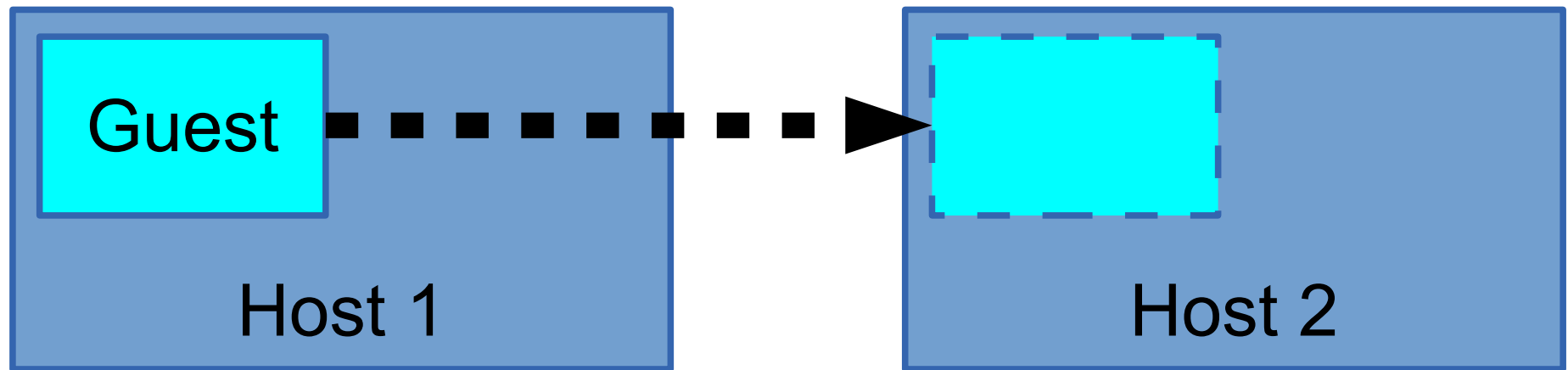


Virtual Machine Migration

Network Startup Resource Center

Migration

- Moving a VM from one host to another



Applications

- Load balancing
 - Move VMs to a less busy host
 - Make use of newly-added capacity
- Maintenance
 - Move VMs off a host before it is shut down
- Recovery from host failure
 - Restart VM on a different host

Types of migration

- Cold migration
 - Shutdown VM on host 1, restart on host 2
- Warm migration
 - Suspend VM on host 1, copy across RAM and CPU registers, continue on host 2 (some seconds later)
- Live migration
 - Copy across RAM *while VM continues to run*
 - Mark "dirty" (changed) RAM pages & re-copy
 - Brief suspension for final copy (<< 1 sec)

