CSE 487/587
Data Intensive Computing
Lecture 20: SAN/NAS
Part 2 of Storage

Lot of info borrowed from James Plank, EMC, MS

Vipin Chaudhary

vipin@buffalo.edu

716.645.4740
305 Davis Hall

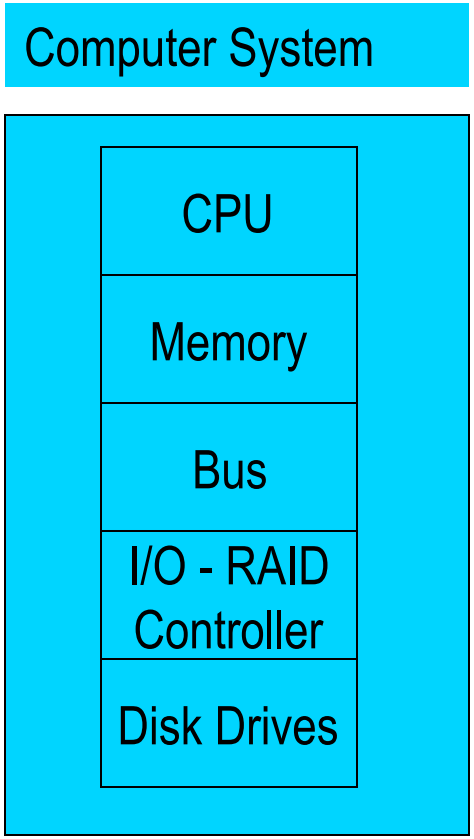
Overview of Lecture Series

- Basics – DAS, SAN, NAS
- RAID
- Erasure codes
- SSDs
- Newer Technologies

A Few Storage Basics….

- Where will data finally end up?
- How will it get there?
- What will it pass through?

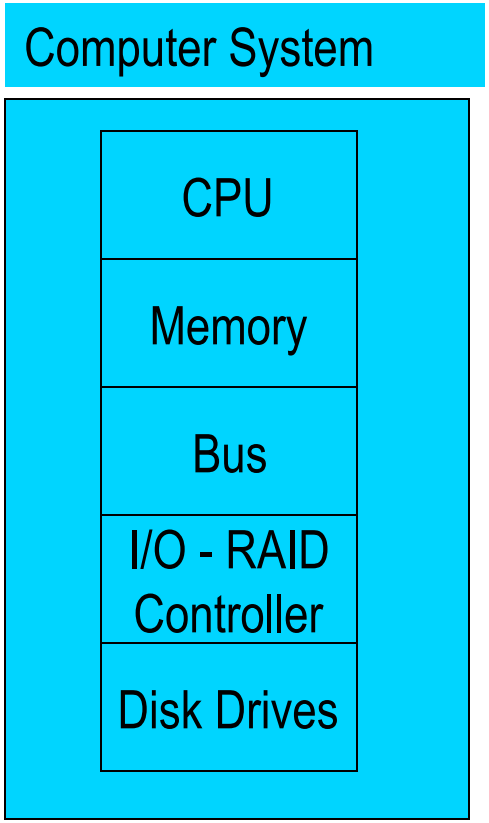
Direct Attached Storage (Internal)



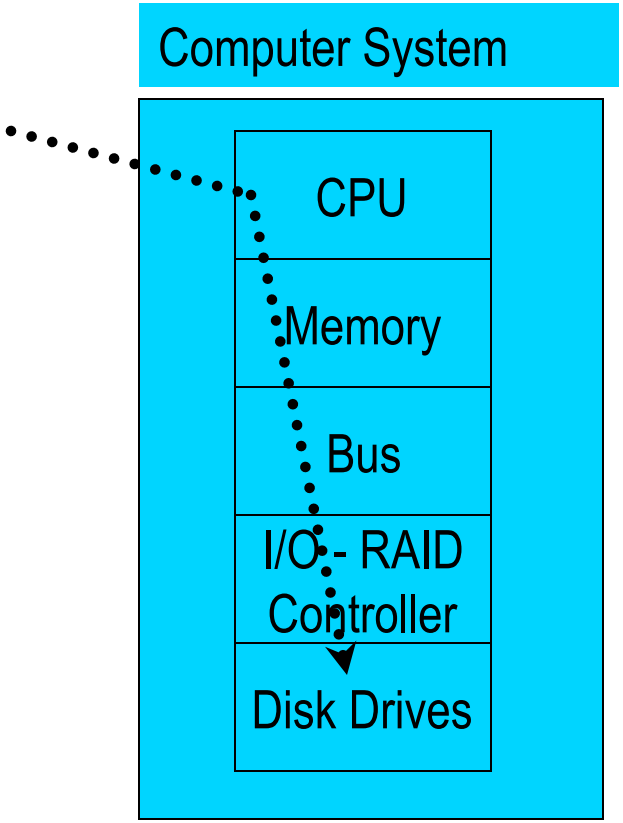
Direct Attached Storage (Internal)

305 Davis Hall
SUNY
Data Intensive
Cse 487/587

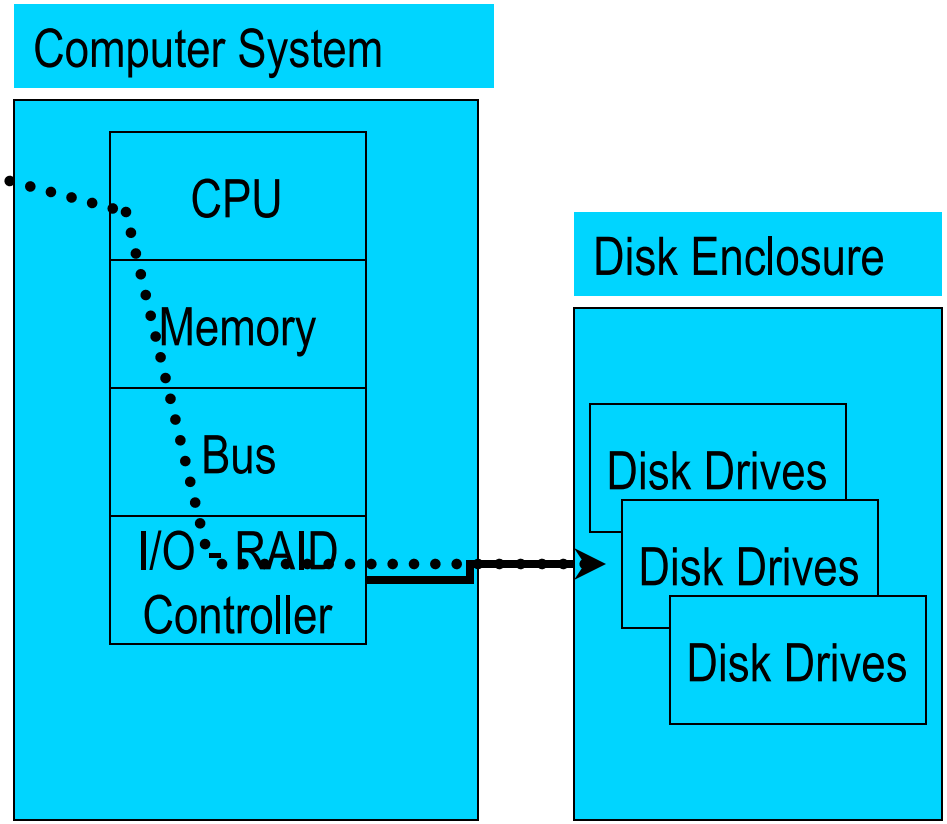
Data



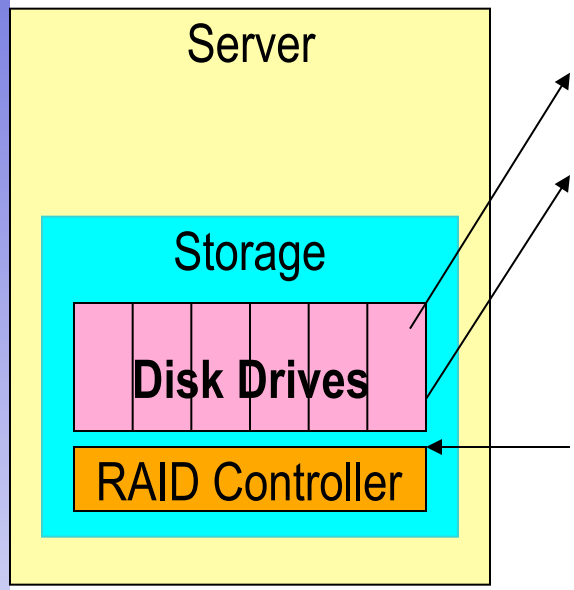
Direct Attached Storage (Internal)



DAS w/ internal controller and external storage



Comparing Internal and External Storage



Internal Storage

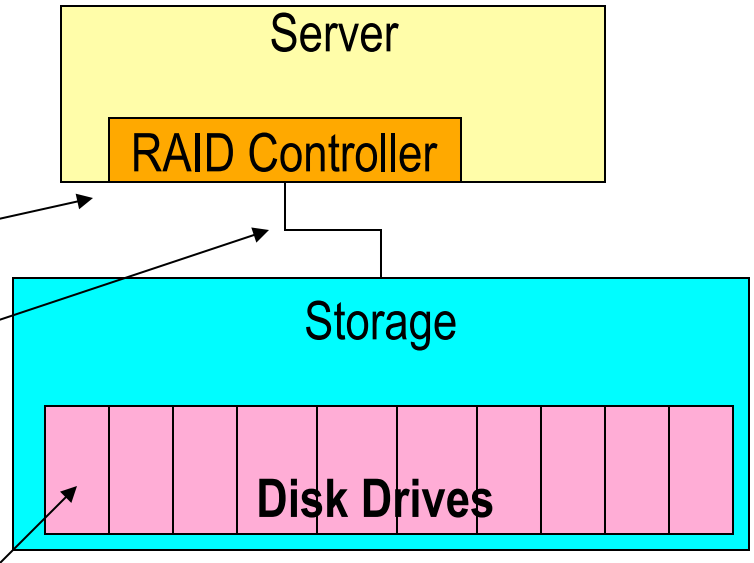
RAID controllers and disk drives are internal to the server

SCSI, ATA, or SATA protocol between controller and disks

RAID controller is internal

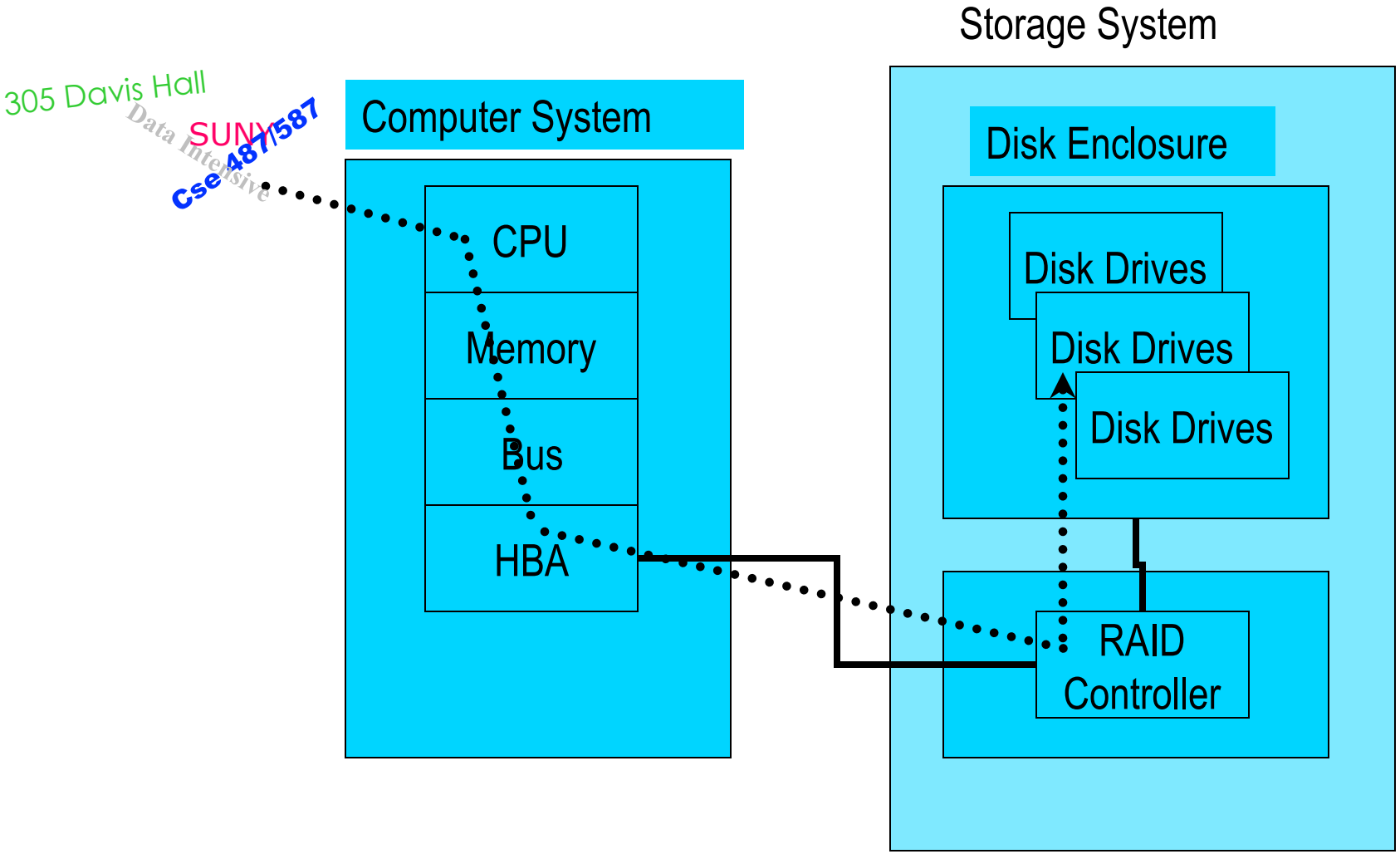
SCSI or SATA protocol between controller and disks

Disk drives are external

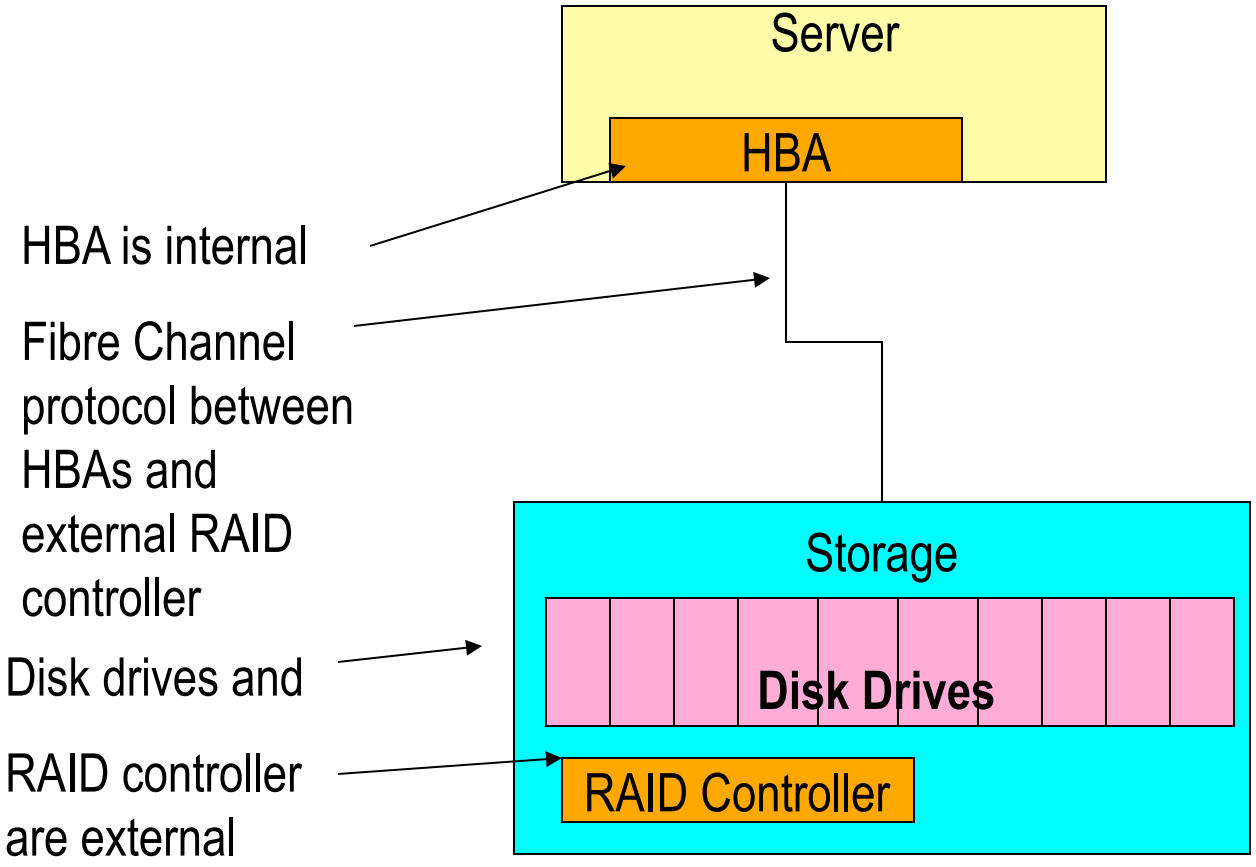


SCSI Bus w/ external storage

DAS w/ external controller and external storage



DAS over Fibre Channel



External SAN Array

I/O Transfer

- RAID Controller
  - Contains the “smarts”
  - Determines how the data will be written (striping, mirroring, RAID 10, RAID 5, etc.)
- Host Bus Adapter (HBA)
  - Simply transfers the data to the RAID controller.
  - Doesn’t do any RAID or striping calculations.
  - “Dumb” for speed.
  - Required for external storage.

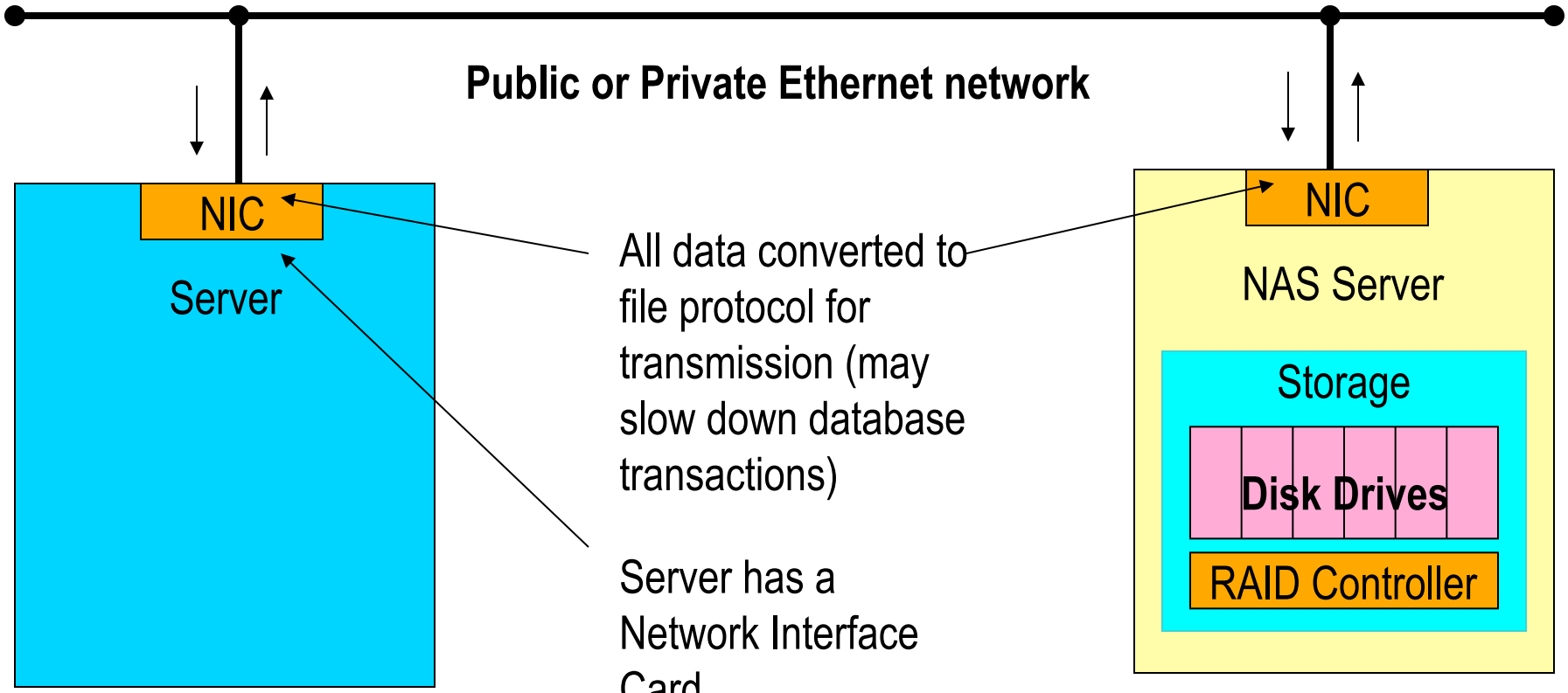
Storage types

- Single Disk Drive
- JBOD
- Volume
- Storage Array
- DAS
- NAS
- SAN

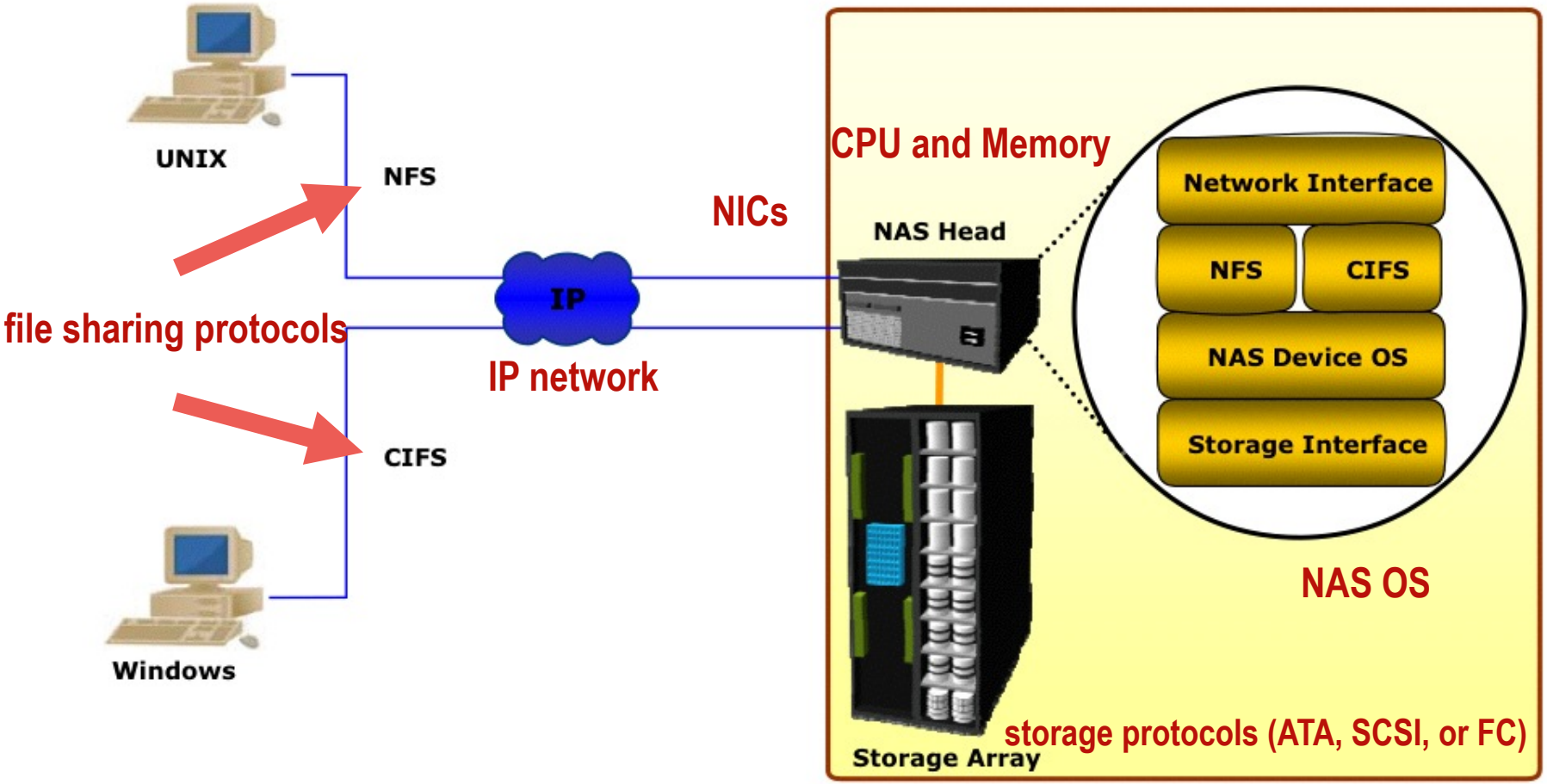
NAS: What is it?

- Network Attached Storage
- Utilizes a TCP/IP network to "share" data
- Uses file sharing protocols like Unix NFS and Windows CIFS
- Storage "Appliances" utilize a stripped-down OS that optimizes file protocol performance

