Converging QEMU and TCMU for Container Storage

...

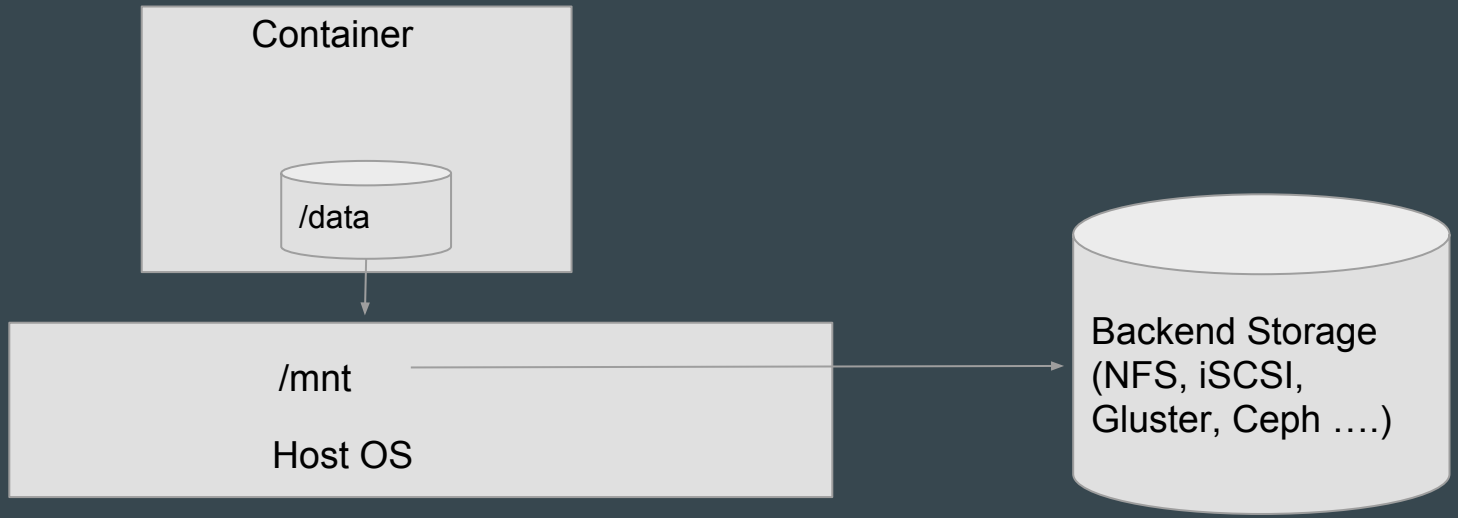
Huamin Chen
Red Hat
@root_fs
https://github.com/rootfs
https://huaminchen.wordpress.com/

