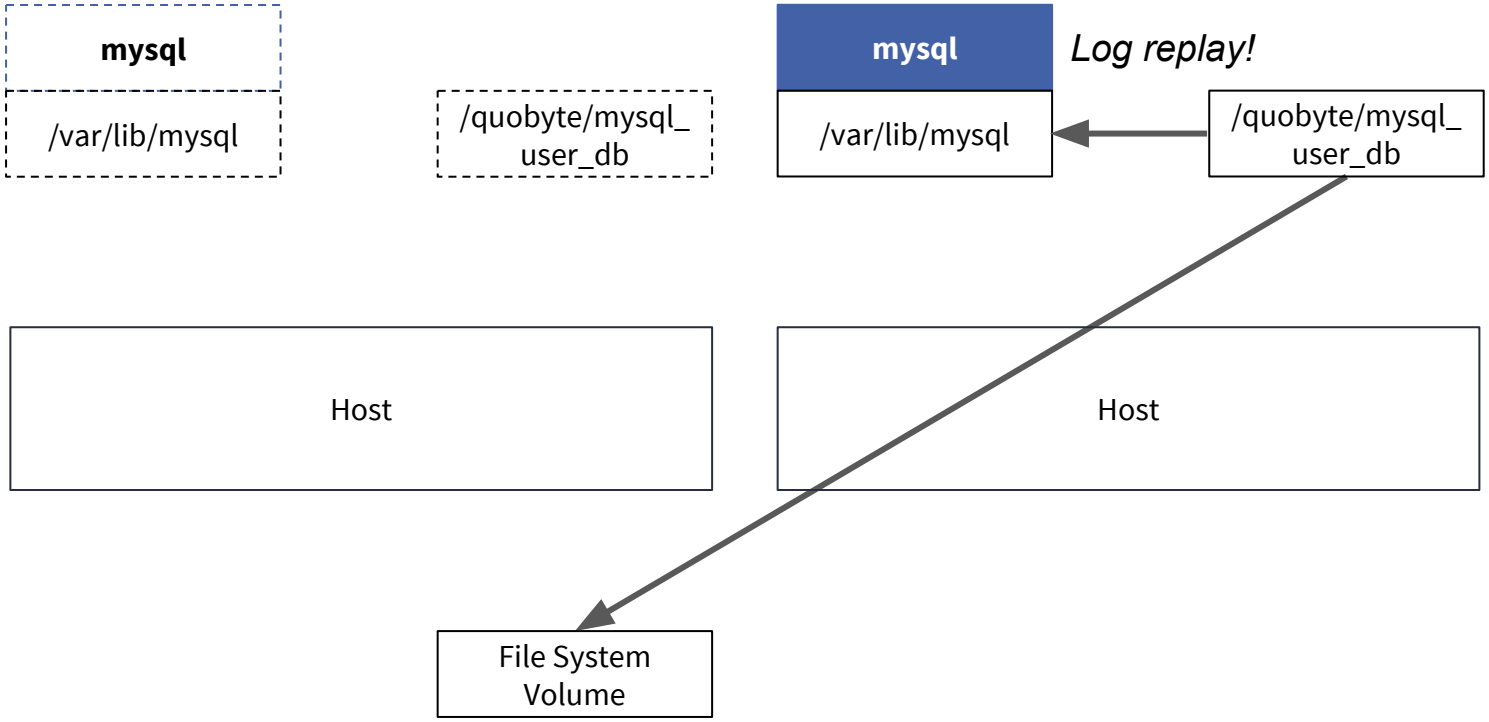
- all data everywhere
- if POSIX: drop-in replacement