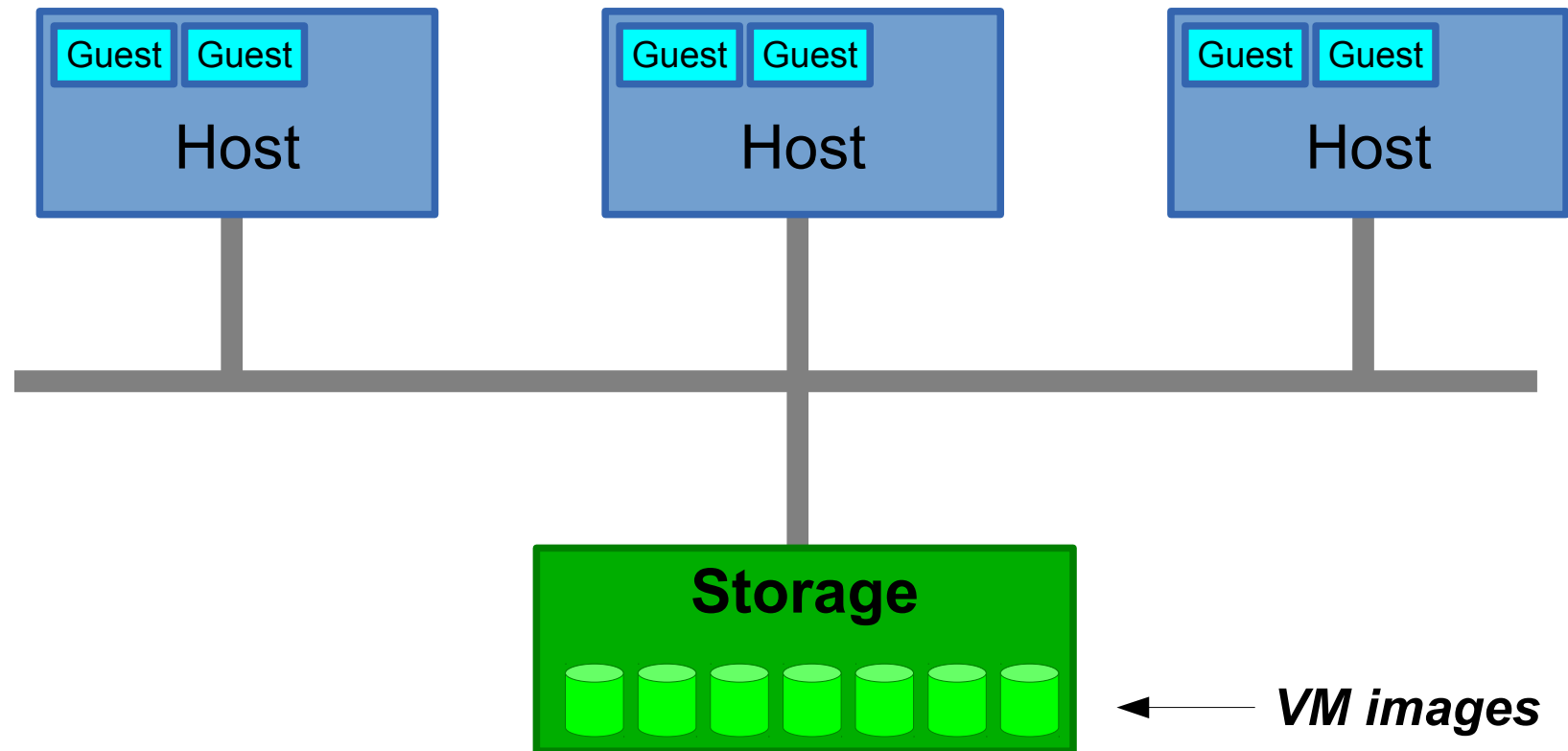
Migration and libvirt

- It does work, but not the ideal tool
- libvirt manages individual hosts, so it doesn't have a cluster-wide view
 - by default won't prevent the same VM starting up in two places at once (very bad!)
 - "sanlock" plugin available
 - can end up with multiple copies of the XML definition file on different hosts

Migration and Storage

- The VM disk image has to be accessible from the new host after the migration
- Just copy the image across?
 - Slow
 - Fine for a cold migration though
- Can we do a "live migration" of storage?
 - Yes (e.g. very recent versions of kvm can do this)
 - Risky
 - Doesn't help recover from node failure