Agenda

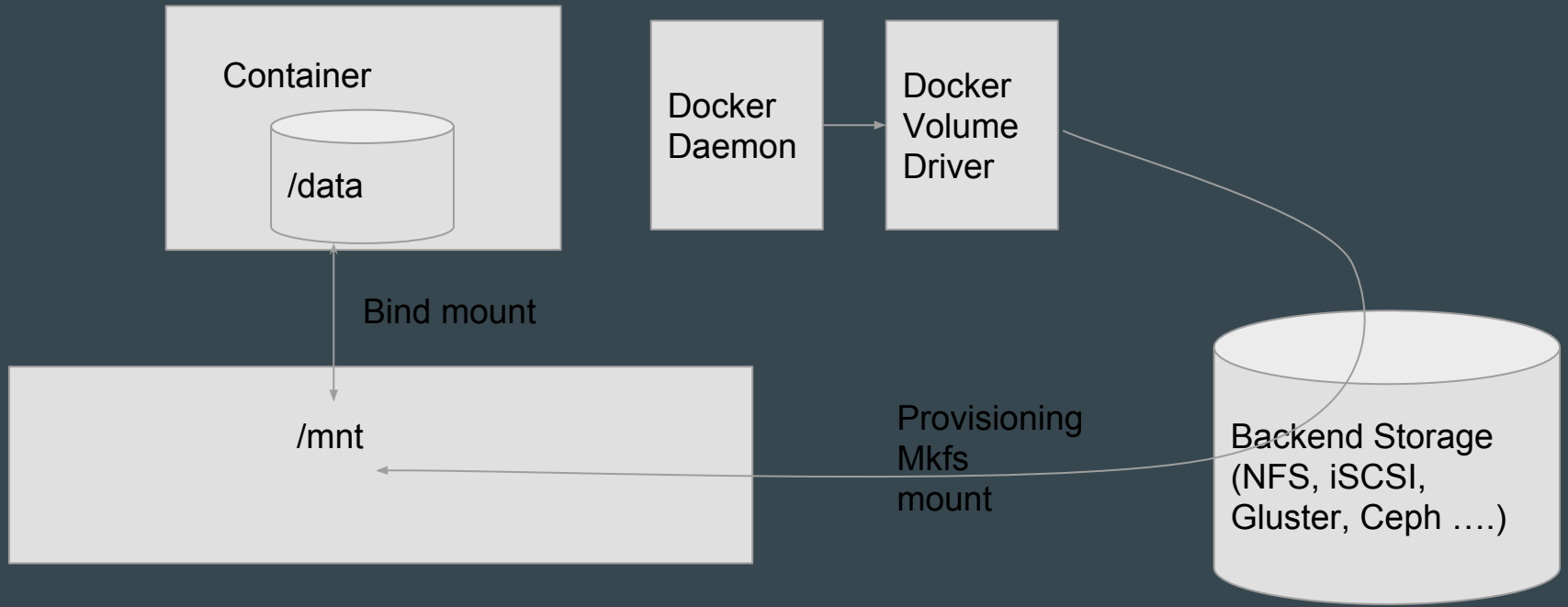
- Background
  - Collaboration with Andy Grover (TCMU) and Fem Zheng (QEMU)
  - Scope
    - Focused on on-premise Network Storage
  - Container Storage
    - Docker and Kubernetes
  - QEMU Block Drivers
- Why Converging?
- Related Works
- Solutions and Status

Docker Volume

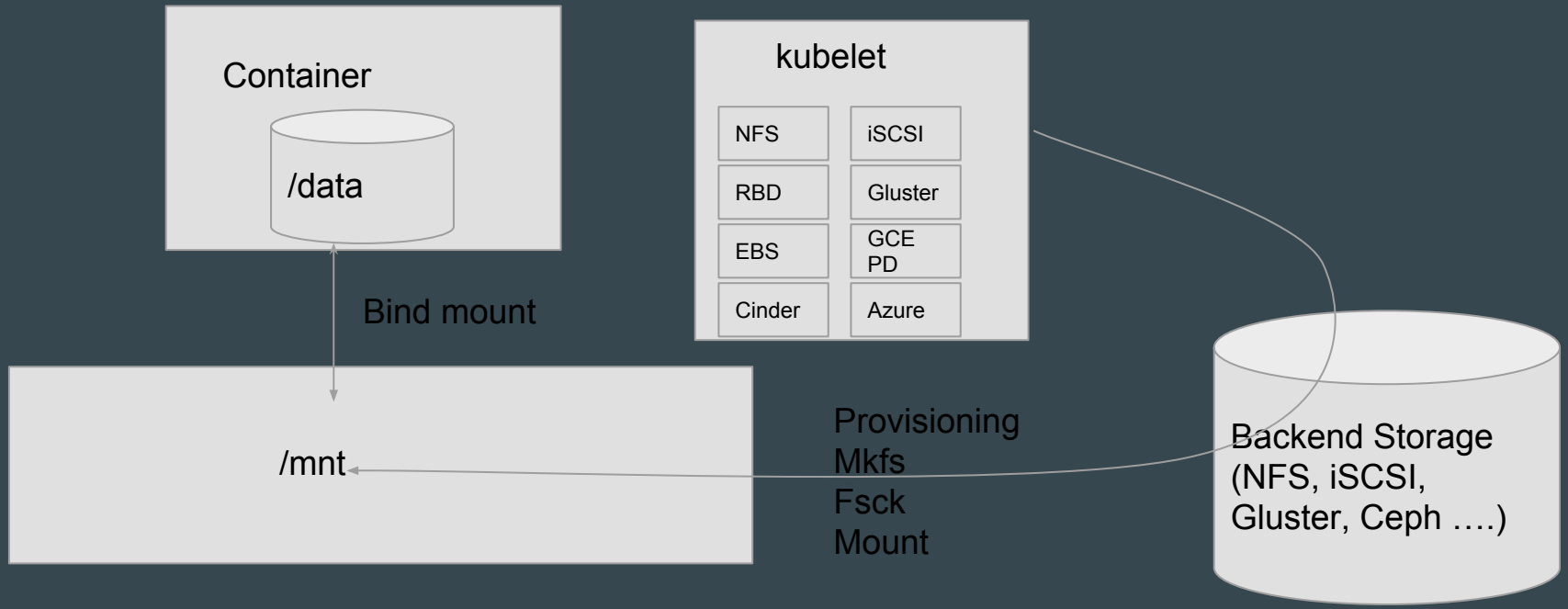
- docker run -v /mnt:/data centos df
- Container volumes are bind mount of those that mounted on the host



