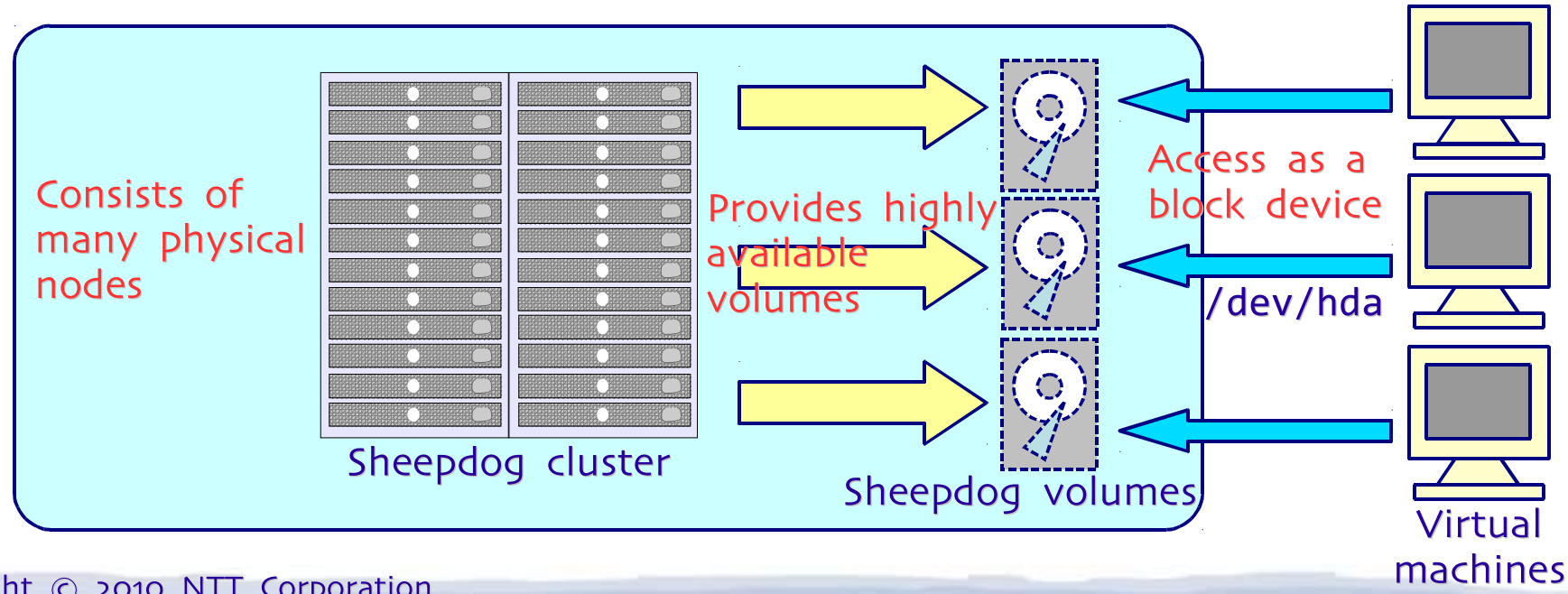
Sheepdog: Distributed Storage System for QEMU/KVM

Kazutaka Morita
NTT Cyber Space labs.

19 January, 2010

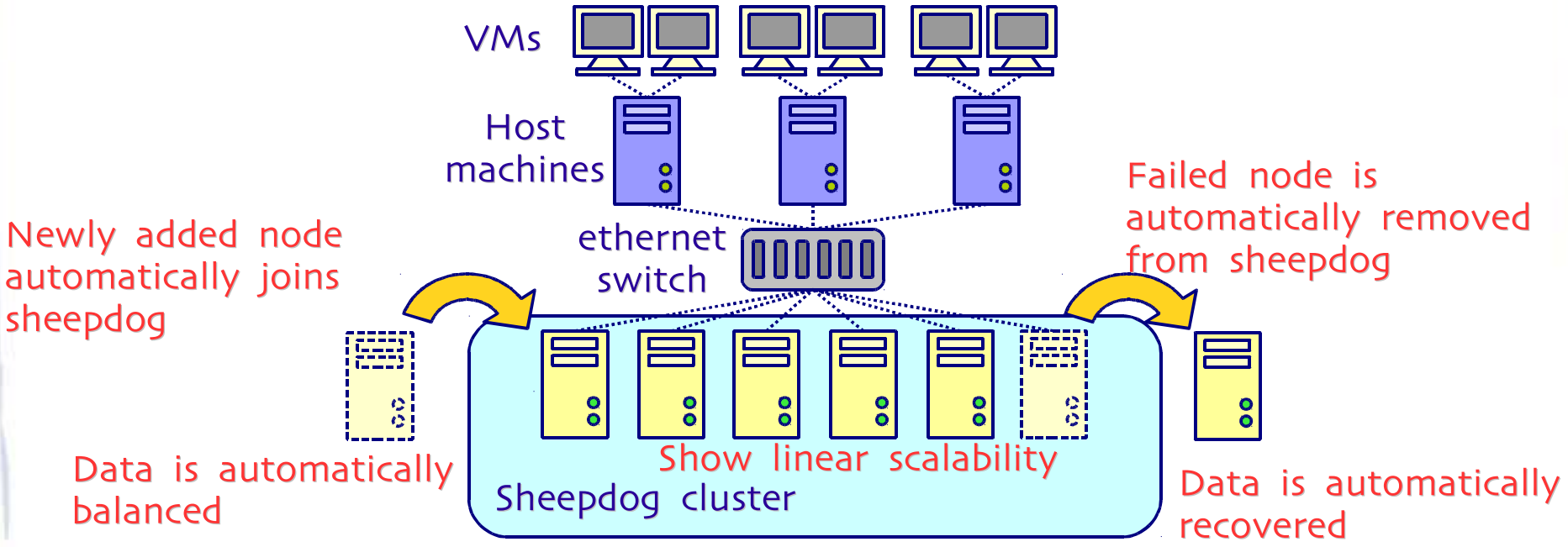
What is Sheepdog?

