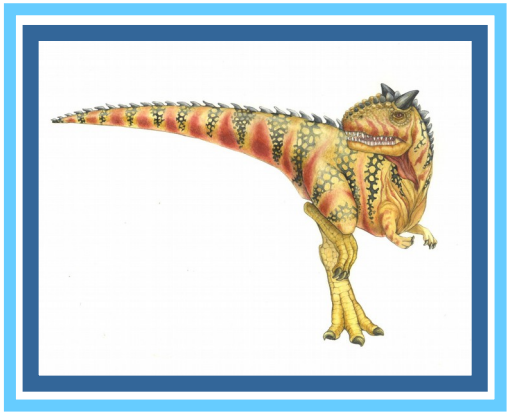
Chapter 11: Mass-Storage Systems



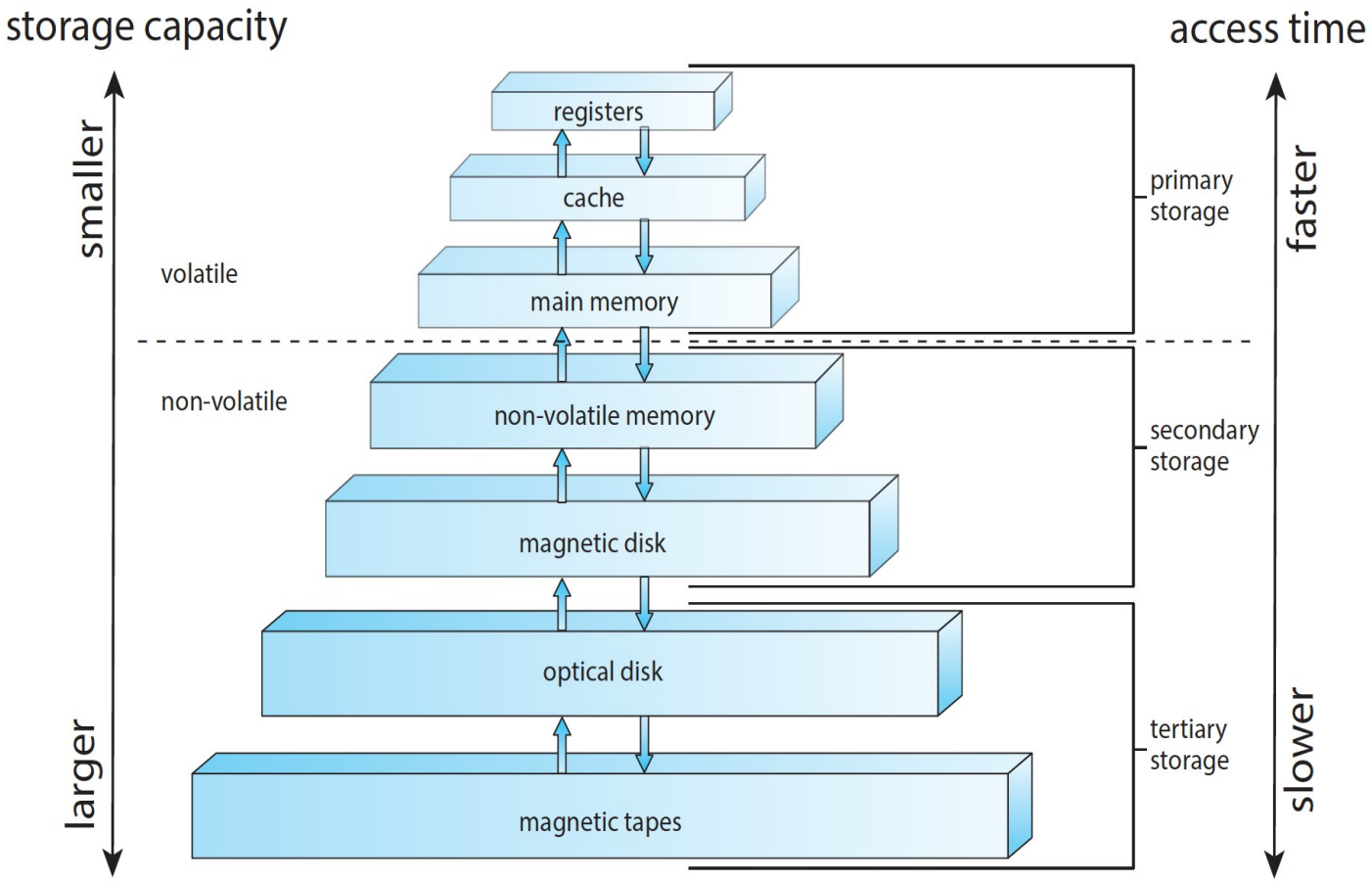
Chapter 11: Mass-Storage Systems

- Overview of Mass Storage Structure
- Storage Attachment
- Secondary Storage I/O Scheduling
- Storage Device Management
- Swap-Space Management
- RAID Structure

Objective

- To describe the physical structure of secondary storage devices and its effects on the uses of the devices
- To explain the performance characteristics of mass-storage devices
- To evaluate I/O scheduling algorithms
- To discuss operating-system services provided for mass storage, including RAID
- To describe the structure of NVM and its effects on the uses of the devices

