Operating System (OS)
CS232

Persistence: Redundant Arrays of Inexpensive Disks (RAIDs)

Dr. Muhammad Mobeen Movania

Outlines

- What are RAIDs?
- What does RAIDs contain?
- Why RAIDs were invented?
- How RAIDs provide reliability?
- Redundancy using striping and mirroring
- Logical to physical mapping
- RAID levels and comparison
- Summary

What are RAIDs?

- Redundant Array of Inexpensive Disks (RAIDs) is a technique to use multiple disks in concert to build a faster, bigger and more reliable disk system.
- Goals
  - Reliability (restore of data on disk failures)
  - Capacity (more than one disks together)
  - Performance (writes can happen in parallel on different disks)

RAIDs contents

- Externally,
  - RAID looks like a disk that contains a group of blocks that can be read or written to.
- Internally
  - RAID consists of multiple disks, memory (both volatile and non-volatile), and one or more processors to manage the system.
- To the user system, RAID appears as a big disk

Why RAIDs were invented?

- Basic idea
  - to combine multiple, small inexpensive disks drive into an array of disk drives which yields performance exceeding that of a Single, Large Expensive Drive(SLED).
- Additionally
  - this array of drives appear to the computer as a single logical storage unit or drive.
- Inventors:
  - In 1987, Patterson, Gibson and Katz at the University of California Berkeley, published a paper entitled “ A Case for Redundant Array of Inexpensive Disks(RAID)”.

How RAIDs provide reliability?

- RAIDs provide reliability through redundancy
- In a single large disk, disk failure results in complete data loss
  - As number of disks per component increases, the probability of failure also increases
- Solution
  - Redundancy (Make a copy of the data on another disk in parallel)

Redundancy through striping

- **Striping:**
  - Key idea: Distribute data across multiple disk
  - Read/write from multiple disk in parallel
  - Method of concatenating multiple drives into one logical drive
  - Two types: bit-level or block level

-chunk size design considerations:
1. small -> files striped across many disks
> position time goes up because worst case
2. big -> vice versa

- Capacity = N.B
- reliability = worst (ask disk failure leads to data loss)
- performance = excellent

Blocks in the same row are called stripes

Extract the most parallelism when requests are made for contiguous chunks of the array

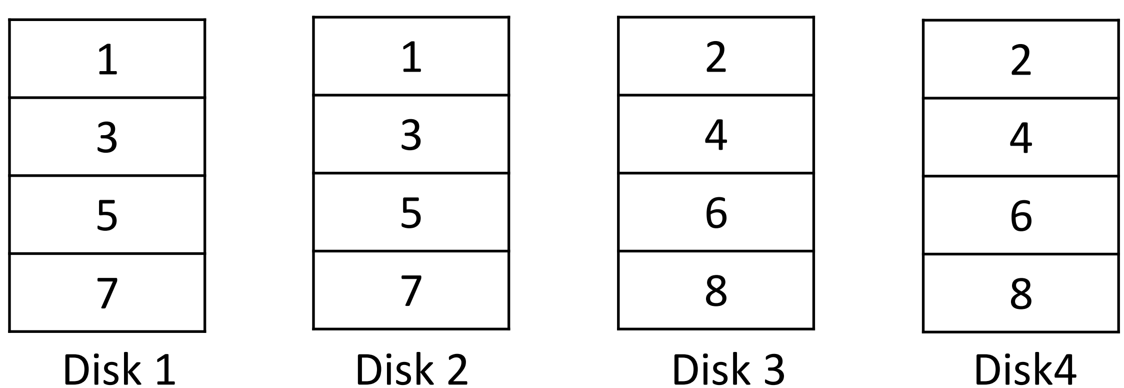
Disk 1	Disk 2	Disk 3	Disk4
1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

- single-request latency: identical to a single disk/

- steady state throughput:
--for sequential workload:
N times S (transfer rate for sequential workload of a single disk)
--for random workload:
N times R (... random