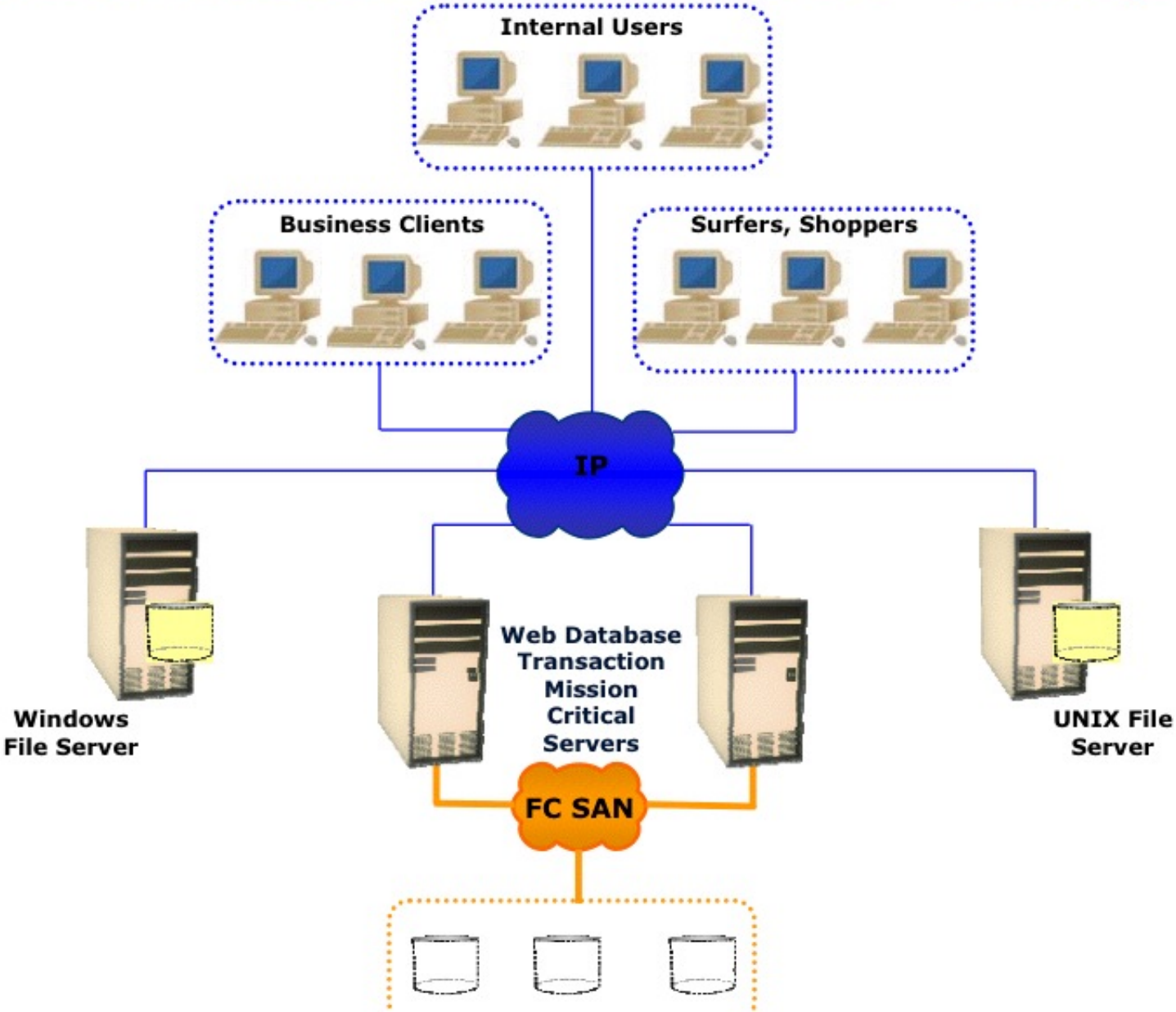
Networked Attached Storage



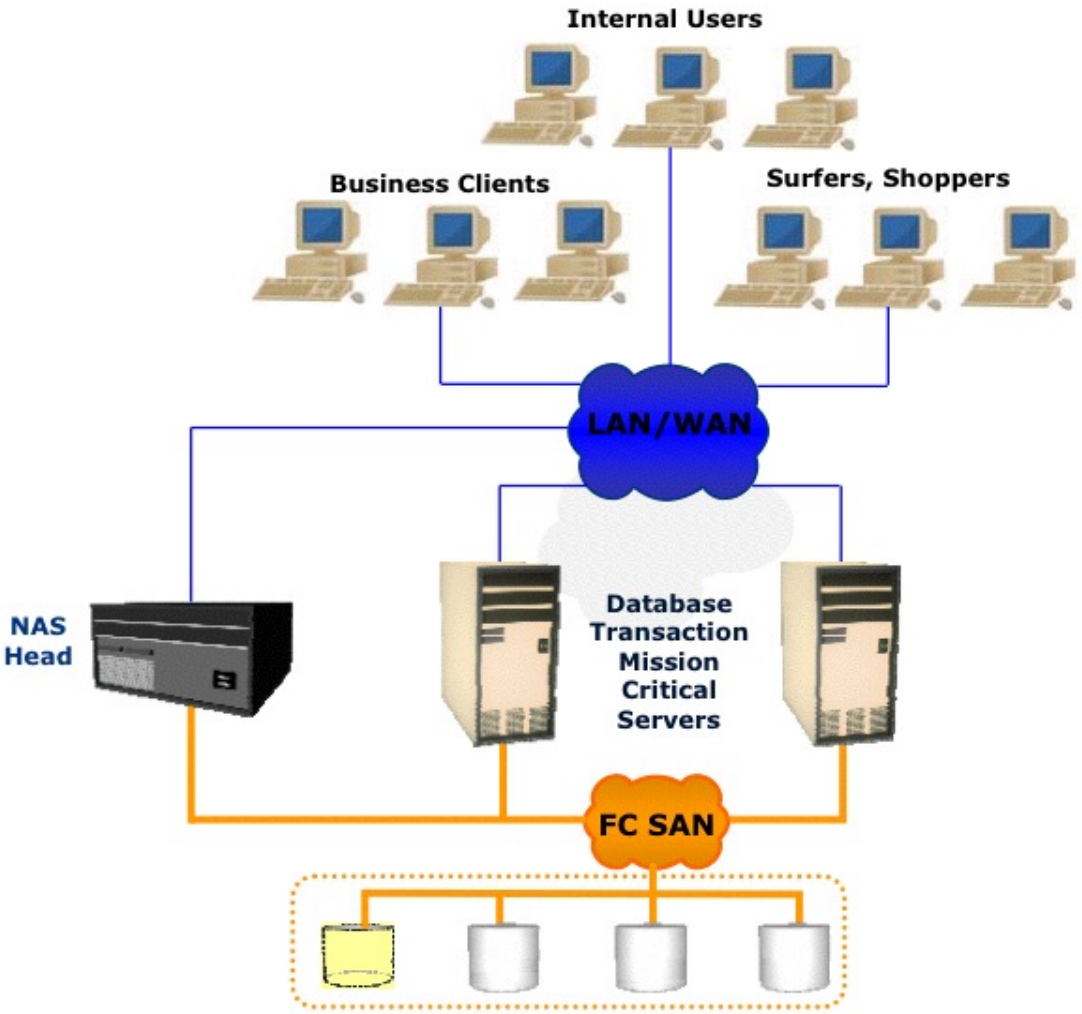
Components of NAS



Traditional File Server Environment – Example 1



Storage Consolidation with NAS

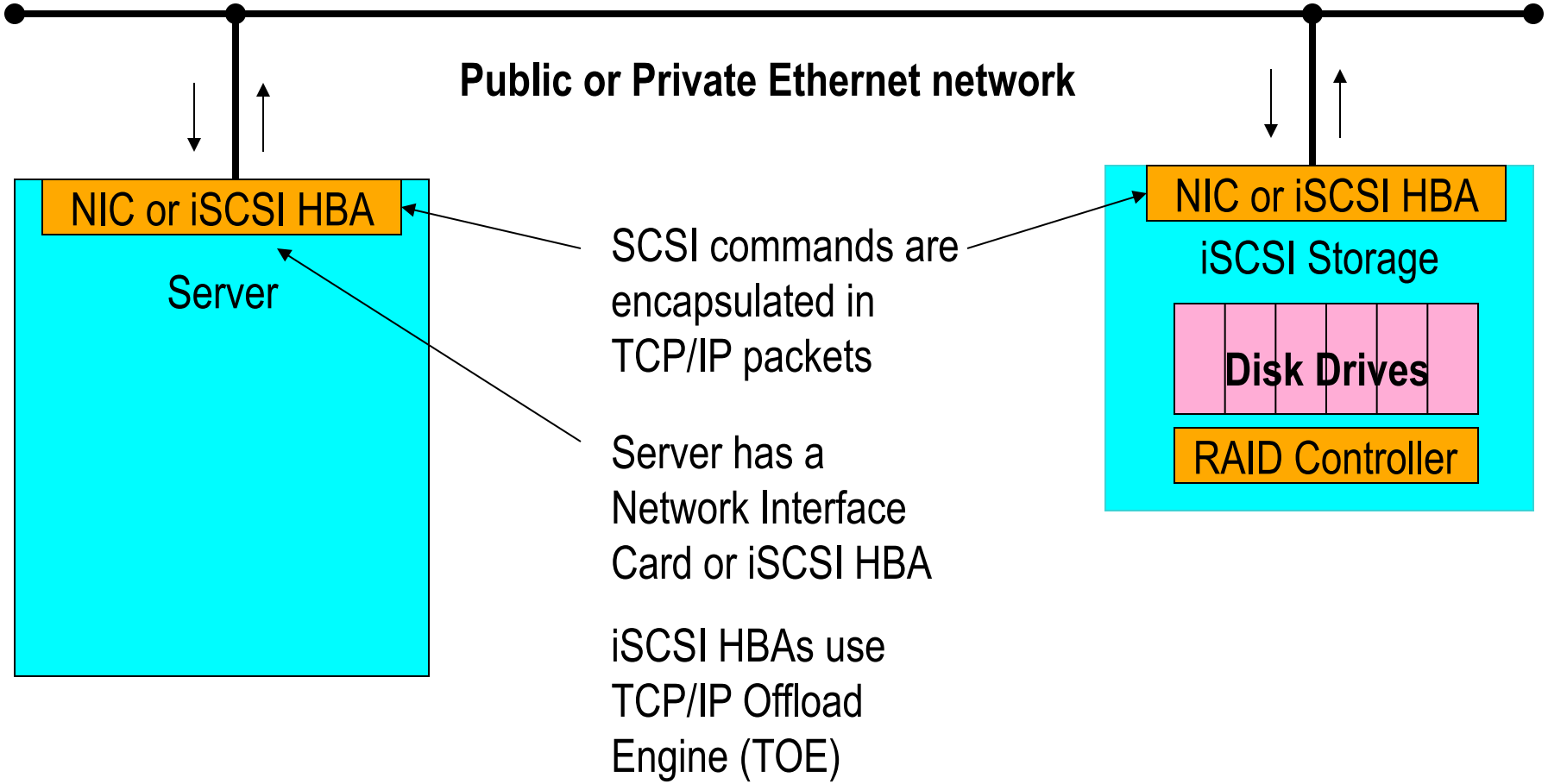


- Benefits:
- Increases performance throughput (service level) to end users
- Minimizes investment in additional servers
- Provides storage pooling
- Provides heterogeneous file servings
- Uses existing infrastructure, tools, and processes

iSCSI: What is it?

- An alternate form of networked storage
- Like NAS, also utilizes a TCP/IP network
- Encapsulates native SCSI commands in TCP/IP packets
- Supported in Windows 2003 Server and Linux
- TCP/IP Offload Engines (TOEs) on NICs speed up packet encapsulation

iSCSI Storage



SCSI commands are encapsulated in TCP/IP packets

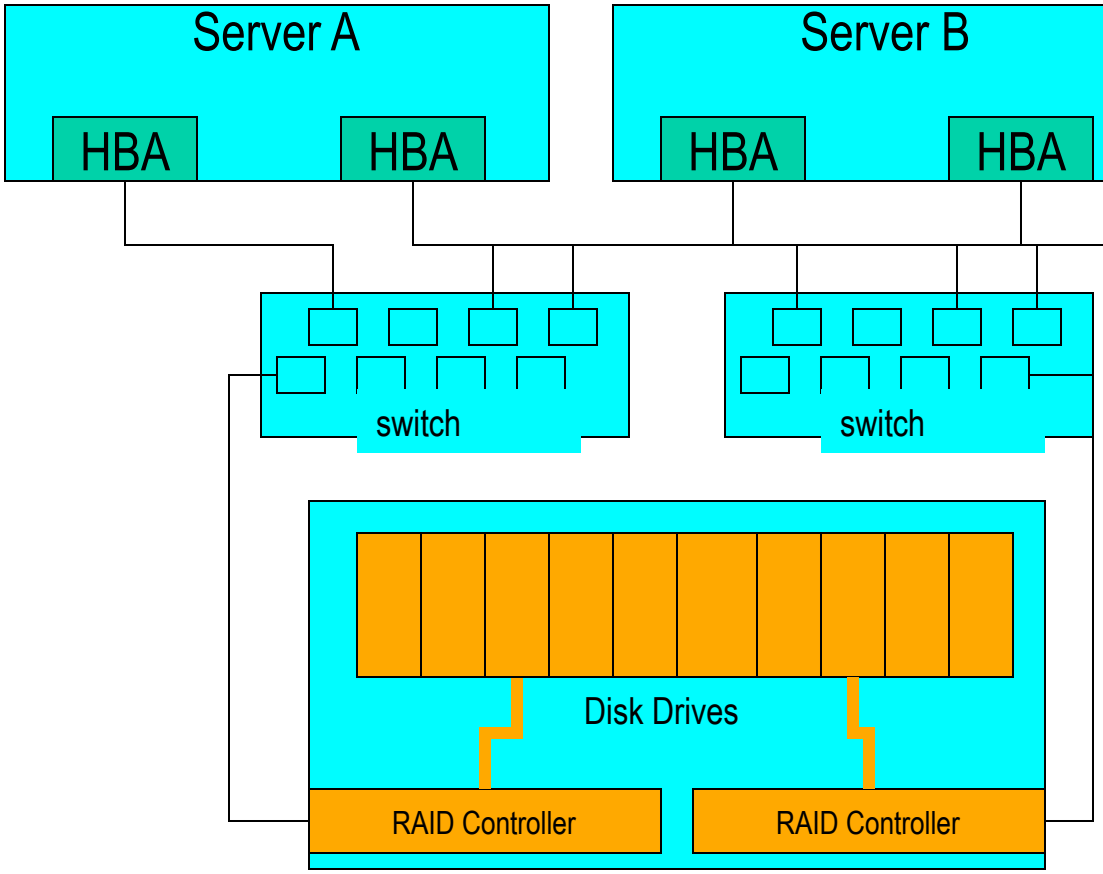
Server has a Network Interface Card or iSCSI HBA

iSCSI HBAs use TCP/IP Offload Engine (TOE)

Fibre Channel: What is it?

- Fibre Channel is a network protocol implemented specifically for dedicated storage networks
- Fibre Channel utilizes specialized
  - Switches
  - Host Bus Adapters
  - RAID controllers
  - Cables

Fibre Channel Components



FC Storage Array

- **Servers**
  - Host Bus Adapters
- **Cables**
  - Fiber optic or copper
- **Fibre Channel Switches**
  - Two switches for redundancy
- **Fibre Channel Storage Array**
  - Two RAID Controllers for redundancy
  - 4–100+ disk drives per array
- **A true storage network**
  - Multiple servers
  - Multiple switches
  - Multiple Storage Arrays

SAN: What is it?

- Storage Area Network
- A network whose primary purpose is the transfer of data between storage systems and computer systems
- Fibre Channel is the primary technology utilized for SANs
- Recently, SANs have been implemented with dedicated iSCSI networks

Benefits of SAN/Consolidated Storage

- Reduce cost of external storage (?)
- Increase performance
- Centralized and improved tape backup
- LAN-less backup
- High-speed, no single-point-of-failure clustering solutions
- Consolidation

Fibre Channel Technology

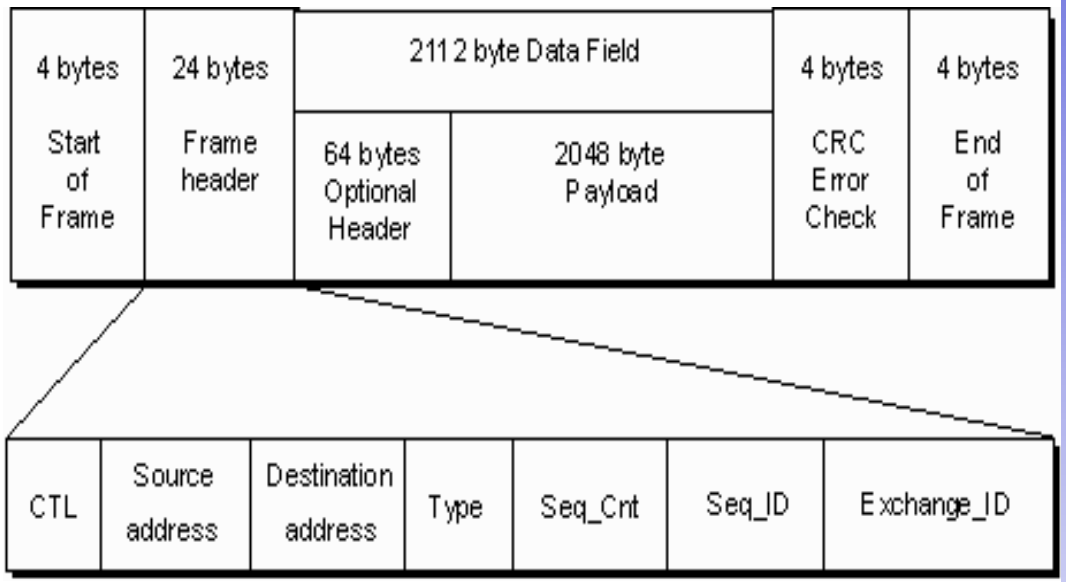
- Provides concurrent communications between servers, storage devices, and other peripherals
- A gigabit interconnect technology
  - FC1: Over 1,000,000,000 bits per second
  - FC2: Over 2,000,000,000 bits per second
  - FC16: now
  - FC128: projected in 2016?
- A highly reliable interconnect
- Up to 127 devices (SCSI: 15)
- Up to 10 km of cabling (3-15 ft. for SCSI)
- Physical interconnect can be copper or fiber optic

Fibre Channel – (continued)

- Hot-pluggable - Devices can be removed or added at will with no ill effects to data communications
- Provides a data link layer above the physical interconnect, analogous to Ethernet
- Sophisticated error detection at the frame level
- Data is checked and resent if necessary

Fibre Channel – Frame Dissection

- Up to 2048 byte payload
- 4 byte checksum for each frame



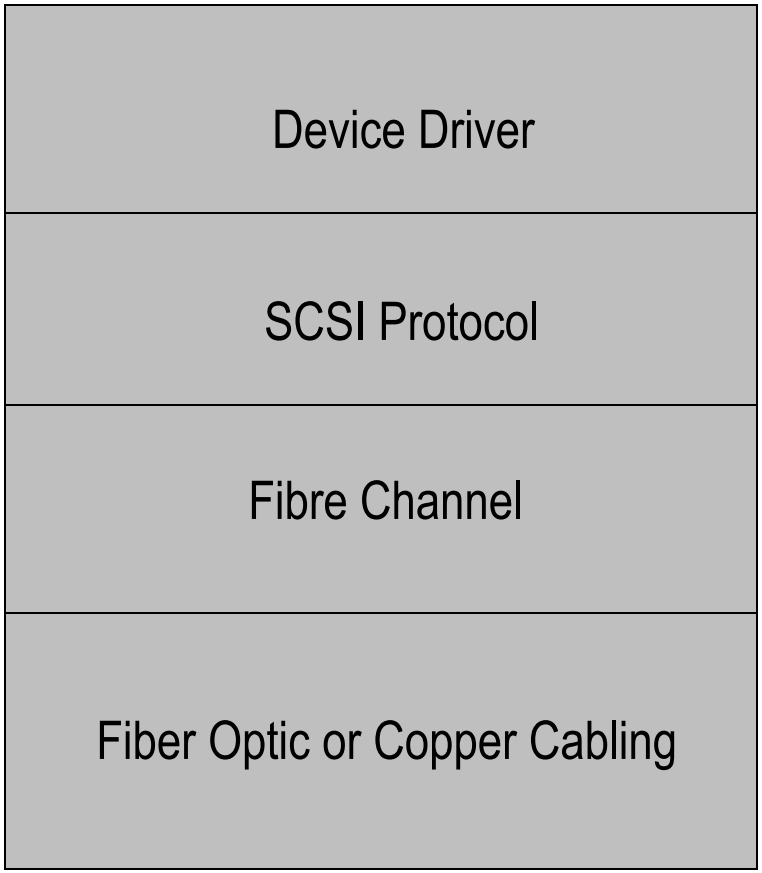
Fibre Channel

- What's with the funny name?
  - Some background history required
  - Originally developed to only support fiber optic cabling
  - When copper cabling support was added, ISO decided not to rename the technology
  - ISO changed to the French spelling to reduce association with fiber optics only medium

Fibre Channel

- How does it work?
  - Serial interface
  - Data is transferred across a single piece of medium at the fastest speed supported
  - No complex signaling required

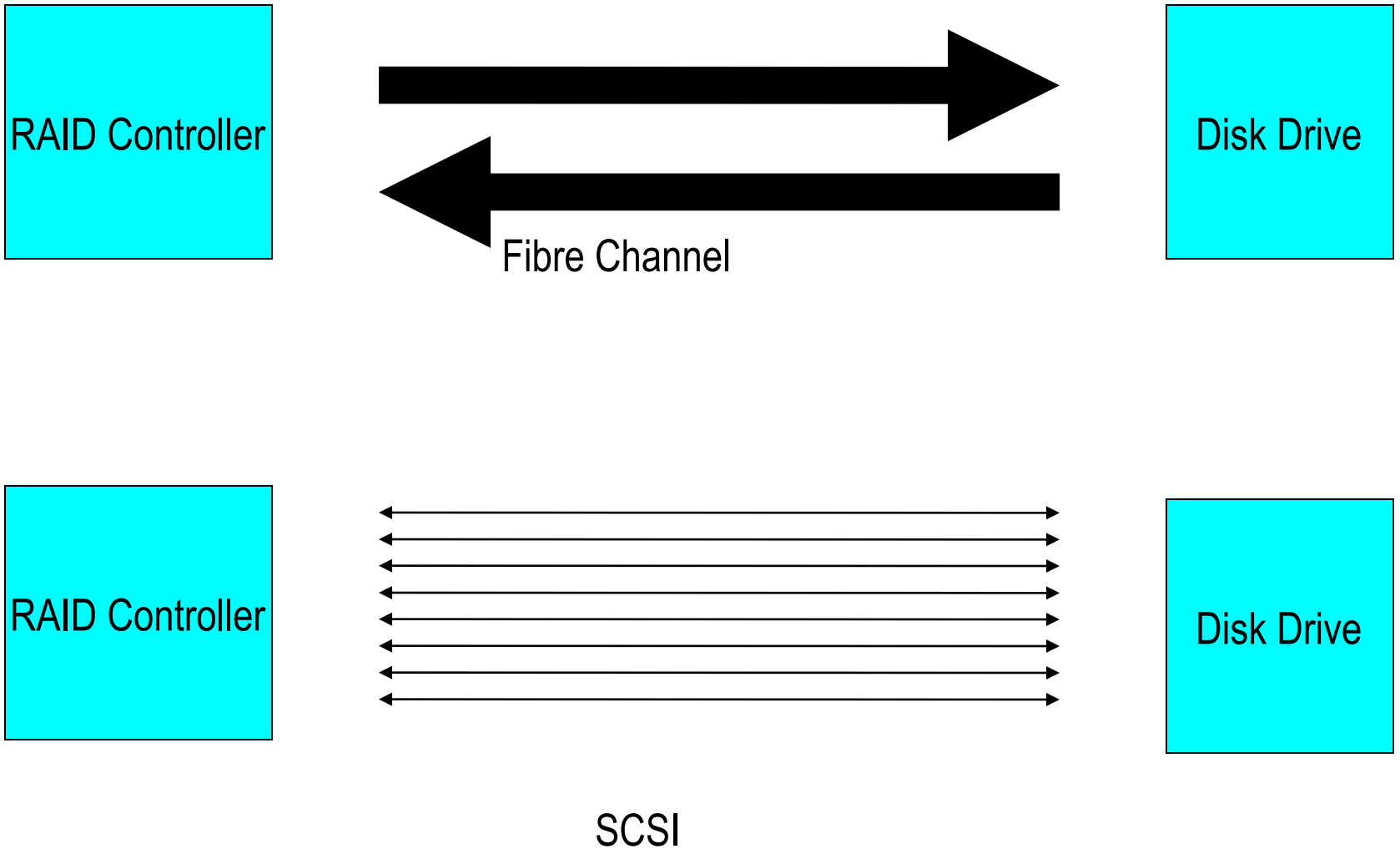
Fibre Channel Interface Layers



SCSI vs. Fibre Channel *Protocol*

- SCSI
  - SCSI protocol vs. SCSI device
  - SCSI is an established, tried and true protocol
  - Provides services analogous to TCP/IP
  - Supported in every major OS on market
- Fibre Channel
  - Fibre Channel runs on top of SCSI
  - No re-inventing the wheel
  - Immediate OS support

SCSI vs. FC Transmission



SCSI vs. Fibre Channel

- Interface for internal storage to external disks
- Potential down time w/ SCSI
- Single bus
- RAID controller is SCSI hardware
- Standards:
  - Ultra2 (80 MB/sec)
  - Ultra 160 (160 MB/sec)
  - Ultra 320 (320 MB/sec)
- Media specific (copper only)
- SCSI Limitations:
  - Cables can't be any longer than 3 feet for single ended; 15 feet for LVD (low voltage differential)
  - No more than 15 devices on a SCSI bus
  - # of disk drives

- Used with SAN
- Lots of built-in redundancy with connections
- Redundant network
- HBA is fibre channel hardware
- Standards:
  - FC16: 16 GB/sec
  - FC128: 128 GB/sec
- Provides a data link layer above the physical interconnect
  - Analogous to Ethernet
  - FC is a network of devices
  - It can be media independent- copper or fibre optic
- Fibre Channel limitations:
  - Cable length: Up to 10 kilometers (more a limitation of cable than FC itself)
  - Up to 127 devices
  - # of disk drives

Fibre Channel vs. iSCSI

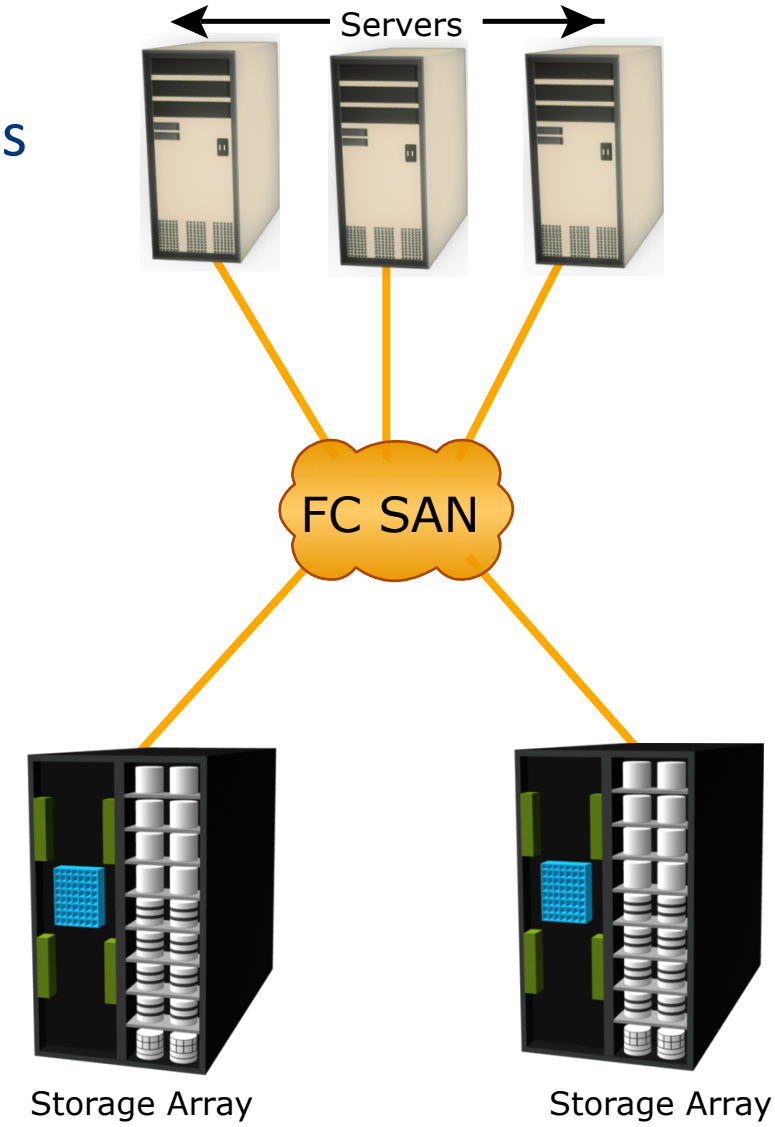
- Fibre Channel
  - The current market leader for shared storage technologies
  - Provides the highest performance levels
  - Designed for mission-critical applications
  - Cost of components is relatively high, particularly per server HBA costs
  - Relatively difficult to implement and manage
- iSCSI
  - Relatively new, but usage is increasing rapidly
  - Performance can approach Fibre Channel speeds
  - A better fit for databases than NAS
  - A good fit for Small to Medium Size Businesses
  - Relatively inexpensive, compared to Fibre Channel
  - Relatively easy to implement and manage

Summary

- How data is routed through a server to I/O
- Types of storage
  - DAS
  - NAS
  - iSCSI
  - SAN
- Benefits of SAN technology
  - Storage consolidation
  - Reduced costs
  - Centralized, LAN-free backup and restore
- The Fibre Channel protocol
  - How it works
  - Fibre Channel protocol vs. SCSI protocol
- Comparing Fibre Channel SANs and iSCSI SANs
  - Fibre Channel SANs offer mission-critical performance, with relatively high costs and high complexity
  - iSCSI SANs offer moderate to high performance at an attractive price/performance ration and are relatively easy to administer

What is a SAN ?

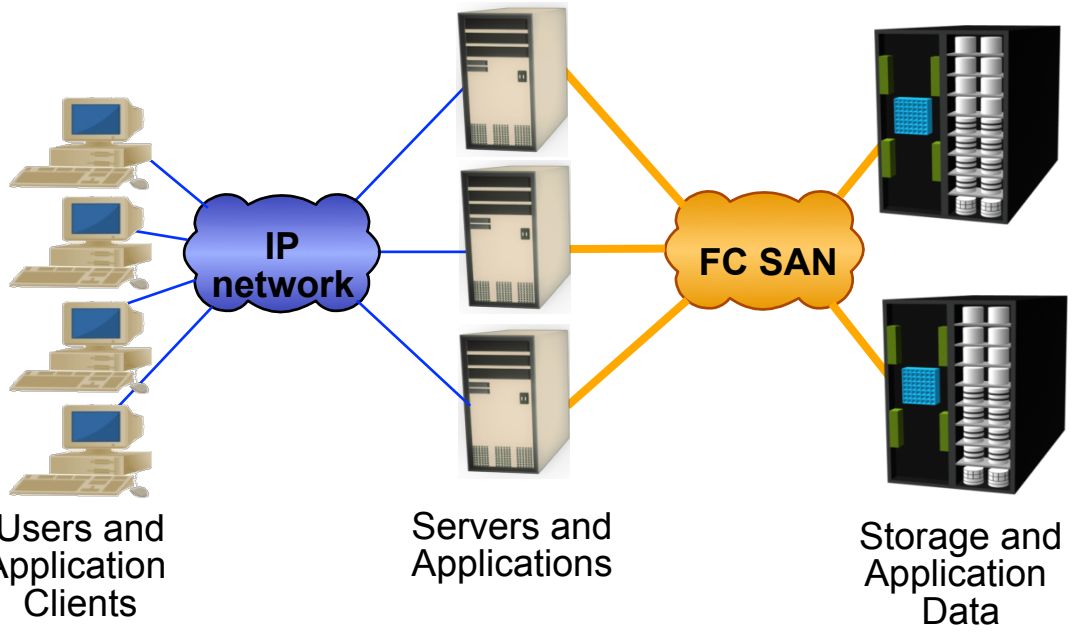
- Dedicated high speed network of servers and shared storage devices
- Provide block level data access
- Resource Consolidation
  - Centralized storage and management
- Scalability
  - Theoretical limit: Appx. 15 million devices
- Secure Access



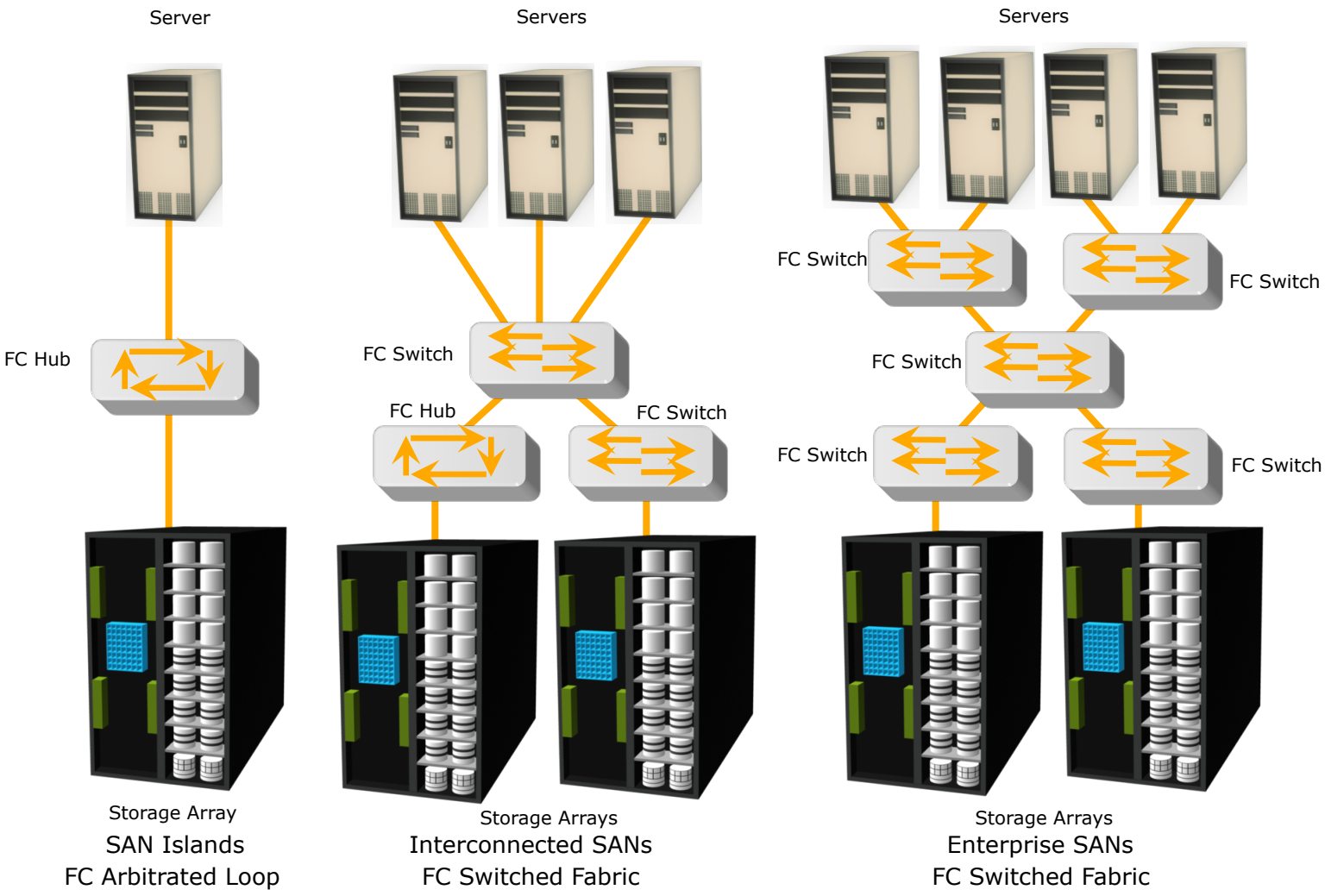
Understanding Fibre Channel

- Fibre Channel is a high-speed network technology that uses:
  - Optical fiber cables (for front end connectivity)
  - Serial copper cables (for back end connectivity)
- Latest FC implementations support 16Gb/s

o Servers are attached to 2 distinct networks

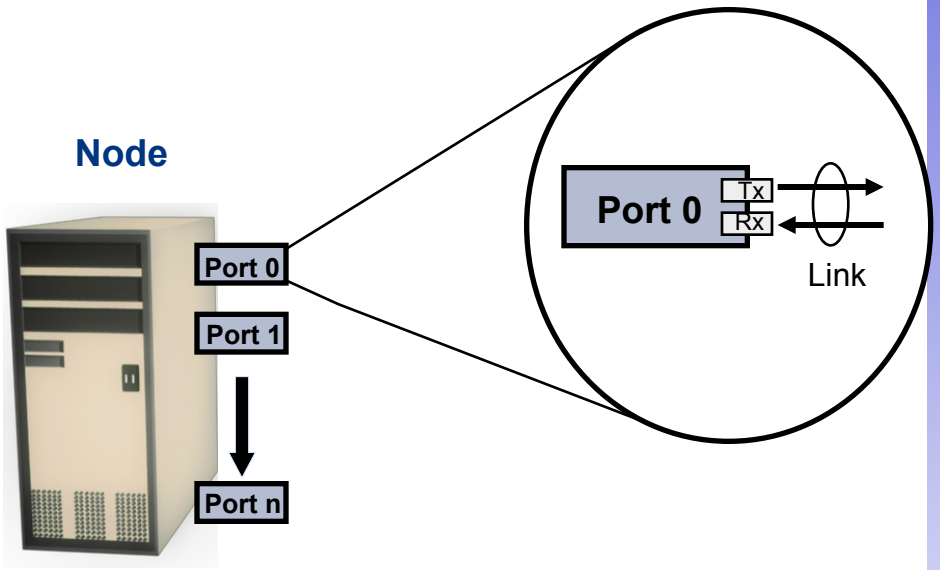


FC SAN Evolution



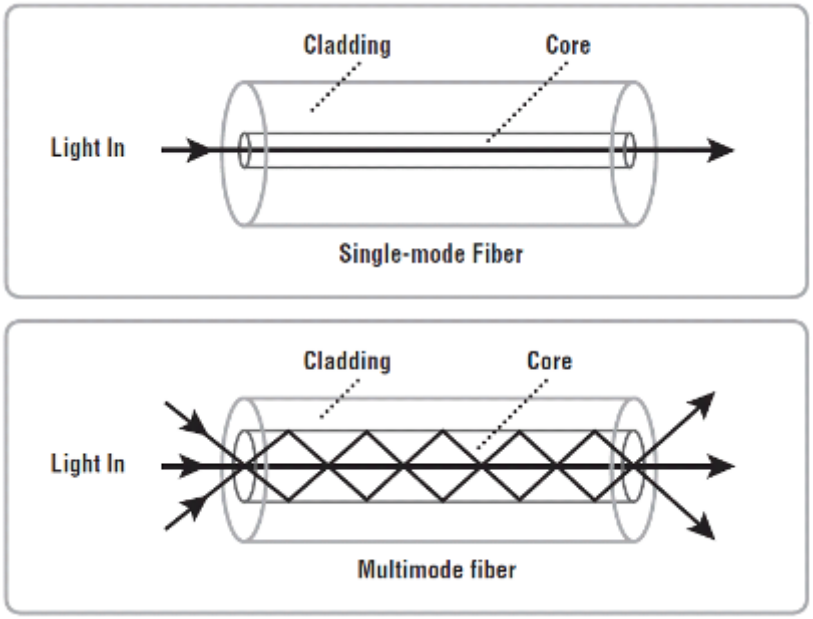
Components of SAN: Node ports

- Examples of nodes
  - Hosts, storage and tape library
- Ports are available on:
  - HBA in host
  - Front-end adapters in storage
  - Each port has transmit (Tx) link and receive (Rx) link
- HBAs perform low-level interface functions automatically to minimize impact on host performance