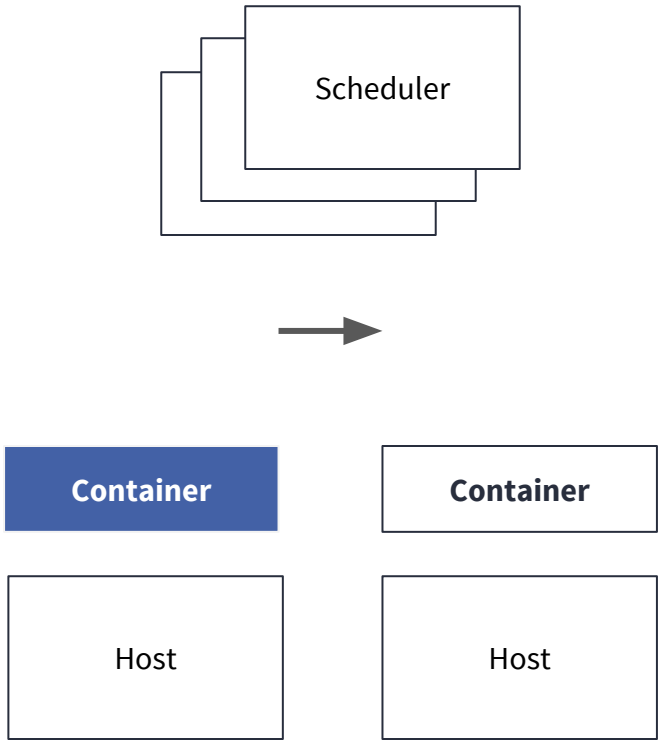
Problem: State



Problem: State

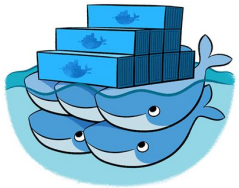


Problem: Rescheduling



Cluster Scheduler:

- decide hosts are dead (timeout)
- start container on live host
- needs to be fault-tolerant itself



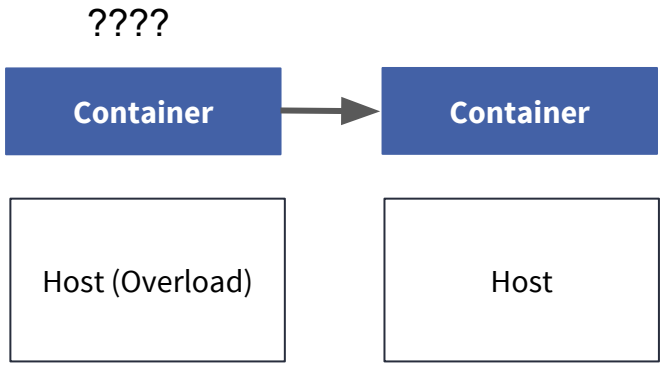
MESOSPHERE

MESOS

CoreOS

Nomad

Problem: Zombies



Scenario:

- failure detection by timeouts
- host detected to be failed
- reschedule
- two instances of same container access the same files
- potential for corruption

Problem: Zombies

Remedies:

- Kill all local containers if master could not be reached
- Careful aligned timeouts, but not safe
- Mutual exclusion via locking / leases
  - Use lock with timeout / lease to guard access, prevent concurrent access
  - Make application aware: block device management or file locking
  - block device: unmount, mount on other host, guard mount with lock
  - file system: Quobyte can automatically lock all open files transparently (implicit locking feature)
- Custom: Task instance numbers
  - instance suicide
  - not exploited yet