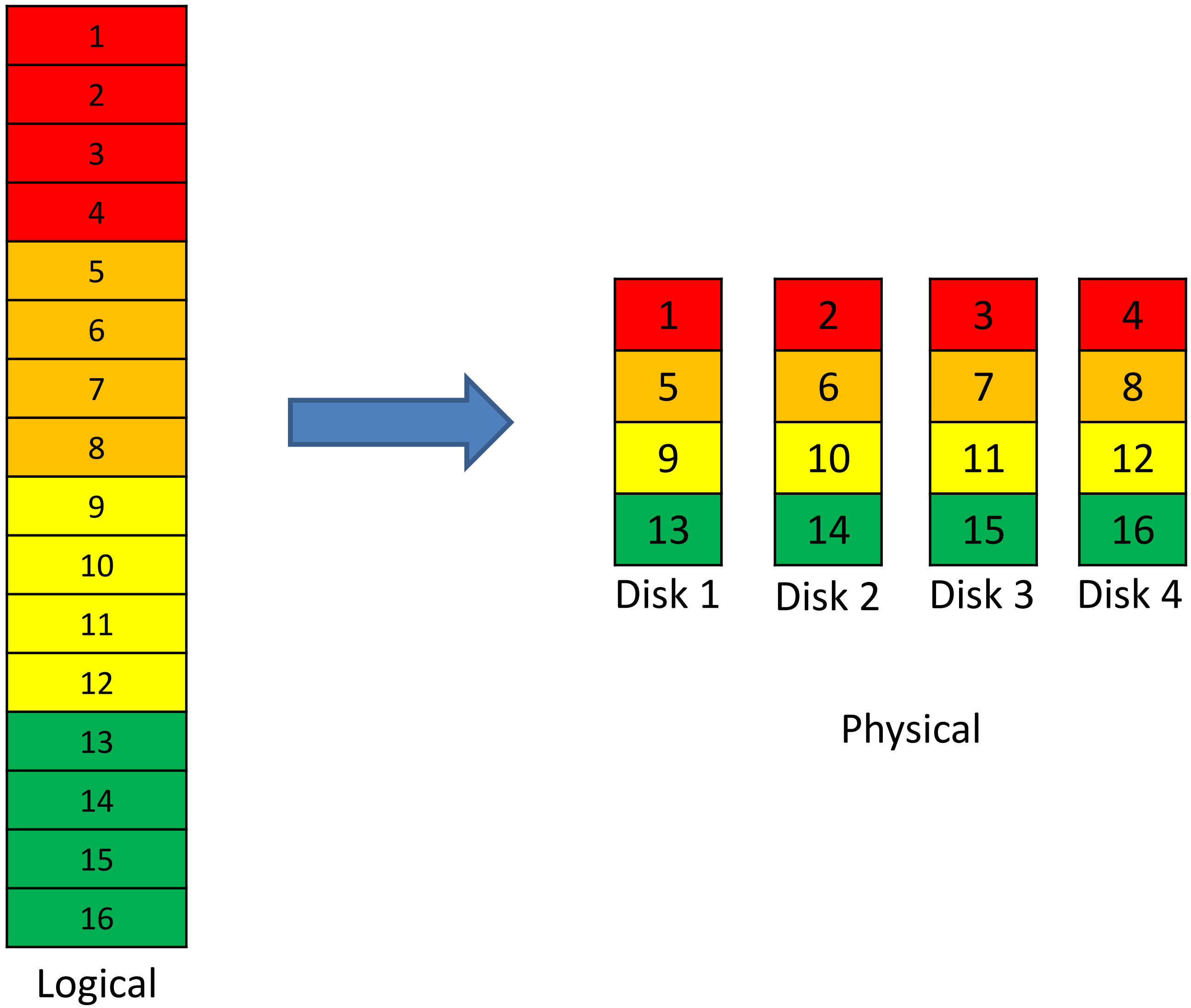
Redundancy through mirroring

- **Mirroring:** Duplicate data on every disk
  - One logical disk consists of two physical disk
  - Every write carried out on both disks
  - If one of the disk fails, data read from the other
  - Data permanently lost only if the second disk fails before the first failed disk is replaced.



Capacity: 0.5 N.BReliability: ca

Logical to physical mapping (Striping)

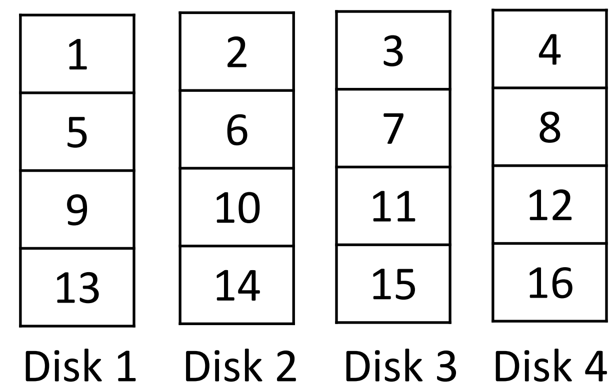


RAID Levels

- Data are distributed across the array of disk drives
- Redundant disk capacity is used to store parity information, which guarantees data recoverability in case of a disk failure
- Levels decided according to schemes to provide redundancy at lower cost by using striping and "parity" bits
- Different cost-performance trade-offs