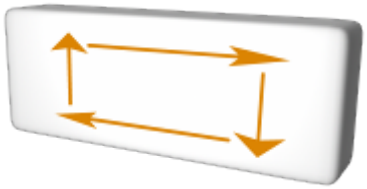
Components of SAN: Cabling

- SAN implementation uses:
  - Copper cables for short distance
  - Optical fiber cables for long distance
- Two types of optical cables
  - Single-mode
    - Can carry single beams of light
    - Distance up to 10 KM
  - Multi-mode
    - Can carry multiple beams of light simultaneously
    - Distance up to 500 meters

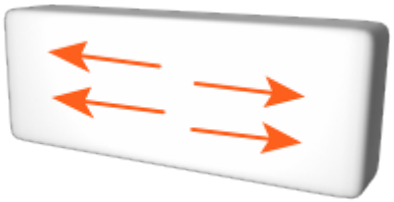


Components of SAN: Interconnecting devices

- Basis for SAN communication
  - Hubs
  - Switches and
  - Directors



FC HUB

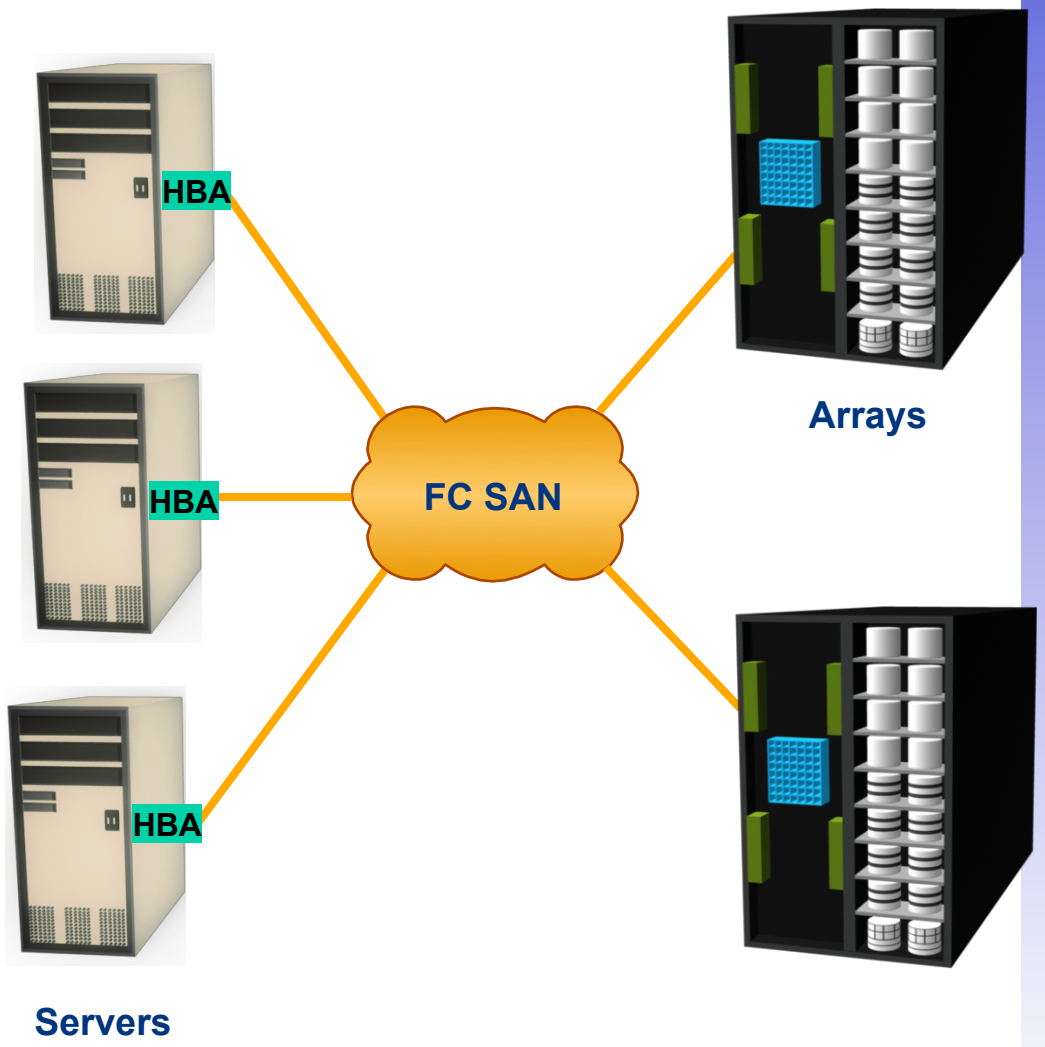


FC Switch

Director

Components of SAN: Storage array

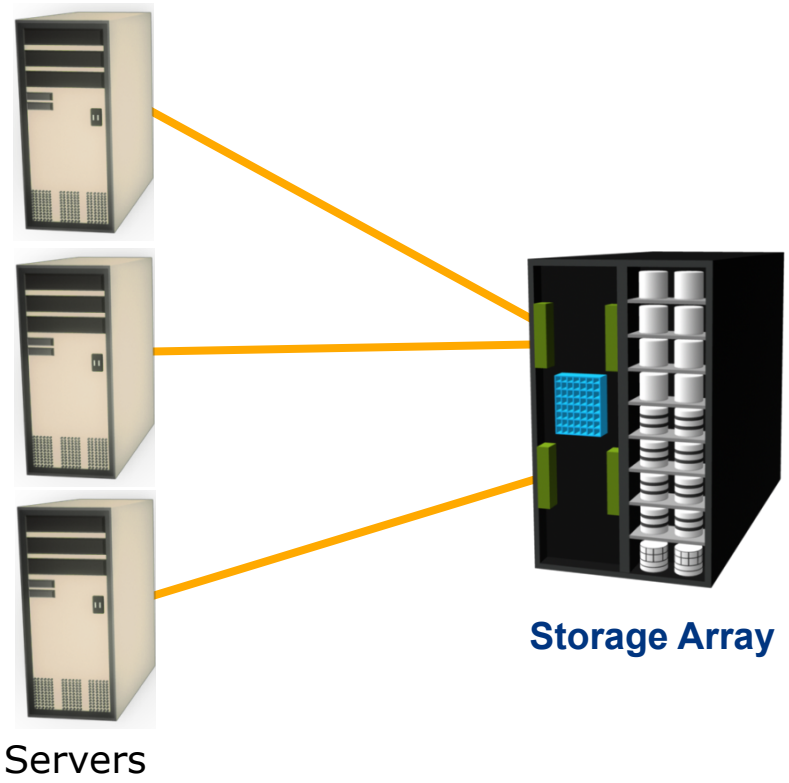
- Provides storage consolidation and centralization
- Features of an array
  - High Availability/Redundancy
  - Performance
  - Business Continuity
  - Multiple host connect



SAN Interconnectivity Options: Point to Point

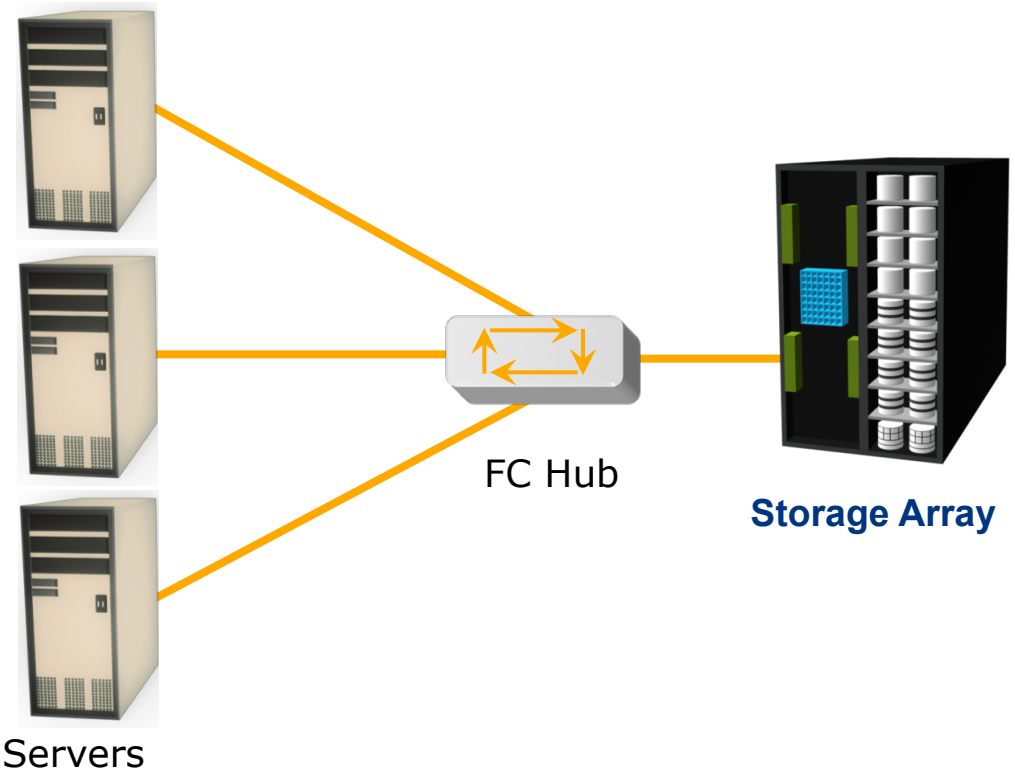
Point to point (Pt-to-Pt)

- Direct connection between devices
- Limited connectivity

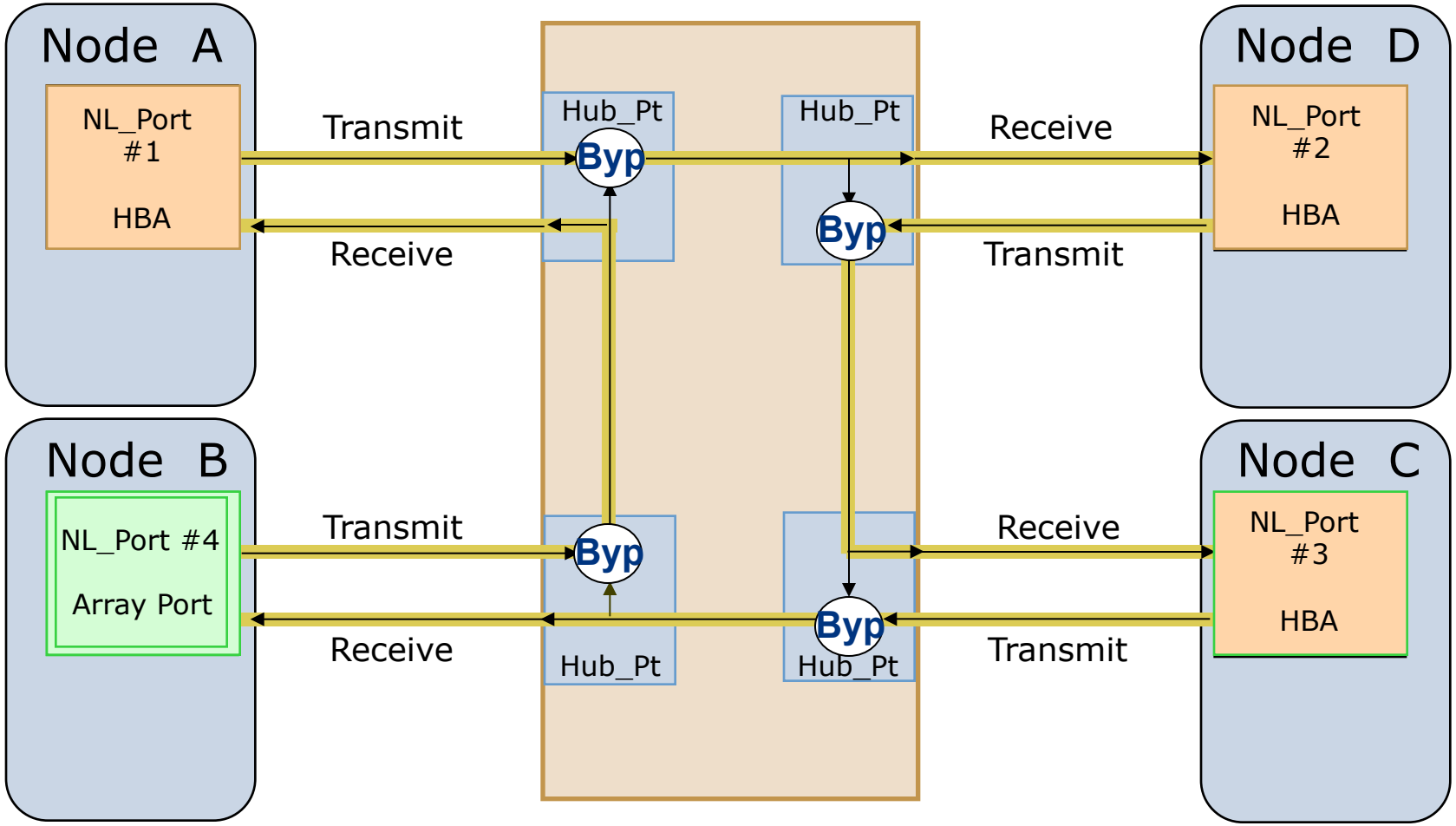


SAN Interconnectivity Options: FC-AL

- Fibre Channel Arbitrated Loop (FC-AL)
  - Devices must arbitrate to gain control
  - Devices are connected via hubs
  - Supports up to 127 devices

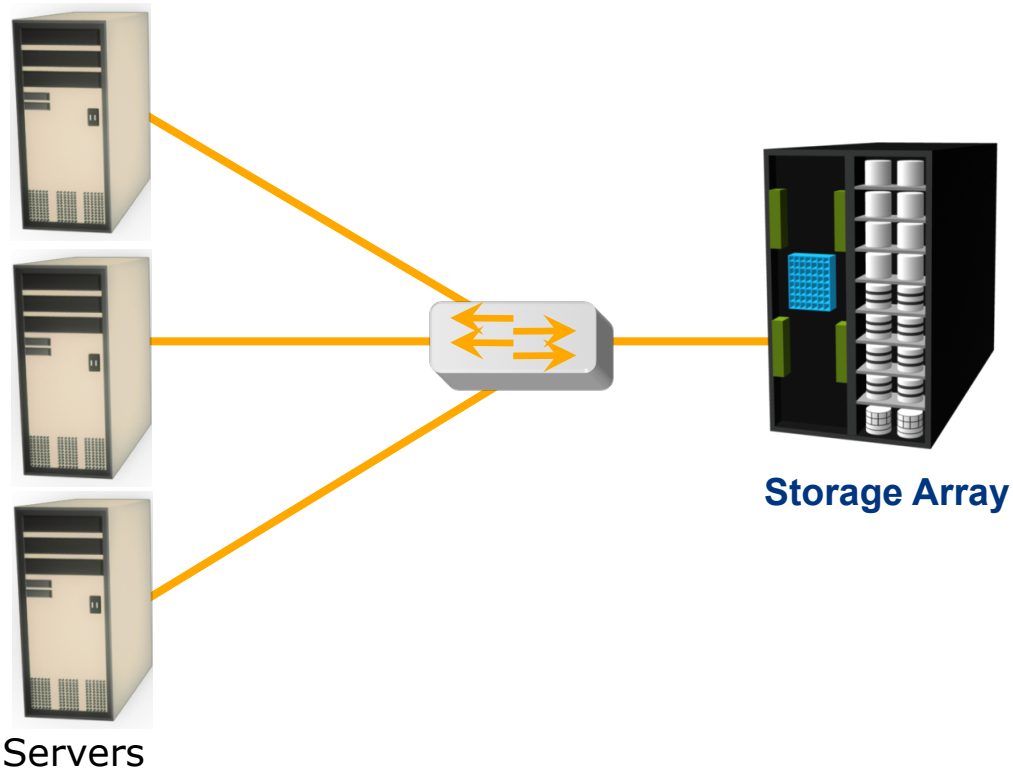


FC-AL Transmission

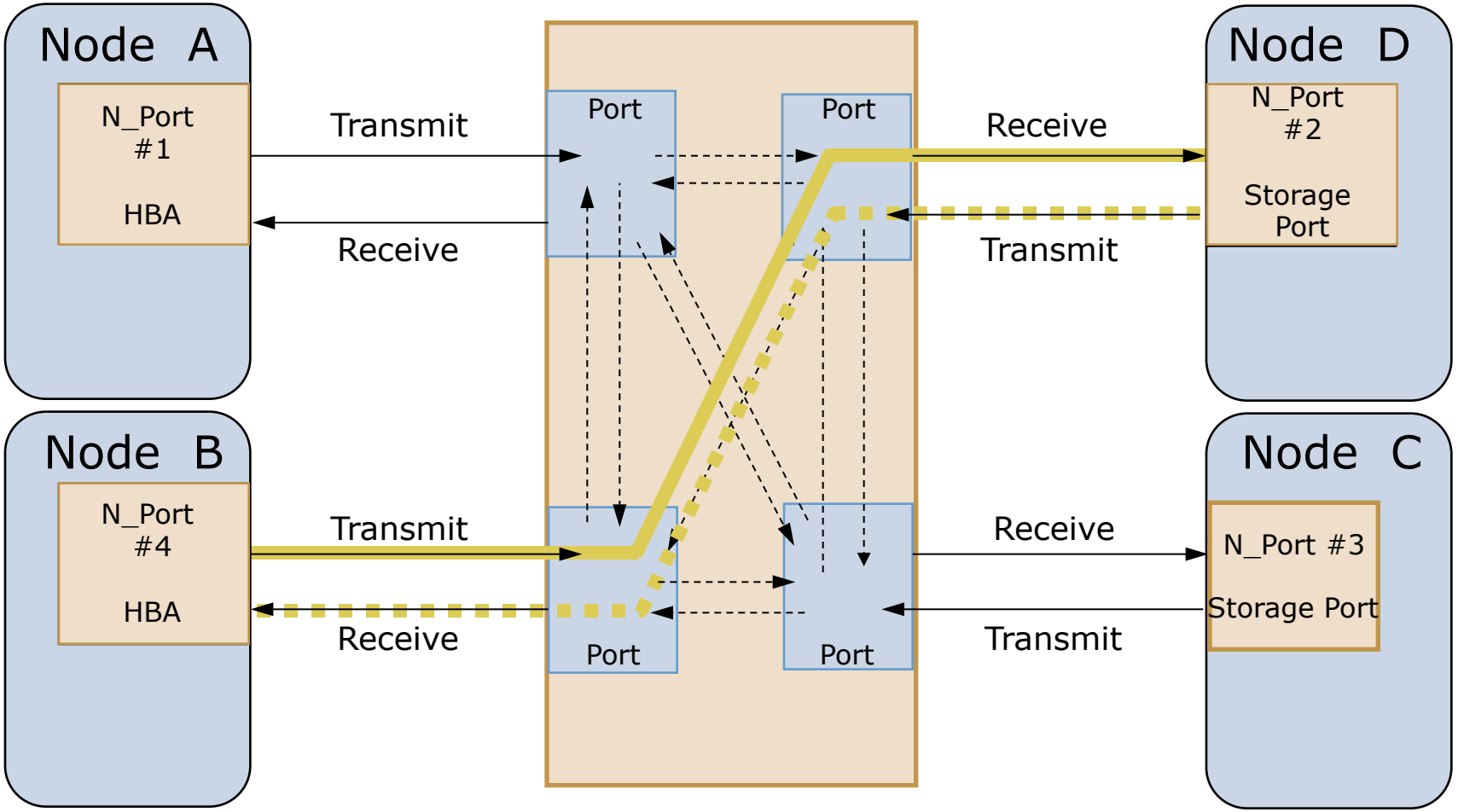


SAN Interconnectivity Options: FC-SW

- Fabric connect (FC-SW)
  - Dedicated bandwidth between devices
  - Support up to 15 million devices
  - Higher availability than hubs



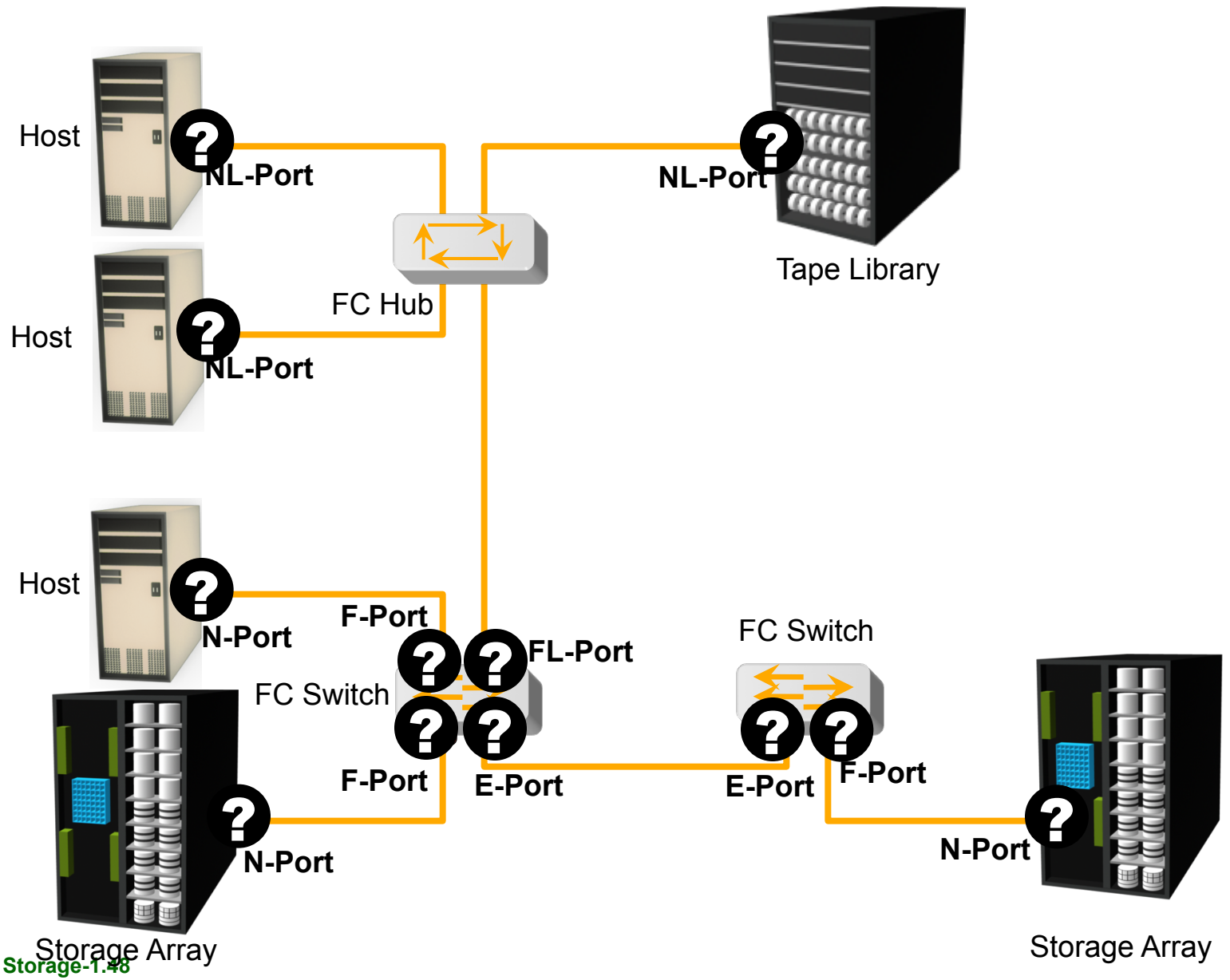
FC-SW Transmission



Port Types

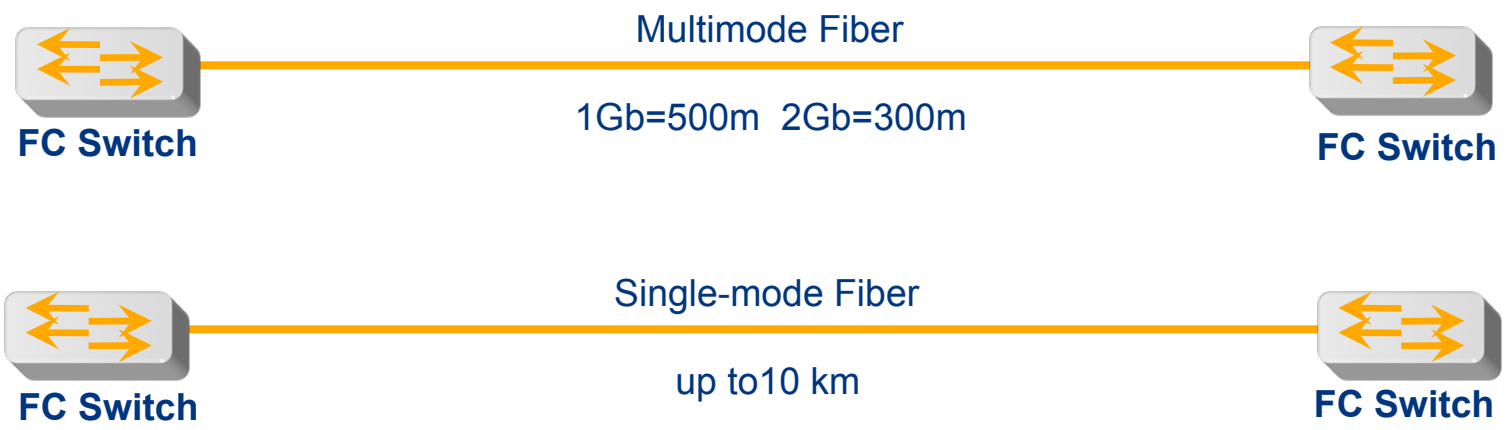
- **N_port** is a port on the node (e.g. host or storage device) used with both FC-P2P or FC-SW topologies. Also known as **node port**.
- **NL_port** is a port on the node used with an FC-AL topology. Also known as **Node Loop port**.
- **F_port** is a port on the switch that connects to a node point-to-point (i.e. connects to an N_port). Also known as **fabric port**. An F_port is not loop capable.
- **FL_port** is a port on the switch that connects to a FC-AL loop (i.e. to NL_ports). Also known as **fabric loop port**.
- **E_port** is the connection between two fibre channel switches. Also known as an **Expansion port**. When E_ports between two switches form a link, that link is referred to as an inter-switch link (**ISL**).
- **B_port** A Bridge Port is a Fabric inter-element port used to connect Bridge devices with E_Ports on a Switch. The B_Port provides a subset of the E_port functionality
- **D_port** is a diagnostic port, used solely for the purpose of running link-level diagnostics between two switches and to isolate link level fault on the port, in the SFP, or in the cable.
- **EX_port** is the connection between a fibre channel router and a fibre channel switch. On the side of the switch it looks like a normal E_port, but on the side of the router it is an EX_port.
- **TE_port *** Is an extended ISL or **EISL**. The TE_port provides not only standard E_port functions but allows for routing of multiple VSANs (Virtual SANs). This is accomplished by modifying the standard Fibre Channel frame (vsan tagging) upon ingress/egress of the VSAN environment. Also known as **Trunking E_port**.
- **VE_Port** an INCITS T11 addition, FCIP interconnected E-Port/ISL, i.e. fabrics will merge.
- **VEX_Port** an INCITS T11 addition, is a FCIP interconnected EX-Port, routing needed via Isan zoning to connect initiator to a target.