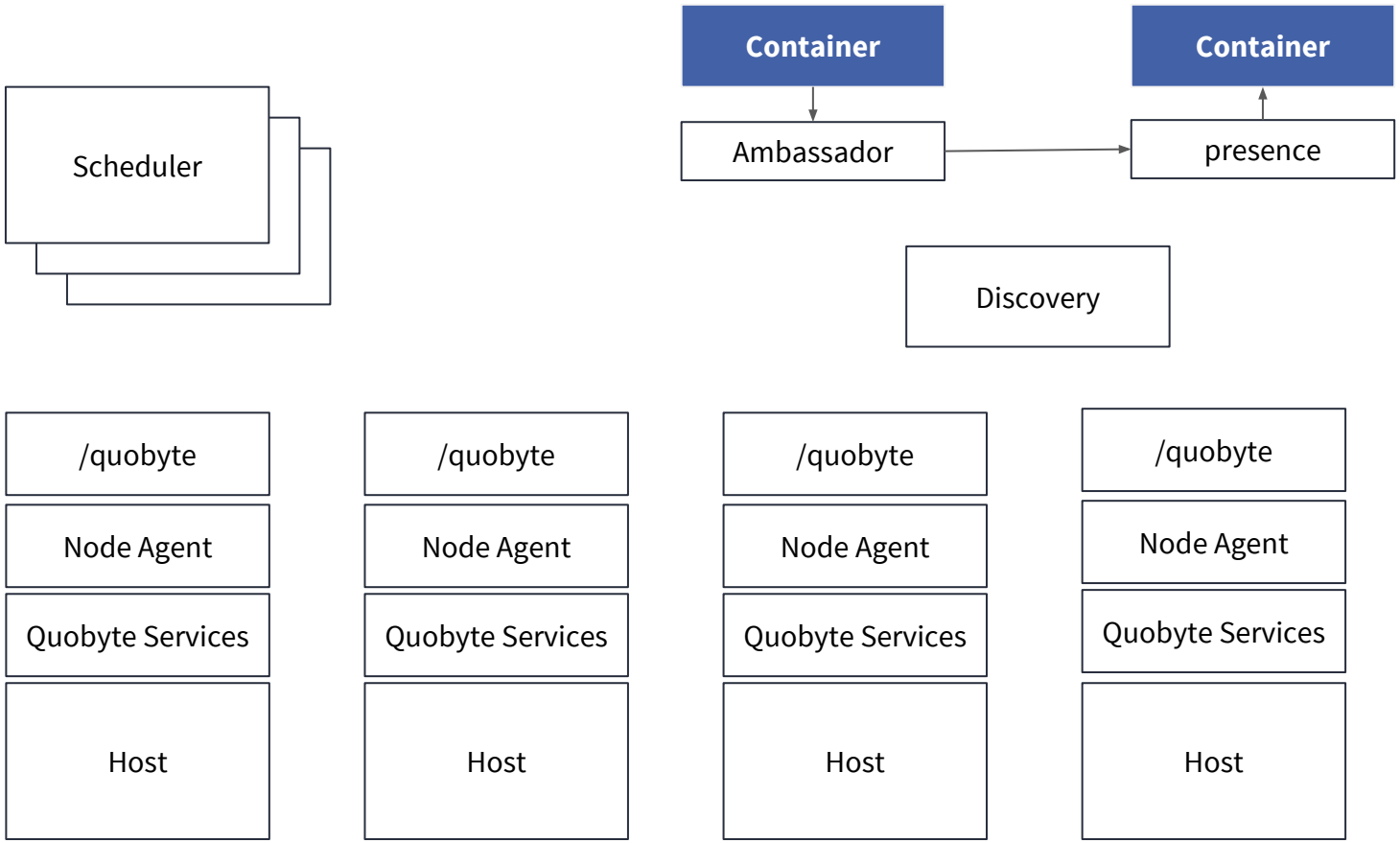
Problem: Discovery and Naming

Container is available - but where do I find it?

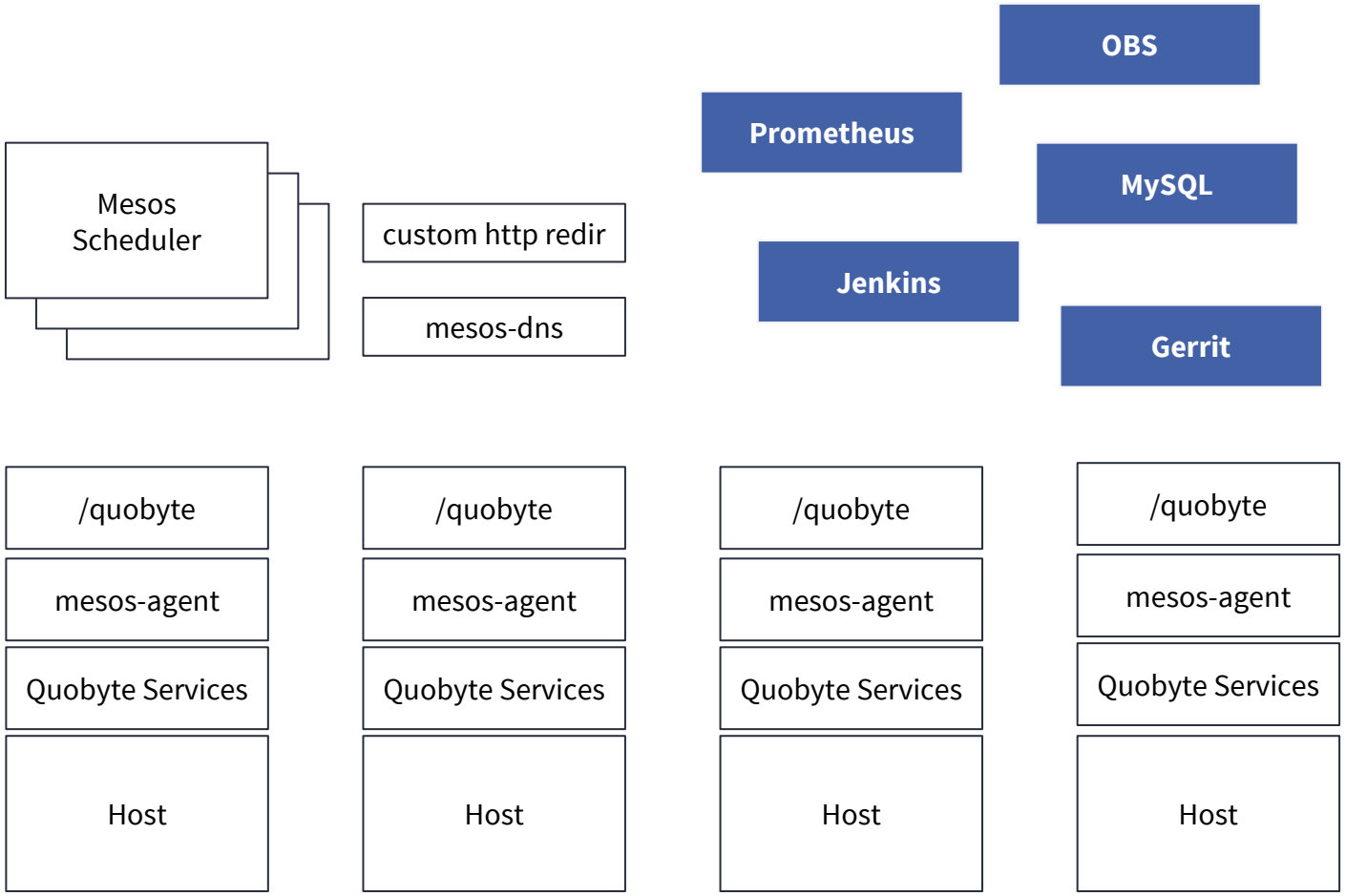
Dynamic discovery as locations change!

