

# EEL-6892 – Virtual Computers

## Lecture 15

# Outline

- Basics of system VMs
- Virtualization in a broader sense
  - Virtual networks
  - Virtual storage
- Applications of virtualization

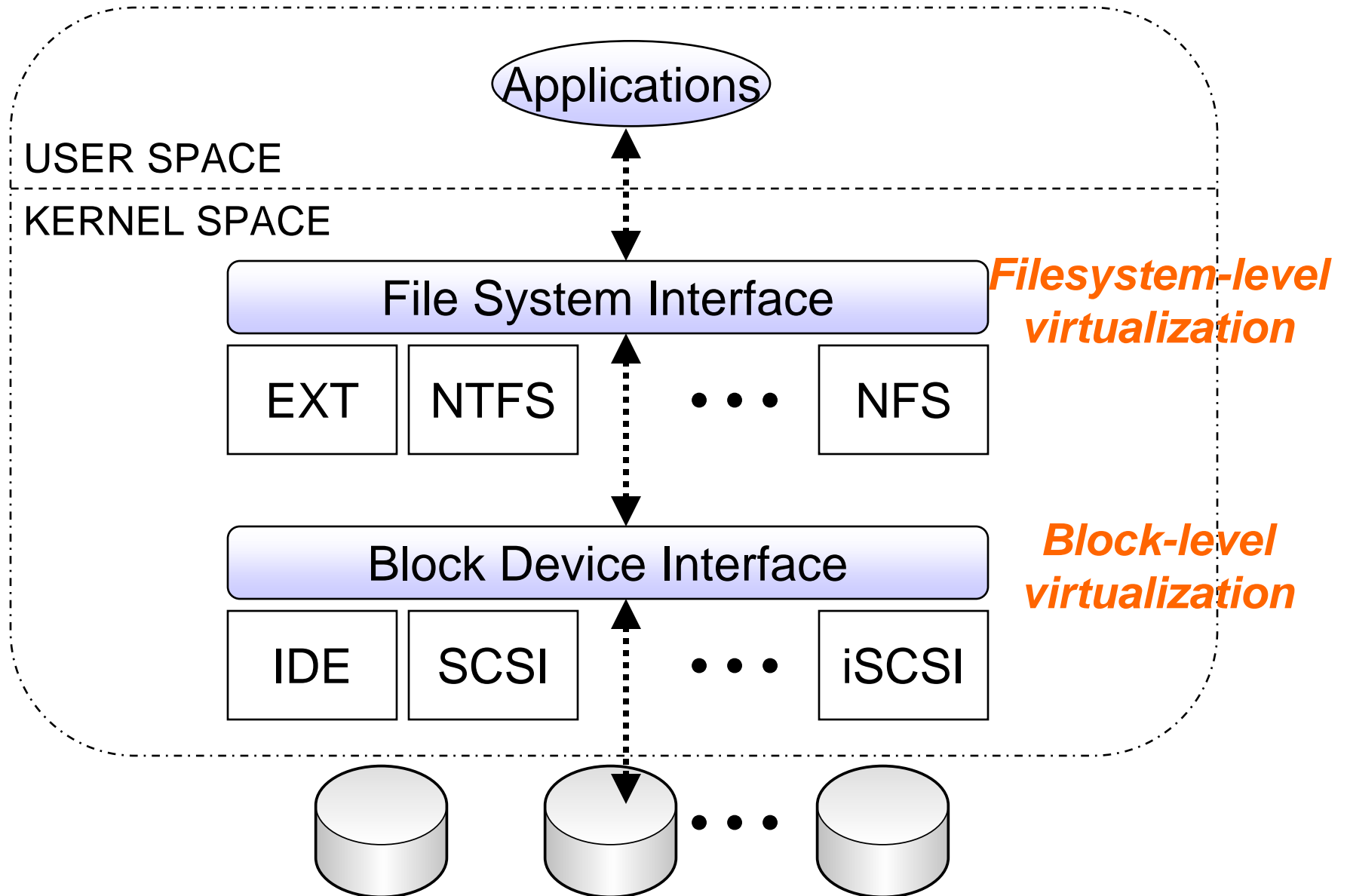
# Virtual storage

- We have studied how storage devices are virtualized for VMs “in a box”
- Beyond storage devices, storage virtualization is another basic form of virtualization
  - For physical and/or virtual machines

# Virtual storage

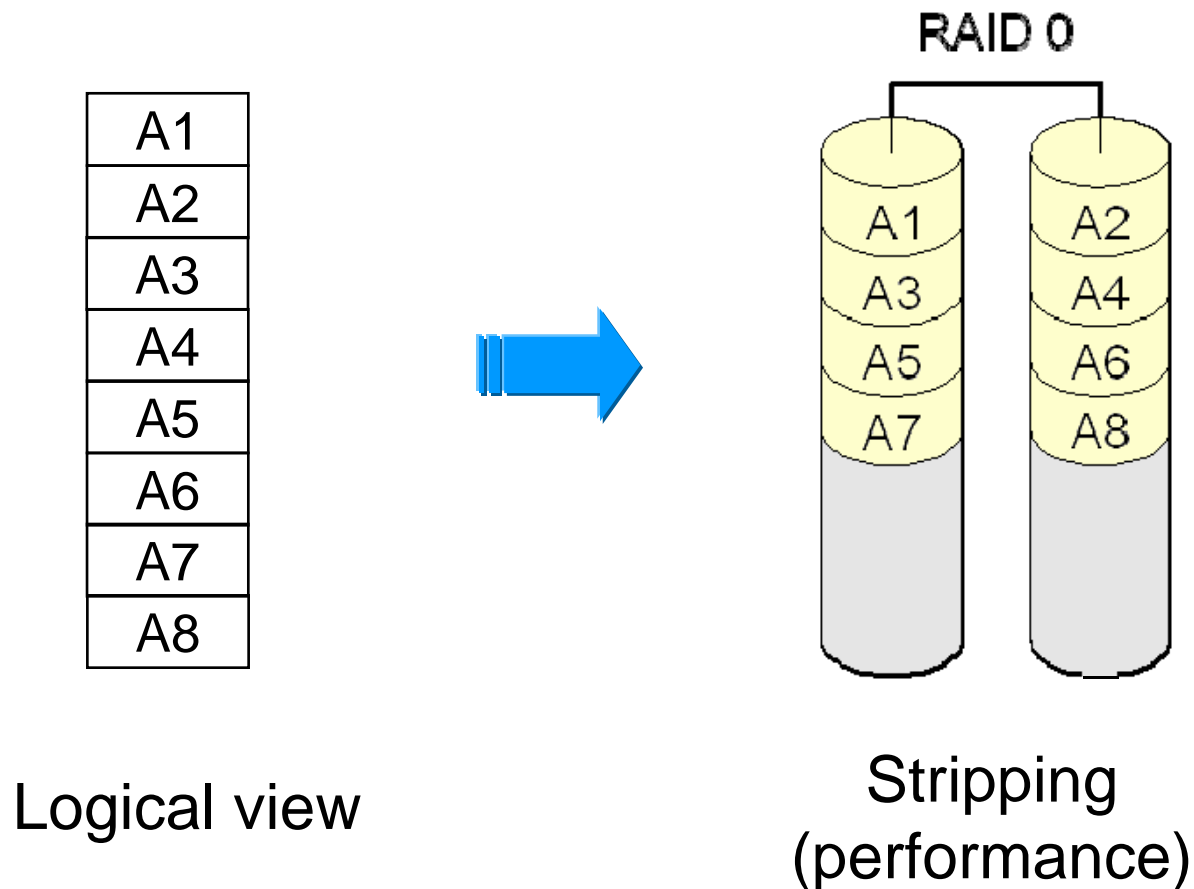
- An abstraction of storage that separates the logical view of data location and path from the physical presence on storage devices
- Two types of virtualization approaches:
  - Filesystem-level virtualization
    - Next lecture
  - Block-level virtualization

