

# **CSE 451: Operating Systems**

## **Spring 2017**

### **Module 18**

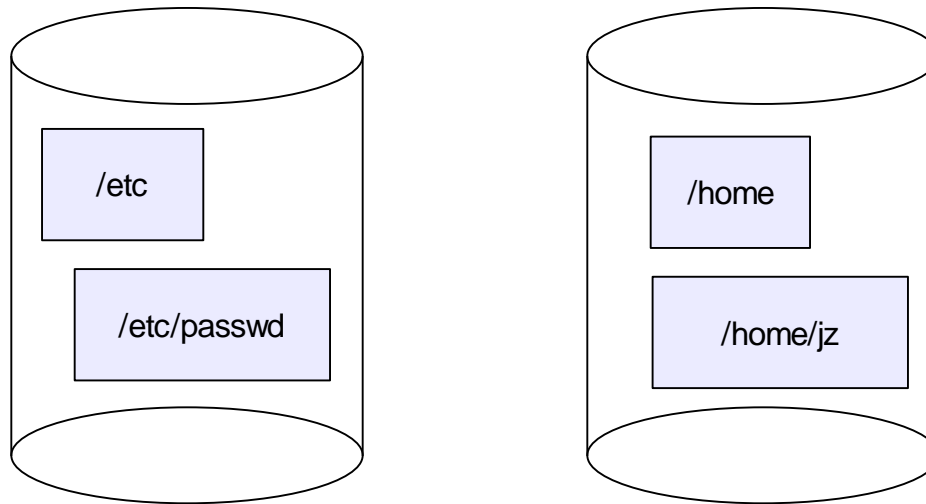
### **Redundant Arrays of Inexpensive Disks**

### **(RAID)**

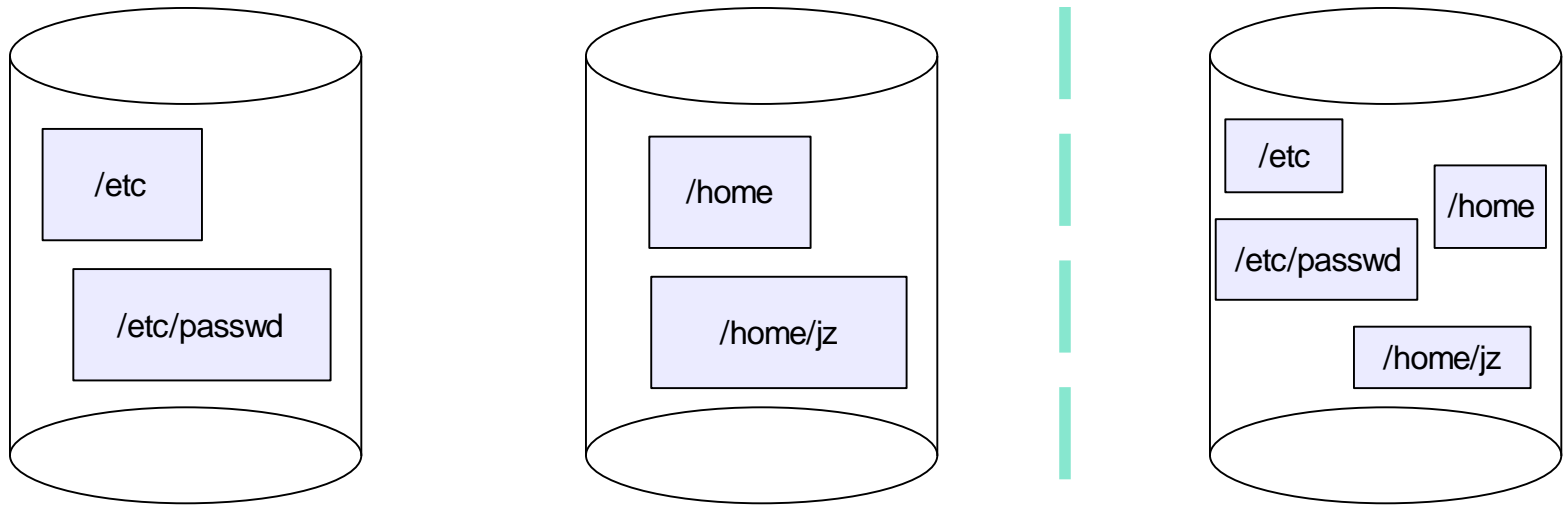
**John Zahorjan**

# Background

- Disks are cheap
  - An individual system can have more than one
- Standard file system implementations manage all of an individual disk/partition



# Two disks vs. one



- How is “peak performance” affected?
  - Are read times cut in half? Is write throughput doubled?
- How is reliability affected?
  - Is it more or less likely a disk failure will cause data loss?
  - How much of your data do you lose?