Docker Volume Driver Model



Kubernetes Volume Plugin Model



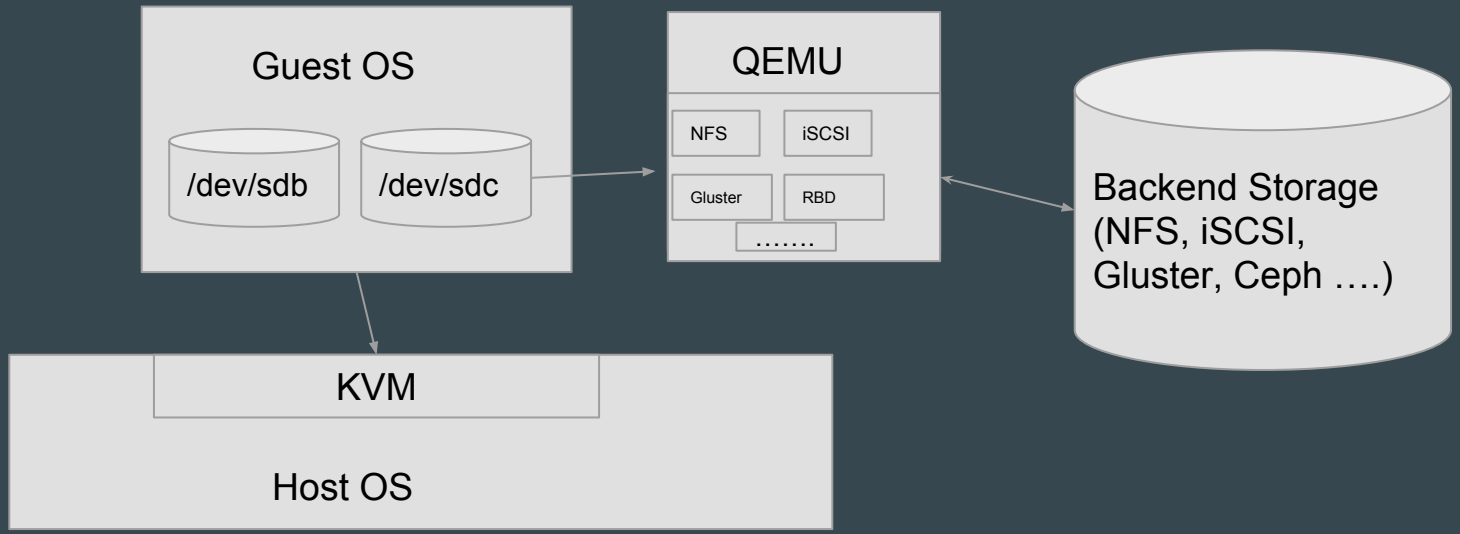
Feel the Pain?