- Distribute storage system for QEMU/KVM
  - Amazon EBS-like volume pool
  - Highly Scalable, available, and reliable
  - Support for advanced volume management



Architecture: fully symmetric

- Zero configuration about cluster nodes
  - Automatically detect added/removed nodes
  - Similar to Isilon architecture

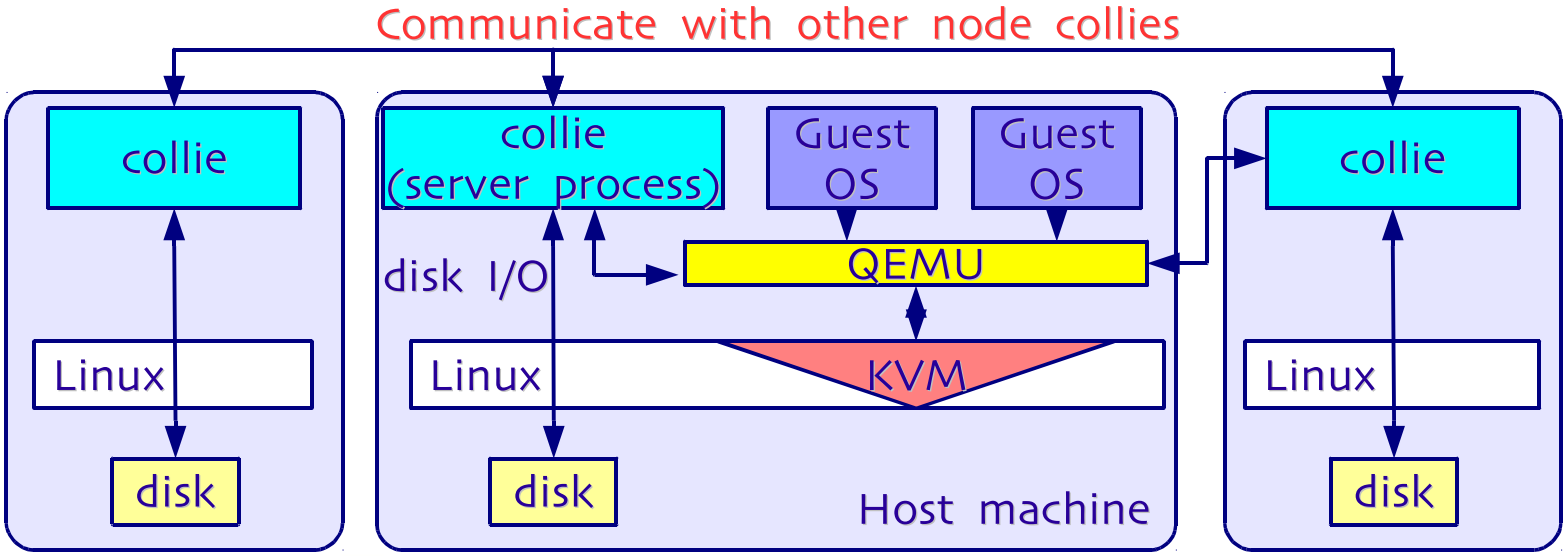


Goals

- Be managed autonomously
  - Automatic data relocation and load balancing
- Scale to several hundreds nodes
  - Linearly scale in performance and capacity
- Provide highly available/reliable volumes
  - Data is replicated to multiple nodes
  - Lost data will be automatically recovered
- Support advanced volume management
  - Snapshot, cloning, and thin provisioning