# The path to storage: where to virtualize?

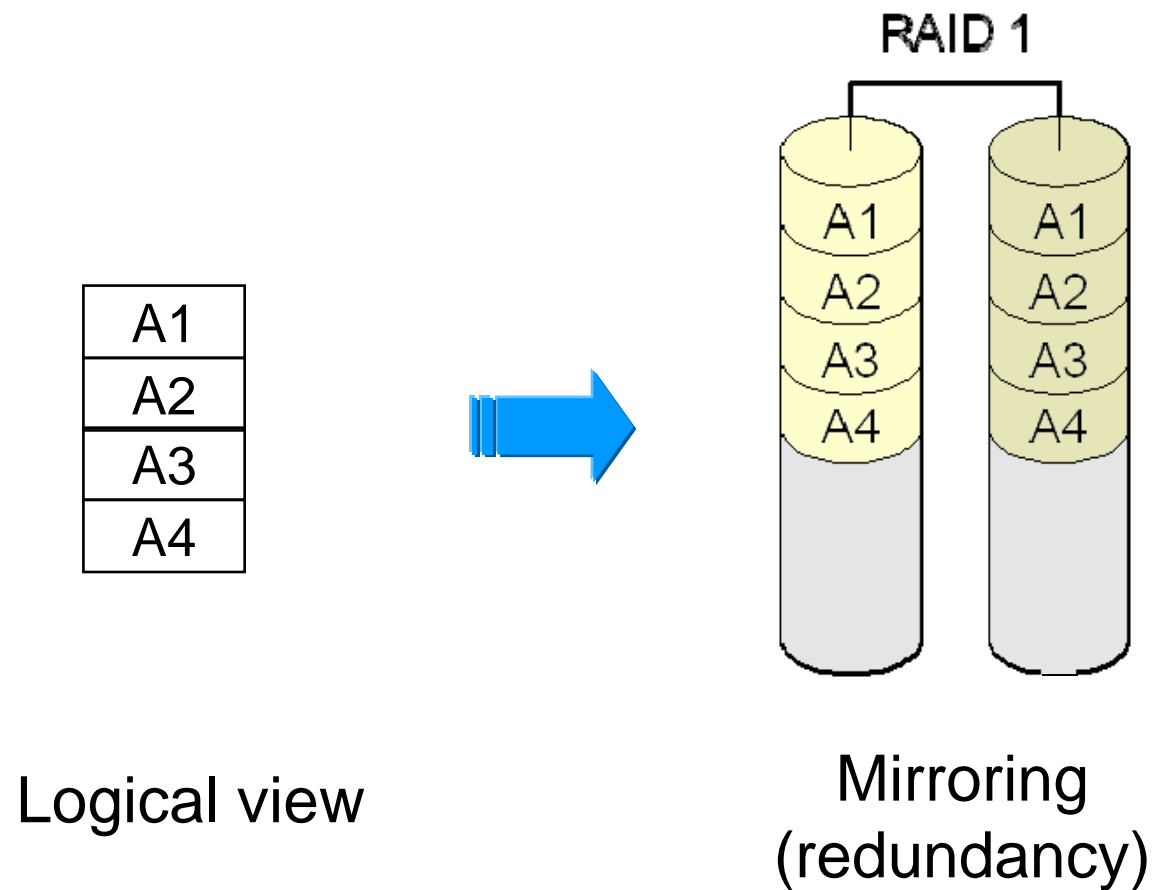


# Virtual storage example: RAID

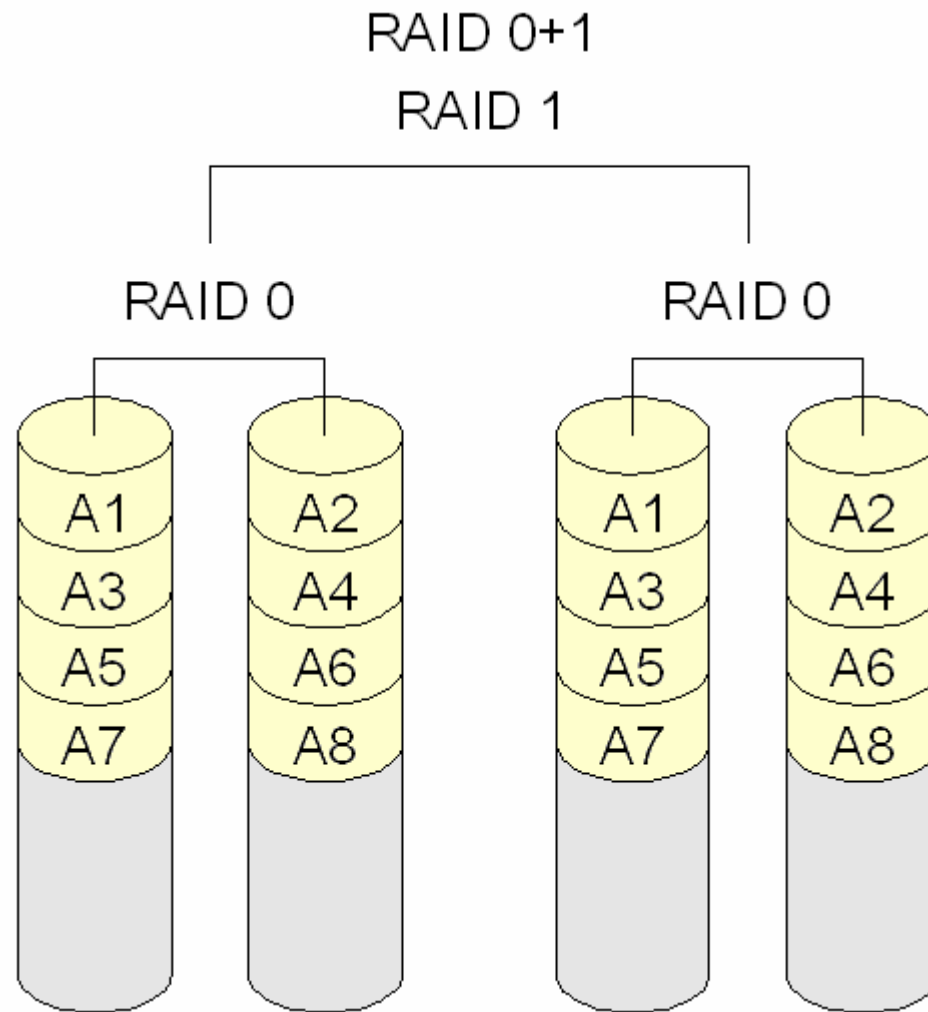
- Redundant array of inexpensive (independent) disks (introduced in 1988)



# Virtual storage example: RAID

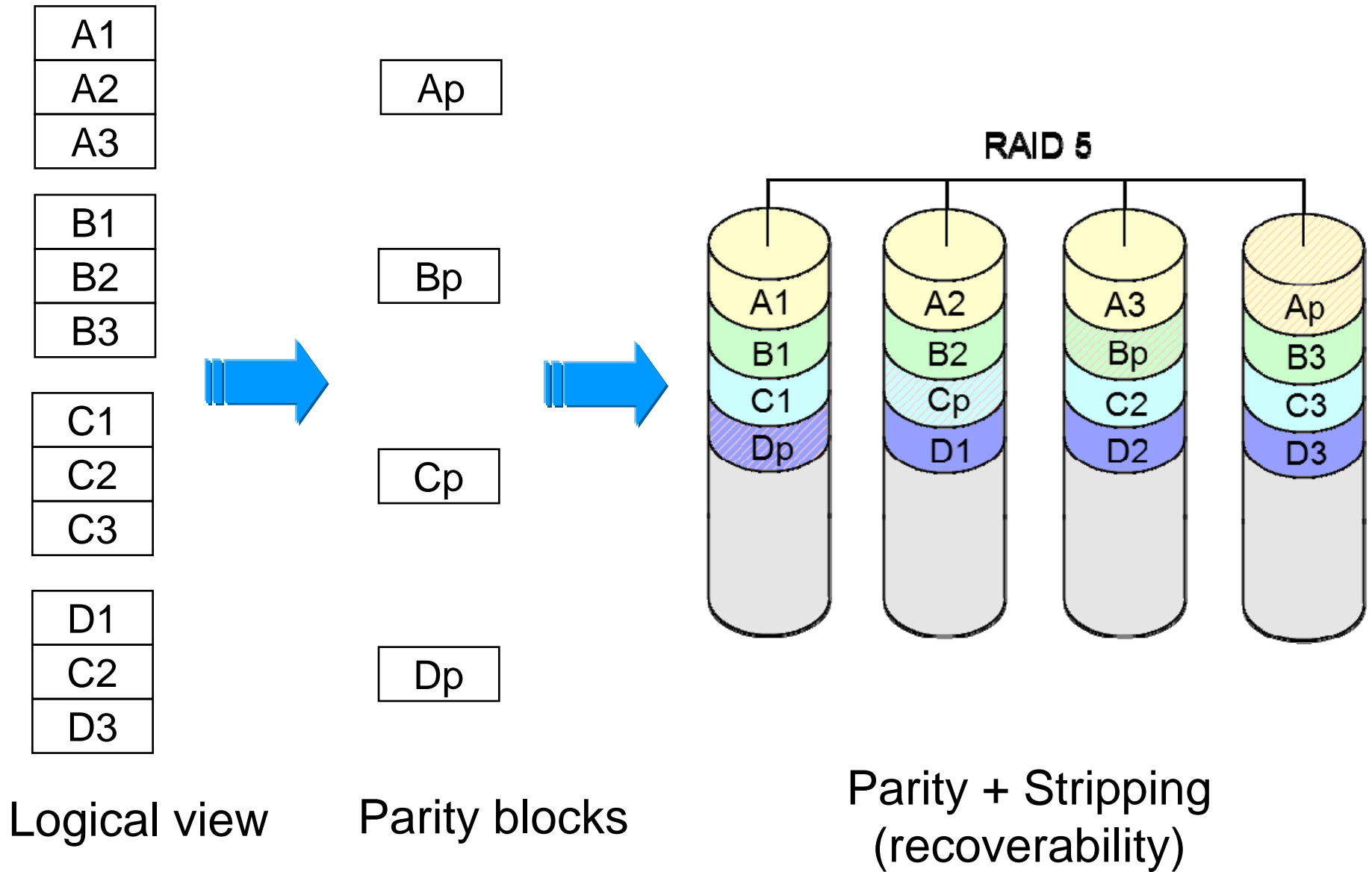


# Virtual storage example: RAID



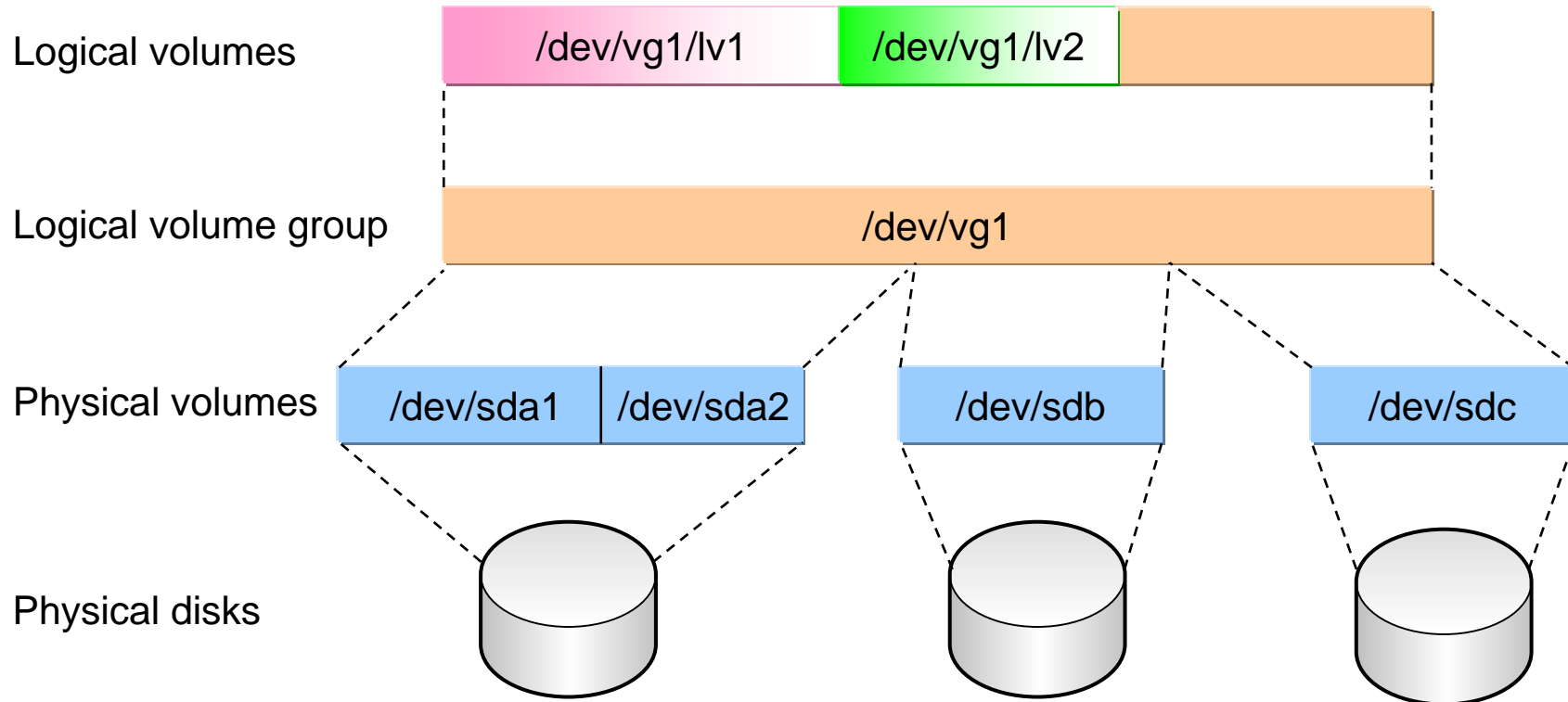
Stripping + Mirroring  
(performance + redundancy)