Port Types



Inter Switch Links (ISL)

- ISL connects two or more FC switches to each other using E-Ports
- ISLs are used to transfer host-to-storage data as well as the fabric management traffic from one switch to another
- ISL is also one of the scaling mechanisms in SAN connectivity



Login Types in a Switched Network

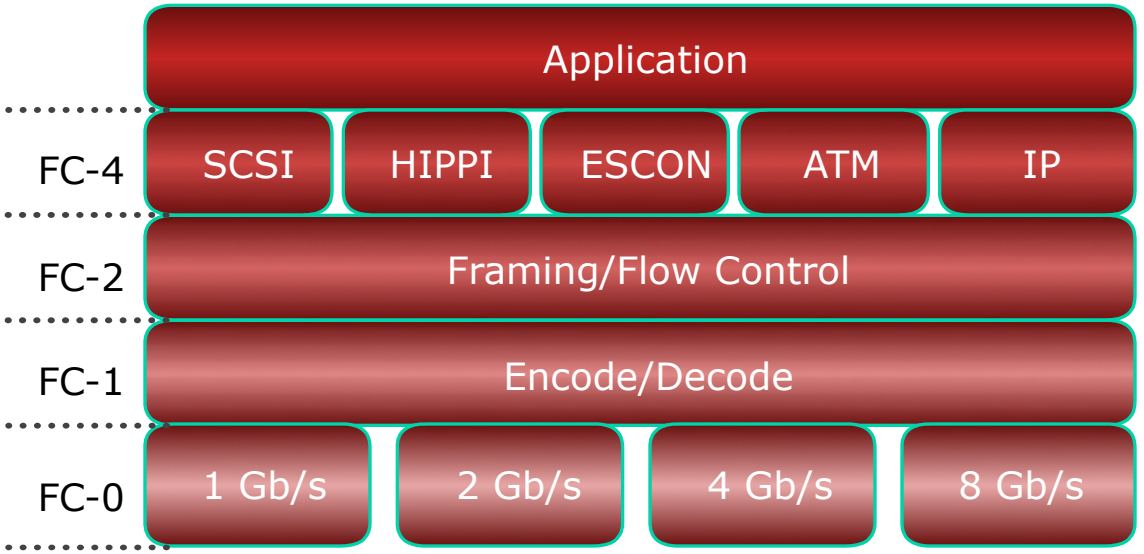
Extended Link Services that are defined in the standards:

- FLOGI - Fabric login
  - Between N_Port to F_Port
- PLOGI - Port login
  - Between N_Port to N_Port
  - N_Port establishes a session with another N_Port
- PRLI - Process login
  - Between N_Port to N_Port
  - To share information about the upper layer protocol type in use
  - And recognizing device as the SCSI initiator, or target

FC Architecture Overview

- FC uses channel technology
- Provide high performance with low protocol overheads
- FCP is SCSI-3 over FC network
  - Sustained transmission bandwidth over long distances
  - Provides speeds up to 8 Gb/s (8 GFC)
- FCP has five layers:
  - FC-4
  - FC-2
  - FC-1
  - FC-0

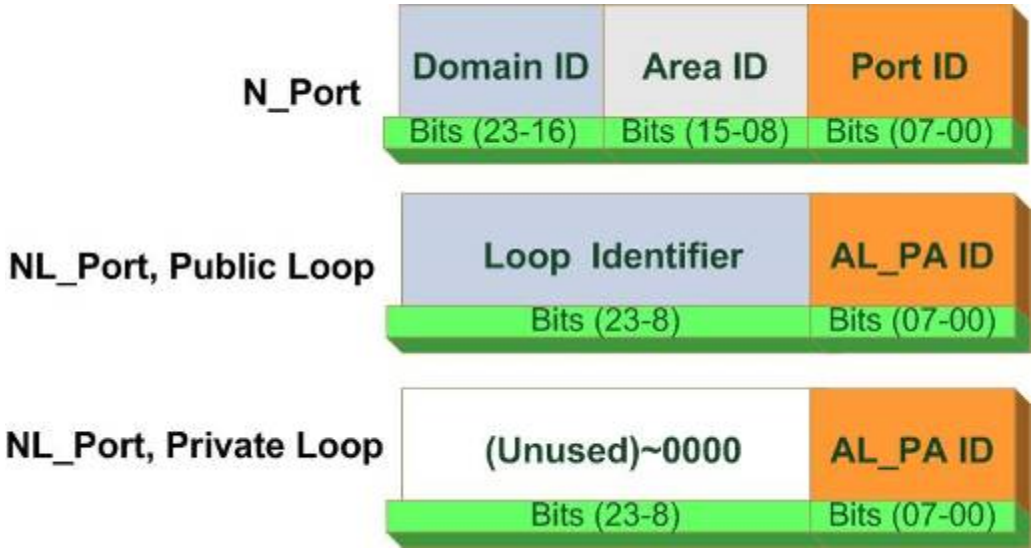
*FC-3 is not yet implemented



Fibre Channel Protocol Stack

FC layer	Function	SAN relevant features specified by FC layer
FC-4	Mapping interface	Mapping upper layer protocol (e.g. SCSI-3 to FC transport
FC-3	Common services	Not implemented
FC-2	Routing, flow control	Frame structure, ports, FC addressing, buffer credits
FC-1	Encode/decode	8b/10b encoding, bit and frame synchronization
FC-0	Physical layer	Media, cables, connector

Fibre Channel Addressing



- FC Address is assigned during Fabric Login
  - Used to communicate between nodes within SAN
  - Similar in functionality to an IP address on NICs
- Address Format:
  - 24 bit address, dynamically assigned
  - Contents of the three bytes depend on the type of N-Port
  - For an N_Port or a public NL_Port:
    - switch maintains mapping of WWN to FC-Address via the Name Server

World Wide Names

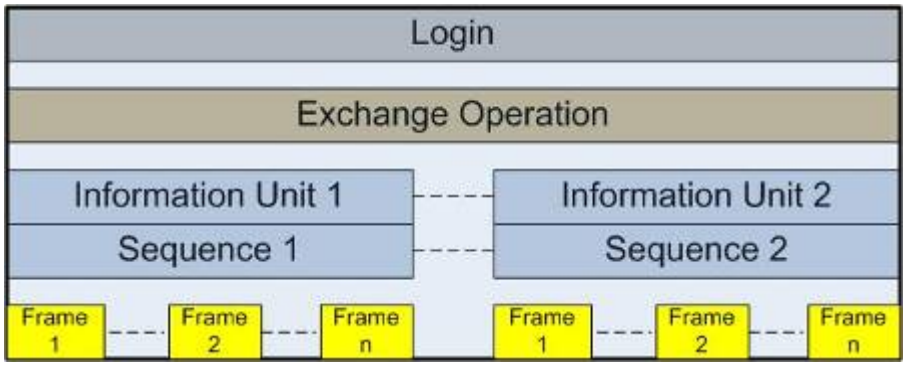
- Unique 64 bit identifier
- Static to the port
  - Used to physically identify ports or nodes within SAN
  - Similar to NIC's MAC address

World Wide Name - Array															
5	0	0	6	0	1	6	0	0	0	6	0	0	1	B	2
0101	0000	0000	0110	0000	0001	0110	0000	0000	0000	0110	0000	0000	0001	1011	0010
	Company ID 24 bits						Port	Model Seed 32 bits							

World Wide Name - HBA															
1	0	0	0	0	0	0	0	c	9	2	0	d	c	4	0
	Reserved 12 bits			Company ID 24 bits						Company Specific 24 bits					

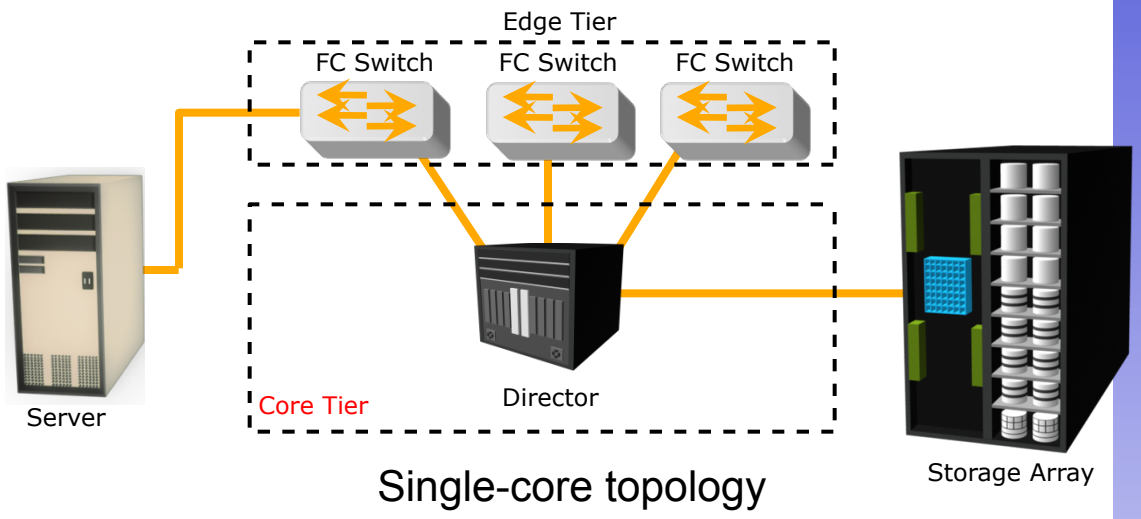
Structure and Organization of FC Data

- FC data is organized as:
  - Exchange operations
    - Enables two N_ports to identify and manage a set of information units
    - Maps to sequence
  - Sequence
    - Contiguous set of frames sent from one port to another
  - Frames
    - Fundamental unit of data transfer
    - Each frame can contain up to 2112 bytes of payload

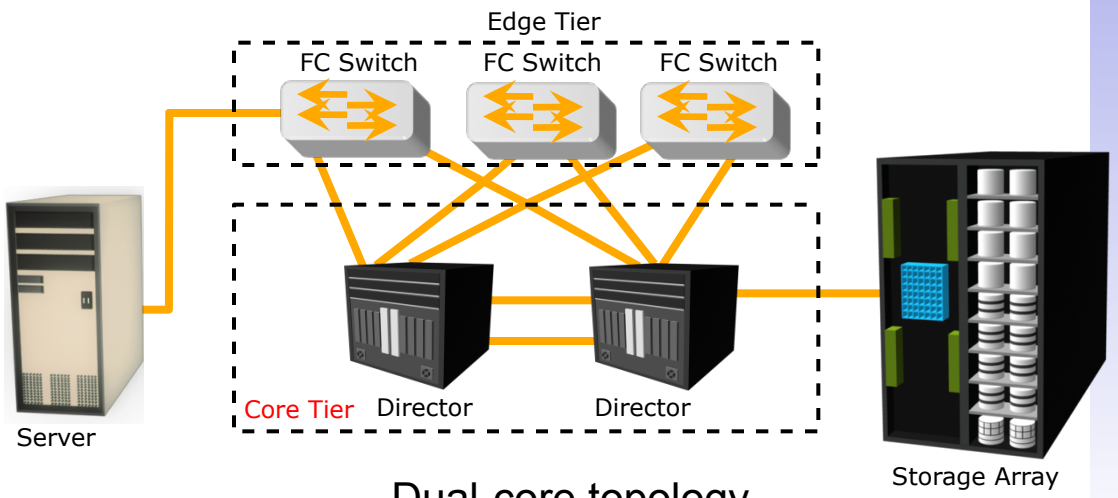


Fabric Topology: Core-Edge Fabric

- Can be two or three tiers
  - Single Core Tier
  - One or two Edge Tiers
- In a two tier topology, storage is usually connected to the Core
- Benefits
  - High Availability
  - Medium Scalability
  - Medium to maximum Connectivity



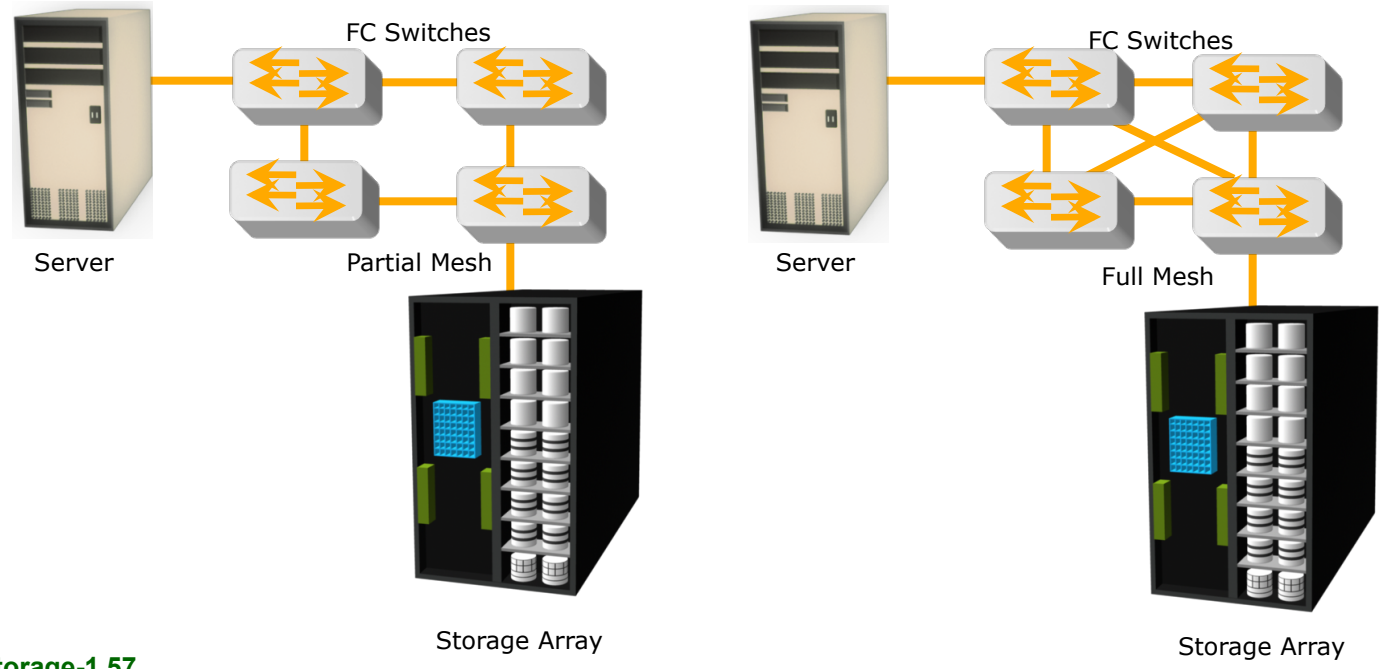
Single-core topology



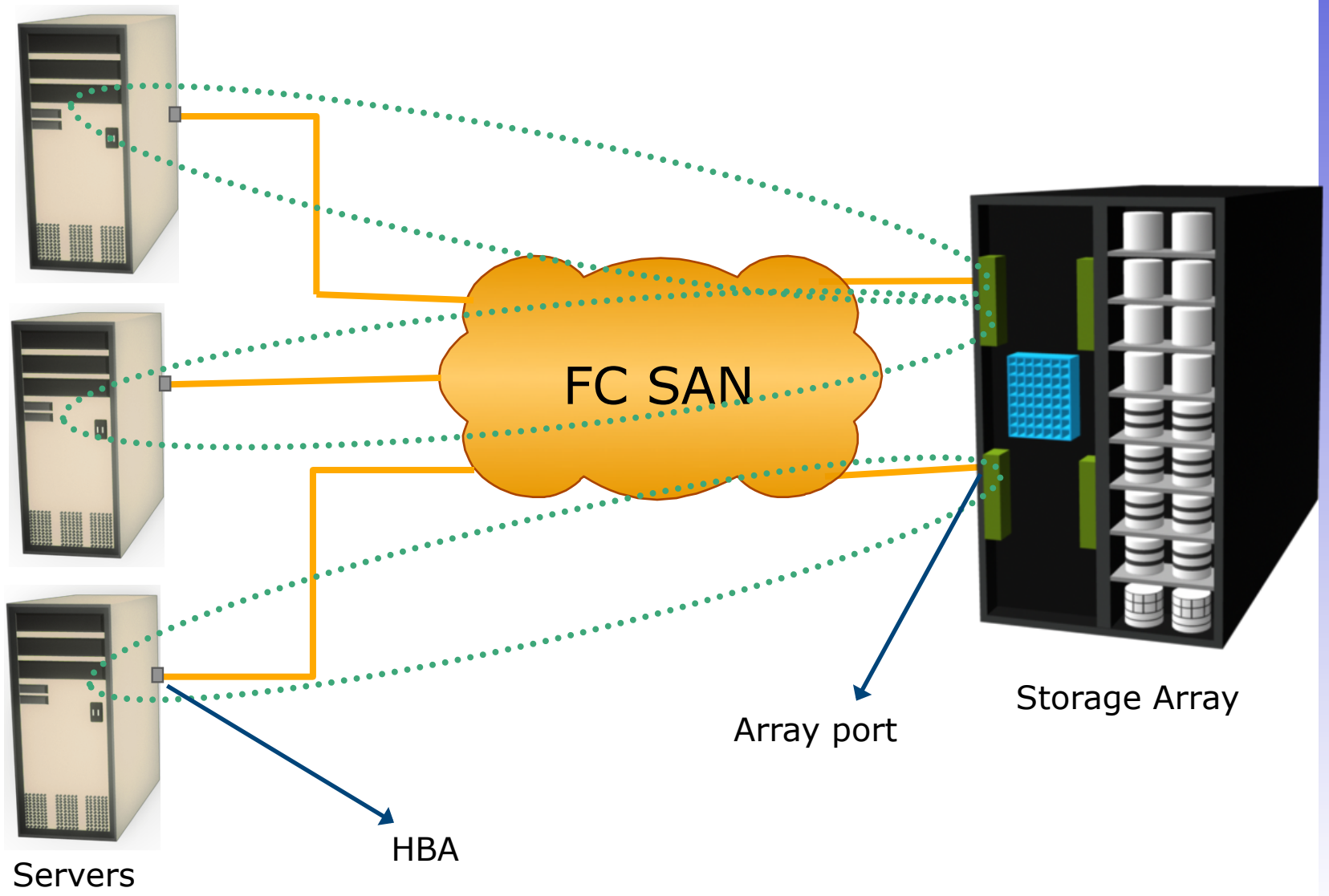
Dual-core topology

Fabric Topology: Mesh

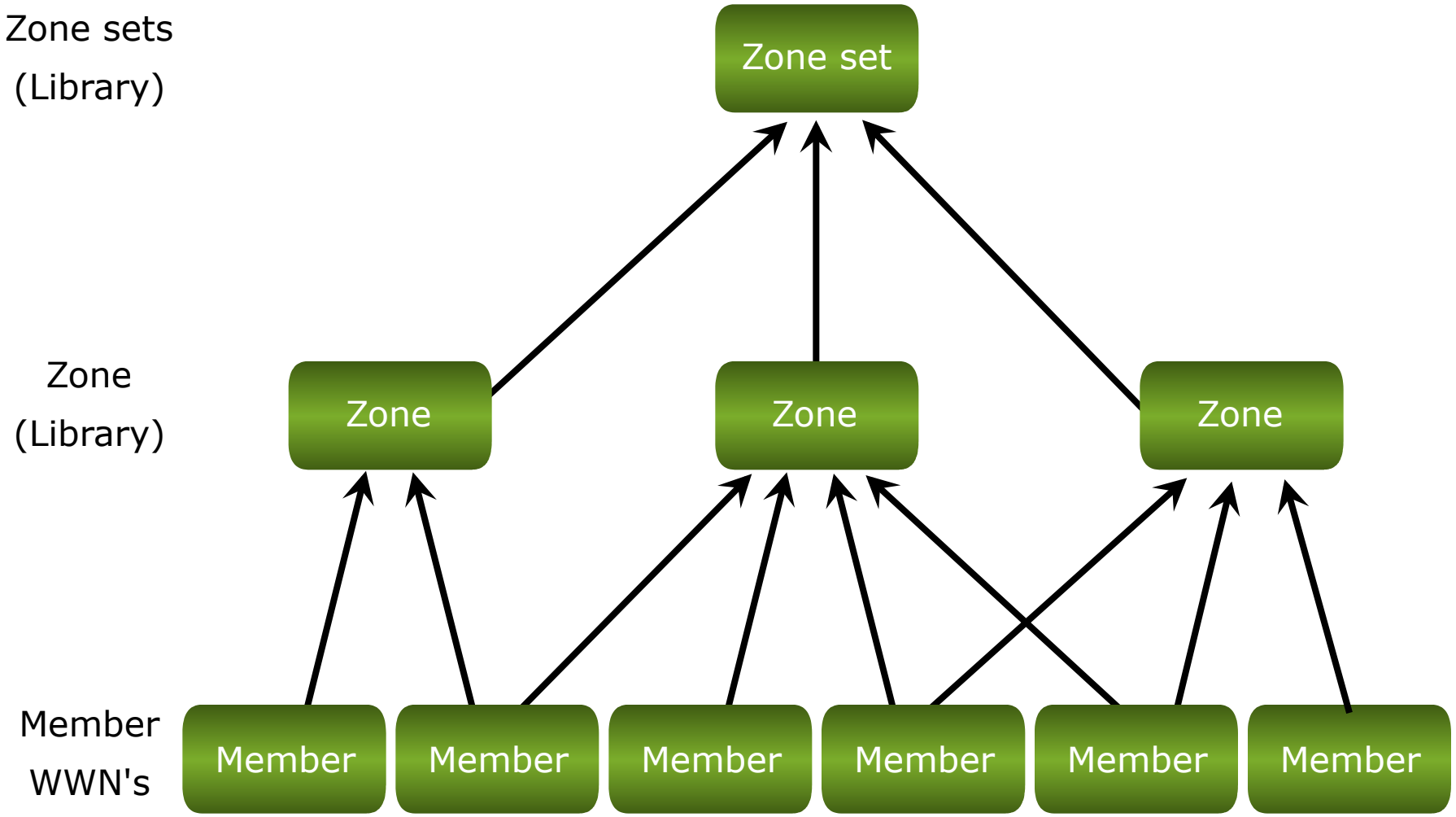
- Can be either partial or full mesh
- All switches are connected to each other
- Host and Storage can be located anywhere in the fabric
- Host and Storage can be localized to a single switch



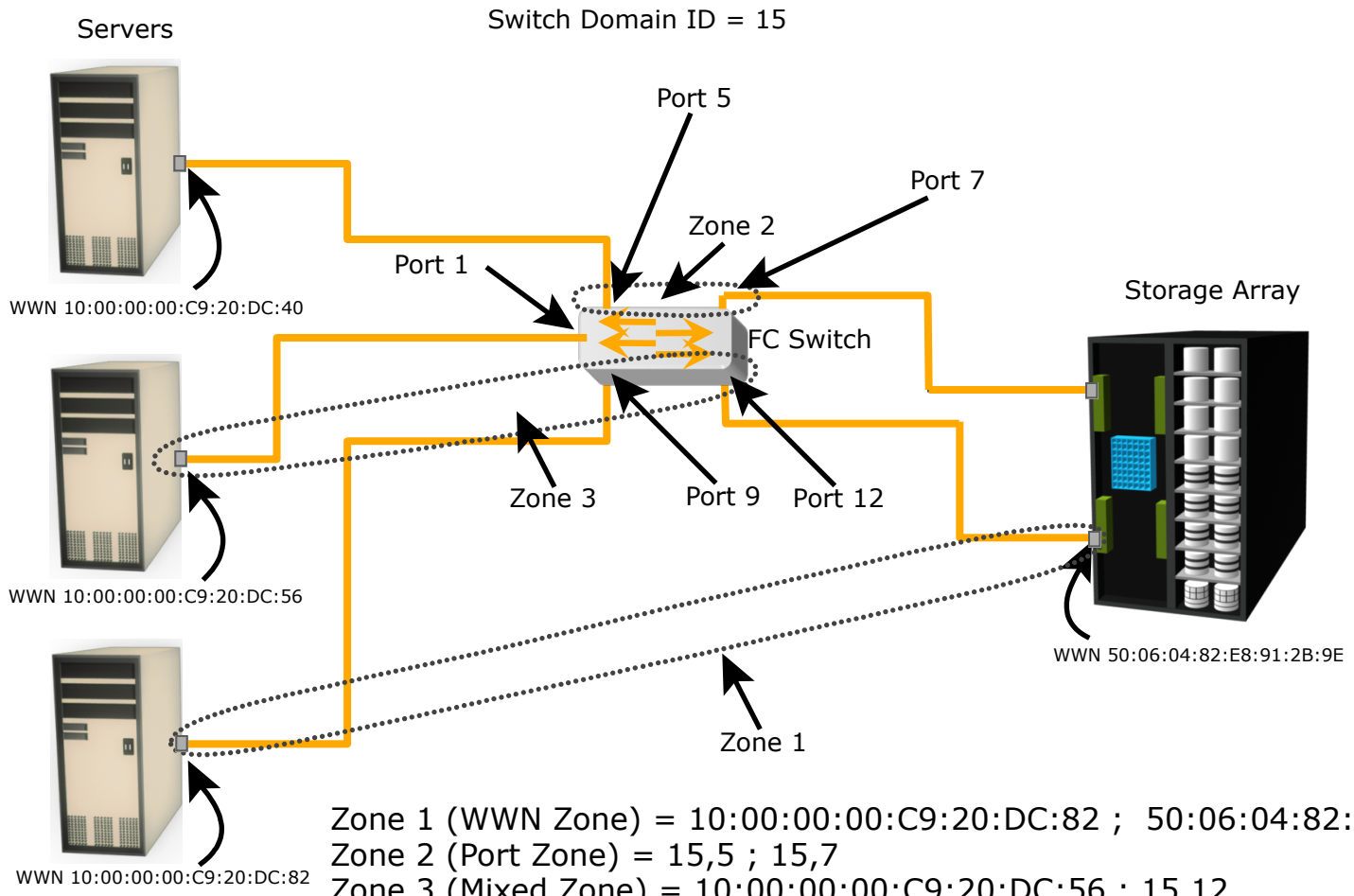
Fabric Management: Zoning



Zoning Components



Types of Zoning



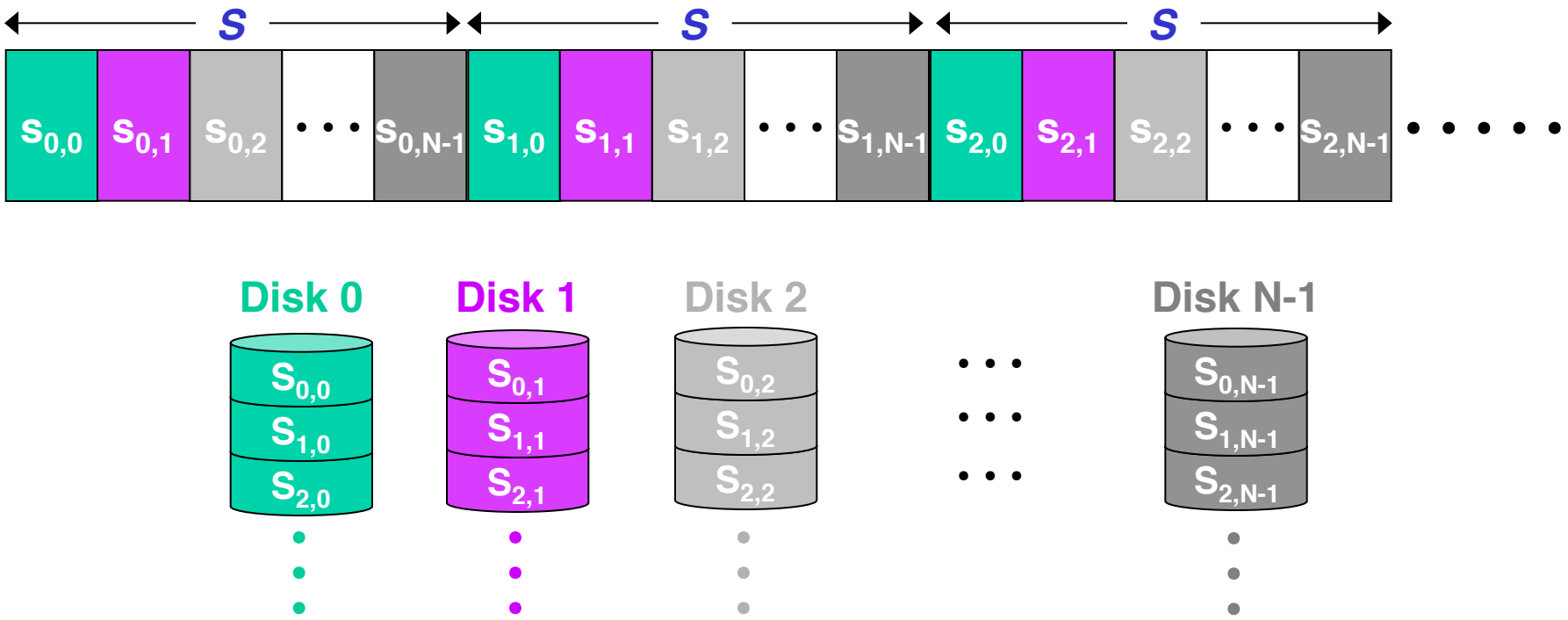
Zone 1 (WWN Zone) = 10:00:00:00:C9:20:DC:82 ; 50:06:04:82:E8:91:2B:9E
Zone 2 (Port Zone) = 15,5 ; 15,7
Zone 3 (Mixed Zone) = 10:00:00:00:C9:20:DC:56 ; 15,12

Motivation for Disk Arrays

- Typical memory bandwidths ≈ 150 MB/sec
- Typical disk bandwidths ≈ 10 MB/sec
- Result: *I/O-bound applications limited by disk bandwidth*
  - (not just by disk latency!)
- Two common disk bandwidth-limited scenarios
  - Scientific applications: one huge I/O-hungry app
  - Servers: many concurrent requests with modest I/O demands

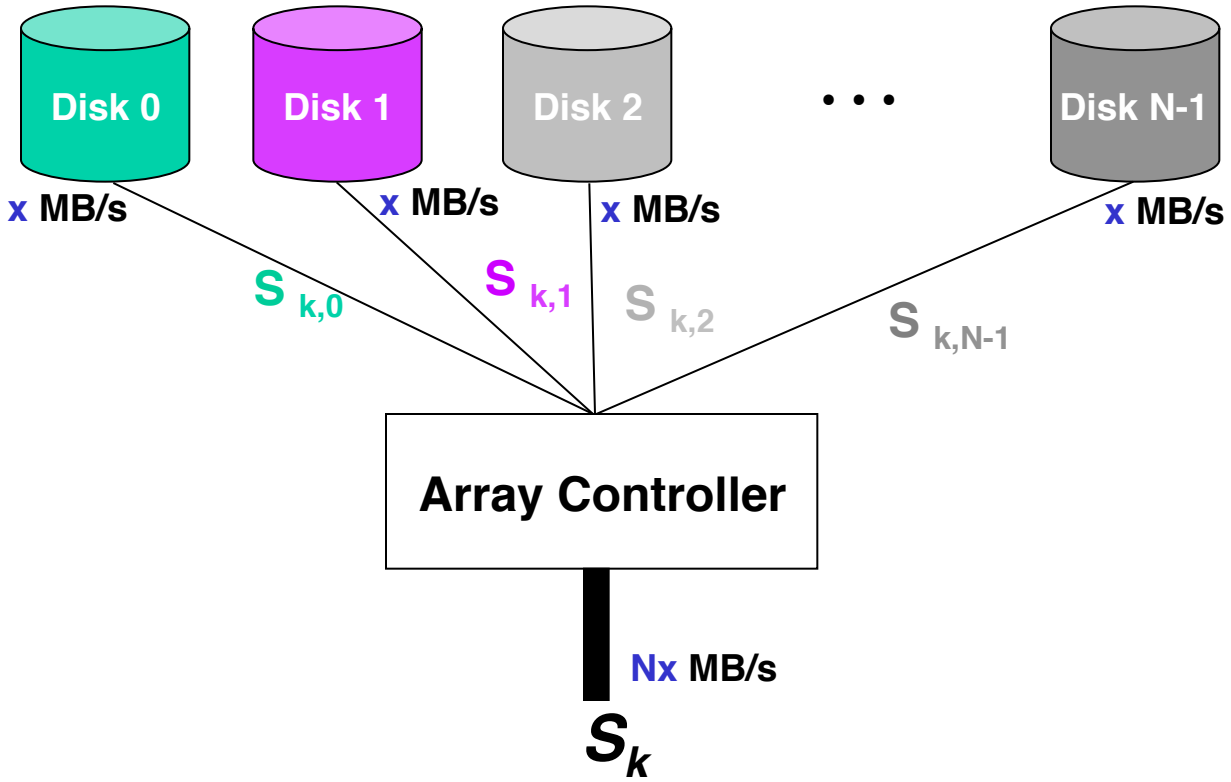
Solution: Exploit Parallelism

- *Stripe* the data across an array of disks
  - many alternative striping strategies possible
- Example: consider a big file striped across N disks
  - *stripe width* is S bytes
  - hence each *stripe unit* is S/N bytes
  - sequential read of S bytes at a time