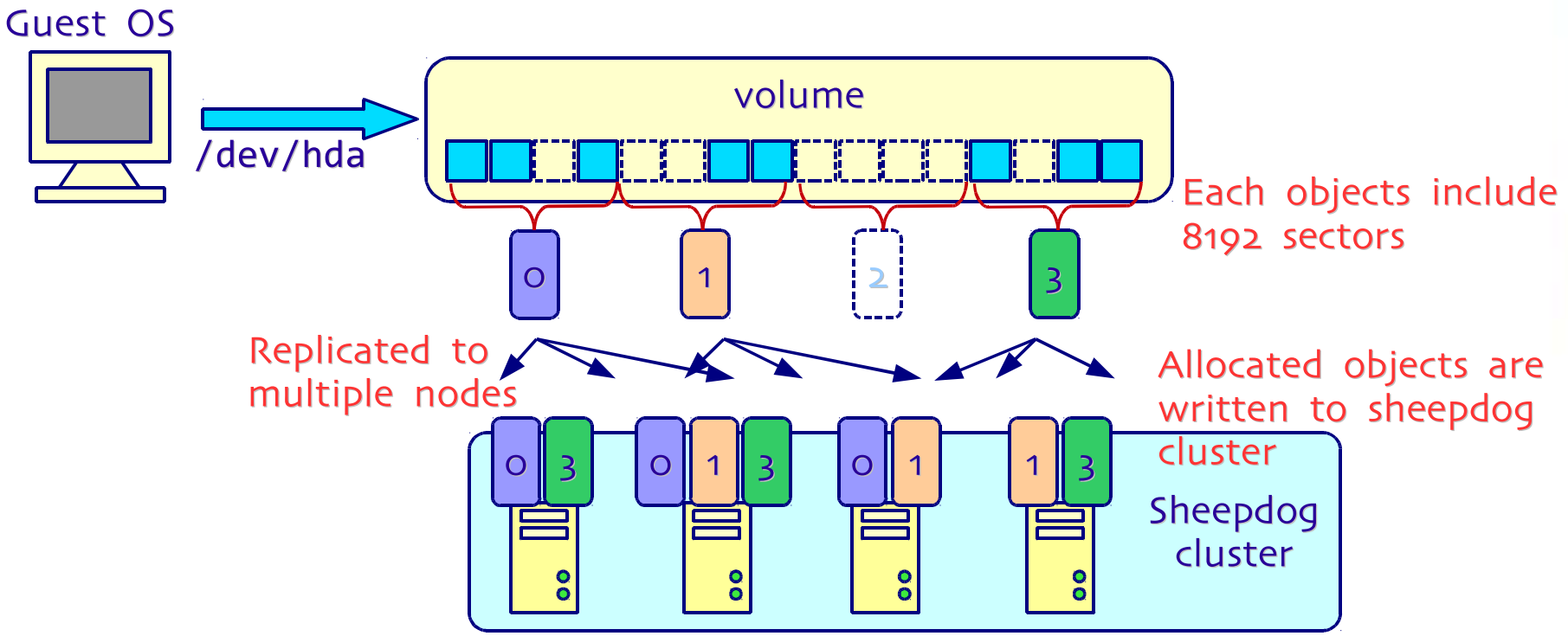
Design: not general file system

- We have simplified the design significantly
  - API is designed specific to QEMU
  - We cannot use sheepdog as a file system
  - One volume can be attached to only one VM at once



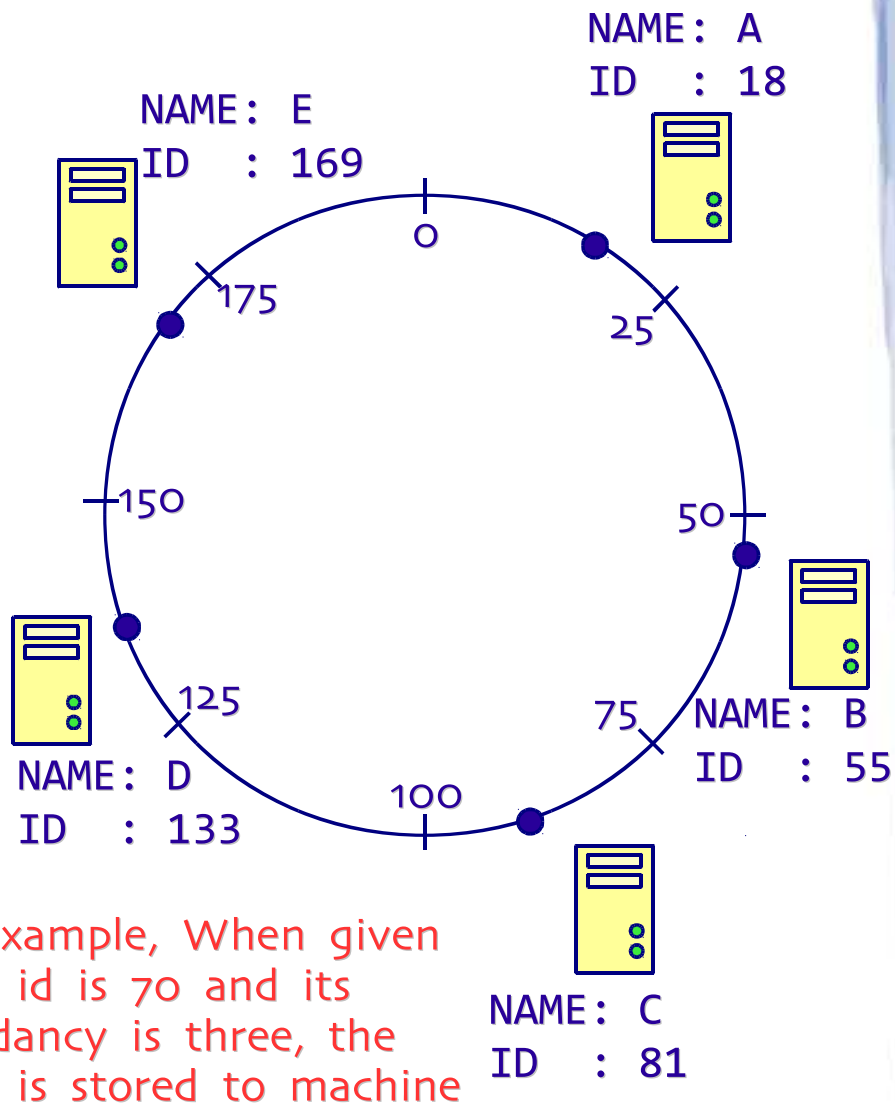
How to store volumes?