# Virtual storage example: RAID



# Virtual storage example: LVM

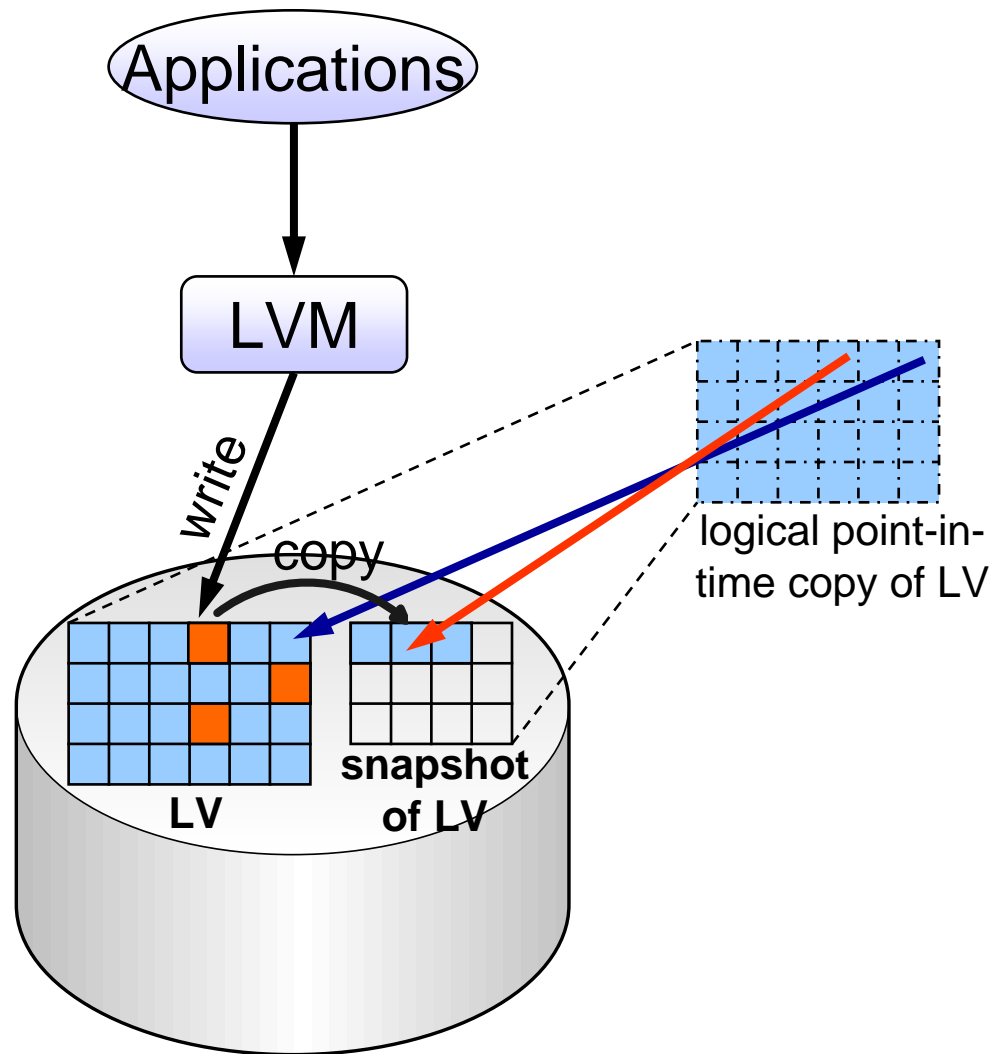
- Logical volume management (existing in most O/Ss: UNIX, Linux, Windows)



# Virtual storage example: LVM

- Online resizing, migration
- Software RAID
- Snapshotting via Copy-on-Write (COW)
  - Copy blocks to be changed before they are overwritten
  - Preserve consistent historical images of the block device

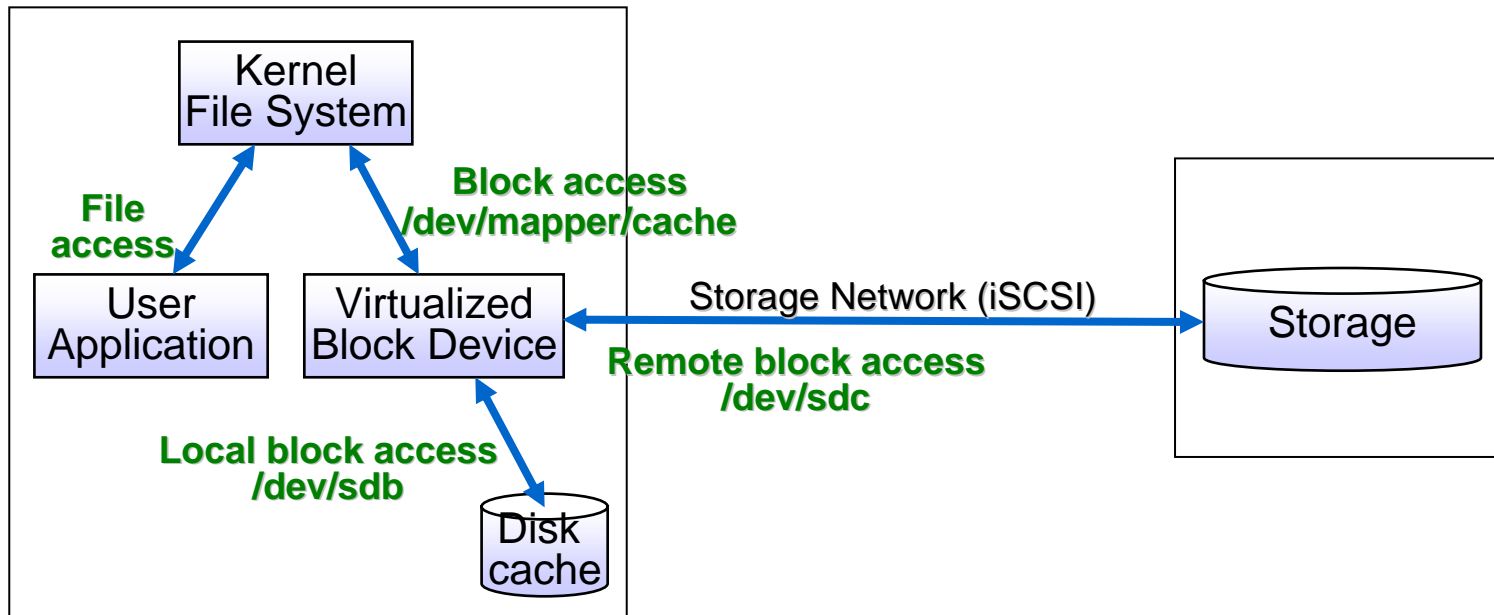
# Virtual storage example: LVM



# Virtual storage example: a demo

- A virtual block device used and served by virtual machines
- Remote block access via iSCSI
- Local block-level caching

# Virtual storage example: a demo

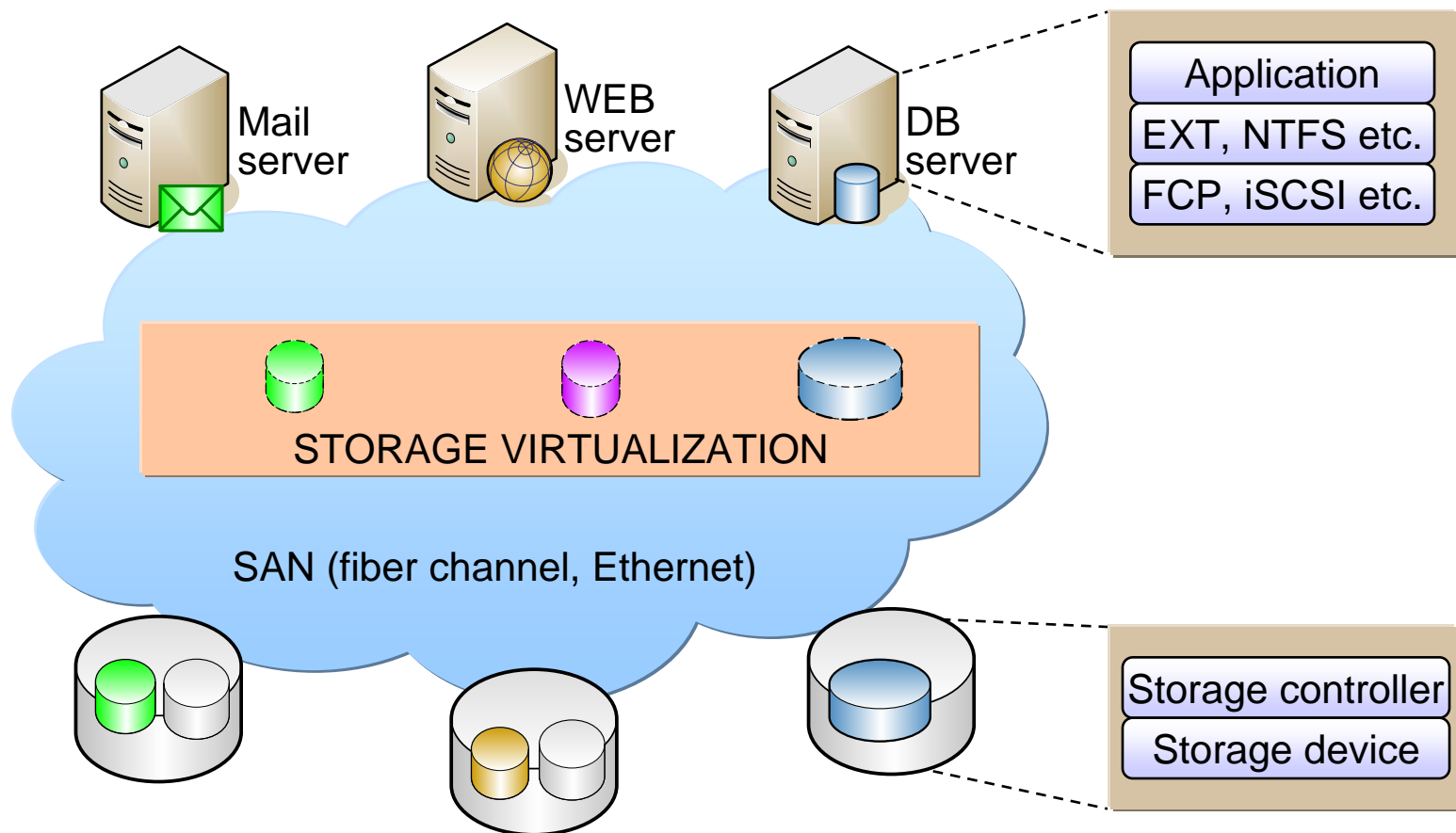


# Benefits of virtual storage

- Similar to virtual machines
  - Resource consolidation
  - Resource sharing with isolation
  - Management decoupling
- Supports services to improve:
  - Utilization: aggregation and allocation
  - Performance: parallel I/O, load-balancing, caching
  - Reliability: redundancy, snapshots