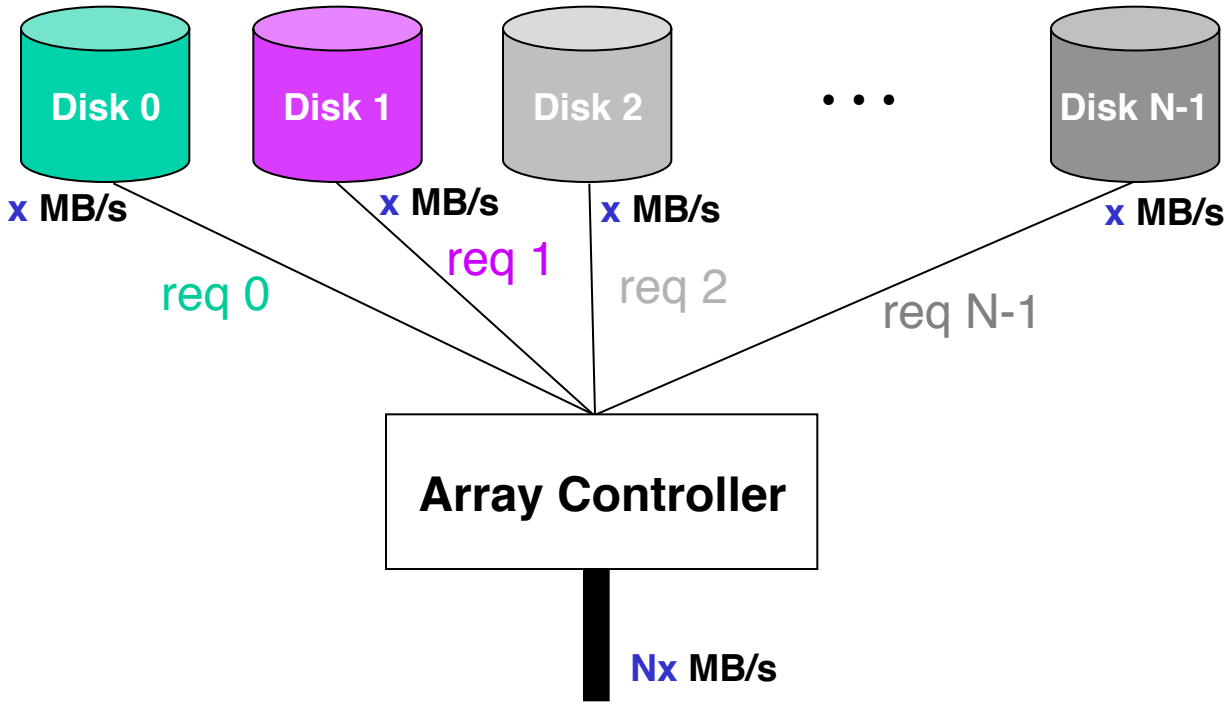


Performance Benefit

- *Sequential read or write of large file*
  - application (or I/O buffer cache) reads in multiples of S bytes
  - controller performs parallel access of N disks
  - aggregate bandwidth is N times individual disk bandwidth
    - (assumes that disk is the bottleneck)



- *N concurrent small read or write requests*
  - randomly distributed across N drives (we hope!)
  - common in database and Web server environments



N independent requests

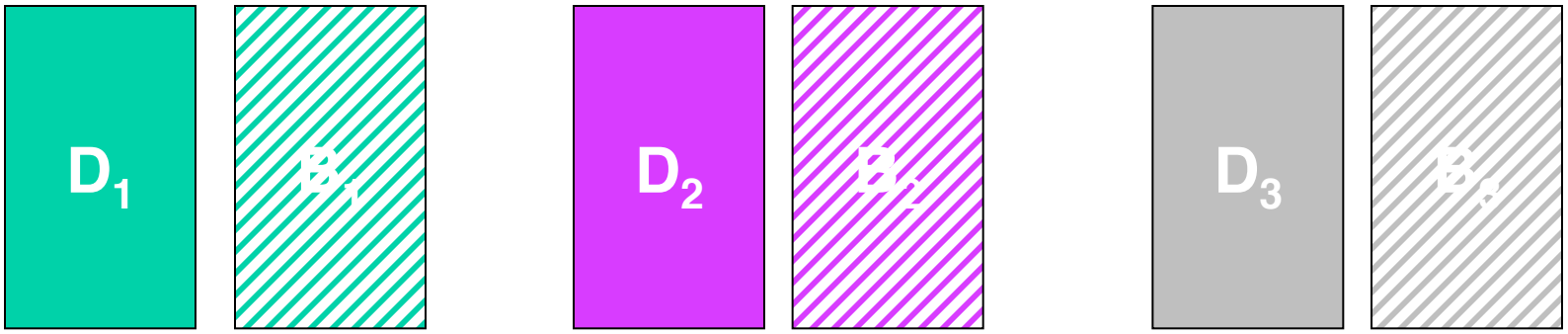
Reliability of Disk Arrays

- As number of disks grows, chances of at least one failing increases
- Reliability of N disks = (reliability of 1 disk) / N
  - suppose each disk has MTTF of 50,000 hours
    - (roughly 6 years before any given disk fails)
  - then some disk in a 70-disk array will fail in (50,000 / 70) hours
    - (roughly once a month!)
- Large arrays without redundancy too unreliable to be useful
  - *"Redundant Arrays of Independent Disks" (RAID)*

RAID Approaches

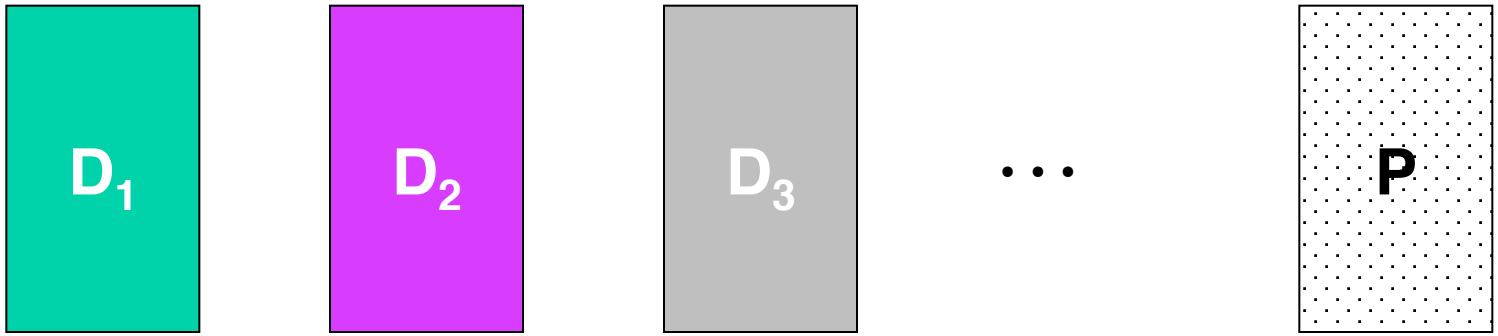
- Many alternative approaches to achieving this redundancy
  - *RAID levels 1 through 5*
  - *hot sparing* allows reconstruction concurrently with accesses
- Key metrics to evaluate alternatives
  - wasted space due to redundancy
  - likelihood of “hot spots” during heavy loads
  - degradation of performance during repair

RAID Level 1



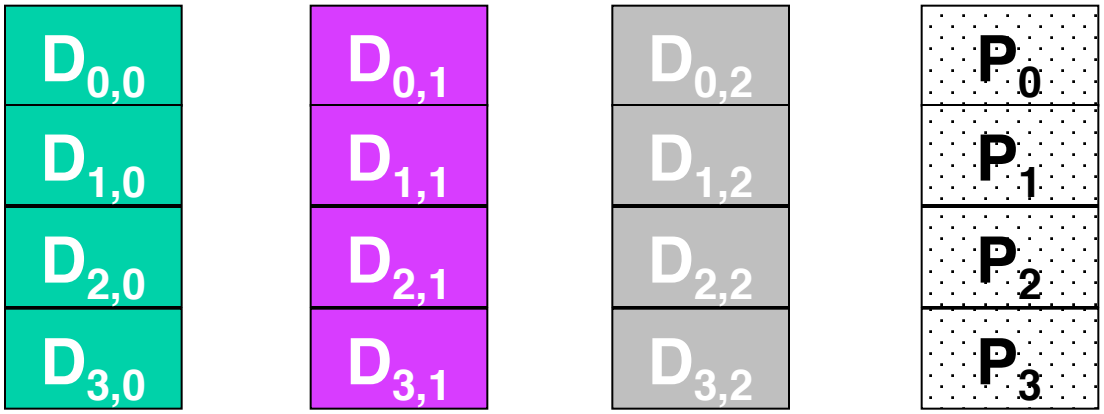
- Also known as *"mirroring"*
- To read a block:
  - read from either data disk or backup
- To write a block:
  - write both data and backup disks
  - failure model determines whether writes can occur in parallel
- Backups can be located far way: safeguard against site failure

RAID Levels 2 & 3



- These are *bit-interleaved* schemes
- In Raid Level 2, P contains memory-style ECC
- In Rail Level 3, P contains simple parity
- Rarely used today

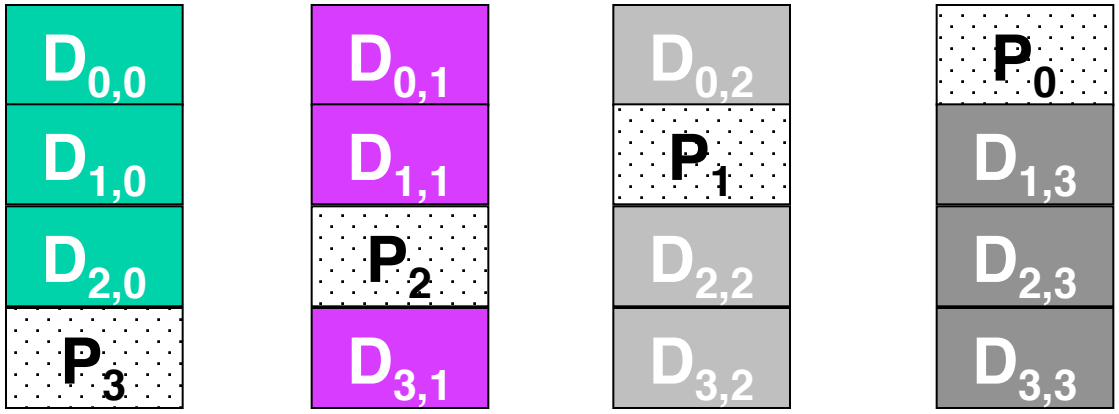
RAID Level 4



$$D_{0,0} \oplus D_{0,1} \oplus D_{0,2} = P_0$$

- *Block-interleaved parity*
- Wasted storage is small: one parity block for N data blocks
- Key problem:
  - parity disk becomes a hot spot
  - write access to parity disk on every write to any block

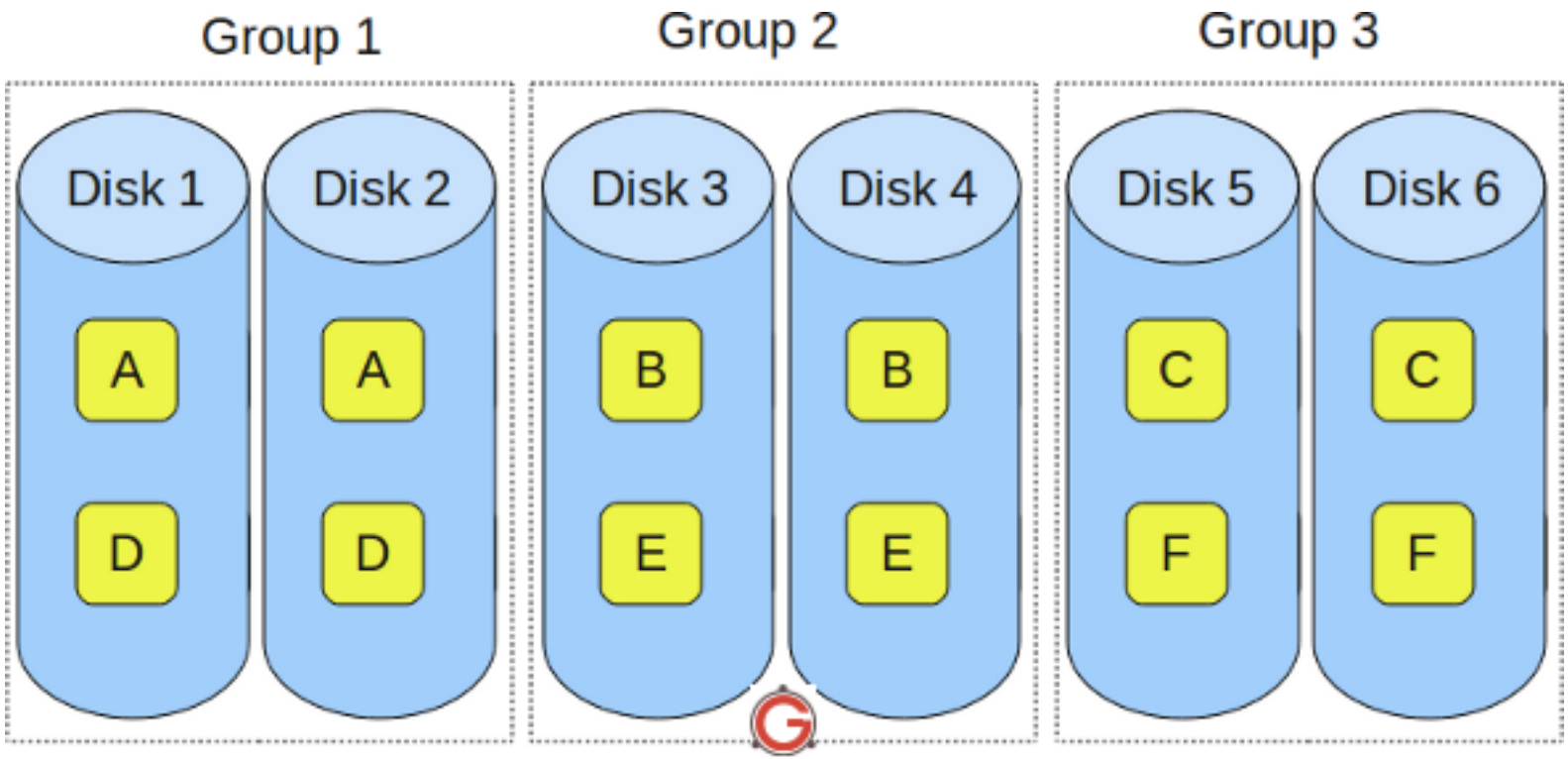
RAID Level 5



$$D_{0,0} \oplus D_{0,1} \oplus D_{0,2} = P_0$$

- *Rotated parity*
- Wastage is small: same as in Raid 4
- Parity update traffic is distributed across disks

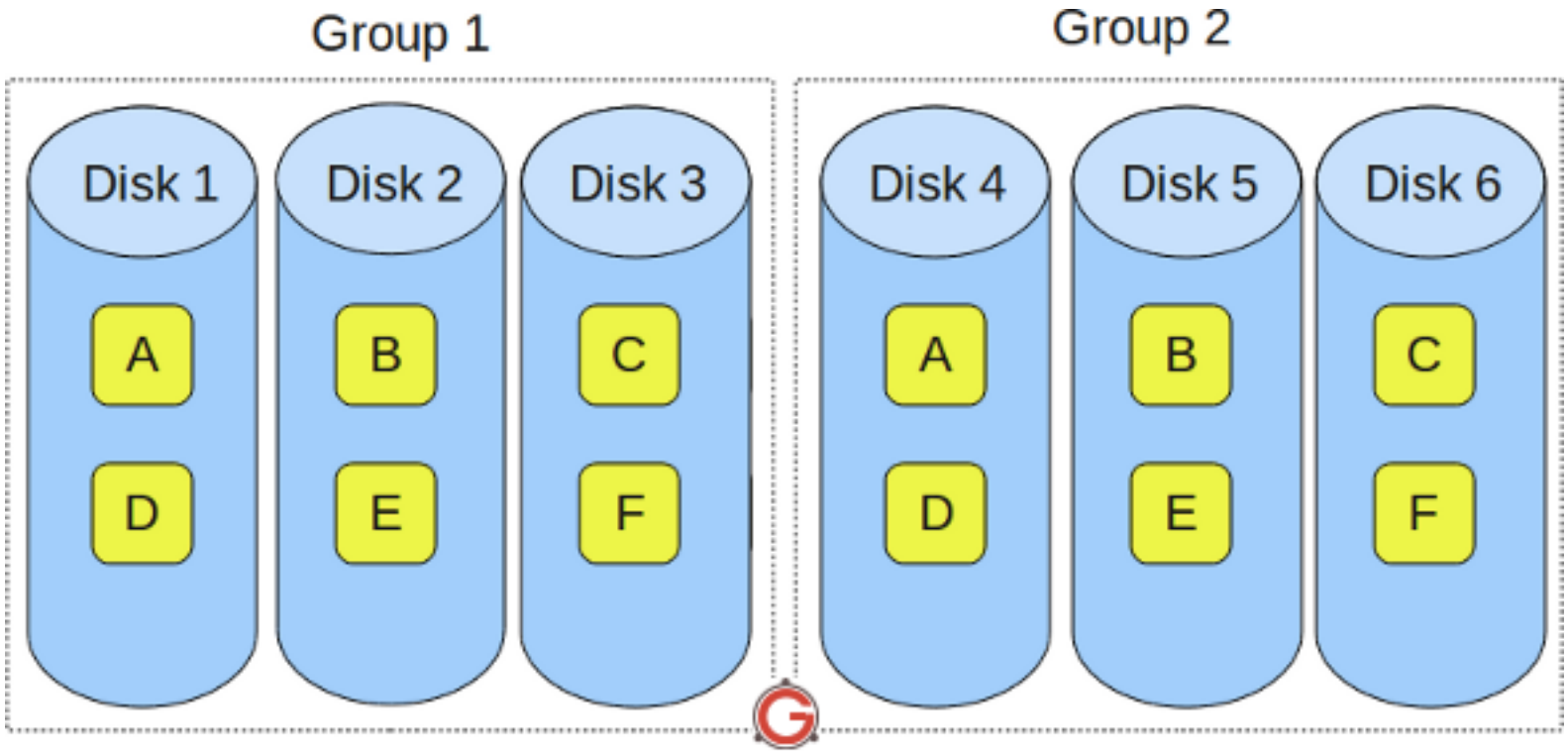
RAID Level 10



RAID 10 – Blocks Mirrored. (and Blocks Striped)

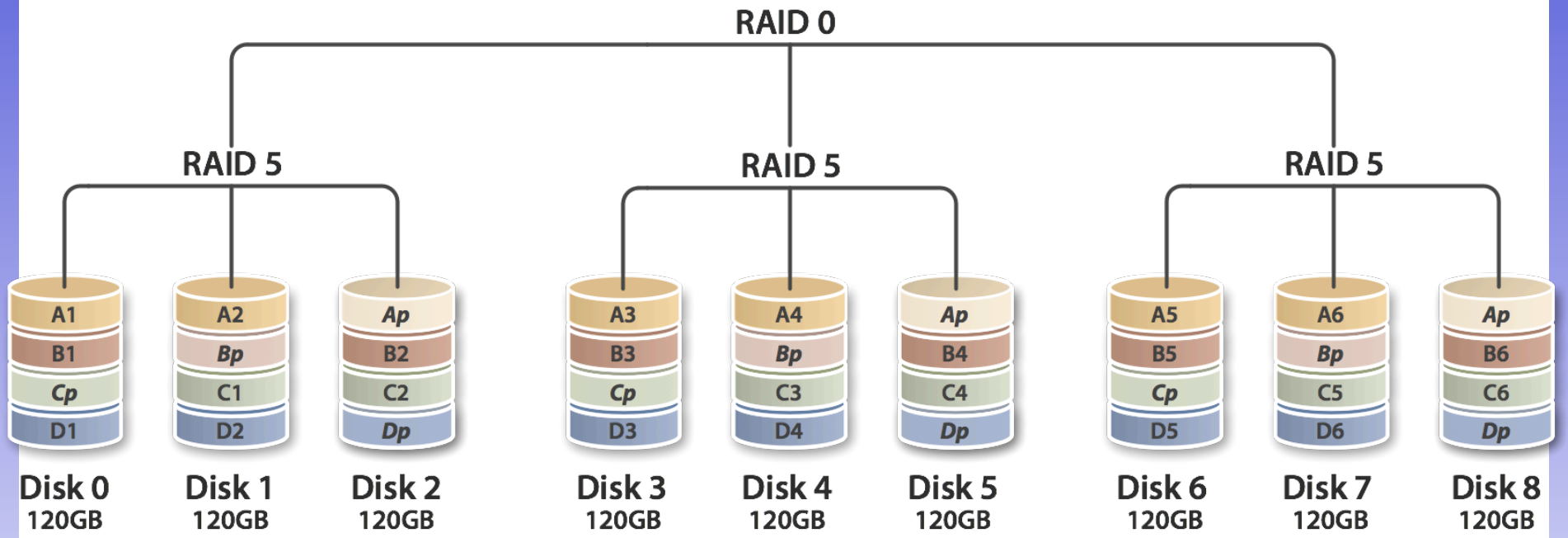
- *Difference between RAID 0+1 and RAID 1+0*

RAID Level 0+1

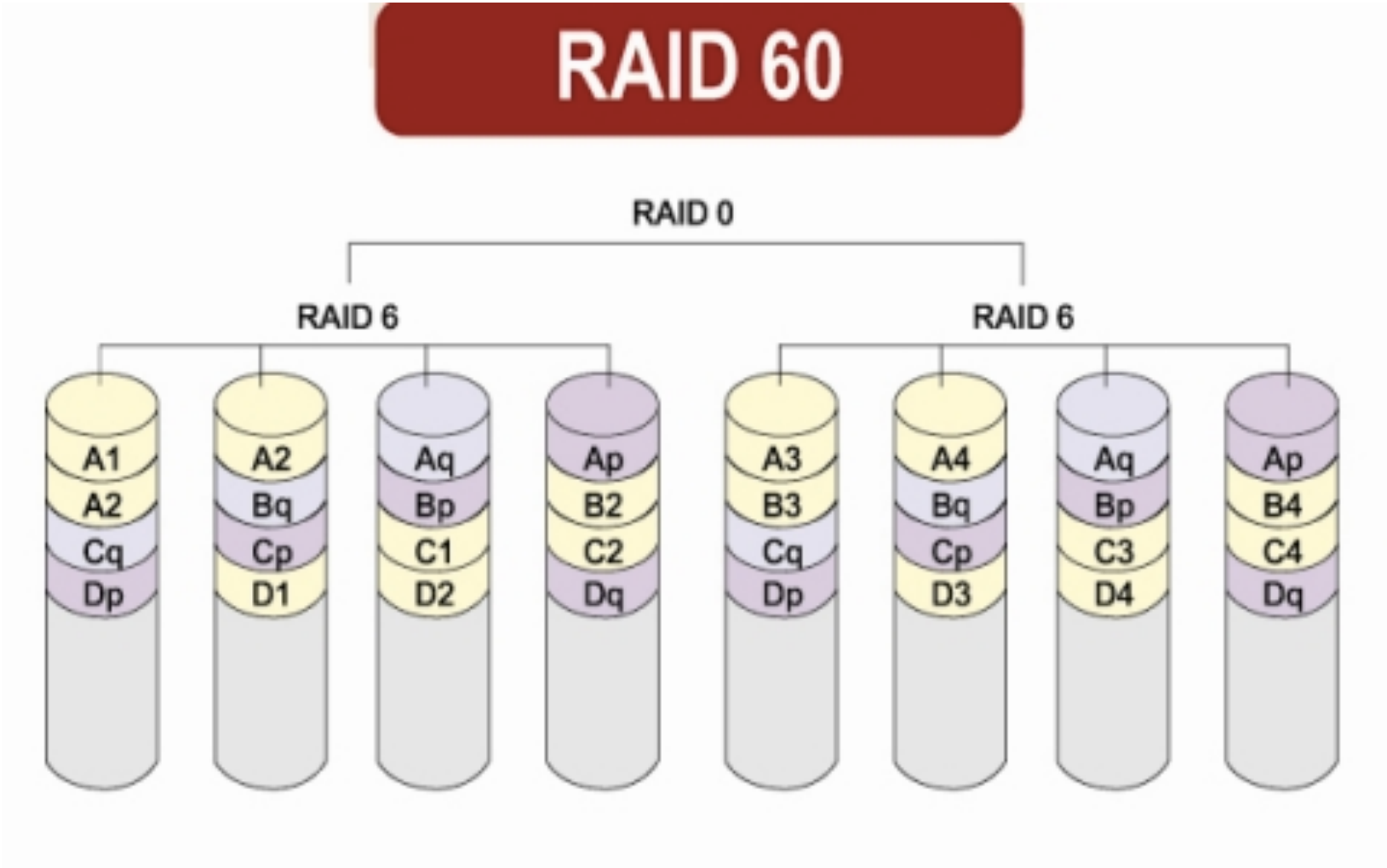


RAID 01 – Blocks Striped. (and Blocks Mirrored)