Traditional solution: shared storage



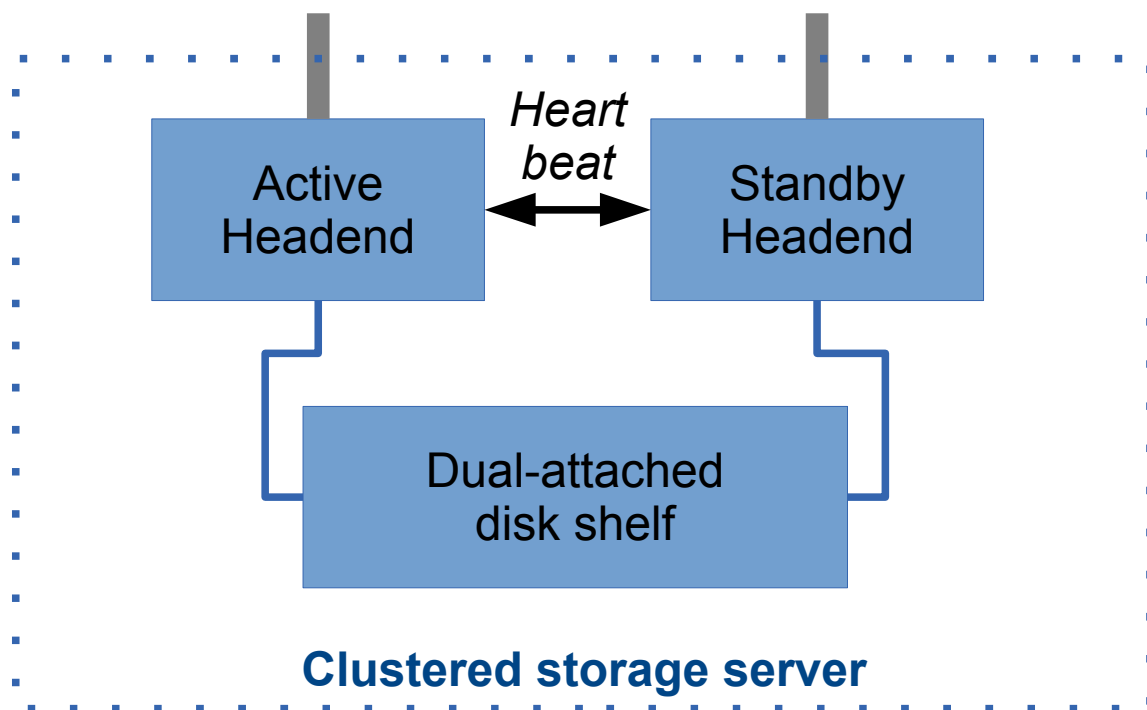
Advantages of shared storage

- Complete mobility of VMs with live migration
- Can scale the compute nodes and the storage nodes independently
- Simpler compute nodes
 - little or no local storage required
- Central point of volume management
- Central point of backup / DR

Disadvantages of shared storage

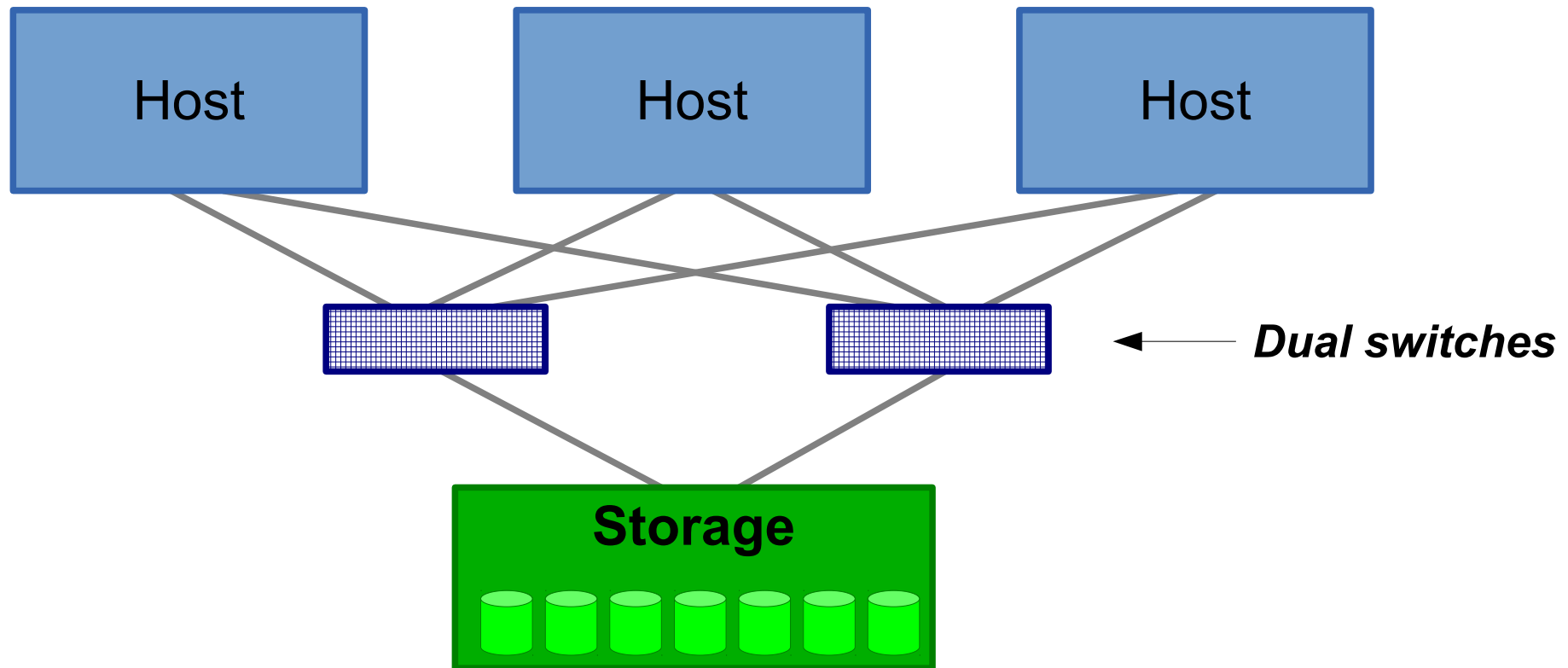
- Storage becomes single point of failure
- Network becomes single point of failure
- Network bandwidth can be a bottleneck
- Network latency can impact performance
- Network security
 - keep storage on a completely separate network
- Risk of accidentally starting two VMs using the same disk image!