- Volumes are divided into 4 MB objects
  - Each object is identified by globally unique 64 bit id, and replicated to multiple nodes



Where to store objects?

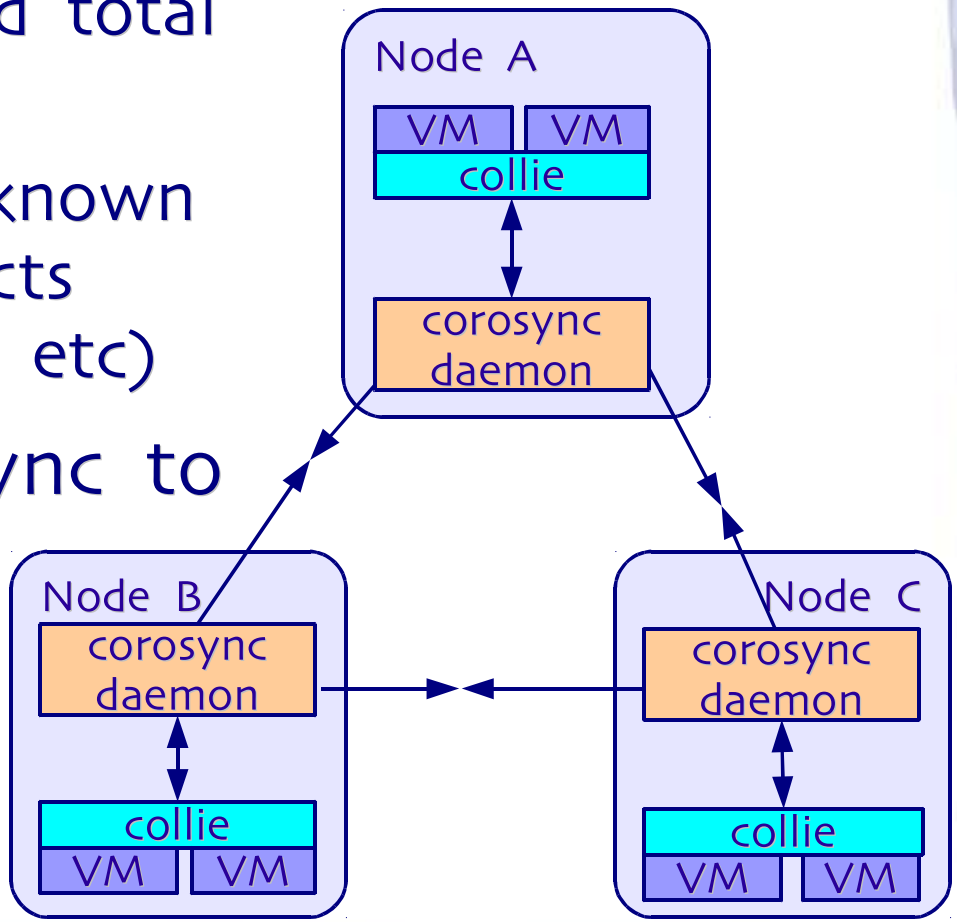
- We use consistent hashing to decide which node to store objects
  - Each node is also placed on the ring
  - addition or removal of nodes does not significantly change the mapping of objects



For example, When given object id is 70 and its redundancy is three, the object is stored to machine B, C, and D.