- Last mile problem
  - Bringing easy storage access to clients is hard
  - No universal control or data plane exists
- Storage vendors need to develop, maintain, and support all orchestration frameworks
  - Kubernetes, Mesos, Docker, OpenStack, vSphere...
- Orchestrators carry long lists of drivers
  - Most drivers do the same thing inside the orchestrators
  - Extensive integration tests
  - Framework level changes are often visible to and impact on all drivers

Why Converging with QEMU?

- Easy data sharing.
  - There are plenty of qcow2 and vmdk images around. Enabling Containers directly to access these images eliminates data migration complexities.
- Deduplicate Driver Development
  - QEMU already supports many storage drivers that Docker and Kubernetes try to support
- Feature Enablement
  - Provisioning
  - Snapshot/restore
  - Resize
  - Encryption
  - Compression
  - Throttling
  - Replication
- Performance
  - Some QEMU block drivers perform better than direct mount.
    - Gluster: no more FUSE bypassing overhead

QEMU Block Drivers



Related Works

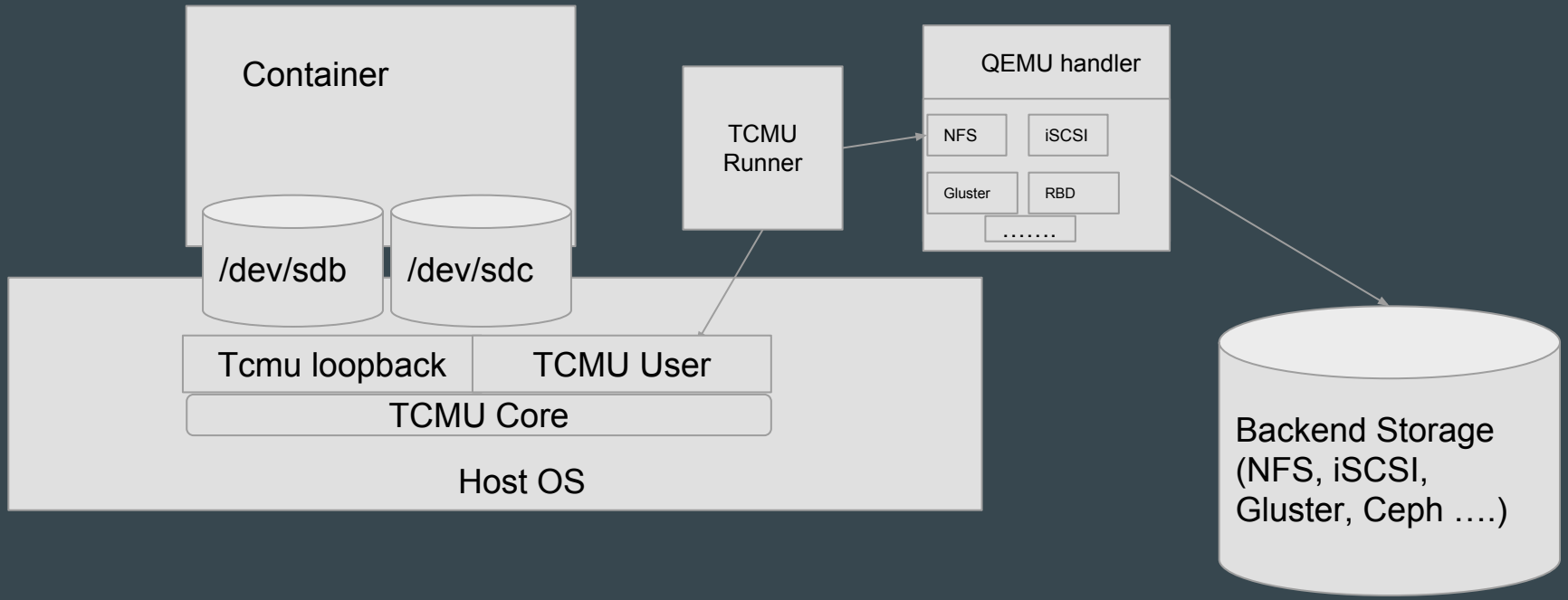
- Network Block Device (nbd): lack of drivers and features