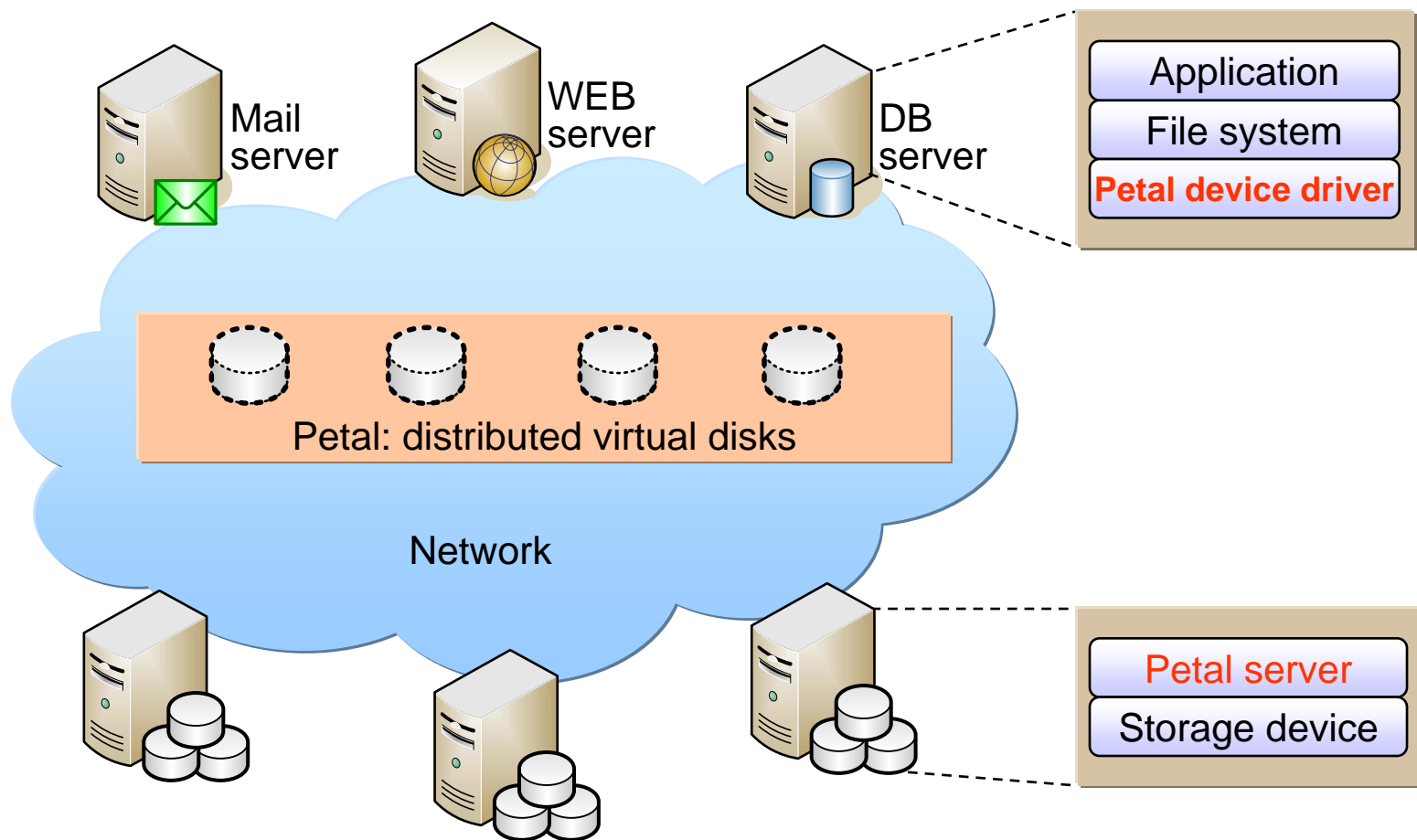
# Storage virtualization systems

- Storage area networks (SAN) virtualization
  - Commercial products from IBM, HP, EMC etc.



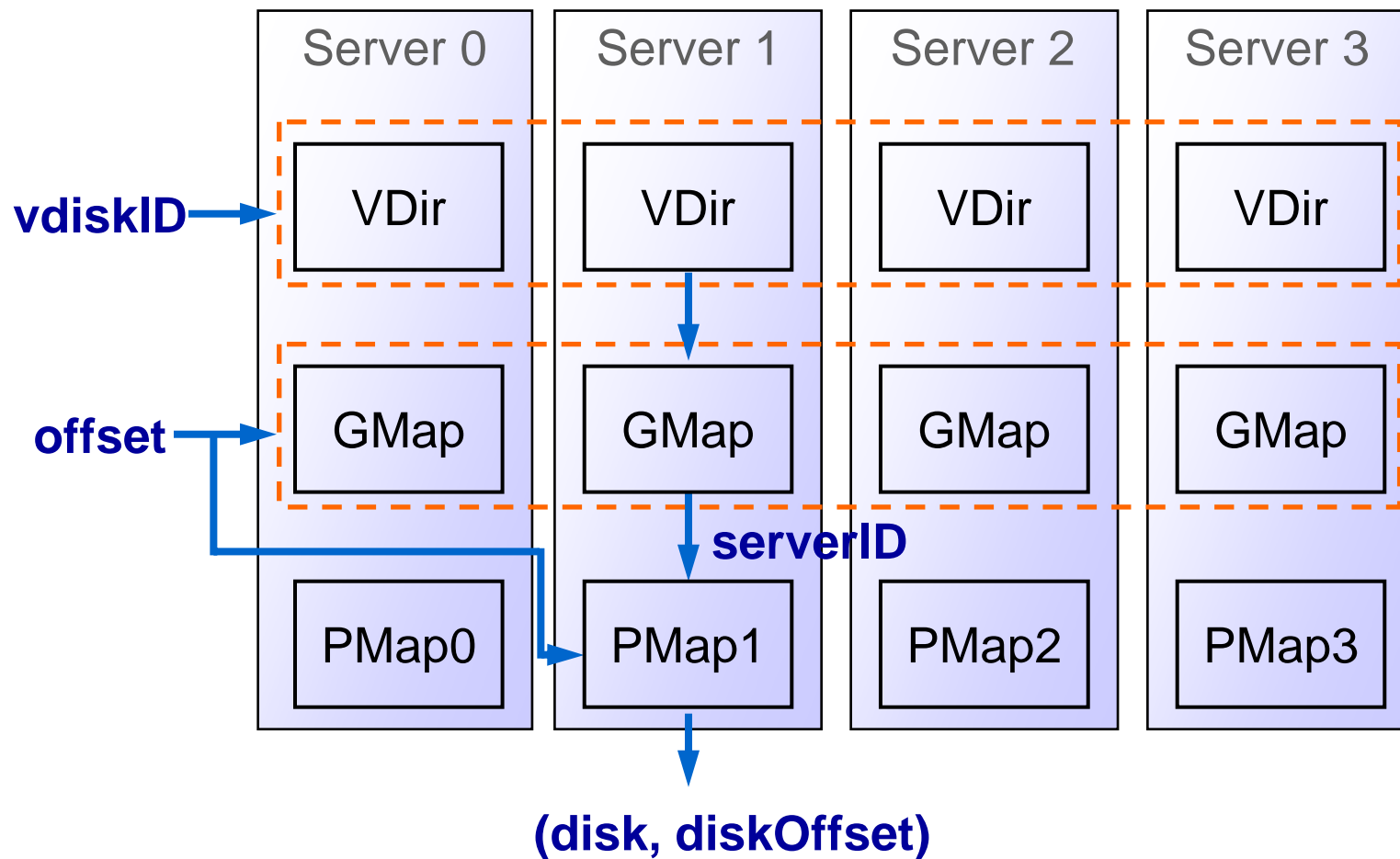
# Storage virtualization systems

- Petal: distributed virtual disks (DEC, 1996)



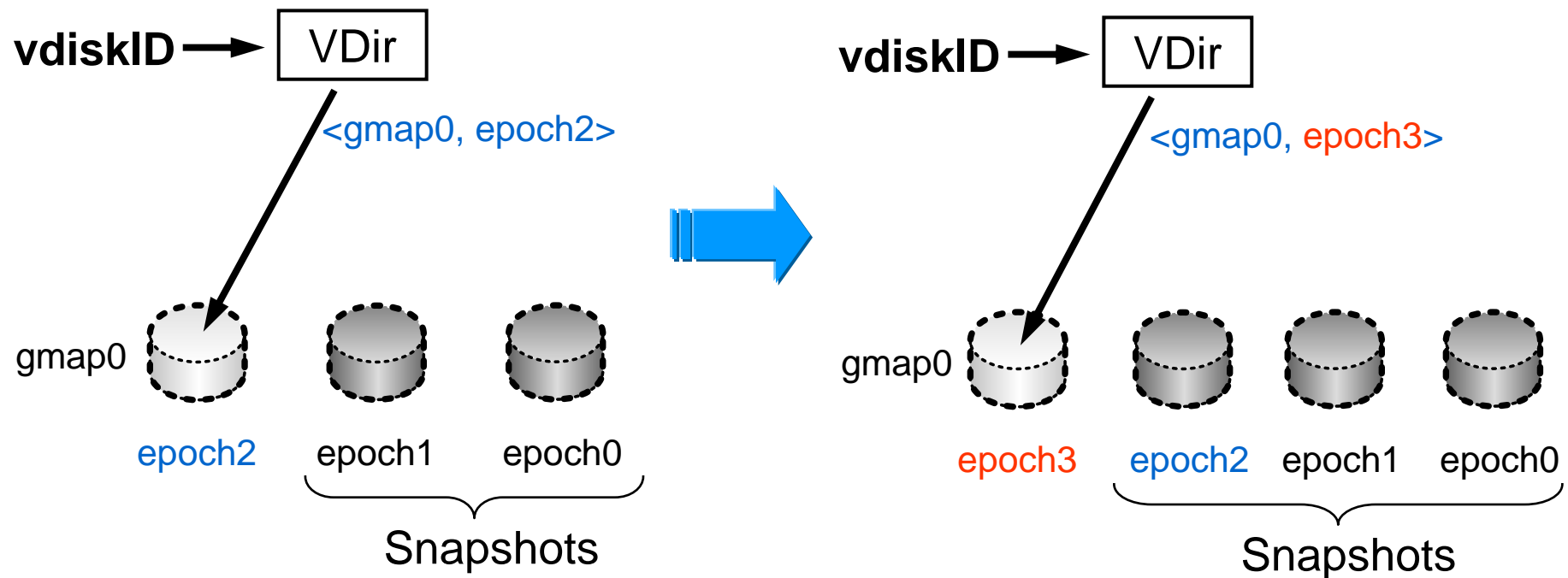
# Virtual to physical translation

- $(\text{vdiskID}, \text{offset}) \rightarrow (\text{server}, \text{disk}, \text{diskOffset})$



# Support for backup

- Read-only snapshot via copy-on-write
- Modified virtual-to-physical translation

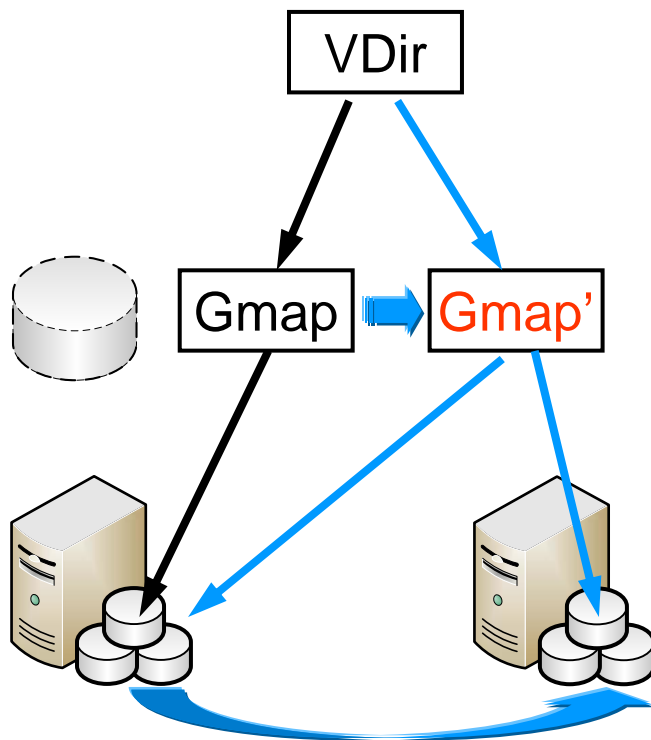


# Virtual disk reconfiguration

- Change of a virtual disk's redundancy scheme or the mapped servers
  - Update virtual-to-physical mappings
  - Redistribute data to servers accordingly
- Challenge: perform reconfiguration incrementally and concurrently while serving normal client requests