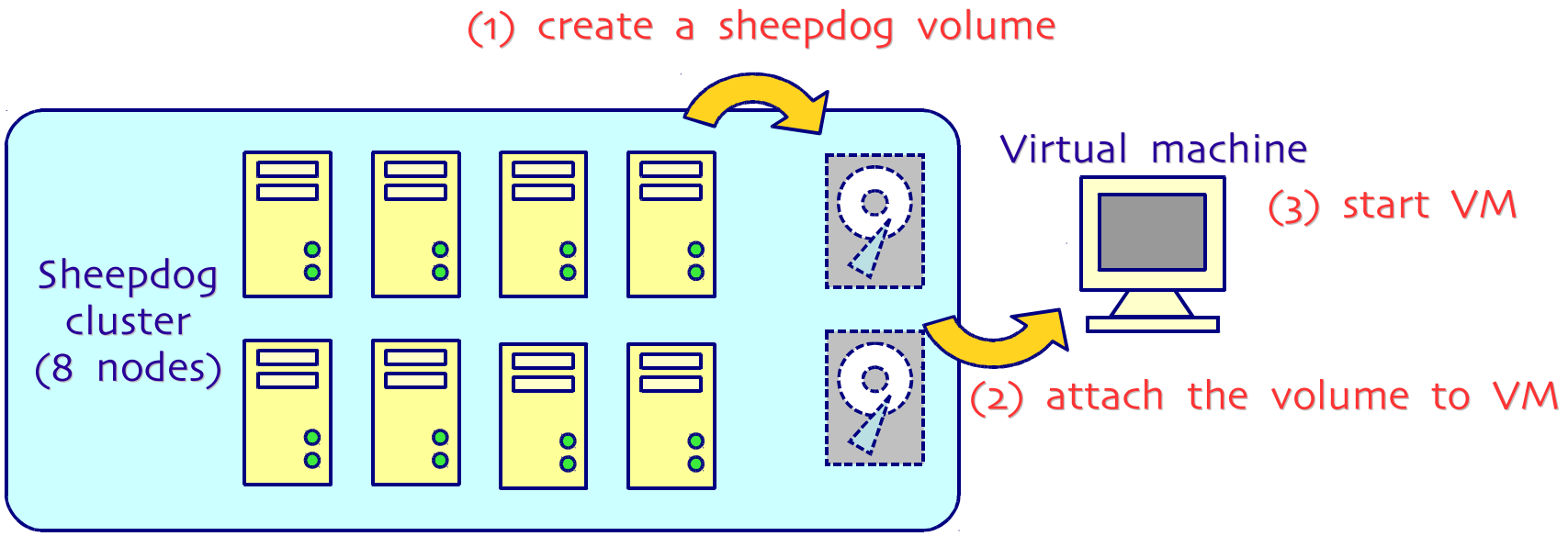
Cluster node management

- corosync
  - Supports reliable and total ordered multi-cast
  - Is adopted by well-known open source projects (Pacemaker, GFS2, etc)
- Sheepdog uses corosync to support
  - lock/release VDI
  - join/leave message
  - master election



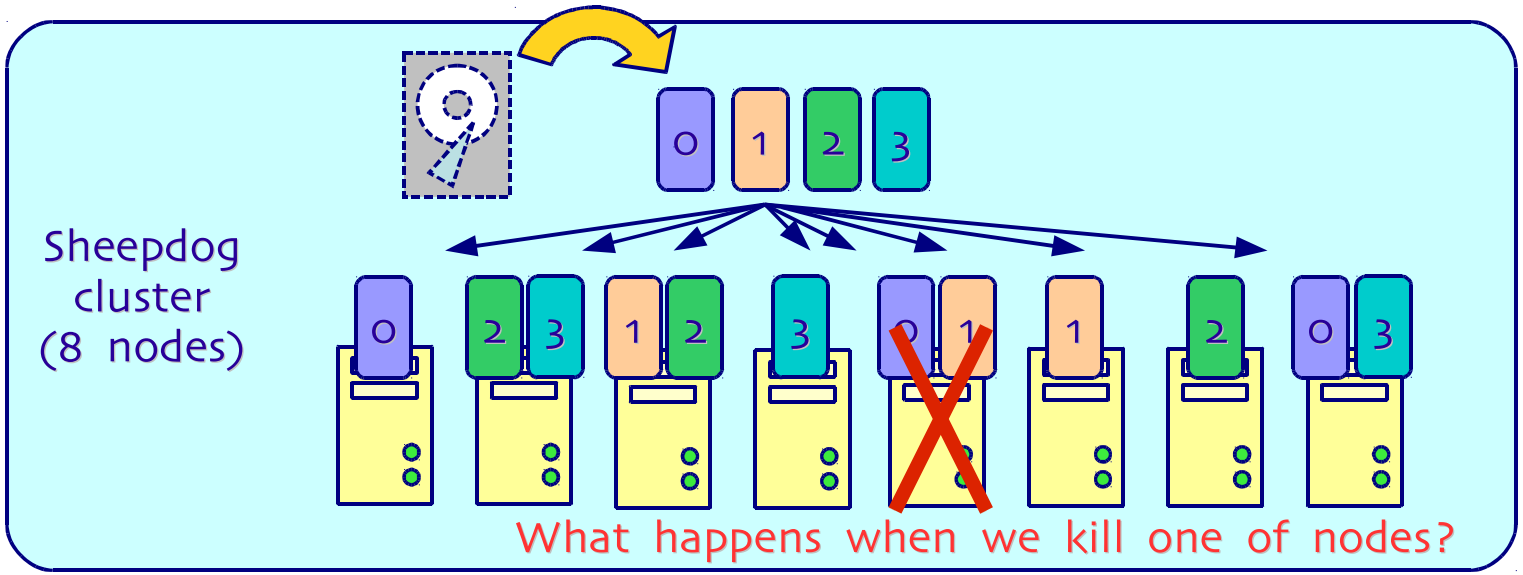
Demonstration (1)

- Let's see how easily you can create sheepdog volumes and attach them to VMs
  - Sheepdog works with virt-manager (libvirt GTK front-end)