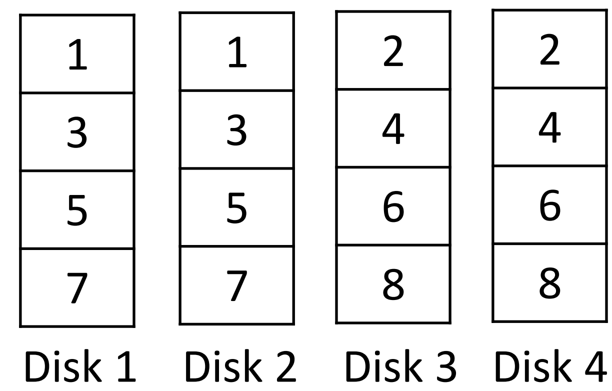
RAID 0

- Striping at the level of blocks
- Data split across drives resulting in higher data throughput
- Pros
  - Performance is very good
- Cons
  - Failure of any disk in the array results in data loss
  - Reliability problems due to no mirroring or parity bits
- Commonly referred to as **striping**



RAID 1

- Introduce redundancy through mirroring
- Pros
  - No data loss if either drive fails
  - Good read performance
- Cons
  - Expensive: Cost/MB is high
- Commonly referred to as **mirroring**



RAID 2

- Uses Hamming (or any other) error-correcting code (ECC)
- Intended for use in drives which do not have built-in error detection
- **Key Idea:** If one of the disks fail the remaining bits of the byte and the associated ECC bits can be used to reconstruct the data
- Not very popular

RAID 3

- Improves upon RAID 2, known as **Bit-Interleaved Parity**
- Pros:
  - Disk Controllers can detect whether a sector has been read correctly.
  - Storage overhead is reduced – only 1 parity disk needed
- Cons:
  - Expense of computing and writing parity
  - Need to include a dedicated parity hardware

RAID 4

- Stripes data at a block level across several drives, with parity stored on one drive - **block-interleaved parity**
- Pros
  - Allows recovery from the failure of any of the disks
  - Performance is very good for reads
- Cons
  - Writes require that parity data be updated each time.
  - Slows small random writes but large writes are fairly fast

RAID 4

Disk 0	Disk 1	Disk 2	Disk 3	Disk 4
0	1	2	3	P0
4	5	6	7	P1
8	9	10	11	P2
12	13	14	15	P3

Figure 38.4: **RAID-4 With Parity**

C0	C1	C2	C3	P
0	0	1	1	XOR(0,0,1,1) = 0
0	1	0	0	XOR(0,1,0,0) = 1

Block0	Block1	Block2	Block3	Parity
00	10	11	10	11
10	01	00	01	10

Capacity: (N - 1).B
Reliability: Tolerate 1 disk failure and not more
Performance:

steady state throughput:
(N - 1). S (seq read)
(N - 1) . S (seq. writes, full stripe writes)
(N - 1) .R (random read)

random writes:
additive parity calculation needs to read all blocks in a stripe, subtractive parity
P_new = (C_old XOR C_new) XOR P_old

Writing to parity disk occurs with every write. So independent disks still create a bottleneck for each other.
R/2

RAID 5

- Addresses small writes problem of RAID 4
- Rotates parity block across drives

Disk 0	Disk 1	Disk 2	Disk 3	Disk 4
0	1	2	3	P0
5	6	7	P1	4
10	11	P2	8	9
15	P3	12	13	14
P4	16	17	18	19

Figure 38.7: **RAID-5 With Rotated Parity**

RAID Capacity, Reliability, Performance

- N disks, B blocks
- Sequential workload (S MB/s)
- Random workload (R MB/s)

	RAID-0	RAID-1	RAID-4	RAID-5
Capacity	$N \cdot B$	$(N \cdot B)/2$	$(N-1) \cdot B$	$(N-1) \cdot B$
Reliability	0	1 (for sure) $\frac{N}{2}$ (if lucky)	1	1
Throughput				
Sequential Read	$N \cdot S$	$(N/2) \cdot S$	$(N-1) \cdot S$	$(N-1) \cdot S$
Sequential Write	$N \cdot S$	$(N/2) \cdot S$	$(N-1) \cdot S$	$(N-1) \cdot S$
Random Read	$N \cdot R$	$N \cdot R$	$(N-1) \cdot R$	$N \cdot R$
Random Write	$N \cdot R$	$(N/2) \cdot R$	$\frac{1}{2} \cdot R$	$\frac{N}{4} R$
Latency				
Read	T	T	T	T
Write	T	T	$2T$	$2T$

Figure 38.8: **RAID Capacity, Reliability, and Performance**

Summary

- RAID transforms a number of independent disks into a large, more capacious, and more reliable single entity
  – Hardware and software are relatively oblivious to the change.
- There are many possible RAID levels
  – The exact RAID level to use depends heavily on what is important to the end-user.
  – Mirrored RAID is simple, reliable, and generally provides good performance but at a high capacity cost
  – RAID-5, in contrast, is reliable and better from a capacity standpoint, but performs quite poorly when there are small writes in the workload.
- Picking a RAID and setting its parameters (chunk size, number of disks, etc.) properly for a particular workload is challenging