# Virtual disk reconfiguration

- A trivial implementation:



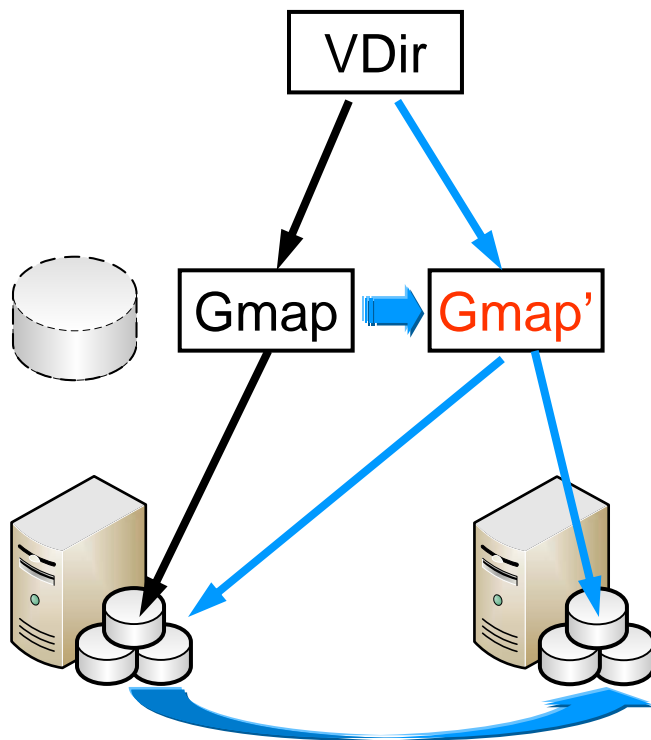
**2.** Change all virtual disk directories to point to the new global map

**1.** Create a new global map with desired redundancy scheme and server mapping

**3.** Redistribute data to the servers according to the translation specified in the new global map

# Virtual disk reconfiguration

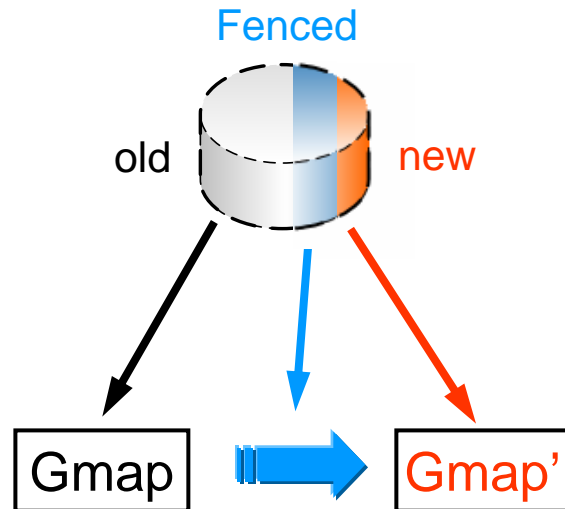
- A trivial implementation:



- For write requests:
  - New Gmap is used
- For read requests:
  - Old Gmap is used on new Gmap miss
  - Too many misses → performance degradation

# Incremental reconfiguration

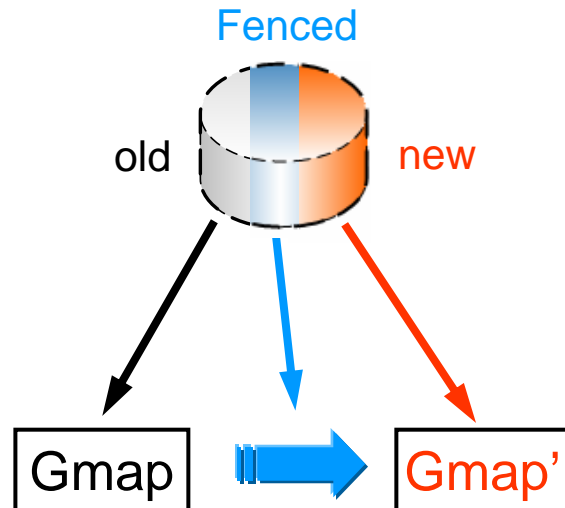
- Relocates only small portions of a virtual disk at a time:



- Requests to new region:
  - Use new Gmap
- Requests to old region:
  - Use old Gmap
- Requests to fenced region:
  - Try new Gmap first;
  - Use old Gmap upon miss

# Incremental reconfiguration

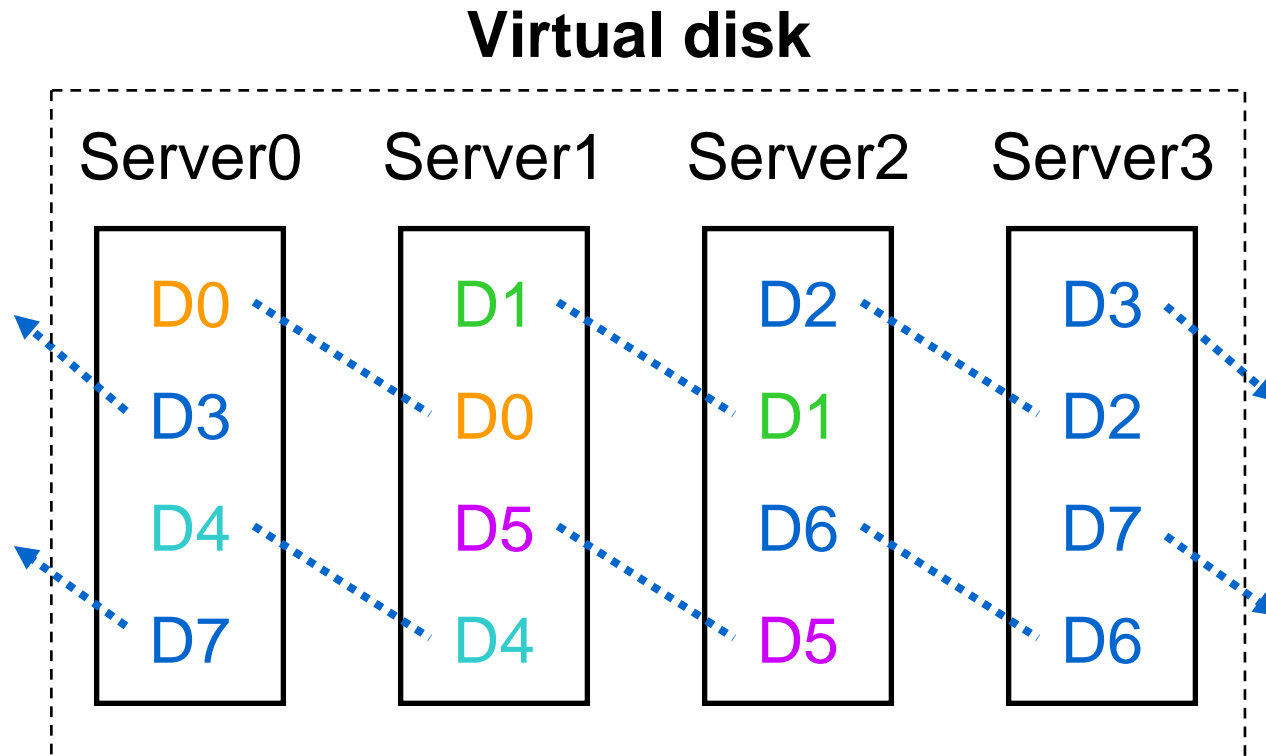
- Relocates only small portions of a virtual disk at a time:



- Fenced region becomes a new region after everything in the region is completely relocated
- Repeatedly fence a portion of the old region until all data redistributed

# Data placement & redundancy

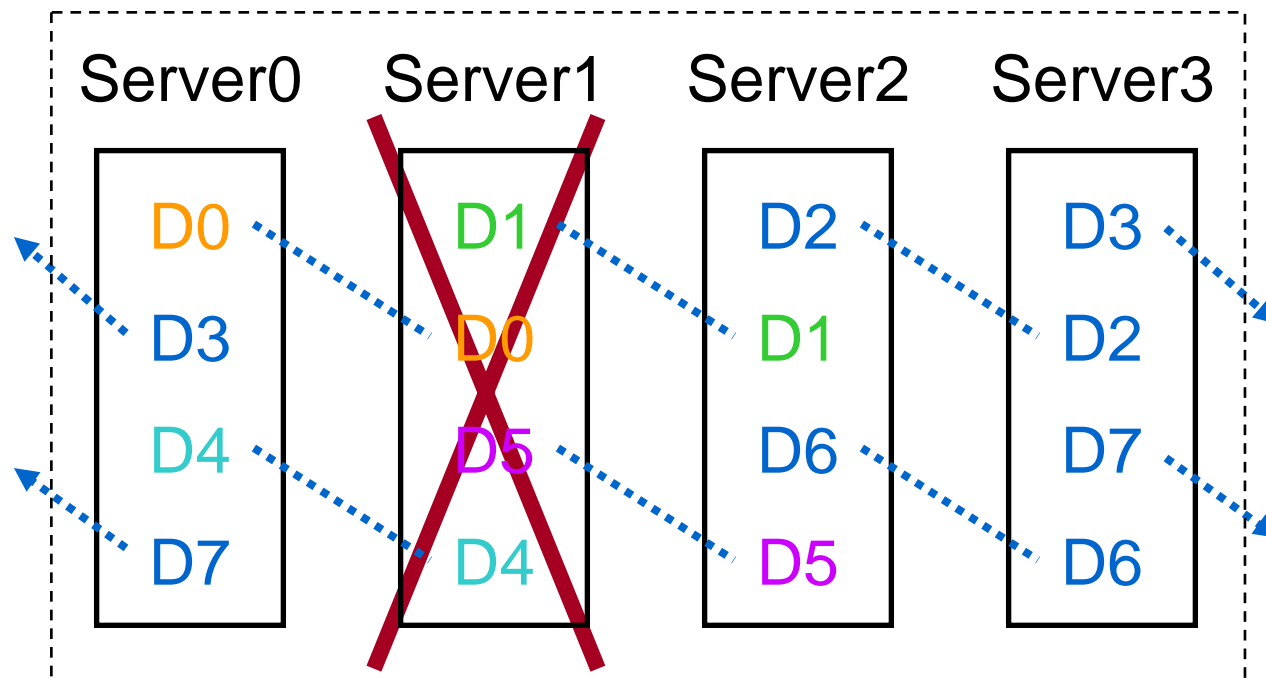
- Chained Declustering
  - Two copies of each data block are stored on neighboring servers