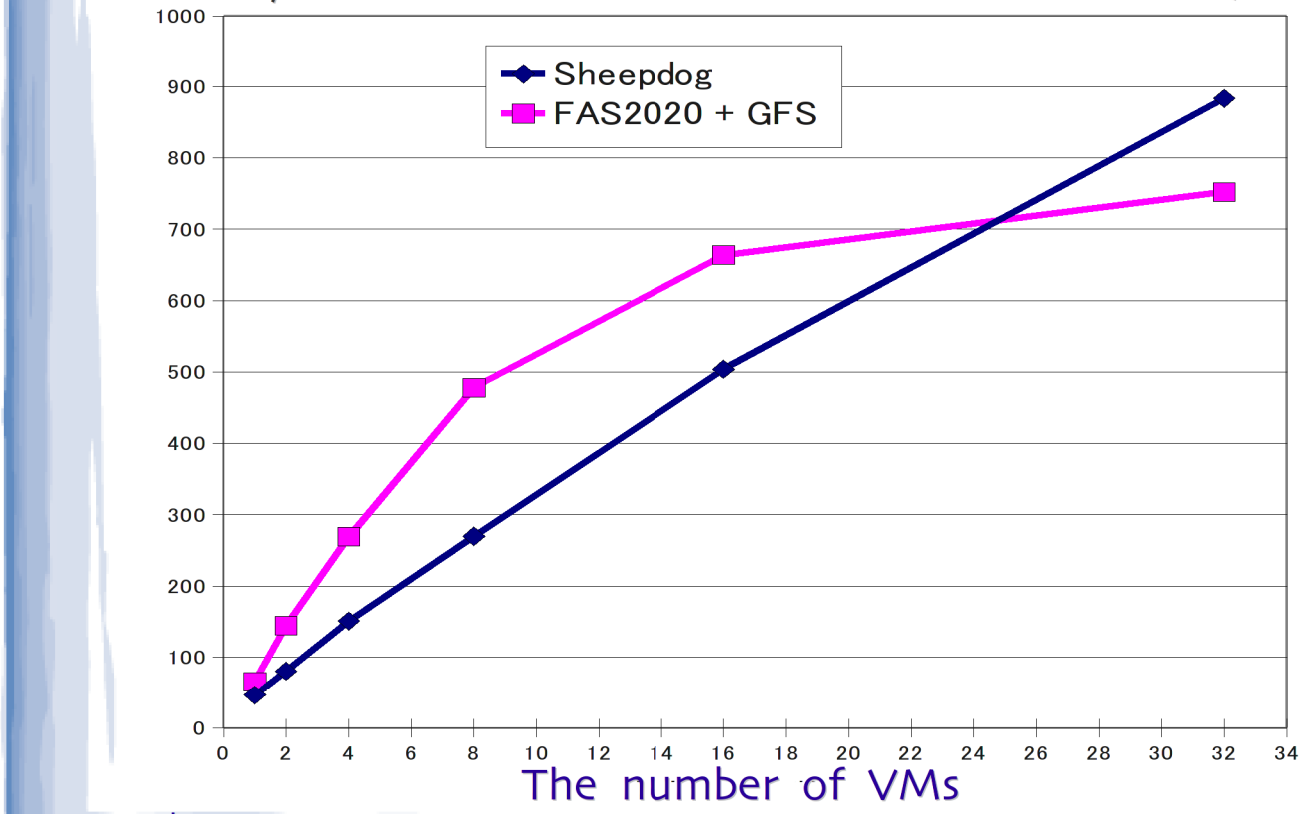
Demonstration (2)

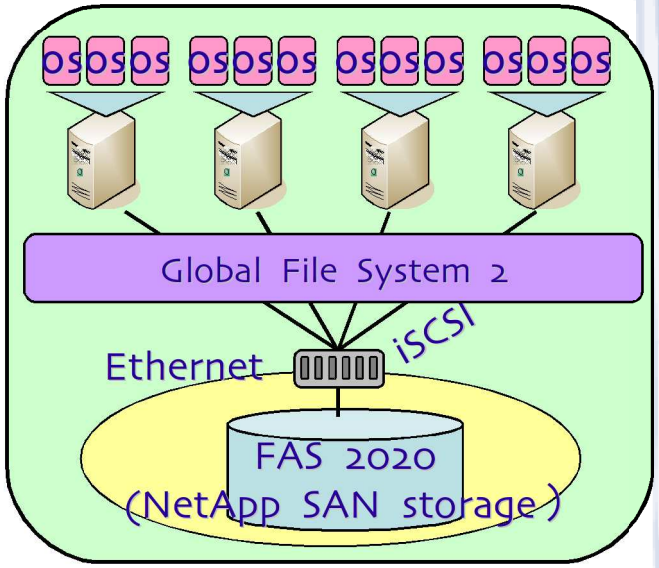
- Let's see what happens when one of sheepdog nodes fall down
  - Sheepdog volumes are divided into 4 MB objects
  - Each objects are replicated to 3 nodes



Scalability evaluation

- Compared with shared storage
  - Blogbench: benchmark tool to reproduce the load of a real-world busy file server





CPU : Core2 Quad 2.4GHz
Memory : 1 GB
Network : 1 Gbps
Disk : SATA 7200 rpm
number of nodes (Sheepdog) : 16
Data redundancy (Sheepdog) : 3

In early development stage

- Supported features
  - Volume snapshot, cloning
  - Live migration
  - Store objects redundantly
  - Scales to tens nodes
- Planed features (aka TODO)
  - volume deletion
  - online snapshot from qemu-monitor
  - Better load balancing
  - scalability improvement