Avoiding storage server SPOF



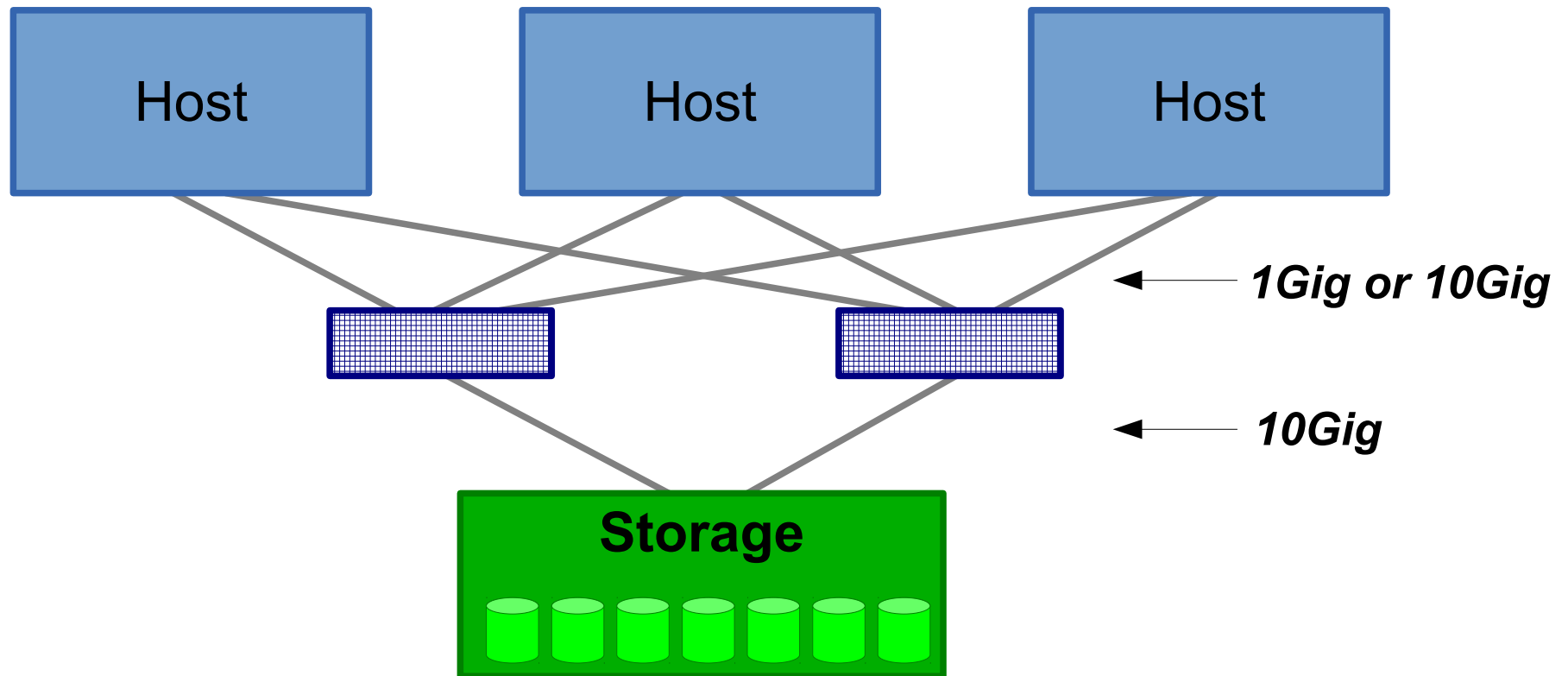
- This is **very hard** to build correctly
- Vendors will sell this to you for \$\$\$