# Data placement & redundancy

- Chained Declustering
  - If server 1 fails, servers 0 and 2 will share server 1's load
  - A uniform load can be maintained by cascading the offloading across the surviving servers

## Virtual disk





# Dynamic load balancing

- Simple client-initiated scheme
  - Tracks its pending requests at each server
  - Sends requests to the server with the shorter queue length
  - If timeout, tries another server
  - Works well if most requests are generated from a few clients

# Data consistency

- Uses locking to avoid conflicting accesses on the copies
- Designates a copy as primary:
  - A write always starts with the primary server (to avoid deadlock)
  - Primary server performs the write on its local copy and also forwards it to the secondary copy
- A failed sever is brought up-to-date during its recovery process

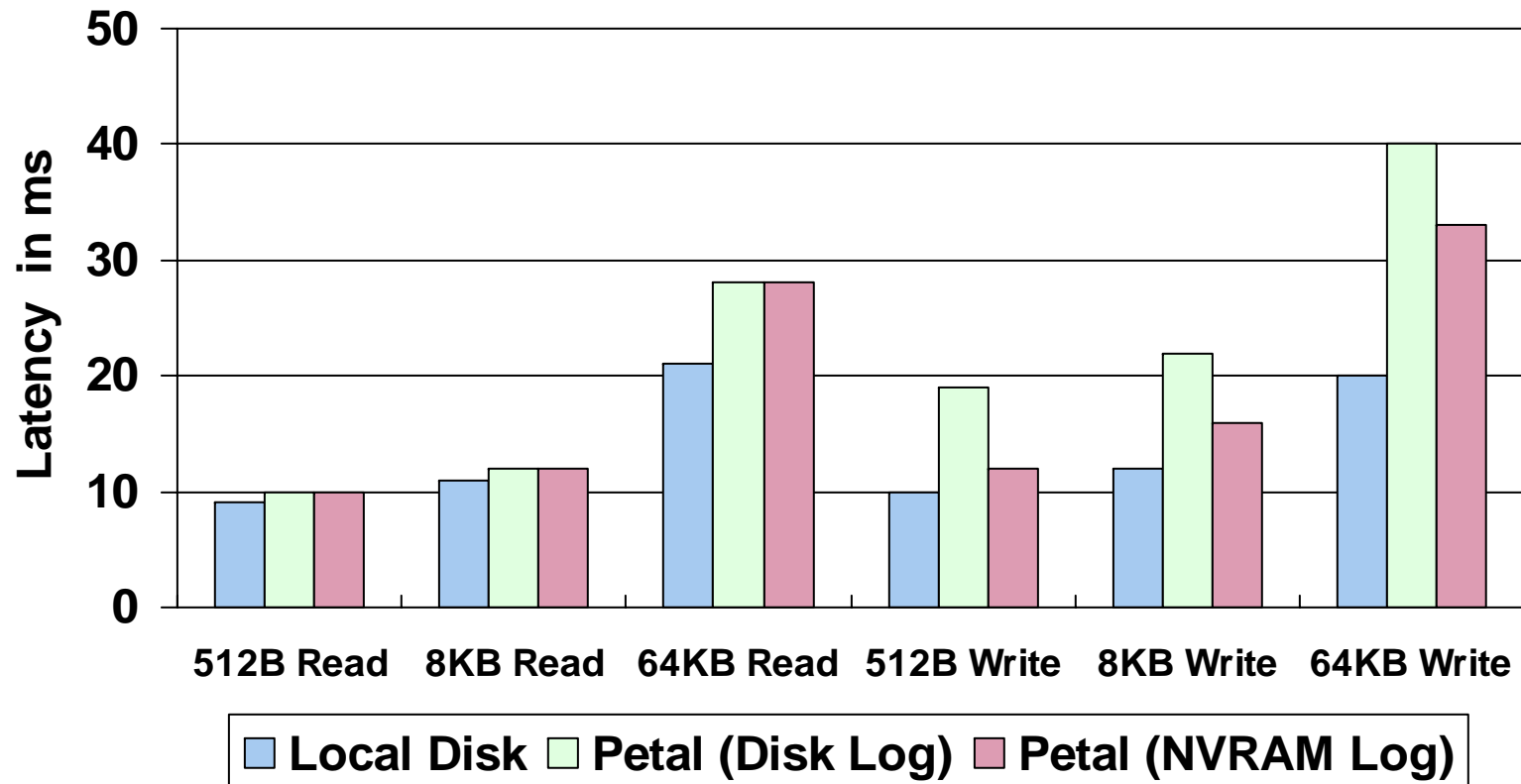
# Evaluation of the prototype

- Digital ATM network.
  - 155 Mbit/s per link.
- 8 AlphaStation Model 600.
  - 333 MHz Alpha running Digital Unix.
- 72 RZ29 disks.
  - 4.3 GB, 3.5 inch, fast SCSI (10MB/s).
  - 9 ms avg. seek, 6 MB/s sustained transfer rate.
- Unix kernel device driver.
- User-level Petal servers.

\* This and the following slides on Petal evaluation are taken from Edward K. Lee's original presentation. These materials are owned by the authors of the work.

# Client request latency

Chain-declustered virtual disk.  
Random requests.

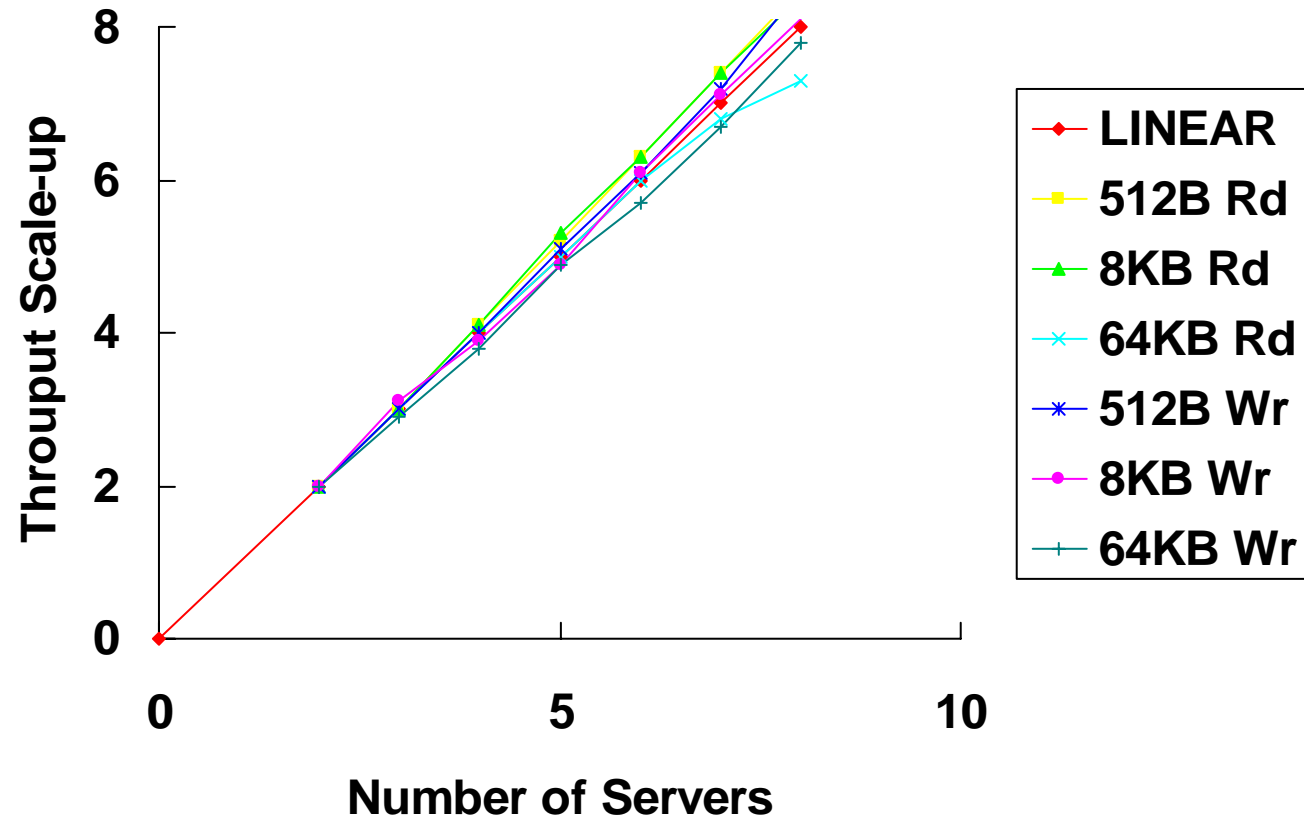


# Failure mode performance

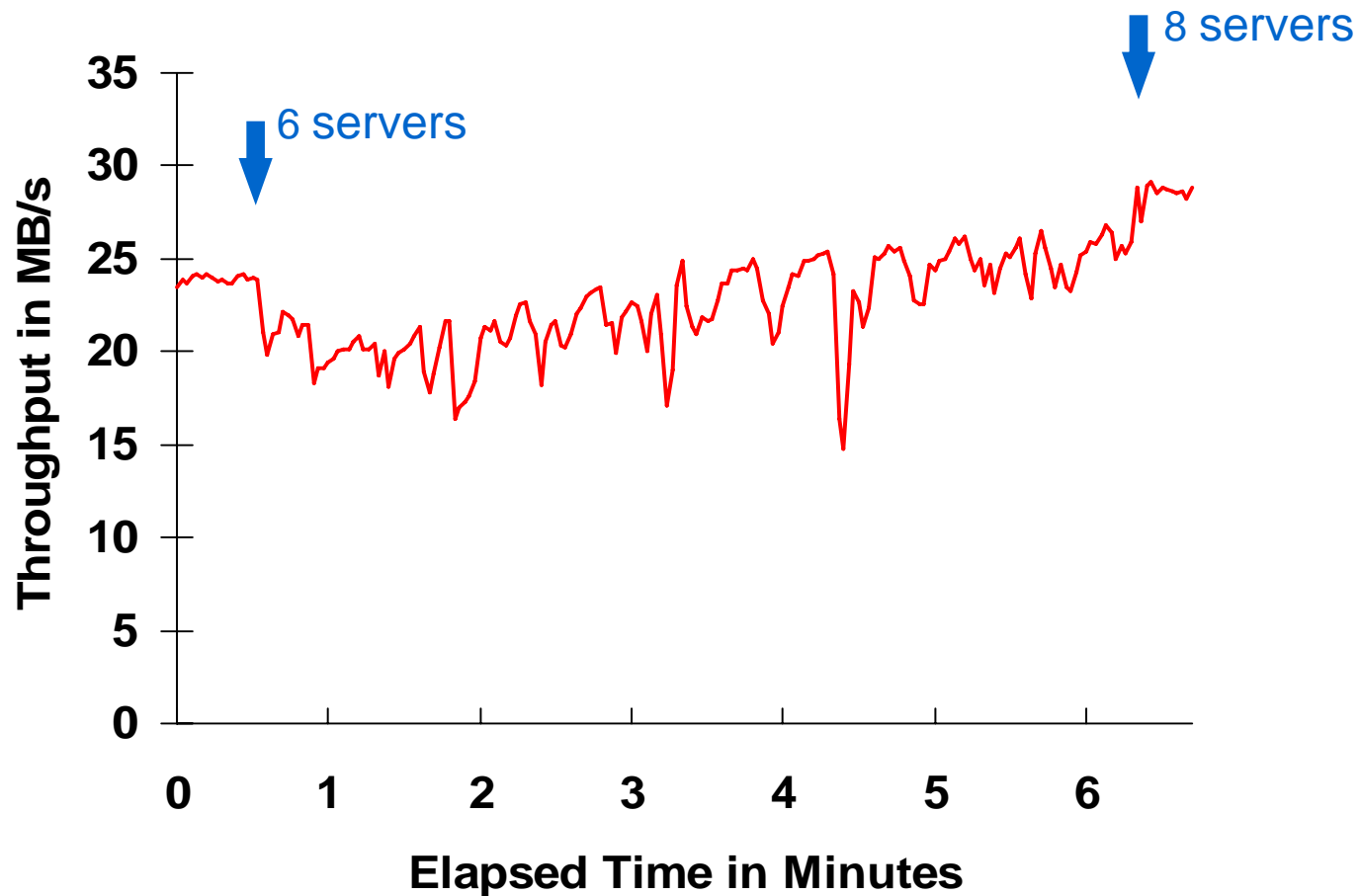
1 out of 8 servers failed  
 $7/8 = 88\%$