- Custom version: use lock server like Zookeeper or etcd
- DNS
  - mesos-dns
  - DNS failover is often slow due to caching
- HTTP
  - http APIs .. use http redirector
- General TCP-based protocols need sort of software-defined networking
  - Giantswarm Ambassador pattern
  - Mesos IP-per-Container

Container Infrastructure @



Container Infrastructure @ Quobyte

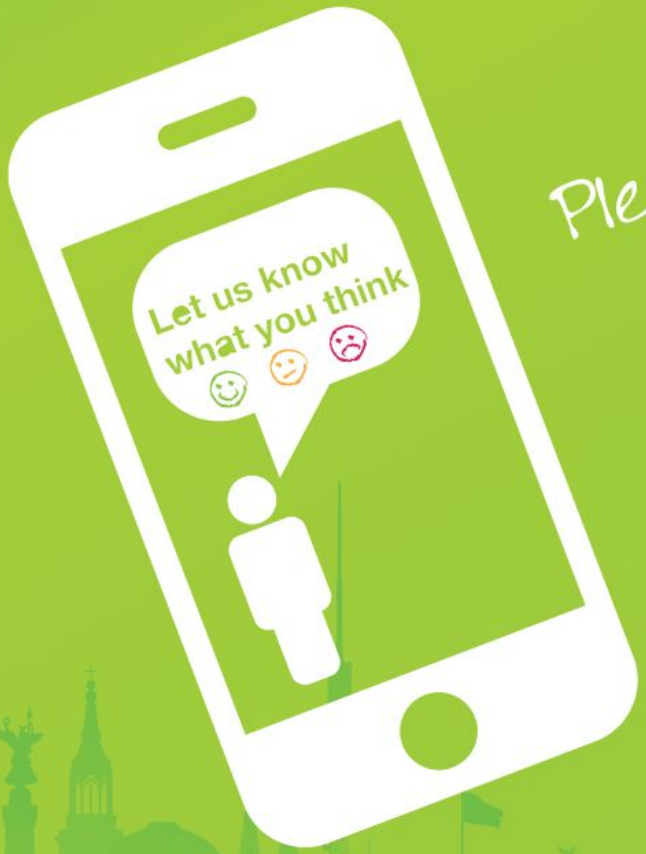


Conclusion

- Full fault-tolerance desirable for infrastructure of any size
- Containers and external state are a good base
- Many pieces are there:
  - containers
  - schedulers
  - storage systems
  - dynamic naming / SDNs
- and can be put together for fault-tolerance for many use cases

Felix Hupfeld
@drhu

goto; conference



Please

Remember to rate session

Thank you!

Follow us on Twitter @GOTOber

www.gotober.com