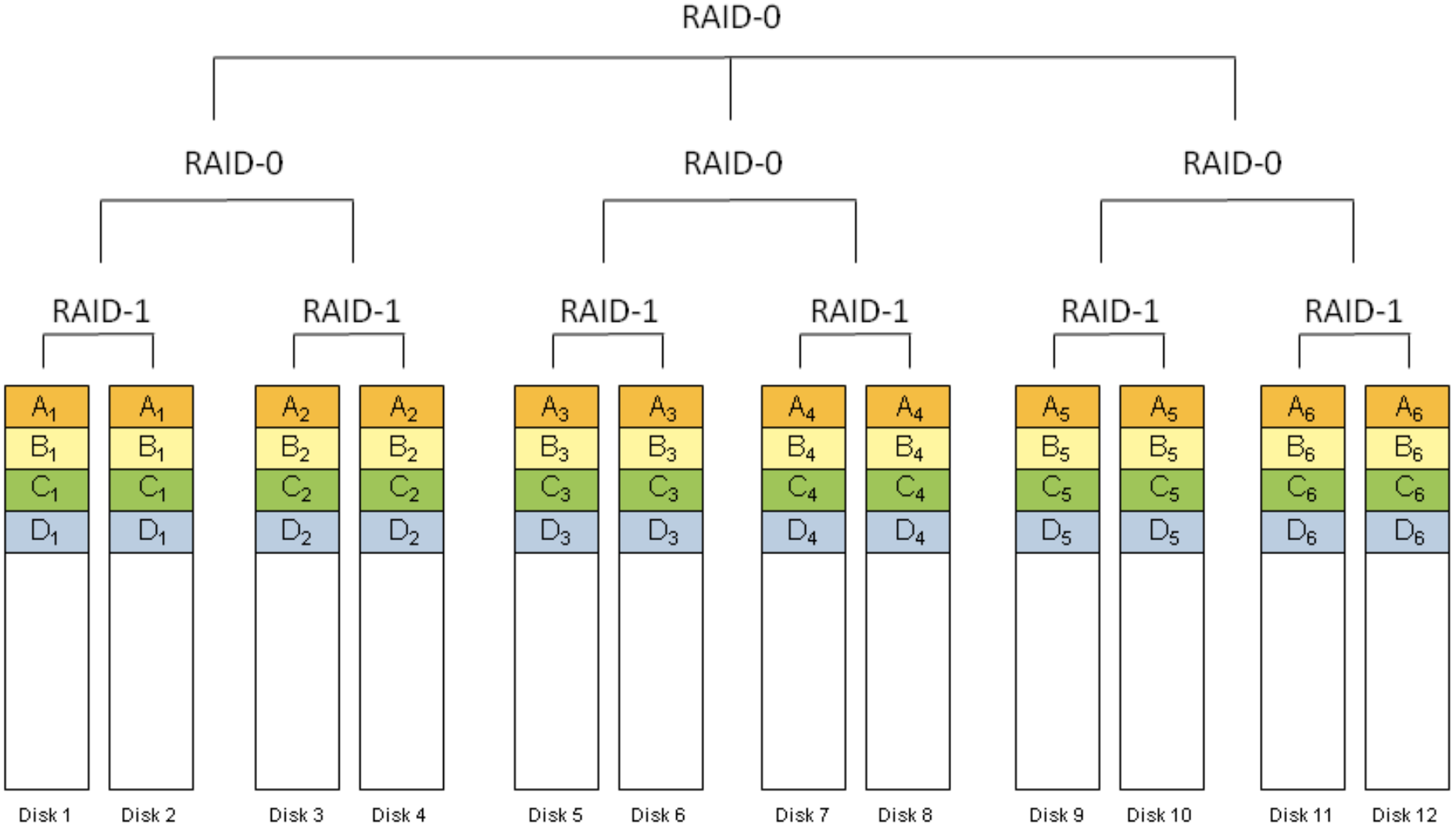
RAID Level 50



RAID Level 60

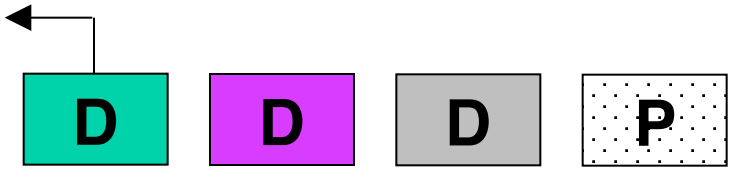


RAID Level 100

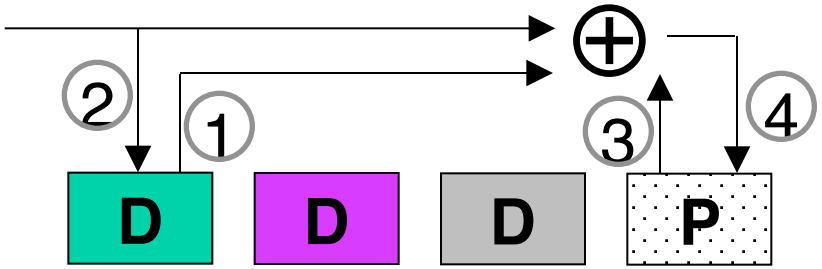


RAID 5 Actions

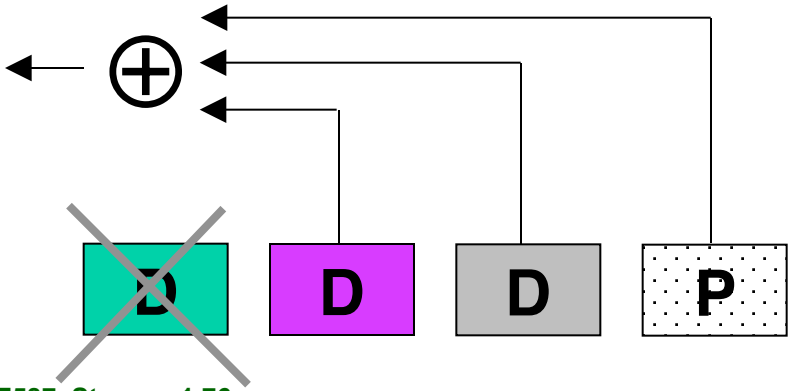
Fault-free Read



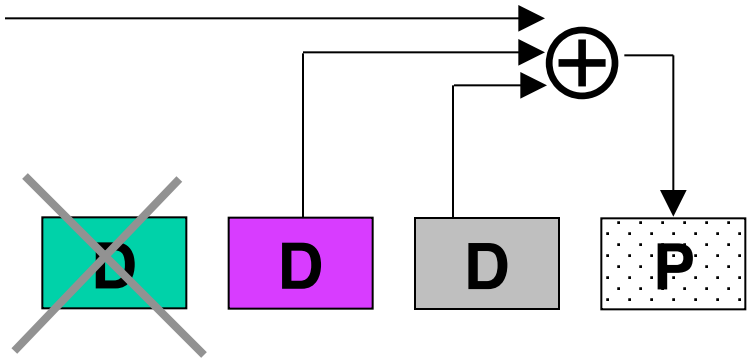
Fault-free Write



Degraded Read

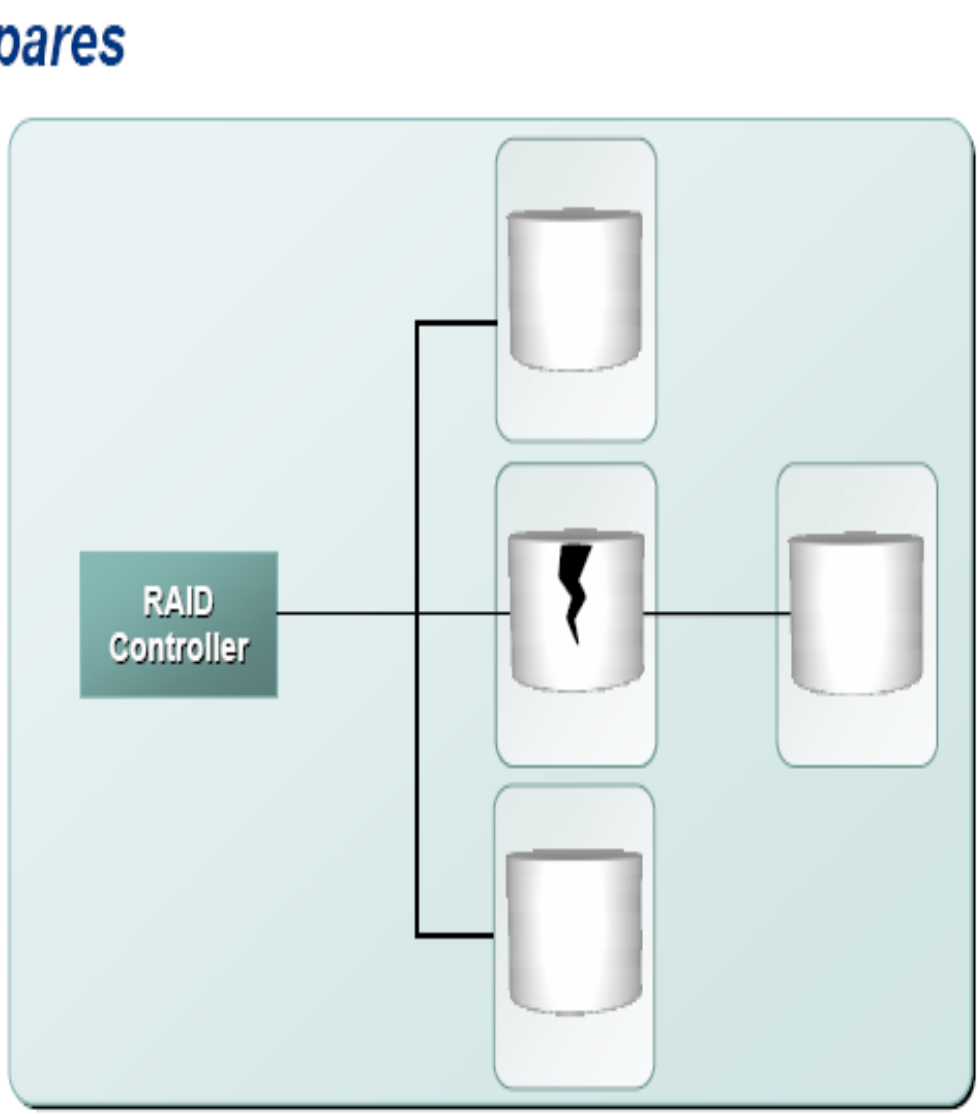


Degraded Write



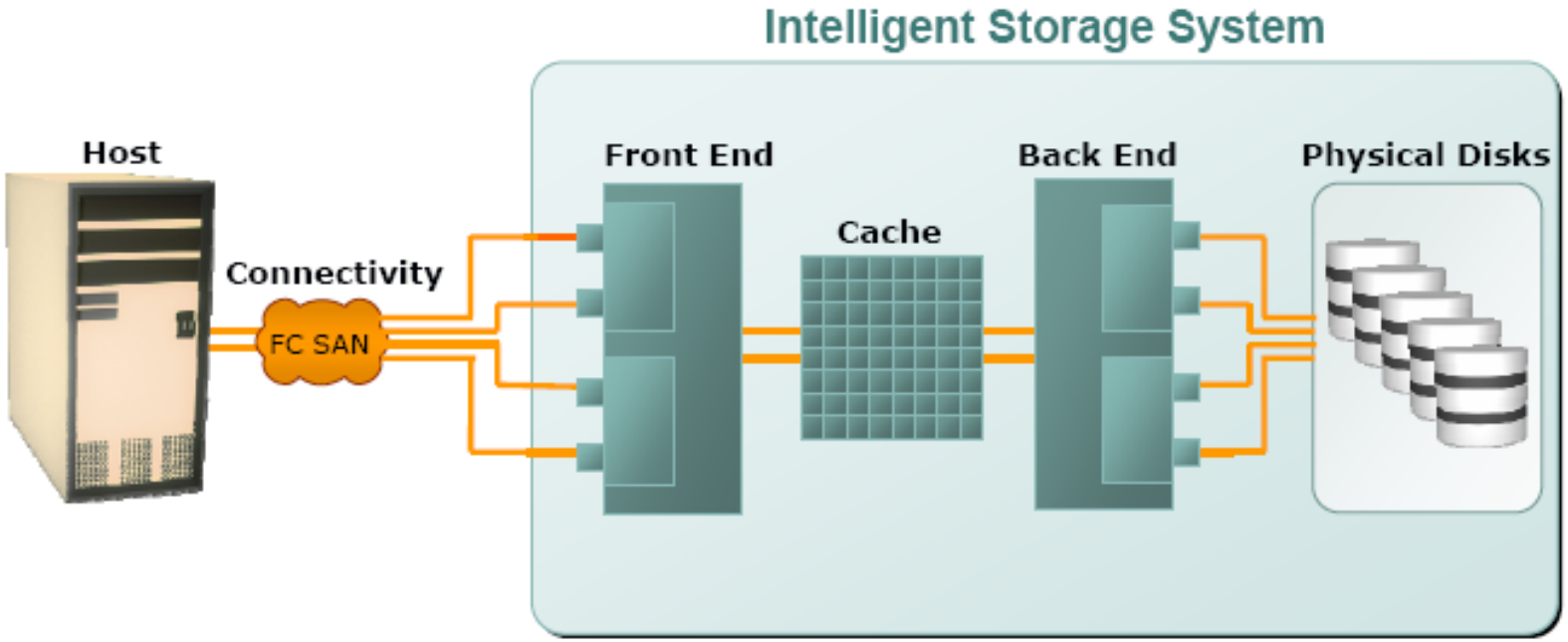
Hot Spare

- A hot spare refers to a spare HDD in a RAID array that temporarily replaces a failed HDD of a RAID set.
  - When the failed HDD is replaced with a new HDD, The hot spare replaces the new HDD permanently, and a new hot spare must be configured on the array, or data from the hot spare is copied to it, and the hot spare returns to its idle state, ready to replace the next failed drive.
  - A hot spare should be large enough to accommodate data from a failed drive.
  - Some systems implement multiple hot spares to improve data availability.
  - A hot spare can be configured as automatic or user initiated, which specifies how it will be used in the event of disk failure



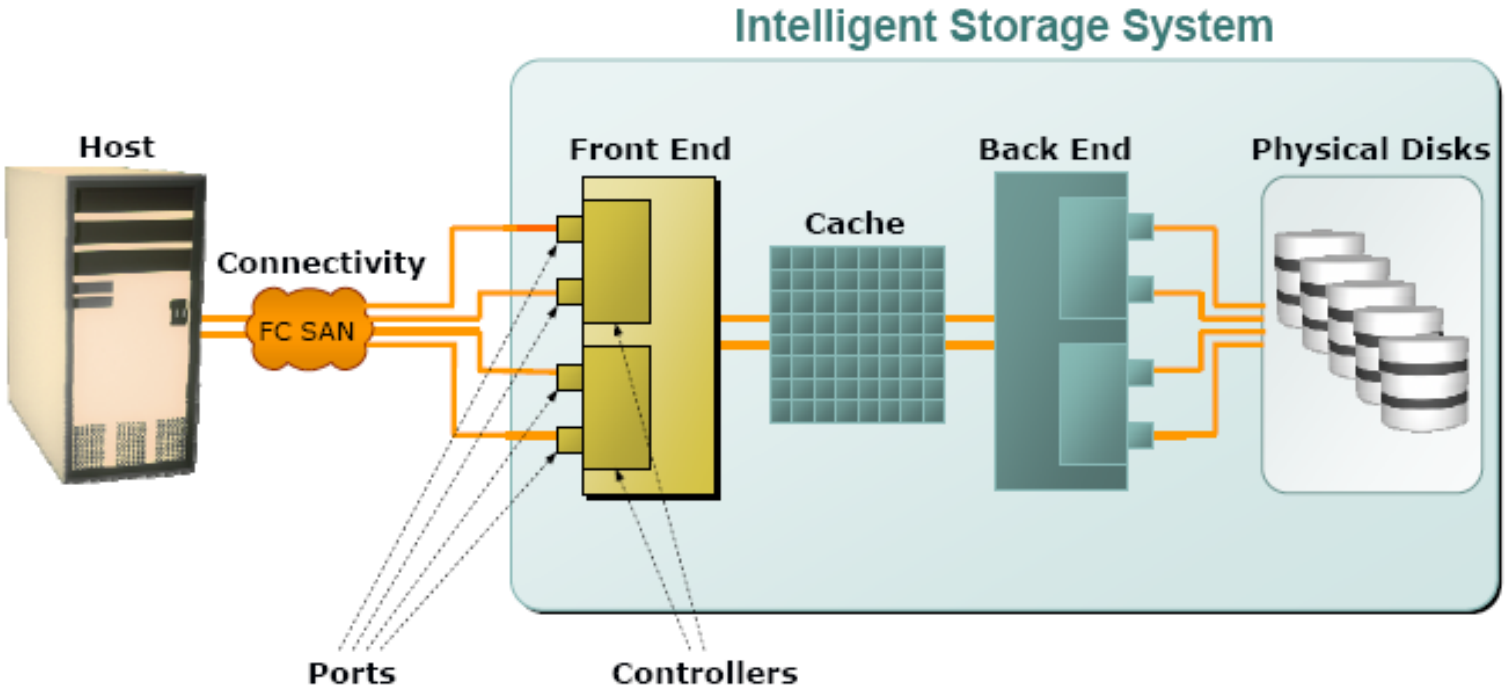
Intelligent Storage System

- An intelligent storage system consists of four key components: *front end, cache, back end,* and *physical disks*.



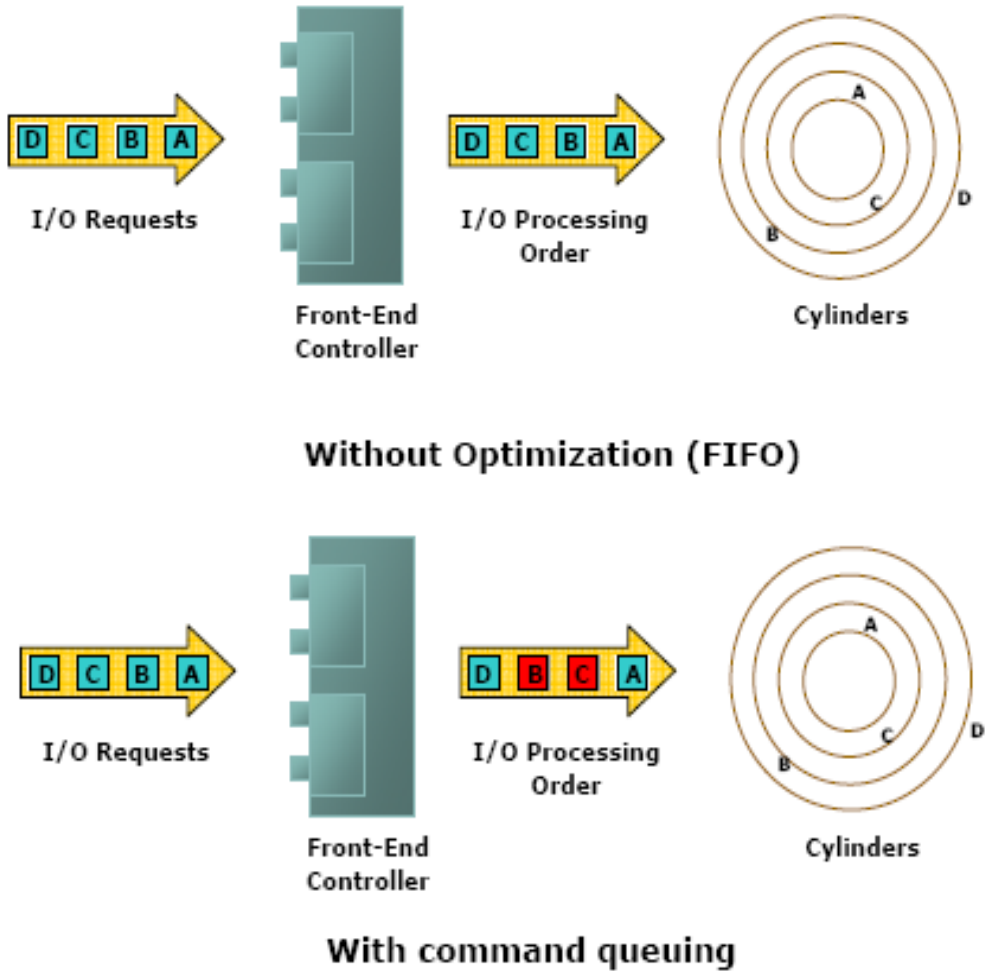
Intelligent Storage System

- The front end provides the interface between the storage system and the host.
  - two components: front-end ports and front-end controllers
- The *front-end ports* enable hosts to connect to the intelligent storage system, and has processing logic that executes the appropriate transport protocol, such as SCSI, Fibre Channel, IB, or iSCSI, for storage connections
- *Front-end controllers* route data to and from cache via the internal data bus. When cache receives write data, the controller sends an acknowledgment



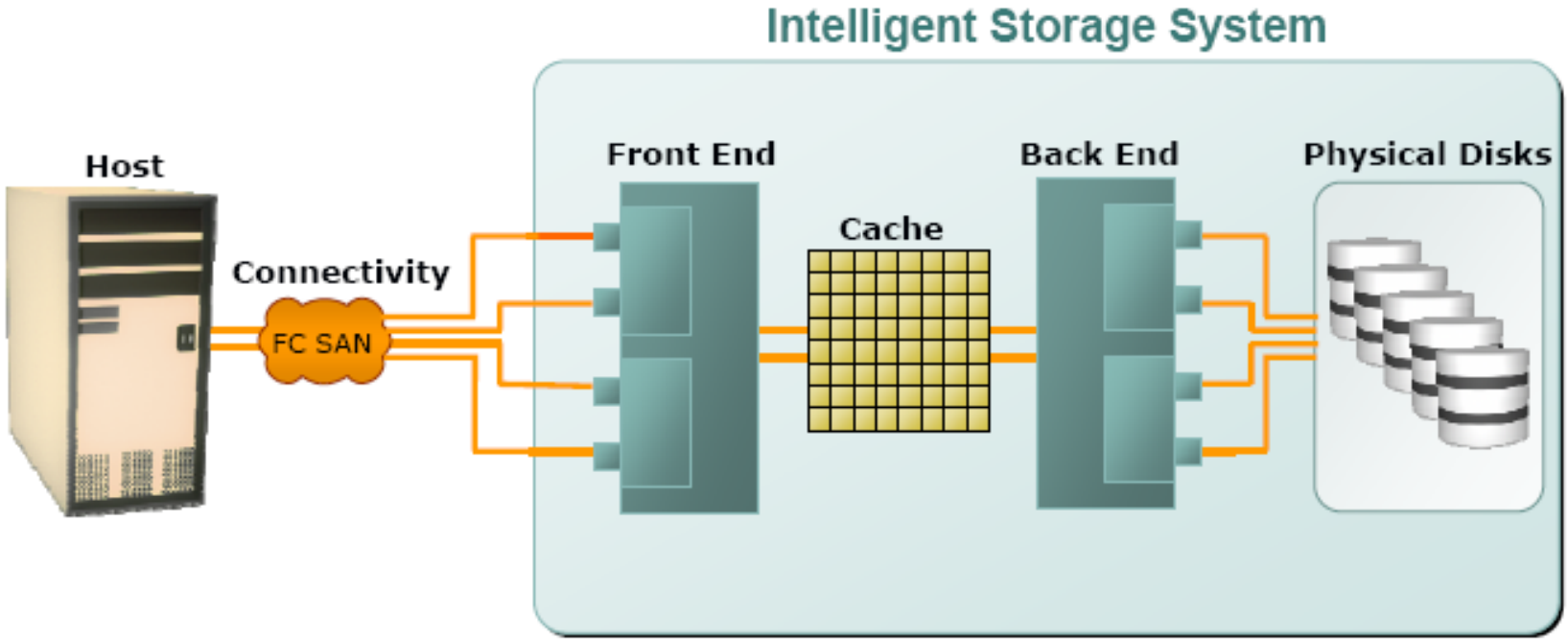
Intelligent Storage System

- Controllers optimize I/O processing by using *command queuing* algorithms
  - technique implemented on front-end controllers
  - determines the execution order of received commands and can reduce unnecessary drive head movements and improve disk performance



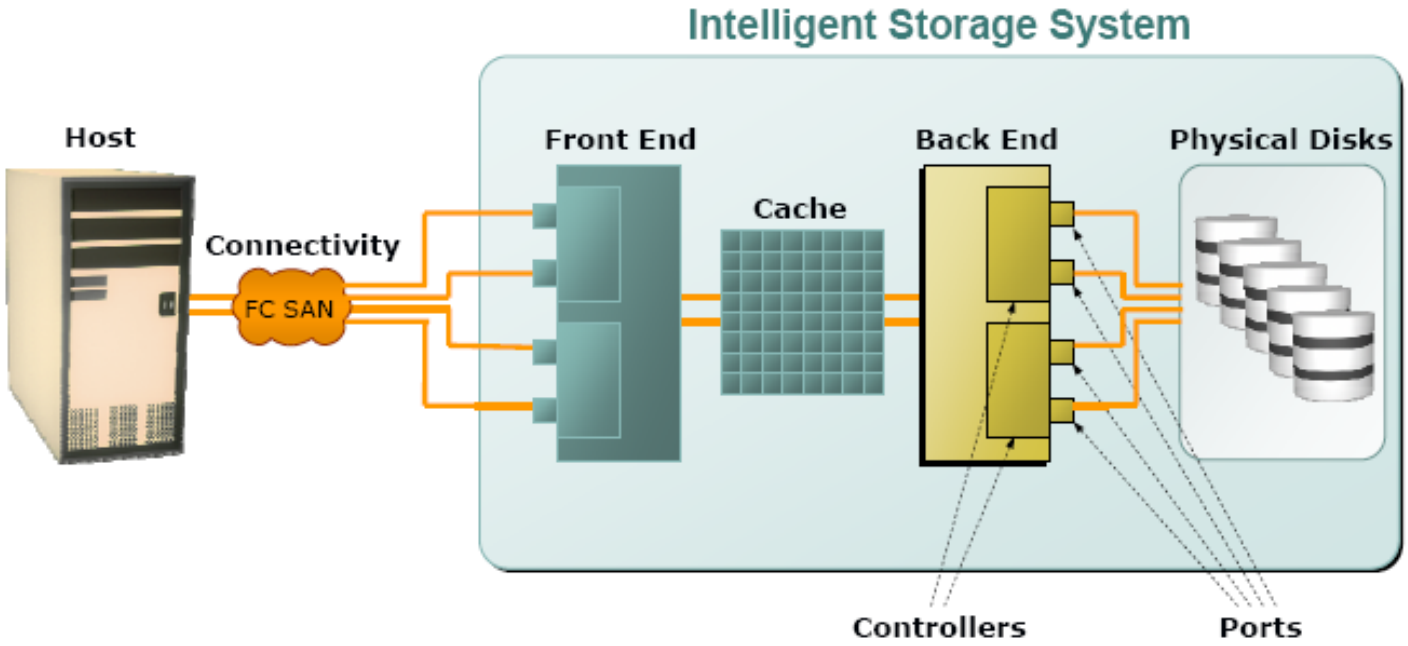
Intelligent Storage System: Cache

- Cache enhances the I/O performance in an intelligent storage system.
  - Accessing data from disk usually takes a few ms from cache takes less than a ms.
  - Write data is placed in cache and then written to disk
- *Cache mirroring*: Each write to cache held in two different memory locations on two independent memory cards
- *Cache vaulting*: Cache exposed to risk of uncommitted data loss due to power failure
  - using battery power to write the cache content to the disk storage vendors use a set of physical disks to dump the contents of cache during power failure



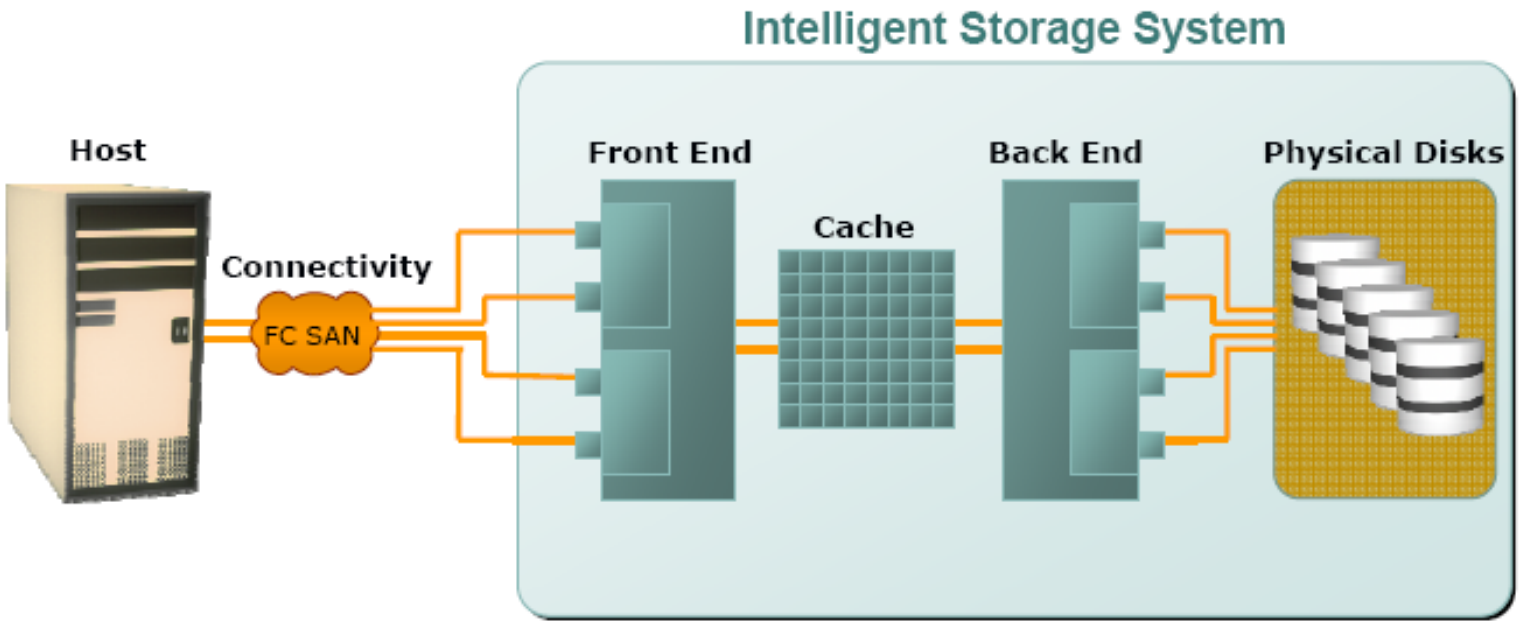
Intelligent Storage System: Back End

- Two components: back-end ports and back-end controllers
  - Physical disks are connected to ports on the back end.
- The back end controller communicates with the disks when performing reads and writes and also provides additional, but limited, temporary data storage.
- The algorithms implemented on back-end controllers provide error detection and correction, along with RAID functionality.
- Multiple controllers also facilitate load balancing



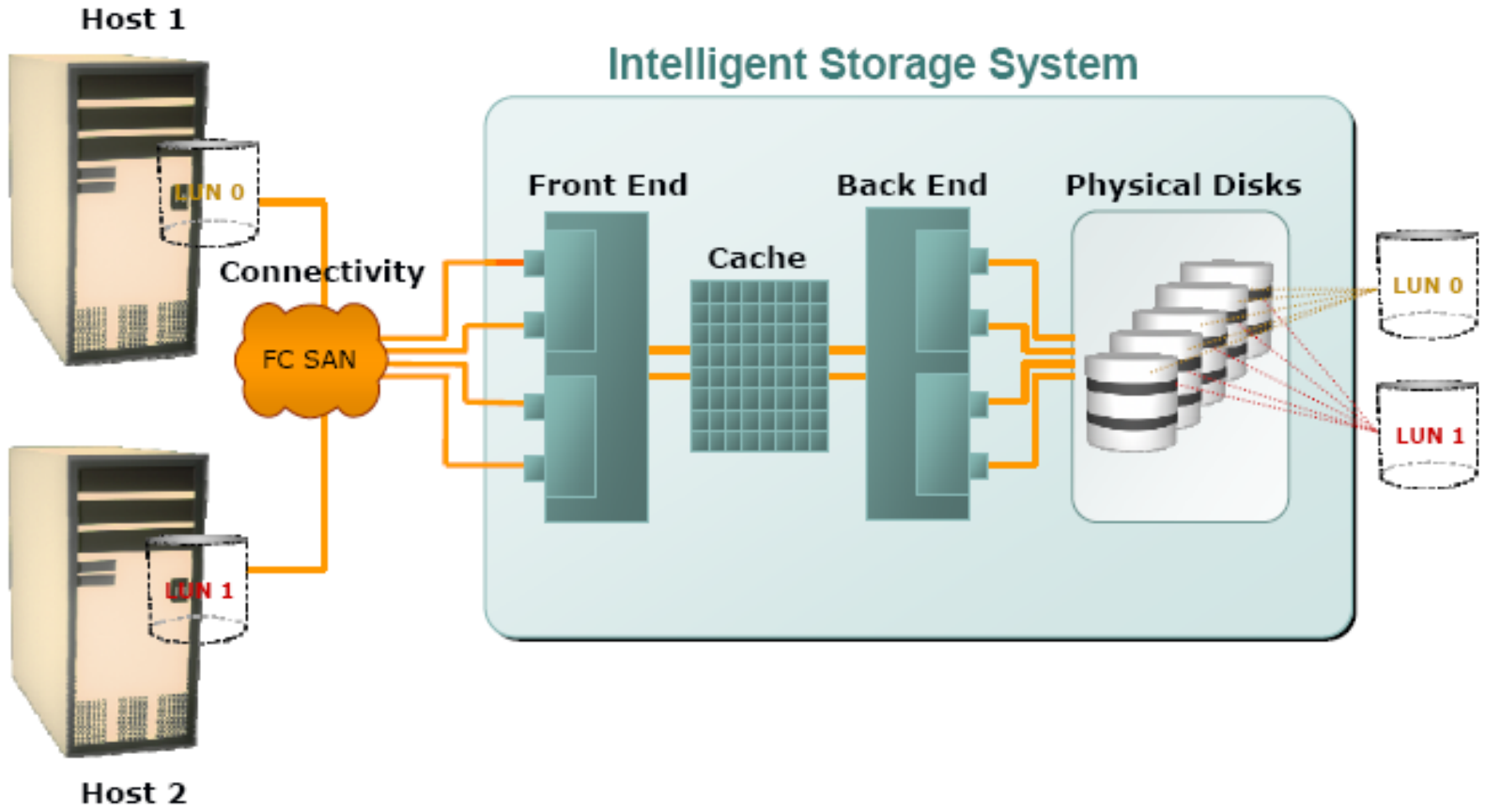
Physical Disks

- Disks are connected to the back-end with either SCSI, IB, Fibre Channel interface



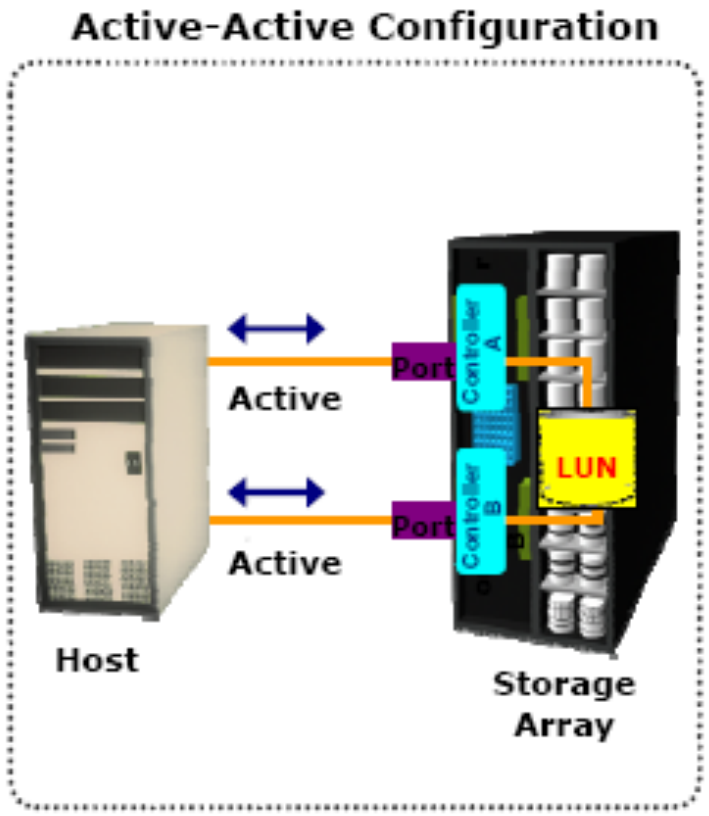
What is LUNs

- Physical drives or groups of RAID protected drives can be logically split into volumes known as logical volumes, commonly referred to as *Logical Unit Numbers* (LUNs)



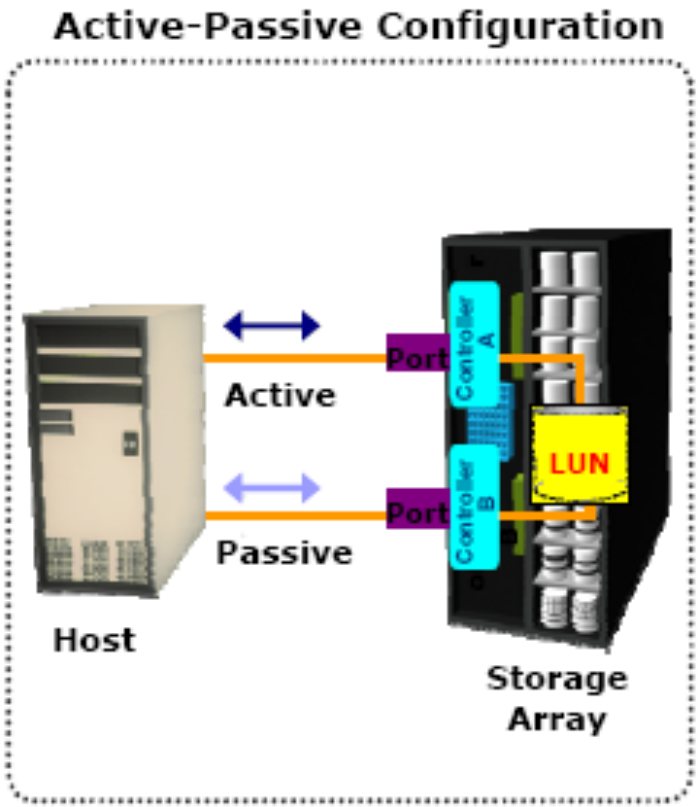
High-end Storage Systems

- High-end storage systems, referred to as *active-active arrays*, are generally aimed at large enterprises for centralizing corporate data
- These arrays are designed with a large number of controllers and cache memory
- An active-active array implies that the host can perform I/Os to its LUNs across any of the available Paths



Midrange Storage Systems

- Also referred as Active-passive arrays
- Host can perform I/Os to LUNs only through active paths
- Other paths remain passive till active path fails
- Midrange array have two controllers, each with cache, RAID controllers and disks drive interfaces
- Designed for small and medium enterprises
- Less scalable as compared to high-end array