	Failed	Normal	% of Normal
512B Read	3.4 MB/s	3.8 MB/s	87 %
8KB Read	47.1 MB/s	53.7 MB/s	88 %
64KB Read	106.7 MB/s	115.2 MB/s	93 %
512B Write	0.88 MB/s	0.89 MB/s	99 %
8KB Write	22.9 MB/s	23.1 MB/s	99 %
64KB Write	48.4 MB/s	49.3 MB/s	98 %

# Throughput scaling



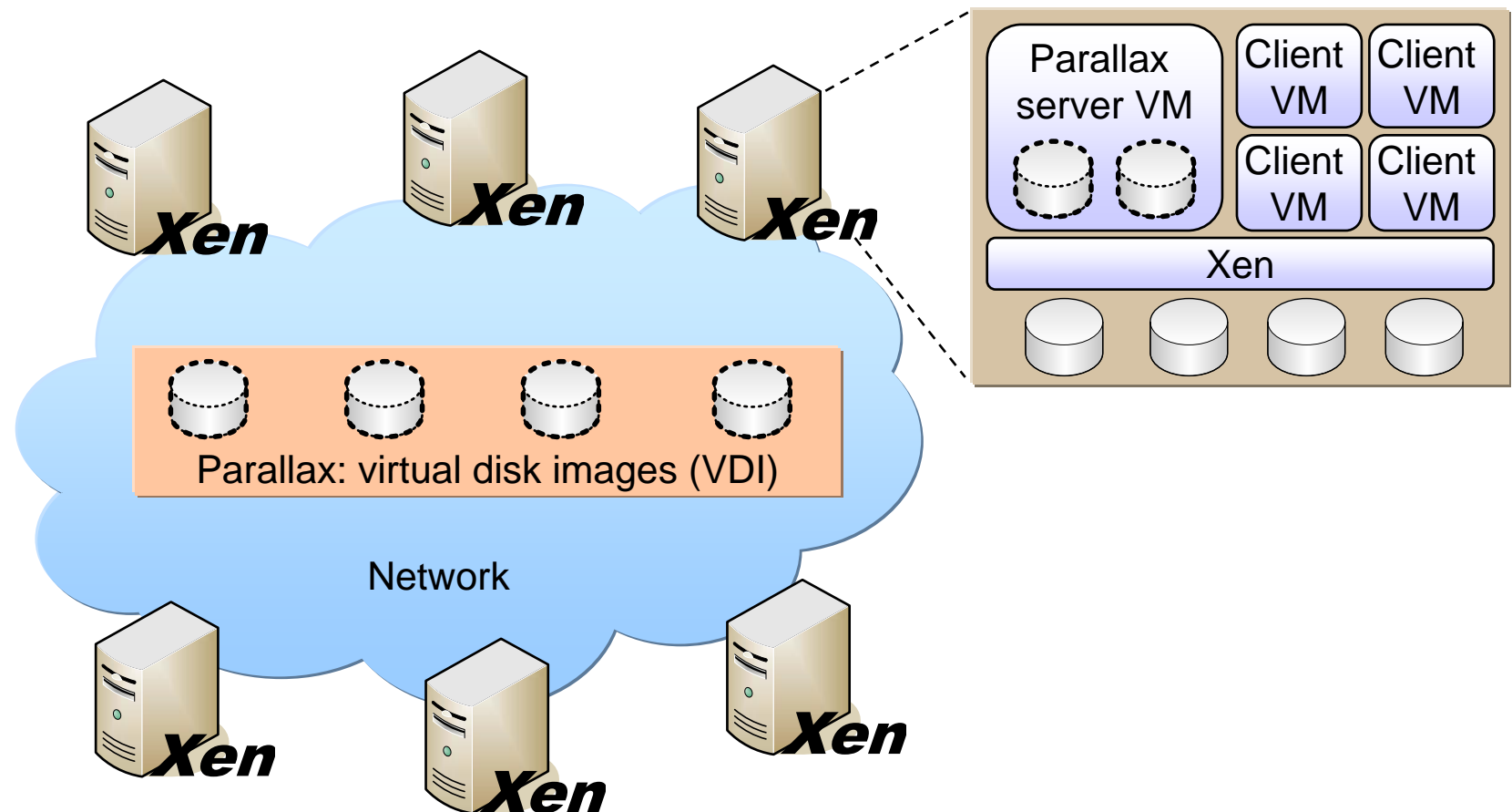
# Virtual disk reconfiguration



virtual disk w/ 1GB of allocated storage  
8KB reads & writes

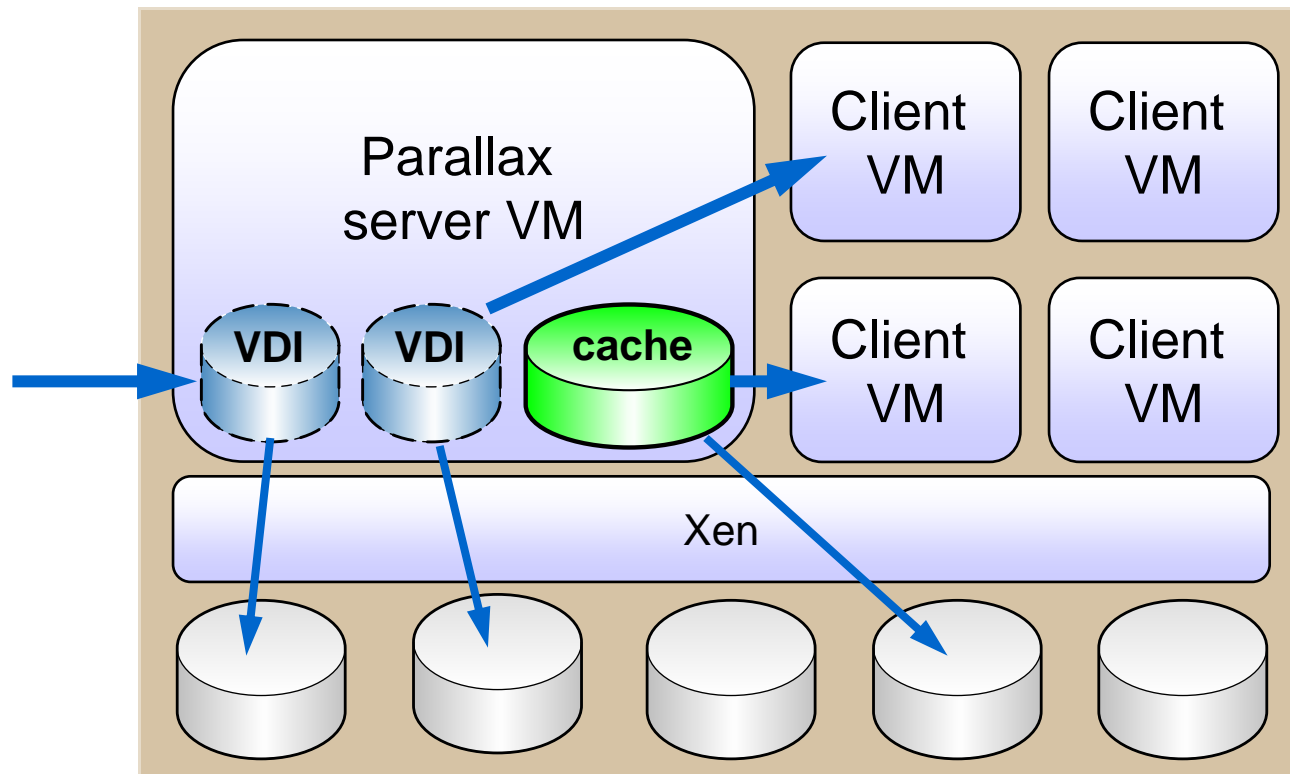
# Storage virtualization systems

- Parallax: managing storage for a “million” (virtual) machines (Xen, 2005)



# Basic design of Parallax

- Parallax VM manages local physical storage
  - A contribution to a storage pool shared by cluster
  - A persistent cache for locally hosted VMs