# Basic Idea

- By *striping* individual files across multiple disks, we can use parallel I/O to improve access time even when overall I/O demand is bursty/low
  - There's only one file to read right now. Get it fast.
- But, striping reduces reliability
  - 10 disks have about 1/10th the MTBF (mean time between failures) of one disk
- So, we want striping for performance, but we need something to help with reliability

# Reliability through Redundancy

- To achieve this level of reliability, add **redundant data** that allows a disk failure to be tolerated
  - We'll see how in a minute
- At the scales we're currently considering (tens of disks), it's typically enough to be resilient to the failure of a single disk
  - What are the chances that a second disk will fail before you've replaced the first one?
    - Er, it has happened to us!
- So:
  - Obtain performance from striping
  - Obtain reliability from redundancy (which reduces useful total data capacity **and** steals back some of the performance gain)

# RAID

- A RAID is a Redundant Array of Inexpensive Disks
- Disks are small and cheap, so it's easy to put lots of disks (10s, say) in one box for increased storage, performance, and availability
- Data plus some redundant information is striped across the disks in some way
- How striping is done is key to performance and reliability

# RAID Implementation

- Option A: hardware
  - The hardware RAID controller deals with this
    - From the OS's perspective, the multi-disk RAID looks like one big array of blocks
- Option 2: software
  - A low level layer of the OS knows there are multiple disks, but presents them to upper layers as a single block device
    - That is, it does what the hw RAID controller does
- What's good about each?
  - What's bad?
- It doesn't matter to what follows which approach is used

# Some RAID tradeoffs

- Granularity

- fine-grained: stripe each file over all disks
  - high throughput for the file
  - limits transfer to 1 file at a time
- course-grained: stripe each file over only a few disks
  - limits throughput for 1 file
  - allows concurrent access to multiple files

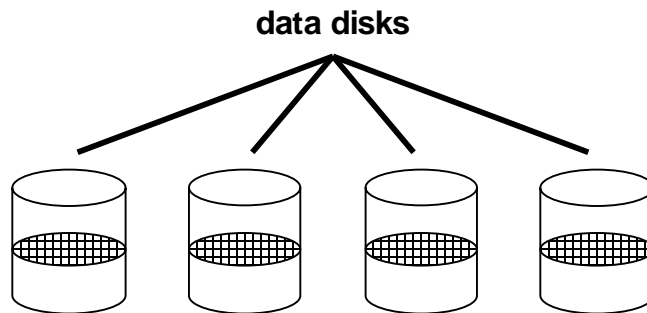
- Redundancy

- uniformly distribute redundancy information on disks
  - avoids load-balancing problems
- concentrate redundancy information on a small number of disks
  - partition the disks into data disks and redundancy disks



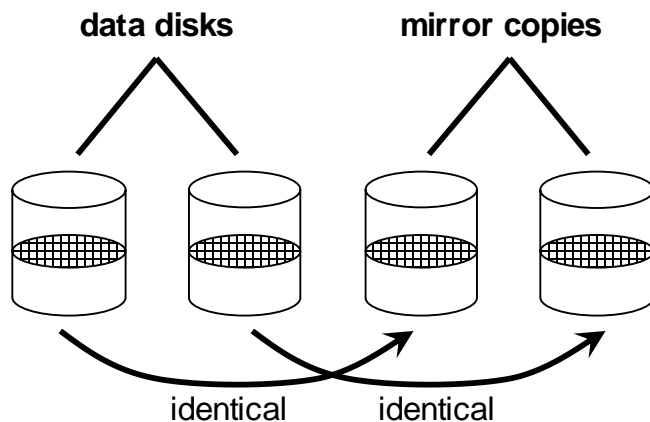
# RAID Level 0: Non-Redundant Striping

- RAID Level 0 is a non-redundant disk array
- Files/blocks are striped across disks, no redundant info
- High (single-file) read throughput
- Best write throughput (no redundant info to write)
- Any disk failure results in data loss

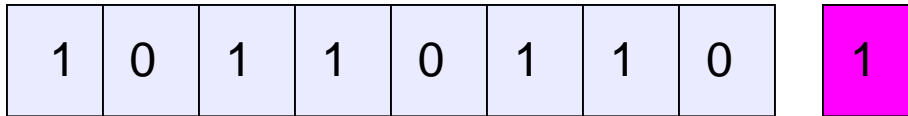


# RAID Level 1: Mirrored Disks

- Files are striped across half the disks, and mirrored to the other half
  - 2x space expansion
- Reads: Read from either copy
- Writes: Write both copies
- On single drive failure, just use the surviving disk
- If two disks fail, you rely on luck...