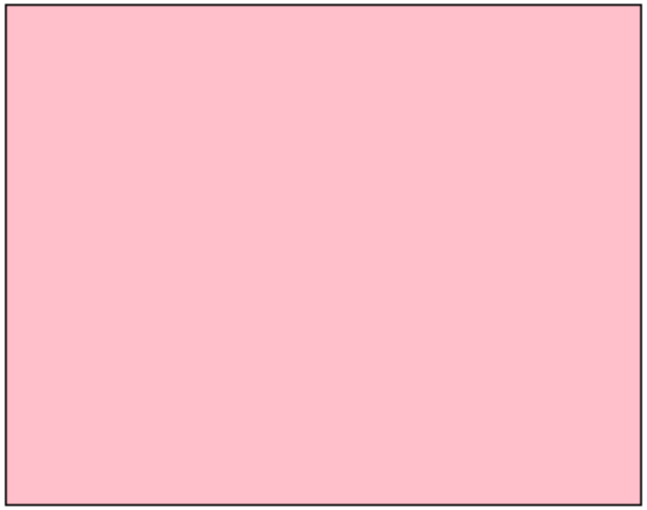


Performance Metrics for Storage

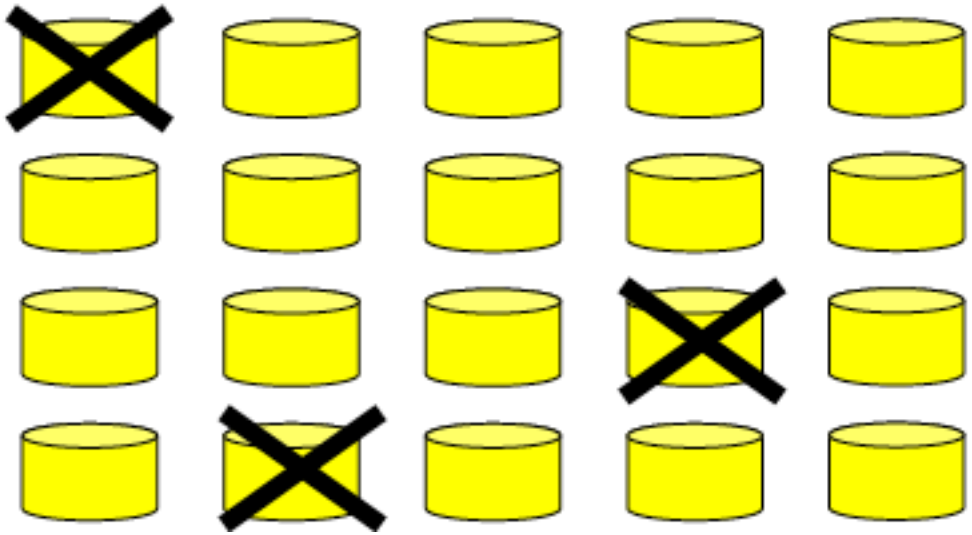
- Storage efficiency: how much redundant information do you store?
- Saturation throughput: how many I/O requests can the system handle before it collapses (or delay increases to infinity)?
- Rebuild time: how fast can you replace information lost due to disk failure?
- Mean time to data loss: under assumptions on failure and usage models of the system, how long do you expect to run without any permanent loss of data?
- Encoding/Decoding/Update/Rebuild complexity: the computation power needed for all these operations; also, how many bytes of data on how many disks do you have to update if you just want to update 1 byte of user data?
- Sequential read/write bandwidth: bandwidth the system can provide for streaming data
- Scale-out (as opposed to "Scale-up"): Add low cost commodity hardware to increase capacity incrementally (scale-out) rather than make up-front large investment in more expensive complex hardware
- High availability: Cost of down time for businesses is large
- Systems cannot be taken down for backing up data
- Rebuild time should be small
- Reliability vs. cost: Replication based schemes for reliability are expensive

Erasure Coding Basics

You've got some data:



And a collection of storage nodes

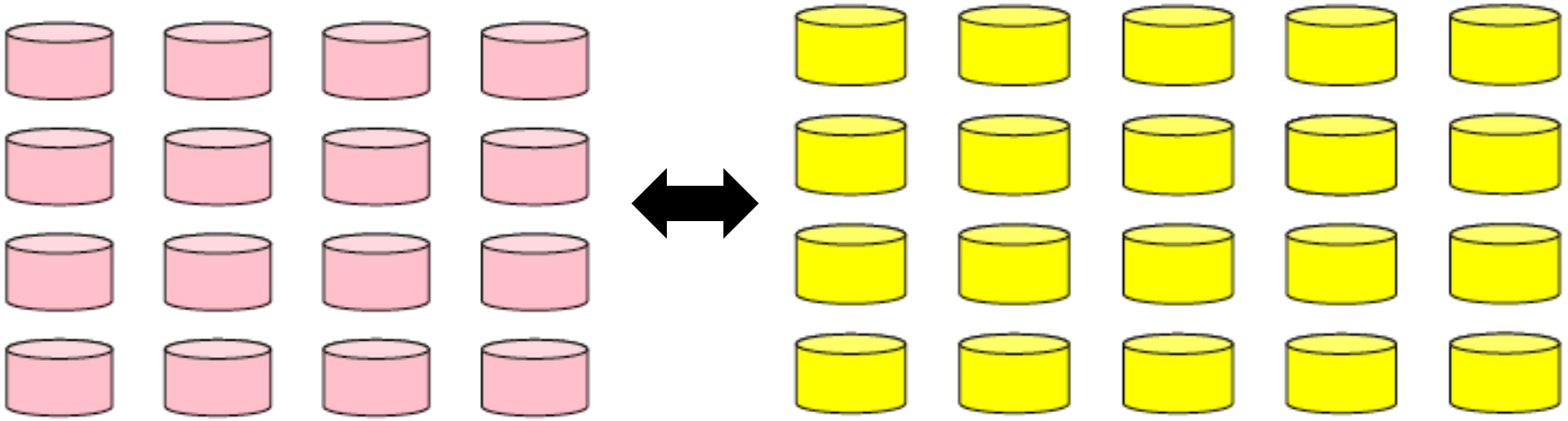


And you want to store the data on the storage nodes so that you can get the data back, even when the nodes fail.

Erasure Coding Basics

More concrete: You have *k* disks worth of data:

And *n* total disks

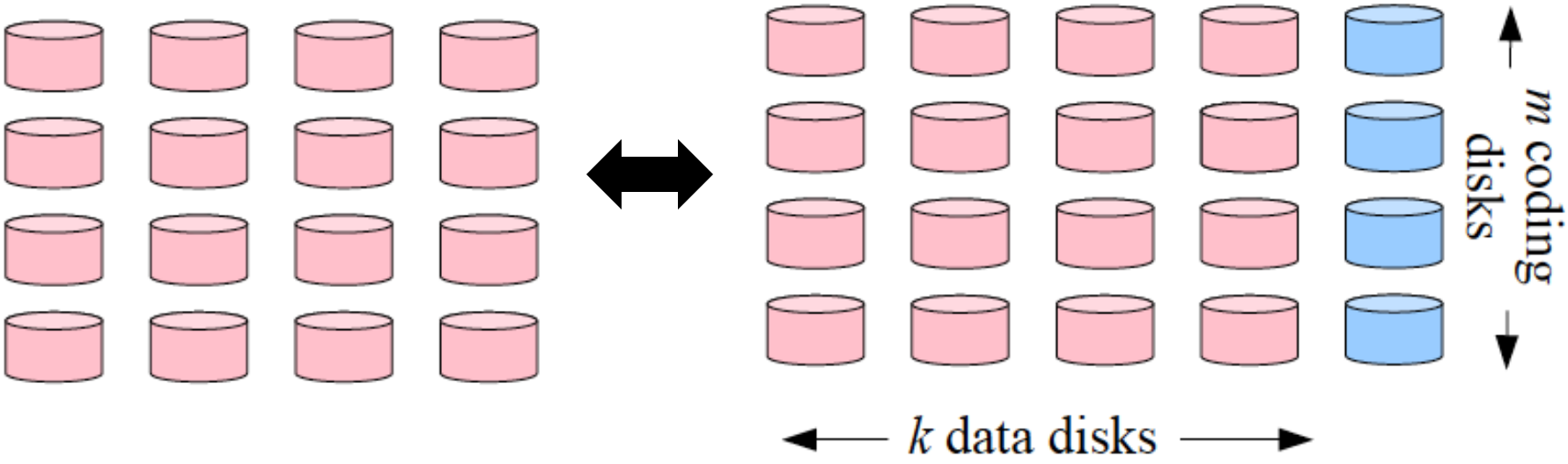


The erasure code tells you how to create *n* disks worth of data +coding so that when disks fail, you can still get the data.

Nomenclature

You have
k disks worth of data:

And *n* total disks
n = *k* + *m*

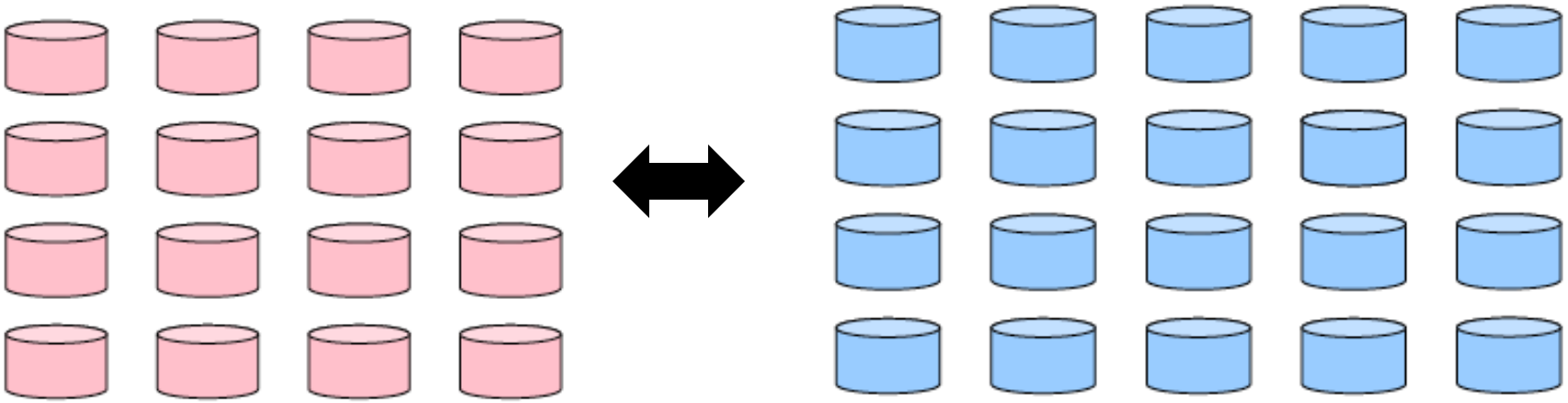


A systematic erasure code stores the data in the clear on *k* of the *n* disks. There are *k* data disks, and *m* coding or "parity" disks.

Nomenclature

You have
k disks worth of data:

And *n* total disks
$n = k + m$

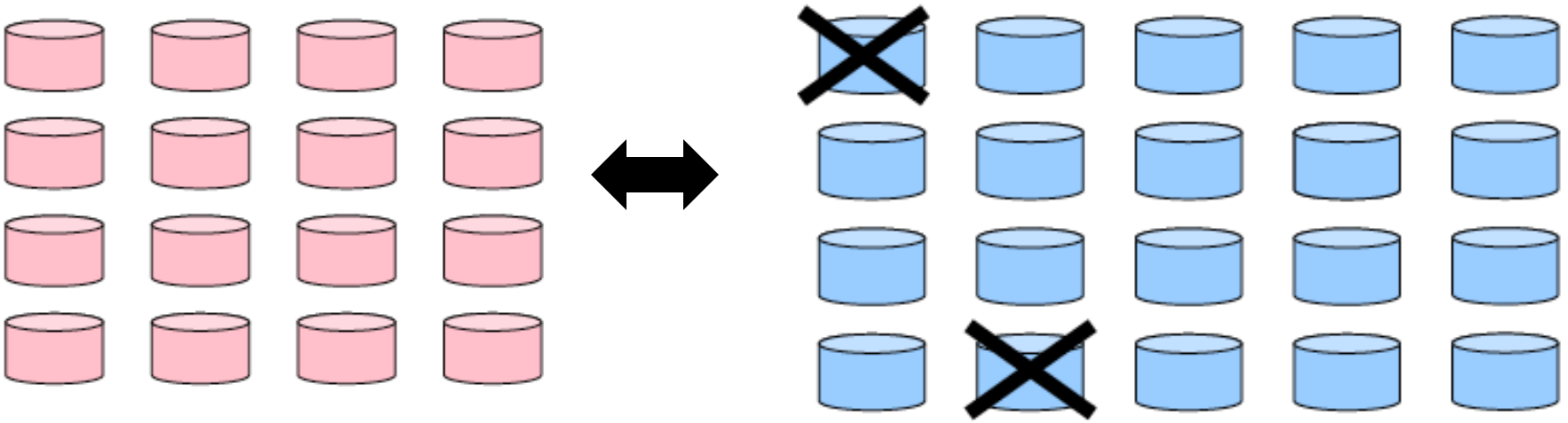


A non-systematic erasure code stores only coding information, but we still use *k*, *m*, and *n* to describe the code.

Nomenclature

You have
k disks worth of data:

And *n* total disks
$n = k + m$

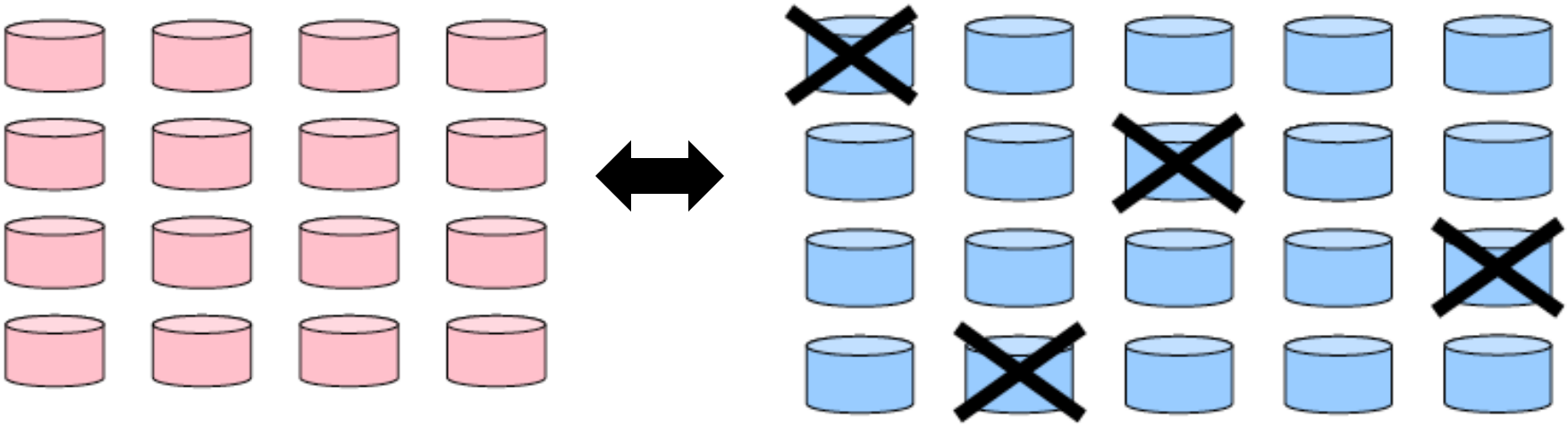


When disks fail, their contents become unusable, and the storage system detects this. This failure mode is called an erasure.

Nomenclature

You have
k disks worth of data:

And *n* total disks
n = *k* + *m*

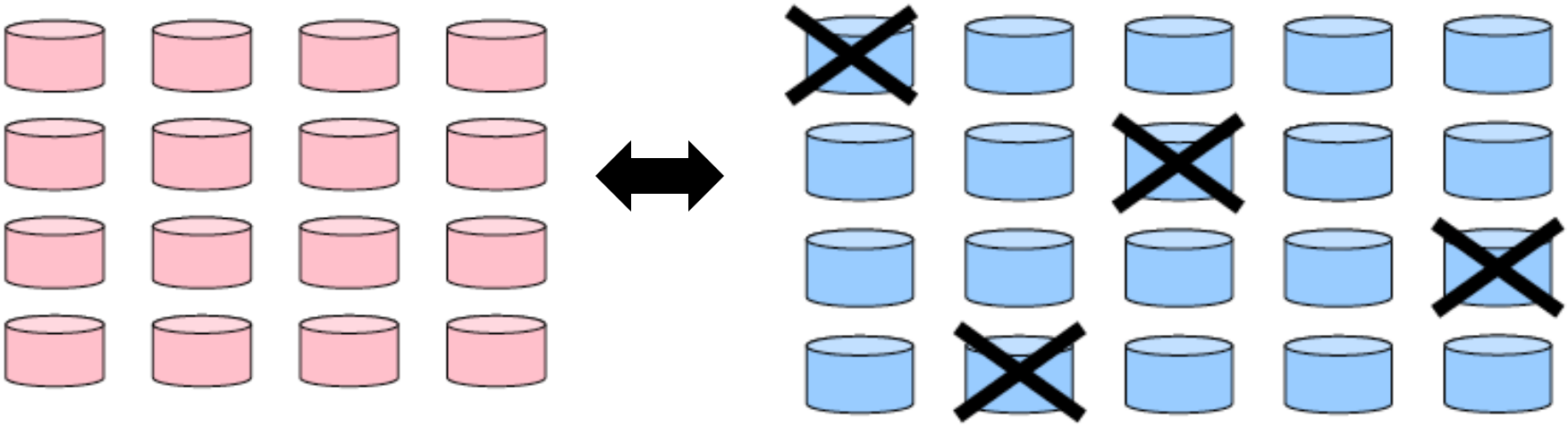


An MDS ("Maximum Distance Separable") code can reconstruct the data from any m failures. That is the best you can do.

Nomenclature

You have
k disks worth of data:

And *n* total disks
n = *k* + *m*

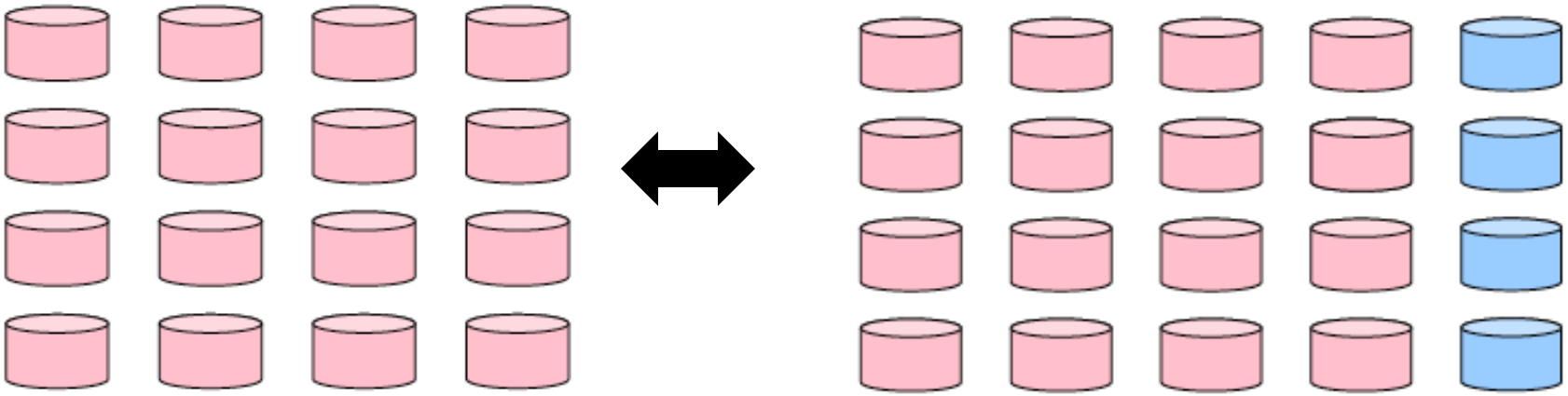


An MDS ("Maximum Distance Separable") code can reconstruct the data from any m failures. That is the best you can do.

Nomenclature

You have
k disks worth of data:

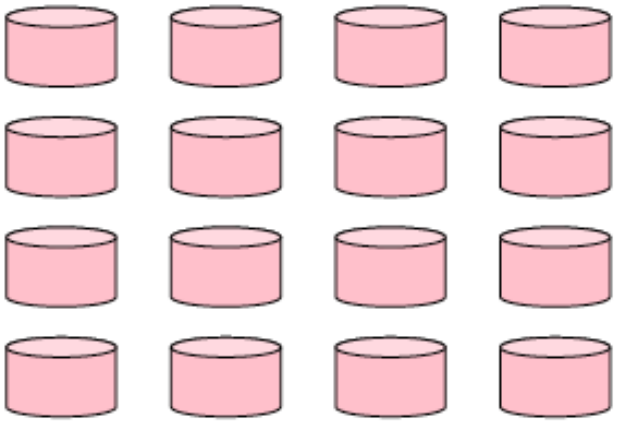
And *n* total disks
$n = k + m$

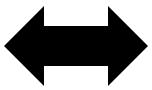


A horizontal code is systematic, and partitions the disks into data disks and coding disks.

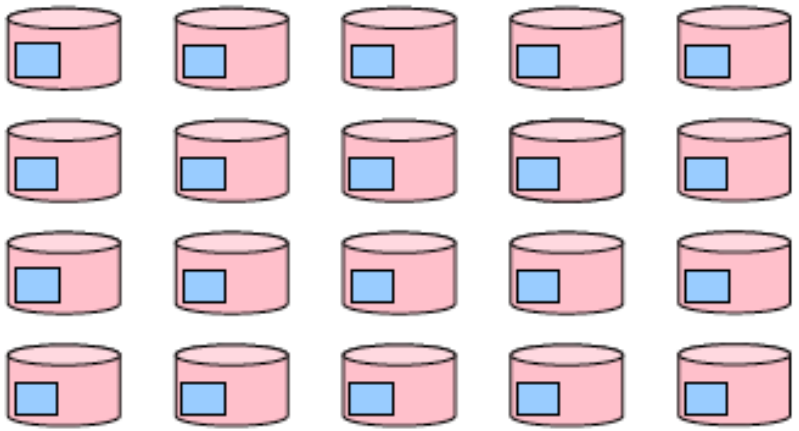
Nomenclature

You have
k disks worth of data:





And *n* total disks
$n = k + m$



A vertical code is systematic, but each disk is required to hold some data and some coding.

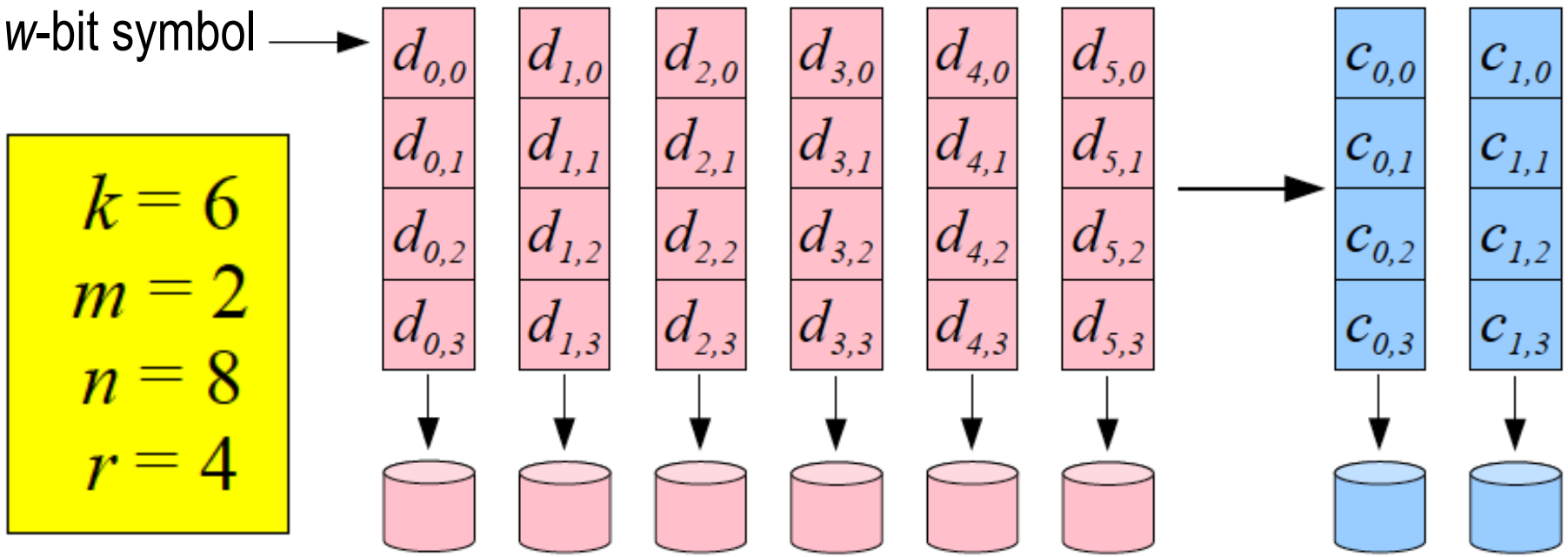
Vertical codes can be MDS, so long as they tolerate all combinations of *m* failed disks.

Two views of a stripe

The Theoretical View:

– The minimum collection of bits that encode and decode together.

– *r* rows of *w*-bit symbols from each of *n* disks:

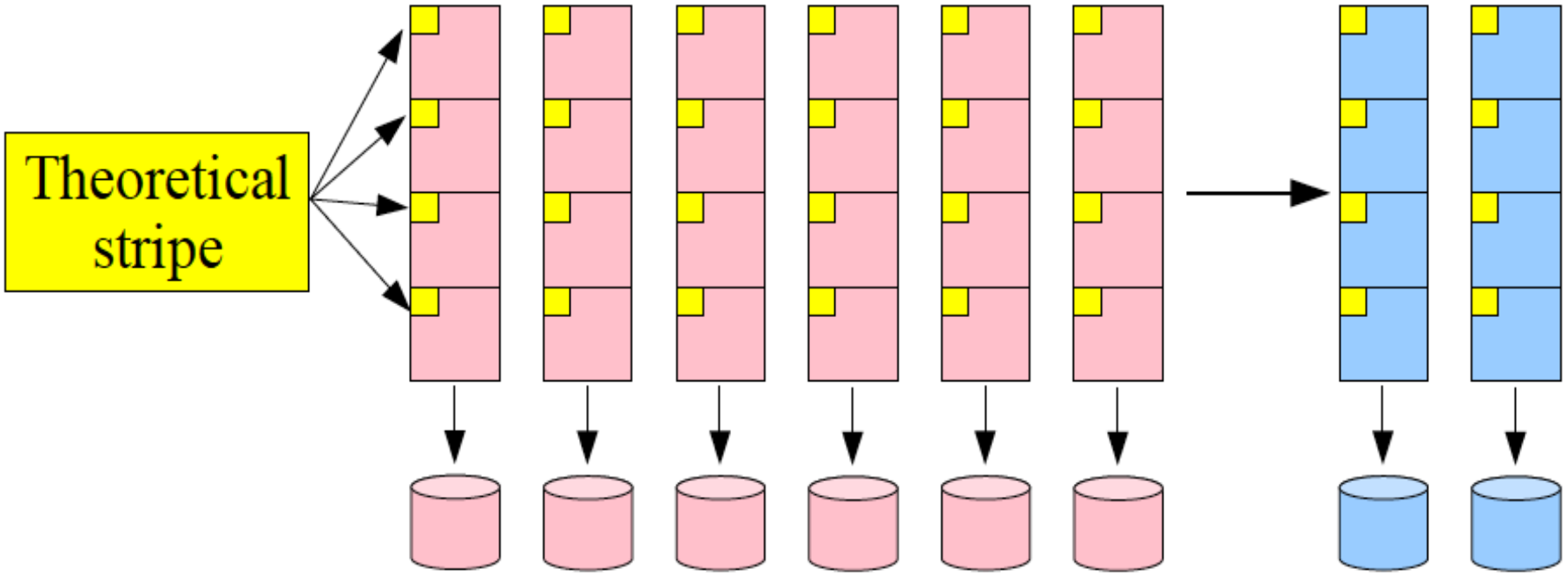


Two views of a stripe

The Systems View:

– The minimum partition of the system that encodes and decode together.

– Group together theoretical stripes for performance

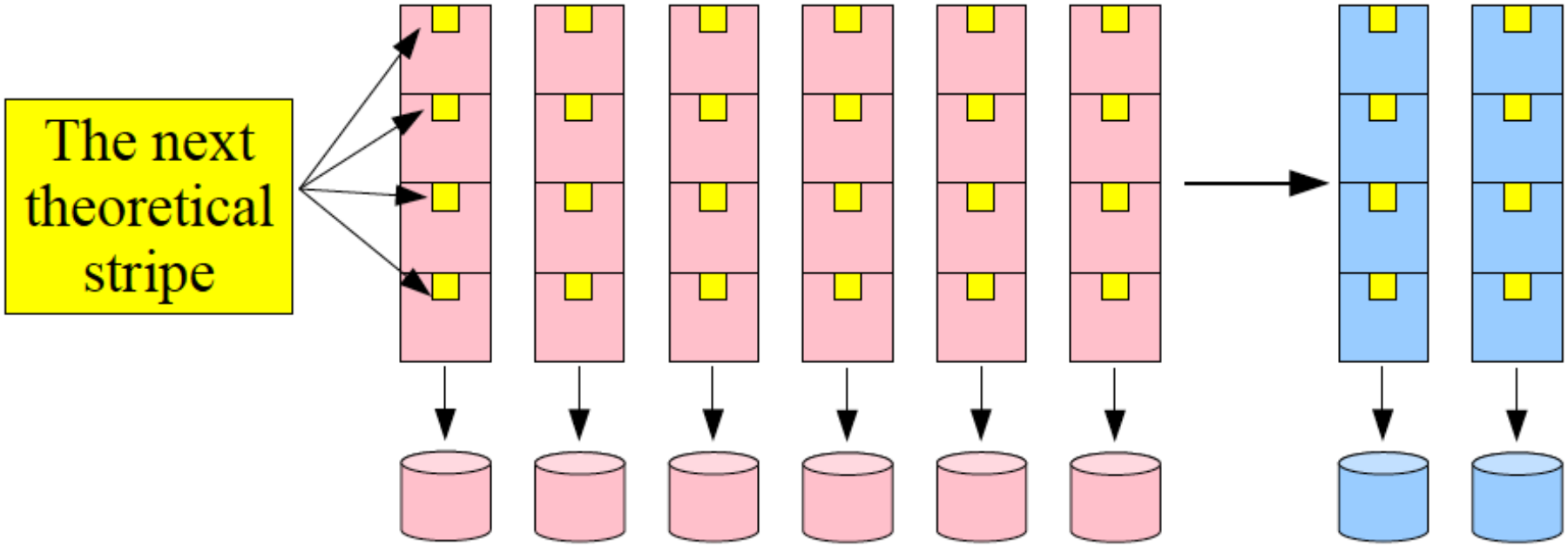


Two views of a stripe

The Systems View:

– The minimum partition of the system that encodes and decode together.