Avoiding network SPOF



Or you can buy a *really expensive* chassis switch with multiple power supplies, line cards, switching fabrics and management cards

Network bandwidth



Note: 1Gbps $\approx$ 100MB/sec $\approx$ throughput of a single hard drive

Latency

- Time between sending a request and receiving the reply
- Some applications are very latency sensitive
 - e.g. a database which writes to disk and waits for confirmation that data has been written
- Networking adds to the latency
 - 10G on CAT6/RJ45 has higher latency than fibre or SFP+ direct-attach cables
 - alternatives to ethernet: fibre channel, infiniband

Questions?

Shared storage protocols

- Fundamentally two types:
- Shared filesystem
 - a.k.a. "Network Attached Storage" or "NAS"
- Shared block device
 - a.k.a. "Storage Area Network" or "SAN"

Shared filesystem

- Client has remote access to server's filesystem
 - requests like "read directory", "open file named X"
- Examples:
 - NFS
 - CIFS (Windows/Samba shares)
- VM images are just files on the server
 - very easy to set up and understand
 - directory of image files, just happens to be remote
 - disk image files may support snapshots

Using shared filesystem

```
# mount s1:/var/lib/libvirt/images /mnt
# ls /mnt
debian1.img
debian2.img
debian3.img
debian4.img
...
```

Limitations of shared filesystem

- Overhead of traversing the kernel filesystem at both client side and server side
- Usual issues with disk images at the server side (e.g. fragmentation)

Shared block device

- Remote requests to read/write block N
 - this is closer to what the VM expects
- Server side can map these requests to a physical drive, a logical volume or an image file
- Examples:
 - iSCSI (standard, heavyweight)
 - **nbd (Linux only)**
 - ggated (FreeBSD only)
 - Fibre Channel (\$\$\$)

nbd: server side

/etc/nbd-server/config

```
[generic]
```

```
    user = nbd
```

```
    group = nbd
```

```
    includedir = /etc/nbd-server/conf.d
```

```
    listenaddr = 10.10.0.241
```

```
    oldstyle = true
```

```
# Repeat for each volume to export
```

```
[disk1]
```

```
    exportname = /data/nbd/disk1.img
```

```
    port = 20001
```

```
    flush = true
```

```
    fua = true
```

"Old style" nbd protocol uses a different TCP port per volume. New servers use named volumes all accessible on the same port (10809)

nbd: client side

```
# nbd-client s1.ws.nsrc.org 20001 /dev/nbd0  
# blockdev --getsize64 /dev/nbd0  
2147483648
```