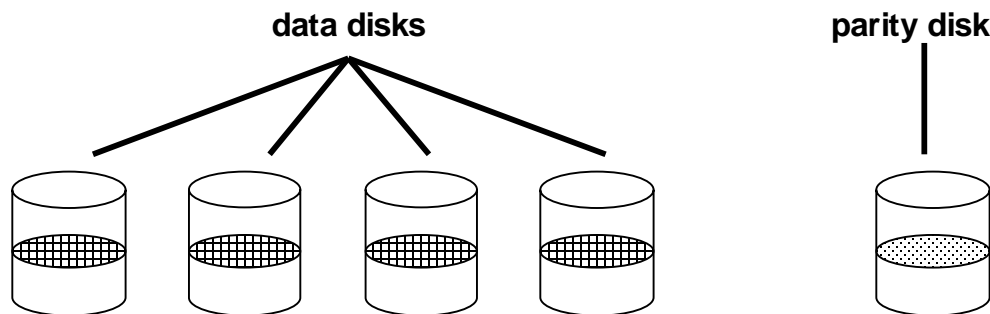
# Prelude to RAID Levels 2-5: A parity refresher



- To each byte, add a bit whose value is set so that the total number of 1's is even
- Any single missing bit can be reconstructed
- More sophisticated schemes, called ECC (error correcting codes), can correct multiple bit errors
  - e.g., based on Hamming codes

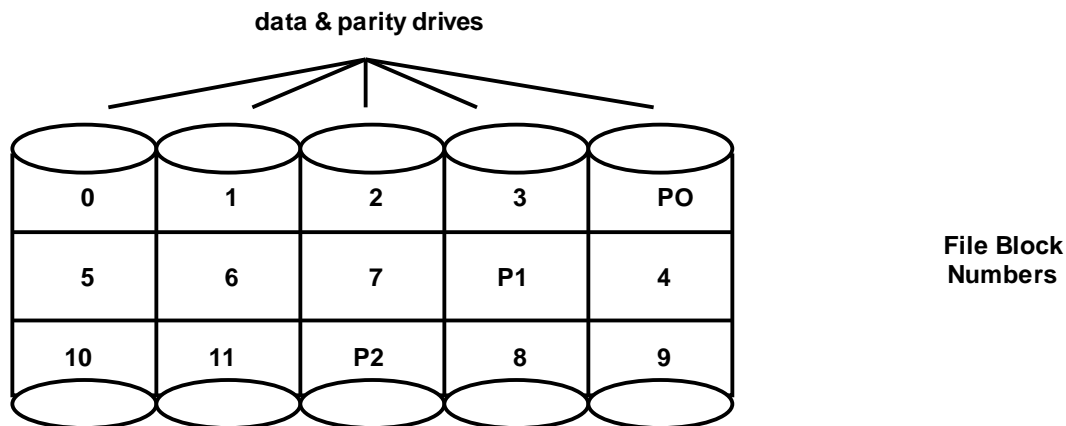
# RAID Levels 2, 3, and 4: Striping + Parity Disk

- RAID levels 2, 3, and 4 use parity or ECC disks
  - e.g., each byte on the parity disk is a parity function of the corresponding bytes on all the other disks
  - details between the different levels have to do with kind of ECC used, and whether it is bit-level, byte-level, or block-level
- A read accesses all the data disks, a write accesses all the data disks plus the parity disk
- On disk failure, read the remaining disks plus the parity disk to compute the missing data



# RAID Level 5

- RAID Level 5 uses block interleaved distributed parity
- Like parity scheme, but distribute the parity info (as well as data) over all disks
  - for each block, one disk holds the parity, and the other disks hold the data
- Significantly better performance
  - parity disk is not a hot spot



# RAID Level 6

- Basically like RAID 5 but with replicated parity blocks so that it can survive two disk failures.
- Useful for larger disk arrays where multiple failures are more likely.

# Example RAID Storage



Promise 3U rack-mountable 16-disk RAID Storage System

Hot swappable drives

Dual controllers with 4 host interface ports for reliability

Can be ganged together into larger units

# Example RAID Storage



13 hot-swappable drives!