Conclusion

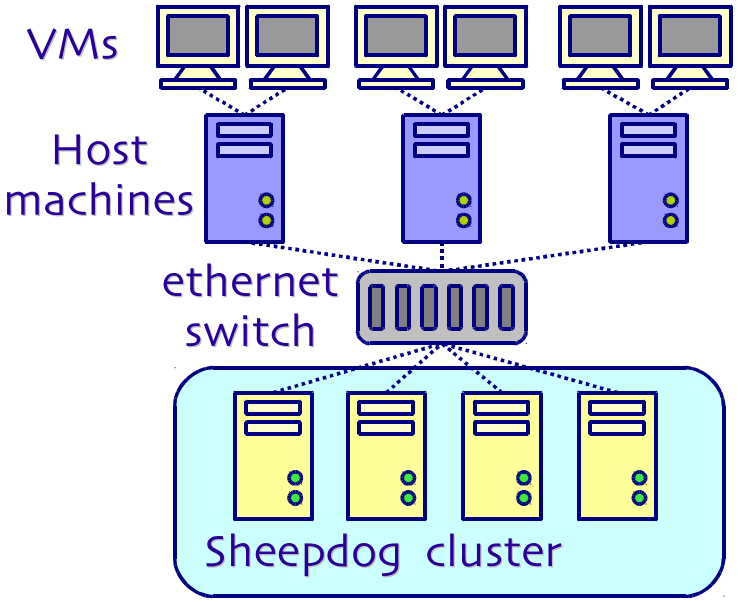
- Sheepdog is highly available storage pool for QEMU/KVM
  - We hope Sheepdog will become the de facto standard of cloud storage system
- Further information
  - Project page
    - http://www.osrg.net/sheepdog/
  - Mailing list
    - sheepdog@lists.wpkg.org

Enjoy!

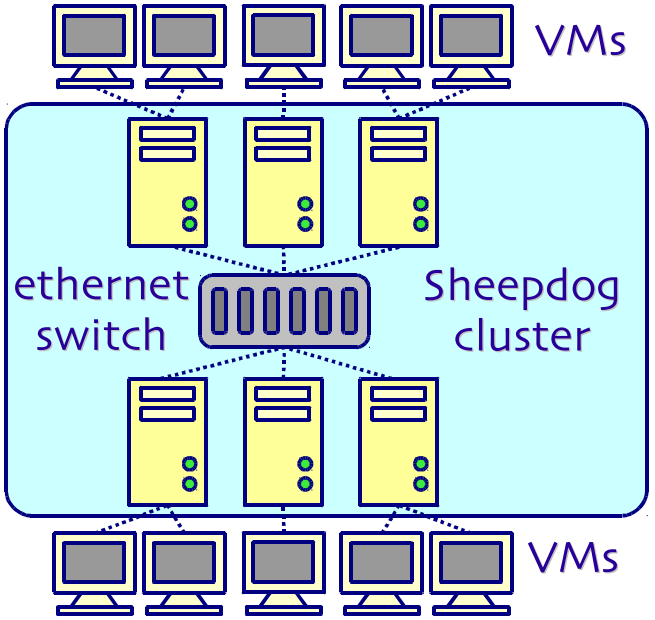
Appendix

Usage

- Alternative to existing network storage
- Storage system of a virtual infrastructure



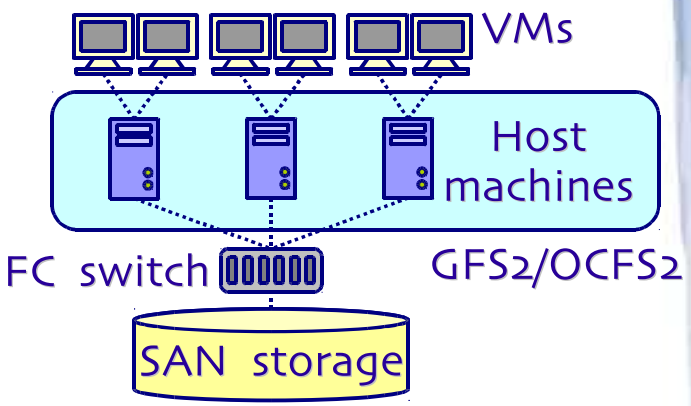
Use sheepdog as a network storage

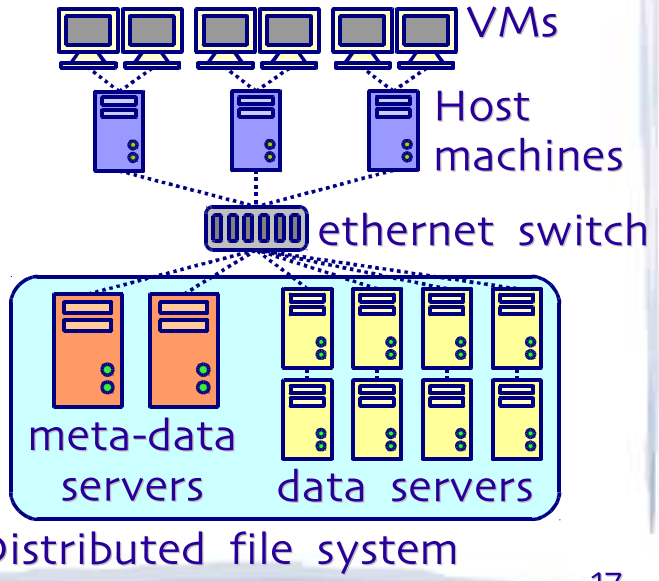


Use sheepdog as a virtual infrastructure

Why another storage system?