...

```
# nbd-client -d /dev/nbd0
```

- You can use /dev/nbd0 just as you would a local block device (mkfs, mount etc)

nbd: direct access from KVM

```
<disk type='network' device='disk'>  
  <driver name='qemu' type='raw' />  
  <source protocol='nbd'>  
    <host name='s1.ws.nsrc.org' port='20001' />  
  </source>  
  <target dev='hda' bus='ide' />  
  <address type='drive' controller='0'... />  
</disk>
```

- KVM knows how to speak the nbd protocol - so it can bypass the kernel's nbd client

nbd limitations

- Changing the config file requires a server restart, which can impact on active clients
 - Clients may see I/O errors
 - kvm's built-in nbd client fails to reconnect?
 - `apt-get install -t wheezy-backports qemu-kvm`
 - Still doesn't seem to work right
 - Pre-create logical volumes and export them before you need them?
- No security, apart from optional "IP allow"
 - Keep all nbd traffic on a separate network!

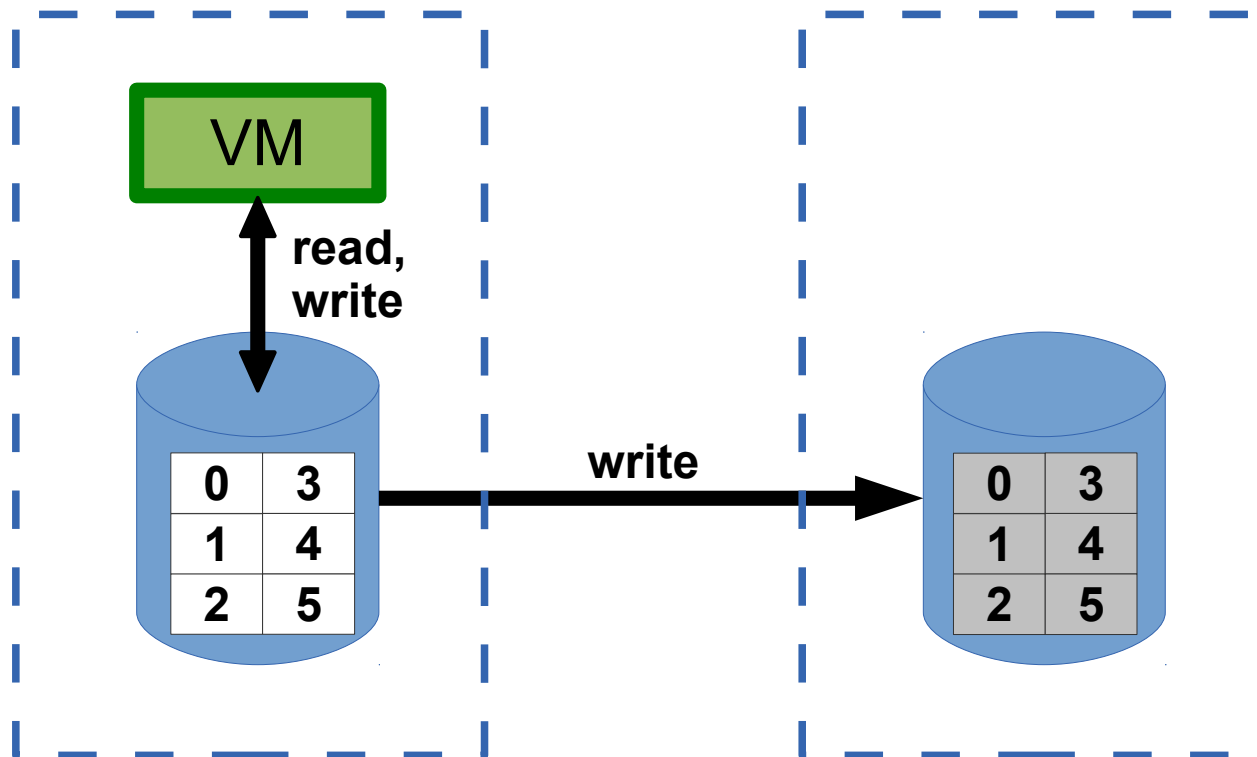
nbd tricks

- Alternative nbd server implementations
 - xnbd-server
 - nbd proxy for migrating disk images
 - flexnbd-c
 - separate IPv6 address per volume, migration support
- Test them before deployment!