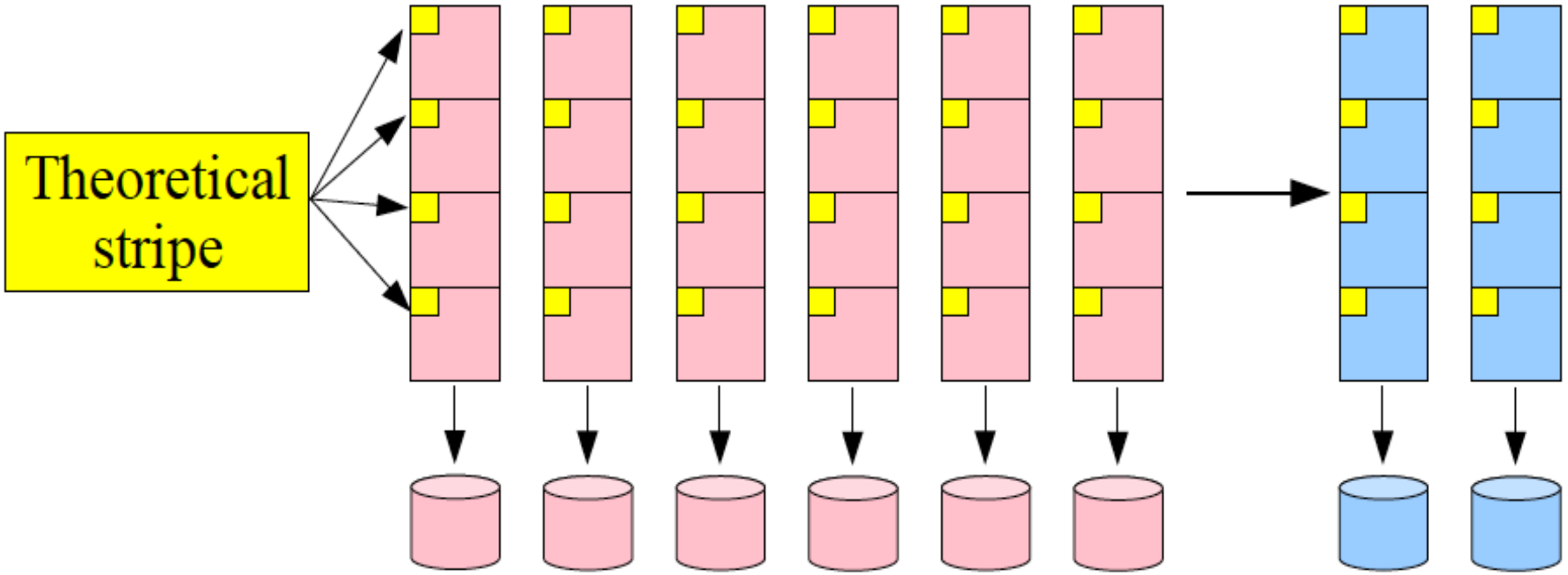
– Group together theoretical stripes for performance



Two views of a stripe

Why the two views:

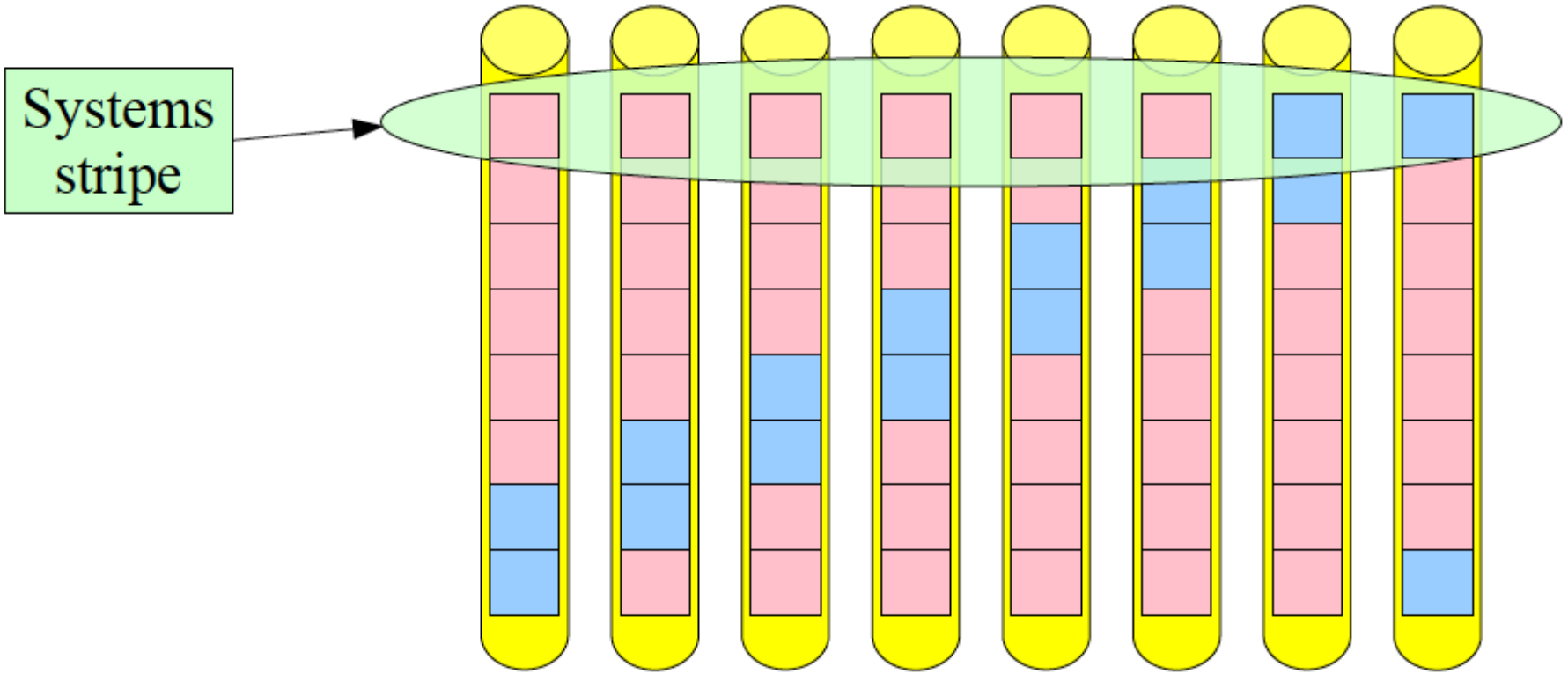
- Because *w* is small (often 1 bit), and operating over collection of symbols is much faster than operating over single symbol



Two views of a stripe

Why the two views:

- You can also balance load by partitioning system into many systems stripes and rotating IDs.



Arithmetic for Erasure Codes

- When w = 1: XOR's only.
- Otherwise, Galois Field Arithmetic GF(2^w)
  - *w* is 2, 4, 8, 16, 32, 64, 128 so that words fit evenly into computer words.
  - Addition is equal to XOR.
    - Nice because addition equals subtraction.
  - Multiplication is more complicated:
    - Gets more expensive as w grows.
    - Buffer-constant different from a * b.
    - Buffer * 2 can be done really fast.
    - Open source library support.

Arithmetic for Erasure Codes

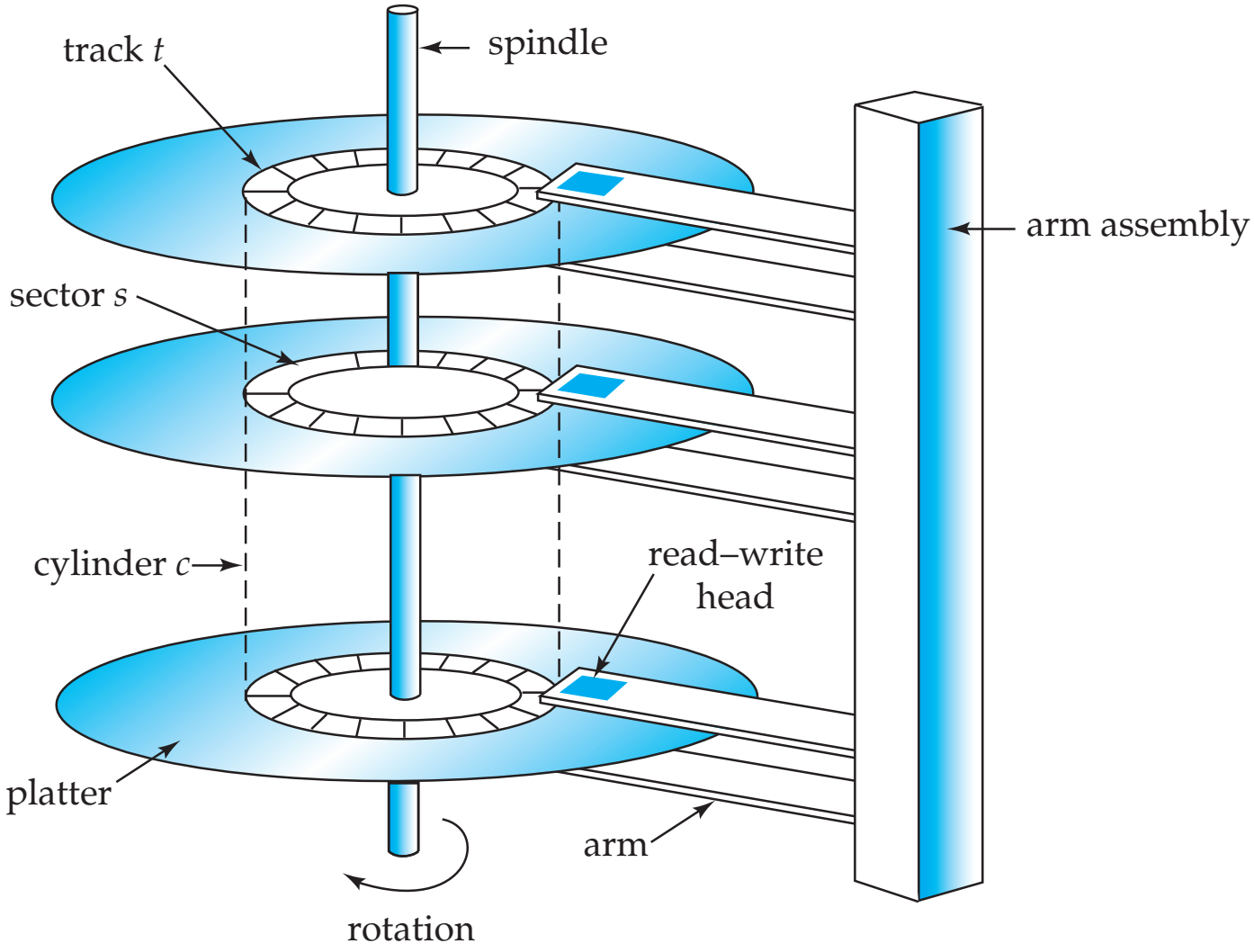
- Multiplication of a*b in GF(2^w)
  - Table lookup for w = 2, 4 ,8.
  - Three table lookups + a little arithmetic for w = 16
  - 16 table lookups + XORs for *w* = 32
  - More expensive for w = 64, 128.

- • Multiplication of a buffer by a in GF(2^w)
  - Used to be roughly a factor of 4 worse than XOR for *w* = 2, 4, 8, 16.
  - SIMD instructions
  - *w* = 32 slightly slower. Others slower still.

Reed Solomon Codes: Properties

- MDS Erasure codes for any n and k.
  - That means any $m = (n-k)$ failures can be tolerated without data loss.
- $r = 1$
  - (Theoretical): One word per disk per stripe.
- w constrained so that $n \leq 2^w$.
- Systematic and non-systematic forms.

Magnetic Hard Disk Mechanism



NOTE: Diagram is schematic, and simplifies the structure of actual disk drives

Magnetic Disks

- **Read-write head**
  - Positioned very close to the platter surface (almost touching it)
  - Reads or writes magnetically encoded information.
- Surface of platter divided into circular **tracks**
  - Over 50K-100K tracks per platter on typical hard disks
- Each track is divided into **sectors.**
  - A sector is the smallest unit of data that can be read or written.
  - Sector size typically 512 bytes
  - Typical sectors per track: 500 to 1000 (on inner tracks) to 1000 to 2000 (on outer tracks)
- To read/write a sector
  - disk arm swings to position head on right track
  - platter spins continually; data is read/written as sector passes under head
- Head-disk assemblies
  - multiple disk platters on a single spindle (1 to 5 usually)
  - one head per platter, mounted on a common arm.
- **Cylinder** *i* consists of i^{th} track of all the platters

Magnetic Disks (Cont.)

- Earlier generation disks were susceptible to head-crashes
  - Surface of earlier generation disks had metal-oxide coatings which would disintegrate on head crash and damage all data on disk
  - Current generation disks are less susceptible to such disastrous failures, although individual sectors may get corrupted
- **Disk controller** – interfaces between the computer system and the disk drive hardware.
  - accepts high-level commands to read or write a sector
  - initiates actions such as moving the disk arm to the right track and actually reading or writing the data
  - Computes and attaches **checksums** to each sector to verify that data is read back correctly
    - If data is corrupted, with very high probability stored checksum won't match recomputed checksum
  - Ensures successful writing by reading back sector after writing it
  - Performs remapping of bad sectors

Performance Measures of Disks

- **Access time** – the time it takes from when a read or write request is issued to when data transfer begins. Consists of:
  - **Seek time** – time it takes to reposition the arm over the correct track.
    - Average seek time is 1/2 the worst case seek time.
    - 4 to 10 milliseconds on typical disks
  - **Rotational latency** – time it takes for the sector to be accessed to appear under the head.
    - Average latency is 1/2 of the worst case latency.
    - 4 to 11 milliseconds on typical disks (5400 to 15000 r.p.m.)
- **Data-transfer rate** – the rate at which data can be retrieved from or stored to the disk.
  - 25 to 100 MB per second max rate, lower for inner tracks
  - Multiple disks may share a controller, so rate that controller can handle is also important
    - E.g. SATA: 150 MB/sec, SATA-II 3Gb (300 MB/sec)
    - Ultra 320 SCSI: 320 MB/s, SAS (3 to 6 Gb/sec)
    - Fiber Channel (FC2Gb or 4Gb): 256 to 512 MB/s

Performance Measures (Cont.)

- **Mean time to failure (MTTF)** – the average time the disk is expected to run continuously without any failure.
  - Typically 3 to 5 years
  - Probability of failure of new disks is quite low, corresponding to a "theoretical MTTF" of 500,000 to 1,200,000 hours for a new disk
    - E.g., an MTTF of 1,200,000 hours for a new disk means that given 1000 relatively new disks, on an average one will fail every 1200 hours
  - MTTF decreases as disk ages

Comparing Cost and Performance of Replication and Erasure Coding

- Storage overhead: Additional storage required for providing data protection
  - Less overhead => storage efficient
- Data loss can occur when either a disk has died or when a disk is still working, but the blocks that hold the data being read have gone bad. For our purposes, we consider either case to have the same effect on our system.
- Data spread across DCs to provide full data availability

DCs	Minimum overhead for N+1
2	100% (100/100)
3	50% (50/50/50)
4	33% (33/33/33/33)
5	25% (25/25/25/25/25)

Replication vs Erasure Coding

- p_L = probability that a single disk is dead, permanently unreadable
- estimate of pL depends on the mean time between failures as well as the time required to replace the disk.
- For example, suppose a disk performs reliably on average for 1,000 days, roughly three years
  - If a hot spare is available to replace the disk upon failure and it takes one day to format the disk and copy data to it, p_L =0.001.
  - If there is not a hot spare and it takes a couple days to order and install a replacement disk and a day to fill it, p_L =0.003
- Similarly let p_U be the probability that a disk is unavailable.
  - A disk may be unavailable, yet still alive. For example, a disk may be in perfect order but temporarily disconnected from the network.
- Because disks can be temporarily unavailable without failing, say due to a network outage, p_L is always smaller than p_U.

Replication

- In a replicated system with k replicas of each object, the probability of data loss is $p^k_{L,}$ assuming disk failures are independent.

- Given a tolerable probability of data loss ε, we can solve for the number of disks k needed keep the probability of loss below this level:

$$k = (\log \varepsilon)/\log p_L$$

- same calculation applies to data unavailability if we replace p_L with p_U, again assuming independence

Erasure Coding

- In an m + n erasure-encoded system, each object is divided into m equal-sized fragments.
- In addition n parity fragments are created, each the size of one of the data fragments.
- The data can be read if any m out of the m + n fragments are available.
  - Stated negatively, data will be lost if more than n fragments are simultaneously lost.

- Since we need m out of m + n disks to reconstruct an object, the probability of data loss is the probability of more than n disks being dead simultaneously:

$$\sum_{i=n+1}^{m+n} \binom{m+n}{i} p_L^i (1-p_L)^{m+n-i}$$

- Define k_c =(m + n)/m. This is the redundancy factor for m + n erasure coding
  - analogous to k for replication
- For a fixed redundancy level, an erasure-encoded system has at least the same level of reliability as the corresponding replicated system
- For a fixed level of reliability, an erasure-encoded system requires less disk space or can store more data with the same amount of disk space.

Erasure Coding

- Given a number of data fragments m and an acceptable probability of unavailability ε, you can solve for the smallest value of n such that

$$\sum_{i=n+1}^{m+n} \binom{m+n}{i} p_L^i (1-p_L)^{m+n-i} < \varepsilon$$

- Build a system with probability of data recovery 0.999999 (six nines) using disks that have probability 0.995 of being alive.
  - Triple replication would have a probability of DL equal to $0.005^3 = 1.25 \times 10^{-7}$.
- Suppose you want to use erasure coding with 8 data disks. You can write code to show that an 8 + n system would require n = 3 to keep the probability of DL below 10^{-6}.
  - In fact, an 8 + 3 system has a probability of DL 1.99×10^{-7}.
- A 1-GB video stored in the triple replicated system would require 3 GB of storage.
- In an 8+3, the same object would be stored in 8 data fragments and 3 parity fragments, each 125 MB in size, for a total of 1.375 GB.
- The erasure-coded system would use about half as much disk space and offer the same level of data protection.

Erasure Coding

- Given a value of m and the individual disk reliability, we figured how to choose n to achieve the desired level of protection.
- But how do you choose m?
- Increasing n while holding m fixed increases reliability.
- Increasing m while holding n fixed decreases reliability.
- But in a sense, we gain more reliability by increasing n than we lose by increasing m.
- We can increase reliability by increasing m and n proportionately, keeping the redundancy factor k_c constant. [Proof]
  - For example, a 4 + 2 system will be more reliable than a 2 + 1 system even though both have the same redundancy kc =1.5.
  - So why not make m and n larger and larger, obtaining more and more reliability for free?
    - Of course the increase in m and n is not free, though the cost is subtle.
- Larger values of m and n do result in greater data protection for a fixed level of disk reliability.
- However, while larger values of m and n reduce the chance of complete failure (irrecoverable data loss), they potentially increase latency and reconstruction costs by increasing the chance of at least one recoverable disk failure.

Erasure Coding

- Increasing m and n also increases the total number of data fragments m + n to manage.
  - In practice, erasure-encoded systems use values of m on the order of 10, not on the order of 100.
  - For example, you may see 6 + 3 systems or 12 + 4 systems, but not 100 + 50 systems
    - you can obtain high levels of data protection without resorting to large values of m.
    - erasure-encoded systems typically have on the order of millions of object fragments.
  - Need a database to keep track of each fragment.
  - If this database is kept in memory, it has to be on the order of gigabytes.
  - If each record in the database is on the order of a kilobyte, the table can contain on the order of a million rows, and so the number of fragments is kept on the order of millions in order to fit the corresponding database in memory.
- There is also the time required to find and assemble the fragments.
  - The required time depends greatly on how EC is implemented, but is always some amount of time, and hence an overhead associated with EC that is not required with replication.
  - EC system can be slower than a replicated system, even when all fragments are in the same data center.
  - The more data fragments there are to manage, the more work that is required to rebuild the fragment inventory database when failures occur

Disk Allocation to DCs

- Far more likely that an entire data center would be unavailable than destroyed.
  - Data centers are occasionally inaccessible due to network outages.
  - This causes all disks inside to be simultaneously unavailable.
  - All disks in data center would fail in a natural disaster or act of war.
  - This means that DU is more correlated than DL.
- Assume network failures to data centers are independent, then the same probability calculations apply to data loss and data unavailability.
- In replicated systems a replica resides in each DC
  - If there are d data centers, the probabilities of an object being lost or unavailable are p^d_L and p^d_U respectively.
- In EC systems, the number of fragments per object is typically larger than the number of DCs.
  - A company often has two or three data centers; more than four would be very unusual, especially for a mid-sized company.
  - EC systems using a scheme such as 8+4 are not uncommon.
  - With fewer than 12 DCs, some of these fragments would have to be co-located.
- Because data fragments are inevitably co-located, these fragments have correlated probabilities of being unavailable and so the unavailability probability for the system goes up.

Probability Assumptions on failure

- Disk failures could be correlated for any number of reasons:
  - disks coming from the same manufacturing lot
  - disks operating in the same physical environment, …
- Disk Failures are correlated in time (enough evidence of that)
  - If you have had more failures than usual this week, you are likely to have more failures than usual next week too
- The assumption of independence is more accurate for disks in separate data centers.
  - For replicated systems with each replica in a separate data center, independence is a reasonable assumption.
  - But for EC systems with multiple fragments in each data center, the assumption of independence is less justified for data loss, and unjustified for data unavailability

Probability Assumptions on failure

- Disk failures could be correlated for any number of reasons:
  - disks coming from the same manufacturing lot
  - disks operating in the same physical environment, …
- Disk Failures are correlated in time (enough evidence of that)
  - If you have had more failures than usual this week, you are likely to have more failures than usual next week too
- New findings (Schroeder and Gibson, CMU, and Google, 2007) based on disk replacement (rates) rather than disk failure (rates)
- Disk replacements are not independent and do not follow a Poisson process
  - Longer since the last disk failed => longer till the next disk will fail!
    - Possible explanation: environmental and other factors (temperature etc.) are more important than component specific factors
- Disk replacement rates do not enter a steady state after 1 year (like a bathtub curve); instead, the replacement rates steadily increase over time
- The assumption of independence is more accurate for disks in separate data centers.
  - For replicated systems with each replica in a separate data center, independence is a reasonable assumption.
  - But for EC systems with multiple fragments in each data center, the assumption of independence is less justified for data loss, and unjustified for data unavailability

Cost of disk failure

- For fixed level of reliability, erasure coding requires less disk space than replication.
  - EC cheaper in cost compared to Replication
  - Other factors depend on usage scenarios.
- EC can lower the probability of a catastrophic failure, it increases the probability of a recoverable failure.
  - The probability that one or more disks in a set will fail is approximately the probability of a particular disk failing times the number of disks
- Comparing triple replication to 8 + 3 erasure coding: Both systems had roughly the same probability of data loss.
  - Replication used 3 disks per object while erasure coding used 11
  - So the erasure-coded system is nearly 4 times as likely to experience a single, recoverable disk failure.
- Cost of recoverable disk failure?
  - cost of replacing hard disks is proportional to the total number of disks in use, whether those disks contain replicated objects or erasure-encoded object fragments.
  - Erasure encoding does not increase the number of disk failures.
  - By reducing the number of disks needed, it can reduce the number of failures.
  - However, erasure encoding increases the probability that an object request is affected by a disk failure.

Latency due to disk failure

- In a replicated system, requests could be routed to the nearest available replica and if the nearest replica is not available, the request would fail over to the next nearest replica and so on until a replica is available or the request fails.
- In an m+n erasure-encoded system, an object request could be filled by reconstructing the object from the m nearest fragments. Since object requests more often encounter (recoverable) disk failures in erasure-encoded systems, they will more often involve an increase in latency.
- Suppose a replicated system maintains k copies of an object, each in a different data center, and that requesting data from these centers has latency L1 <L2 < ... < Lk for a given user.
  - Suppose each replica has a probability p of being unavailable.
  - With probability 1 − p the latency in the object request will be L1.
  - With probability p(1 − p) the latency will be L2.
  - The expected latency will be

$$\sum_{i=1}^{k} p^{i-1}(1-p)L_i$$

Latency due to disk failure

- The expected latency is $\sum_{i=1}^{k} p^{i-1}(1-p)L_i$

- If p is very small, the terms involving higher power of p will be negligible and the sum above will be approximately (1 − p)L1 + pL2

- For example, if there is a probability of 0.001 that an object is available at the nearest data center, and the second data center has 100 times greater latency,
  - the expected latency is 10% greater (that is, pL2/L1 =0.001 * 100 = 0.1) than if both replicas were located in the nearest data center.

Latency due to disk failure

- In erasure-encoded system, fragments could be geographically distributed in many ways.
- If latency is a concern, an m + n system would store at least m fragments in a single location so that only local reads would be necessary under normal circumstances.
  - If m fragments are stored in one location, the probability of one local disk failure is mp.
  - This means the probability of a single local failure, and the necessity of transferring data from a more distant data center, is m times greater for an erasure-coded system compared to a replicated system.
- The expected latency increases from approximately $(1 - p)L1 + pL2$ in a replicated system to approximately $(1 - p)L1 + mpL2$ in an erasure-encoded system.
- For 8 + 3 erasure coding , if there is a probability of 0.001 that an object is available at the nearest data center, and the second data center has 100 times greater latency.
  - we pick our units so that the latency of the nearer server is L1 = 1.
  - If m = 8, the expected latency would be 1.099 using replication and 1.799 using erasure coding.
- For active data, objects that are accessed frequently, latency is a major concern and the latency advantage of replication should be considered

Failure Probability Calculation

- The probability that exactly one out of a set of m disks will fail is mp if p is the probability of an individual failure.
- But the probability of one or more failures is $1-(1-p)^m$, which is approximately mp if p is small.

Cost of reconstruction

- When an object request fails, it must be determined exactly what has failed.
  - This is a much simpler task in a replicated system than in an EC system.
- Then the data must be copied to a replacement system
- Cost of rebuilding the disk depends on where the data are coming from? Locally within a data center or from remote DC
  - Replication: content of failed disk copied from closest location
  - EC: With m + n erasure coding, the content of m disks must be brought to one location. Given 6 + 3 encoding, if three fragments are stored in each of three data centers and one disk fails, the content of four disks must be transferred from a remote data center to the site of the failed disk in order to have enough data for reconstruction.
  - Local Reconstruction Codes: Microsoft Azure
- The time necessary for reconstruction feeds back into the data protection calculations.
  - If failed disks can be reconstructed faster, availability improves, increasing the availability of the system or possibly enabling the system to use fewer disks.

Increasing m+n increases reliability

- Let m and n be fixed positive integers.
- Let p be the probability of a single disk being dead at any time.
- Assume p < n/(m + n).
- We start with an m+n EC system and increase m and n proportionately, resulting in a km+kn system.
- As we increase k, the probability of data loss, that is, the probability of more than kn failures, goes to zero as k increases.
  - To see why this is so, let X be a Binomial(km+kn, p) random variable and let Y be a normal random variable with mean k(m+n)p and variance k(m+n)p(1−p)
  - As k increases, the probability distribution of X converges to the probability distribution of Y.
- For any positive k, Prob(X > kn) ≈ Prob(Y >kn) and the error in the approximation decreases with k, becoming exact in the limit.
- Let Z be a standard normal random variable. Then Y has the same distribution as

$$\sqrt{k(m+n)p(1-p)}Z + k(m+n)p$$

Increasing m+n increases reliability

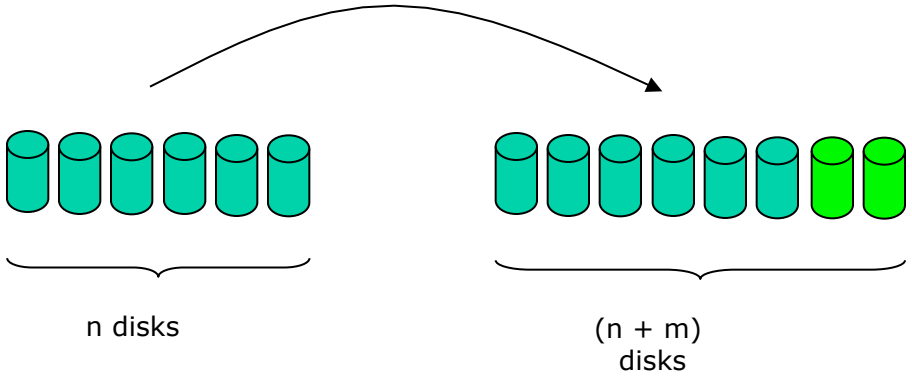
- so we have the following

$$\begin{aligned}\text{Prob}(X > kn) &\approx \text{Prob}(\ Y\ > kn\ ) \\ &= \text{Prob}(\sqrt{k(m+n)p(1-p)Z + k(m+n)p} > kn\ ) \\ &= \text{Prob}\left(Z > \sqrt{k}\ \frac{(1-p)n - mp}{(m+n)p(1-p)}\right)\end{aligned}$$

- The assumption p < n/(m + n) implies that (1 − p)n − mp > 0 and so

$$\lim_{k\to\infty} \text{Prob}\left(\ Z > \sqrt{k}\frac{(1-p)\,n\ -\ mp}{(m+n)\,p\,(1-p)}\right) = 0$$

Erasure Codes



Encode data on n disks onto (n+m) disks such that the whole system can tolerate up to m disk failures

- Reliability
    - Specified by avg # disk failures tolerated
    - If avg # disk failures tolerated = m, then the code is Maximum Distance Separable (MDS)
- Performance (encoding/decoding/update)
- Space usage/Rate: Rate = n / (n+m)
- Flexibility
    - can you arbitrarily add nodes? Change rate?
    - how does this affect failure coverage?

Examples: Reed-Solomon codes, Parity Array codes (EvenOdd coding, X code, Weaver), LDPC codes

Other Issues

- Failure Models:
  - What is a disk failure?
    - Completely unreadable? Mechanical or chip failures, for instance.
    - Latent sector errors?
    - How do you test a drive? Read all sectors and decide faulty if any one operation takes above threshold
      - Result depends on threshold ☹
    - User's point of view: A disk has failed if the user feels it is no longer satisfactory for his/her needs
  - Measures of disk reliability
    - Manufacturers:
      - MTTF: mean time to failure (based on accelerated tests on a large number of disk)
      - AFR: annualized failure rate (percentage of disks expected to fail in a year)
    - More recent measures (from user's PoV)
      - ARR: annualized *replacement* rate (percentage of disks replaced by a user in a year)

Other Issues

- Failure Models (contd.):
  - Traditional assumptions
    - Disk failures are independent and is a Poisson process (that is, time between failures is exponentially distributed with parameter λ; then the MTTF = 1/λ)
    - Bathtub curve (plot of failure rate vs. time looks like a bathtub):
      - "infant mortality": failure rates are high in the first few months to a year or so
      - "useful life period": the failure rate is minimum and stays constant from a year to about 4 or 5 years
      - "wearout period": failure rate again goes up after 4 or 5 years
  - New findings (Schroeder and Gibson, CMU, and Google, 2007) based on disk replacement (rates) rather than disk failure (rates)
    - Disk replacements are not independent and do not follow a Poisson process
      - Longer since the last disk failed => longer till the next disk will fail!
      - Possible explanation: environmental and other factors (temperature etc.) are more important than component specific factors
    - Disk replacement rates do not enter a steady state after 1 year (like a bathtub curve); instead, the replacement rates steadily increase over time
  - Different failure model => different values for performance metrics! => different design of codes for achieving performance!

Other Issues

- Usage models are as important as failure models in estimating performance metrics and designing good codes:
    - **Write once, (almost) never read**: eg. Archival storage
    - **Write once, read many times**: eg. Streaming applications like Youtube
    - **Random short reads and writes**: eg. Systems handling lots of short transactions like shopping sites, or high performance computing
  - A storage system with a certain coding scheme and a certain failure model can perform significantly differently under each of the above usage models
  - Unlikely that there is one coding scheme that is suited for all failure and usage models

Comparing Cloud Storages

http://www.tomshardware.com/reviews/cloud-storage-provider-comparison,3905.html

- More than a feature comparison, looked at the services as three use cases.
  - The first is personal storage, or extending one's own file system to the cloud.
  - The second is sharing, especially of moderately size to large documents. Sharing was looked at in both a one-to-one and group sharing, with people who had accounts on the service and those who did not.
  - Finally, the third use case was extensibility, API and SDK. For this, we explored services' abilities to be a platform or infrastructure backbone to support a small business or gaming-in-the-cloud.