- Ploop (OpenVZ): Kernel-only model; also lack features

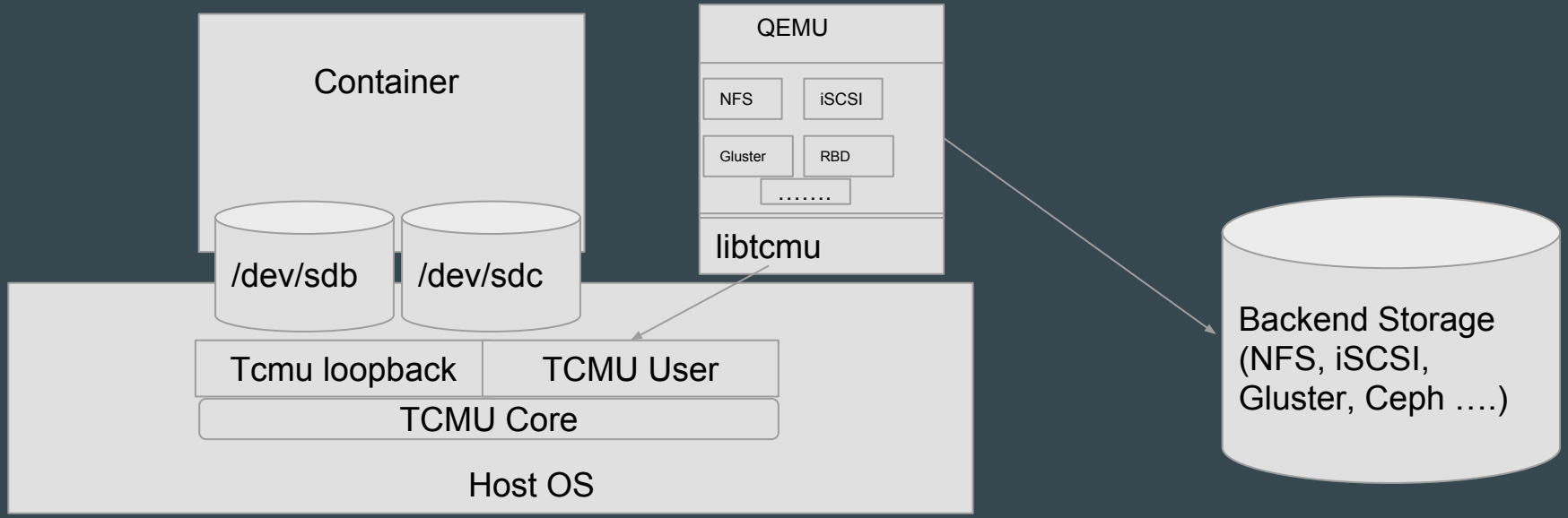
TCM(U)

- TCM (aka LIO) is modularized SCSI target layer separating backend storage and fabrics
- TCMU handles userspace backend

Solution 1: TCMU Centric



Solution 2: QEMU Centric



QEMU-TCMU Integration Status

- Solution 1
  - Decouple QEMU block layer from QEMU monolithic binary
    - Generate a qemu block shared library
  - Implement a QEMU block device handler for tcmu-runner
  - Create TCMU loopback device via targetcli
  - Bind mount to Docker Container

Future Work

- A new standalone qemu-tcmu command
- Support it in frameworks like Kubernetes, Mesos and Docker
- Support it in fabrics like iSCSI, FC, NVMe-OF