Alternatives to shared storage

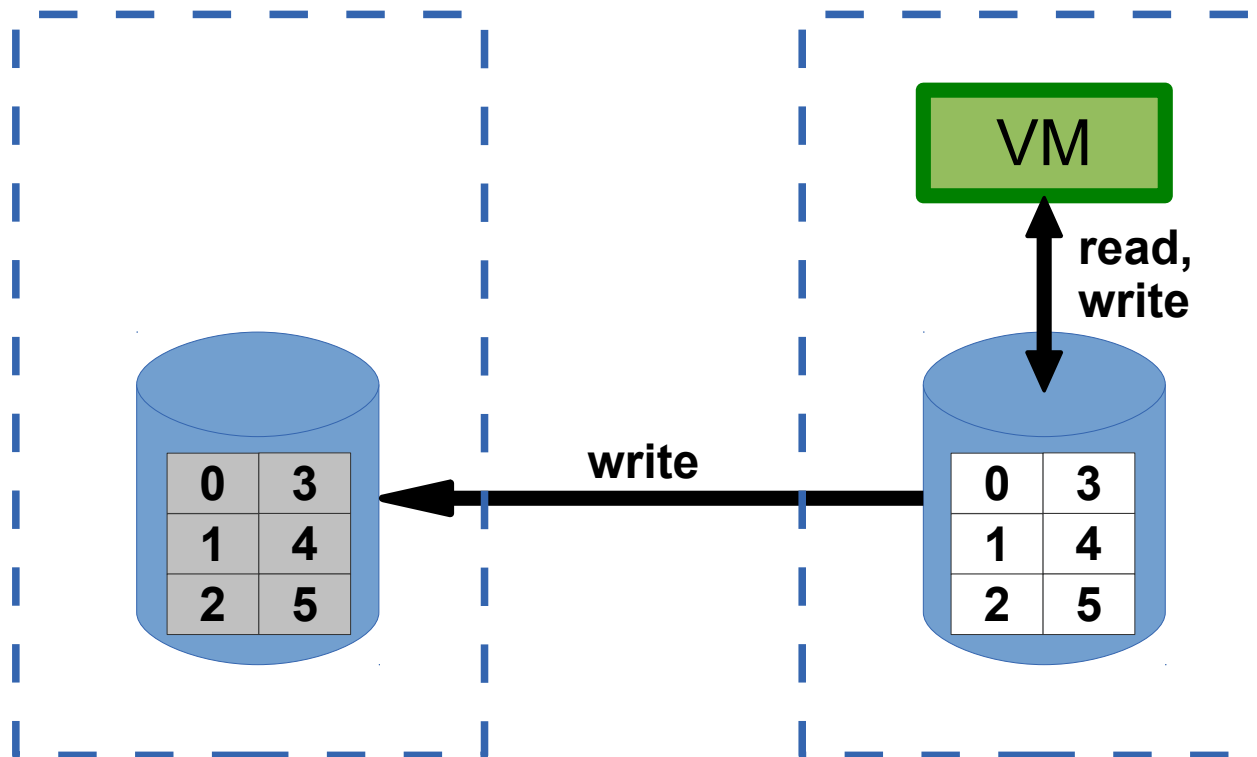
Replicated Storage

- DRBD - as used by Ganeti
 - Primary accesses local disk, writes are replicated

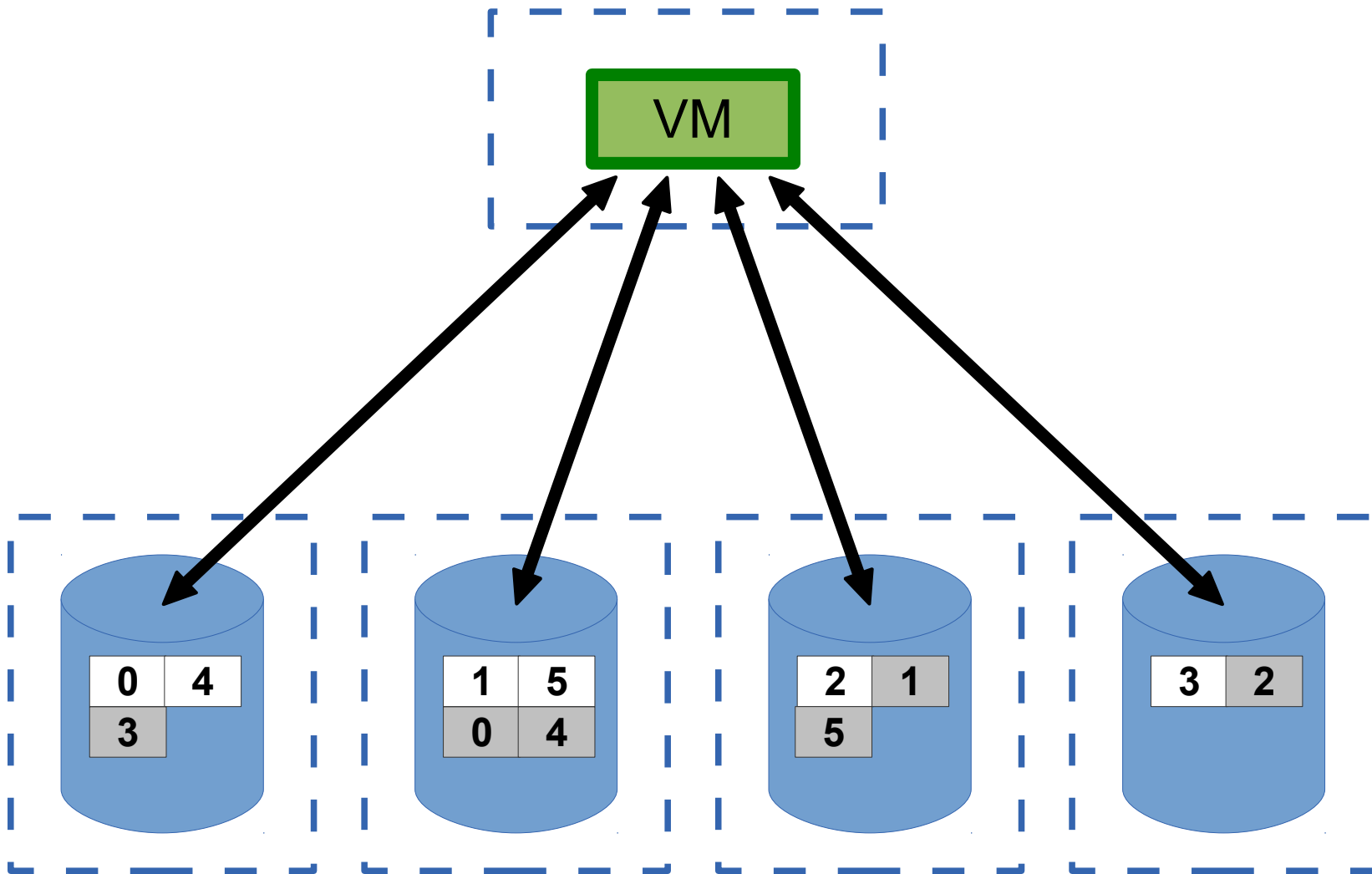


Replicated Storage

- Reverse roles for migration



Distributed Storage



Distributed Storage

- Examples:
 - Ceph (rados/rbd) - general purpose
 - Sheepdog - for KVM images only
 - Glusterfs
- Data is stored on multiple nodes
 - Offers huge scale and resilience against loss of one or more nodes
 - Complexity, balancing, "split brain" problems
 - Still limited by network bandwidth/latency

Summary

- Migration of virtual machines allows load balancing and cluster maintenance
- Live migration makes this invisible to VM users
 - can achieve very high uptime
- Access to storage is the key
- Various options for shared, replicated or distributed storage
- Can be difficult and expensive to build