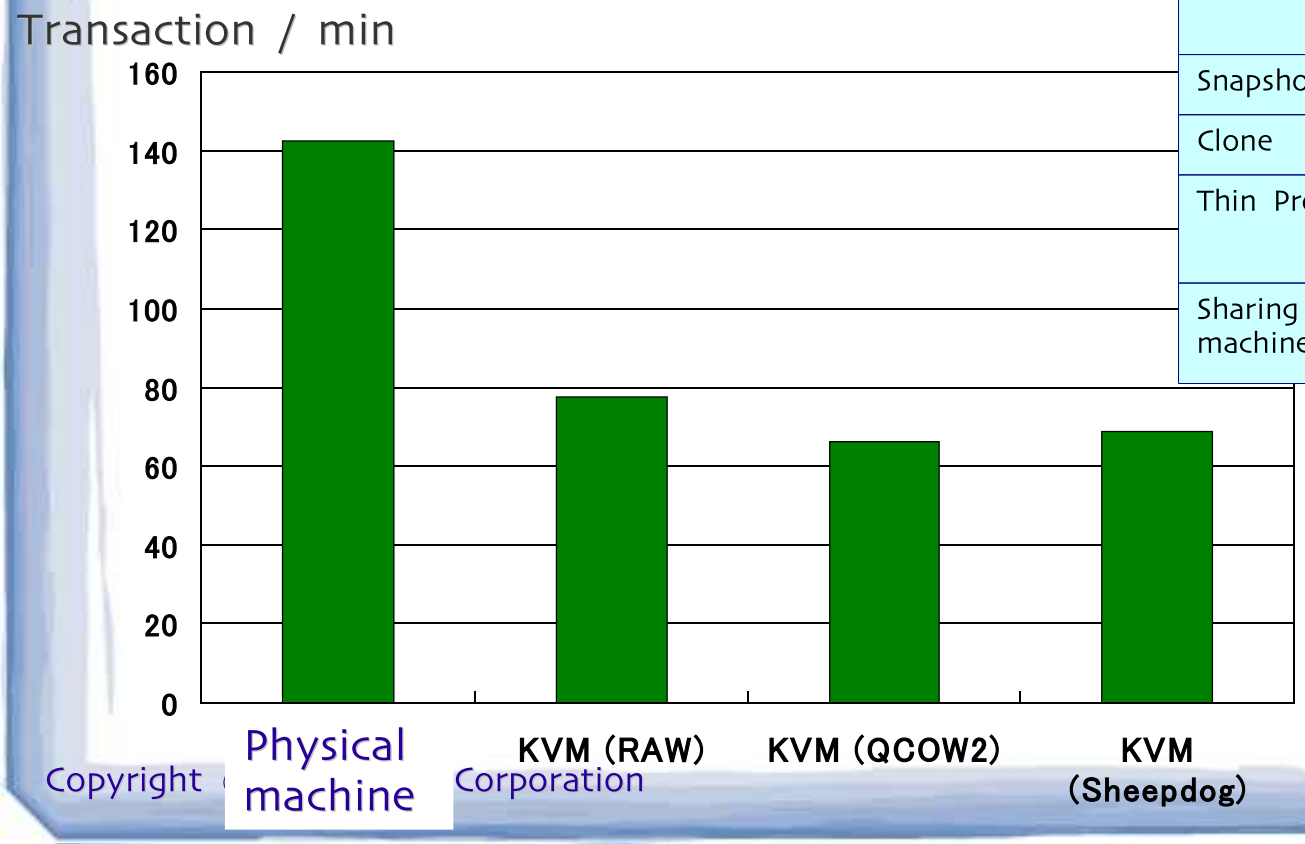
- Why not SAN storage?
  - Large proprietary storage system is too expensive
  - Shared storage could be a single point of failure
- Why not distributed file systems?
  - e.g. Luster, GlusterFS, Ceph
  - Complex configuration about cluster membership, each node role, etc.





Performance evaluation

- Compared with other formats (qcow2, raw)
  - Blogbench: benchmark tool to reproduce the load of a real-world busy file server



	RAW	QCOW2	Sheepdog
Snapshot	×	○	○
Clone	×	○	○
Thin Provisioning	×	○	○
Sharing with other machines	×	×	○

CPU : Core2 Quad 2.4GHz
Memory : 1 GB
Network : 1 Gbps
Disk : SATA 7200 rpm
number of nodes (Sheepdog) : 8
Data redundancy (Sheepdog) : 3

Road map (1)

- Short-term goals (in few month)
  - volume deletion
  - online snapshot from qemu-monitor
  - Support libvirt
  - Support EBS API
  - Support architectures other than i386, x86_64
  - Get delta between snapshots
  - more documentation

Road map (2)

- Long-term goals (in one or two years)
  - performance improvement
  - more scalability
  - guarantee reliability and availability under heavy load
  - tolerance against network partition (split-brain)
  - load balancing corresponding to I/O, CPU, memory load