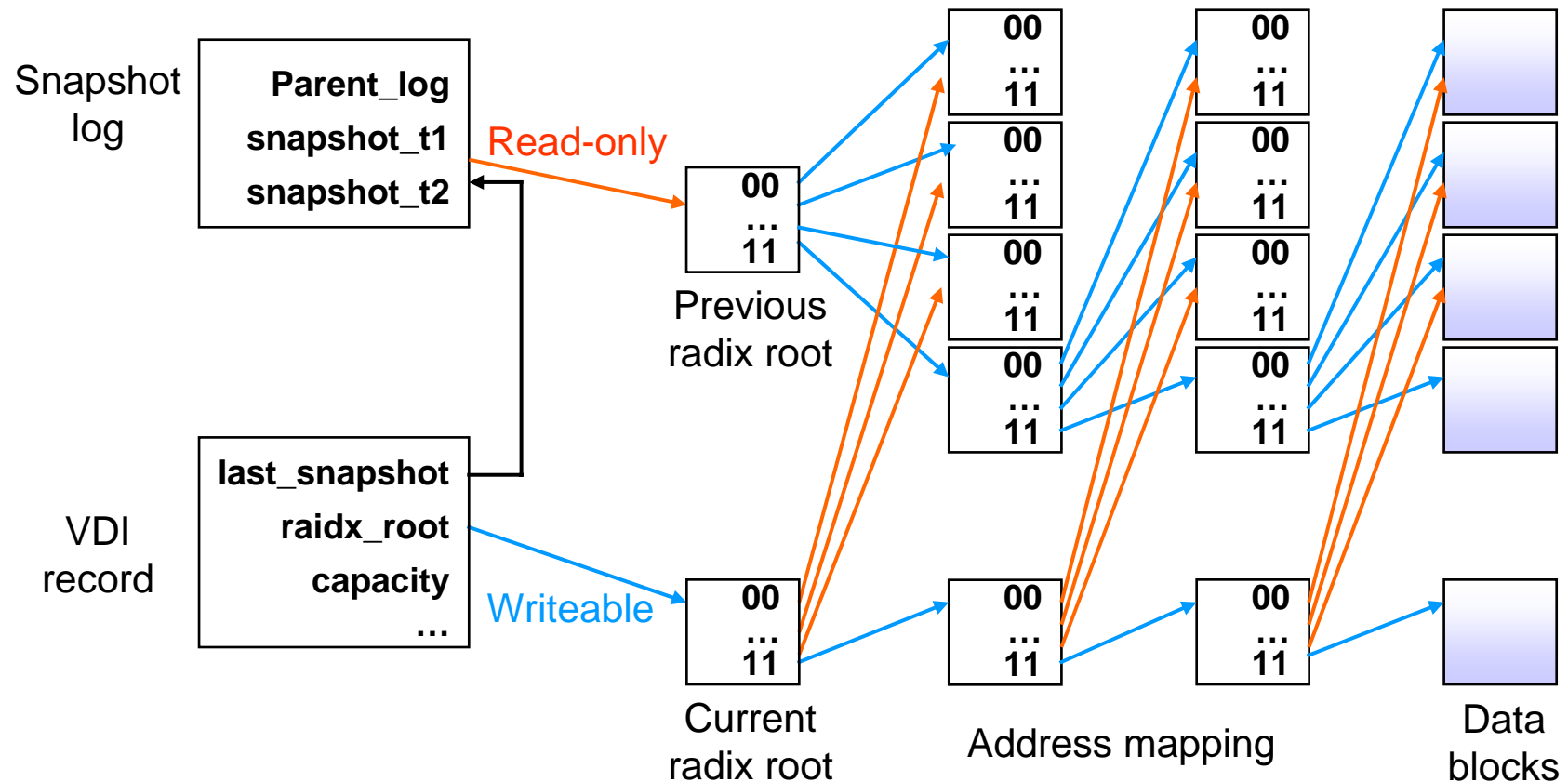


# Virtual disk image (VDI)

- The basic unit of persistent storage management
- Represents:
  - Current, writable persistent state of a virtual disk
  - And a set of VM states at points of its history (immutable snapshots)
- Accessible to cluster-wide physical machines
- Stored with redundancy for reliability

# Snapshot and address mapping

- A virtual disk is described in metadata as a log of snapshots
- Each snapshot points to the root of a radix tree
- Radix trees represent virtual-to-physical mappings



# Block-level sharing

- A VDI's snapshots share read-only common blocks
- VDIs derived from the same template image share read-only common blocks
  - A small set of well-known base images as templates (Fedora Core, FreeBSD, etc.)

# Persistent caching

- Aggressive caching for both data and metadata
  - No write-sharing of VDIs
  - No need for radix tree lookup upon data cache hit
  - Periodical write-back to cluster-wide replicas
  - Reduce server load by leveraging locality

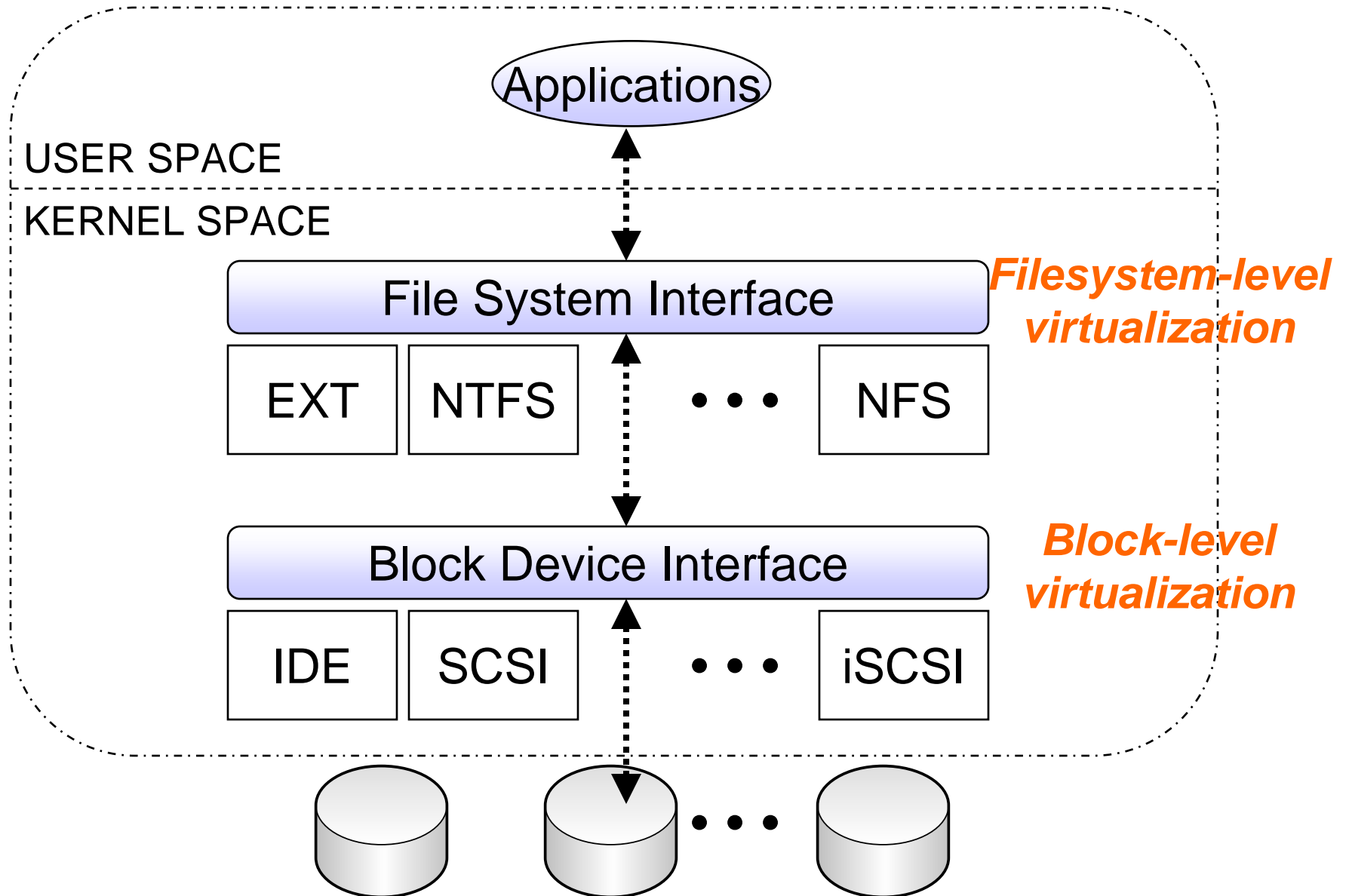
# Improved block-level sharing

- Sharing of blocks that have the same contents
  - Writes to blocks are initially made to cache
  - Content-based hashing are calculated asynchronously (to hide the overhead)
  - Existing block's ID is stored in the radix tree for a duplicate write

# Benefits of Parallax VM

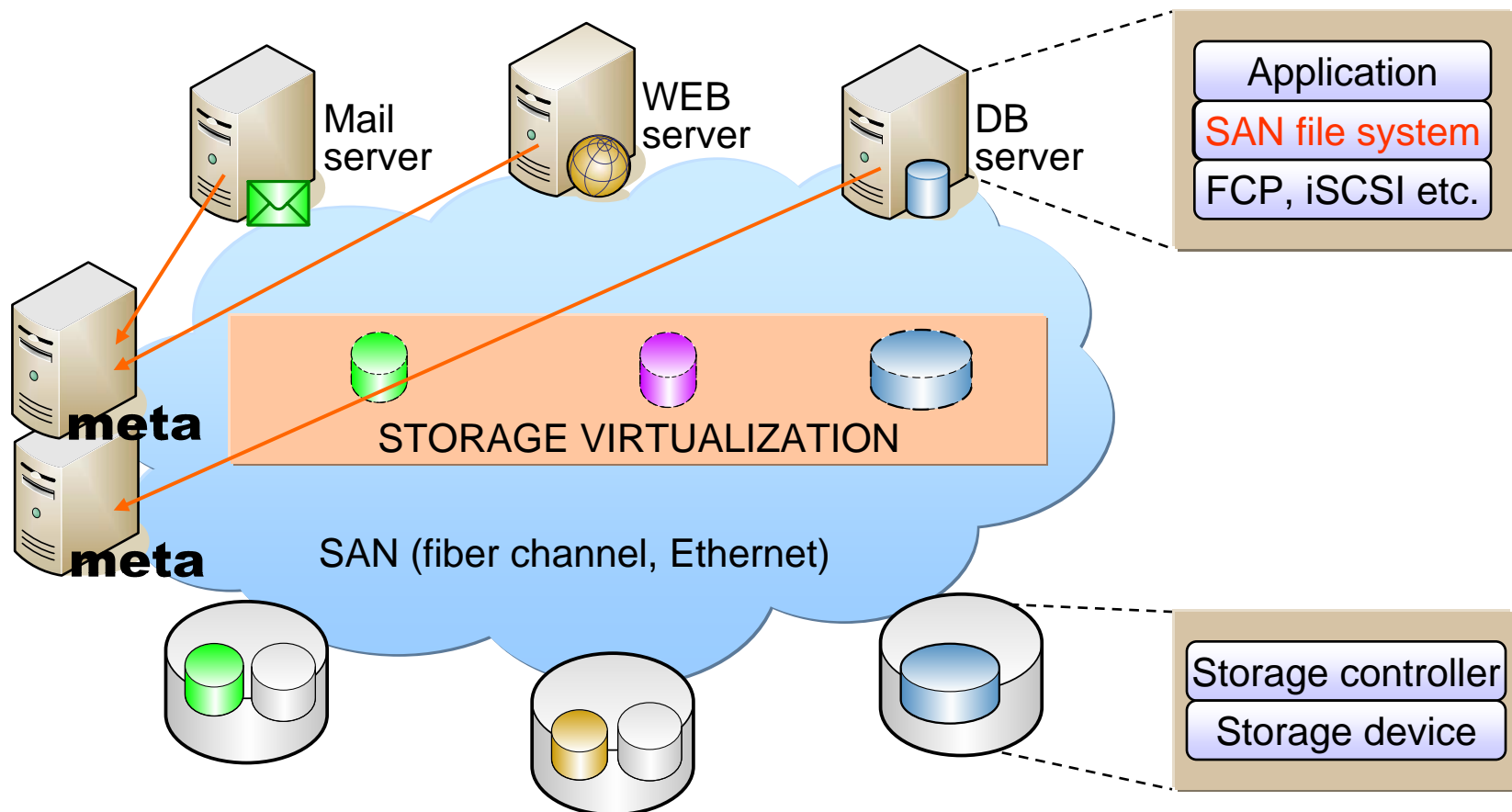
- Separate storage server administration from other administration tasks (client VMs, VMMs)
- Isolated driver VM improves robustness and recoverability
- Fate sharing between Parallax VM and its local clients
  - No need to worry about network failures
  - Non-local clients are not affected on a Parallax VM's failure

# The path to storage: where to virtualize?



# Storage virtualization systems

- Storage area networks (SAN) virtualization
  - SAN file systems enable sharing of data



# Storage virtualization systems

- Petal: distributed virtual disks (DEC, 1996)
  - Frangipani: A Scalable Distributed File System (1997)