Storage-Device Hierarchy

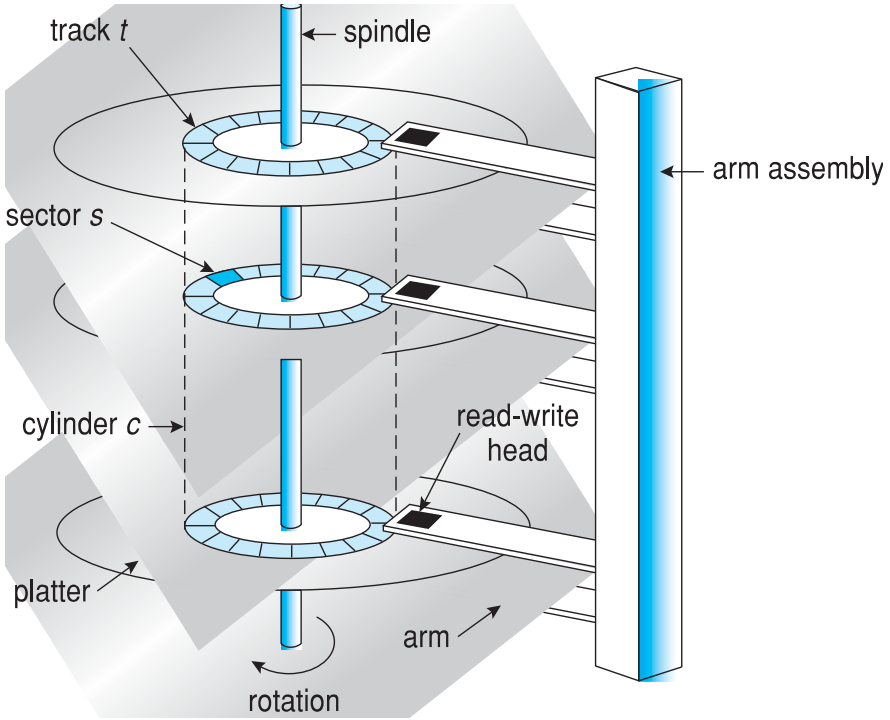


Overview of Mass-Storage Structure

- The bulk of secondary storage for modern computer are
  - Hard disk drives (HDD)
  - Nonvolatile Memory (NVM)
- The most common tertiary storage is magnetic tape.
- In this chapter we describe:
  - The basic mechanism of those devices
  - How operating systems translate their physical properties to logical storage via address mapping.

Hard Disk Drive Moving-head Mechanism

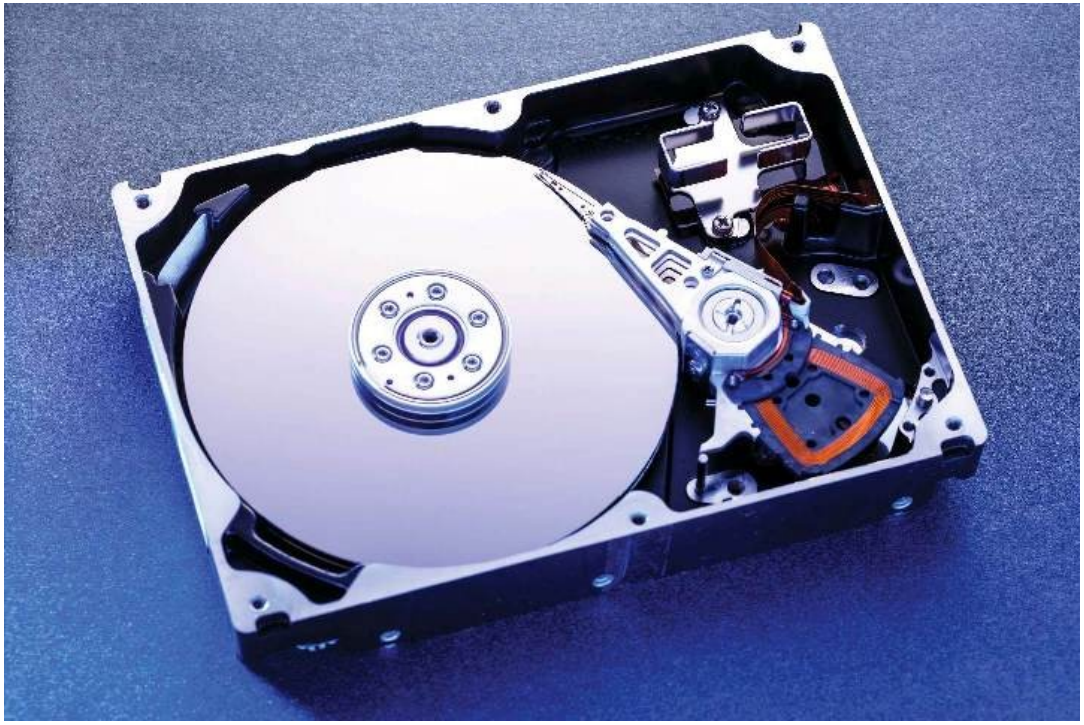
- Platters range from .85” to 14” (historically)
  - Commonly 3.5”, 2.5”, and 1.8”
- Range from gigabytes through terabytes per drive



Hard Disk Drives

- HDDs rotate at 60 to 250 times per second
- **Transfer rate** is rate at which data flow between drive and computer
- **Positioning time** (**random-access time**) is time to move disk arm to desired cylinder (**seek time**) and time for desired sector to rotate under the disk head (**rotational latency**)
- **Head crash** results from disk head making contact with the disk surface
  - That's bad
- Disks can be removable
- Other types of storage media include CDs, DVDs, Blu-ray discs. magnetic tape

A 3.5" HDD with Cover Removed.



The First Commercial Disk Drive

1956
IBM RAMDAC computer included the IBM Model 350 disk storage system

5 million (7 bit) characters
50 x 24” platters
Access time = < 1 second

Performance of Magnetic Disks

- Performance
  - Transfer Rate – theoretical – 6 Gb/sec
  - Effective Transfer Rate – real – 1Gb/sec
  - Seek time from 3ms to 12ms – 9ms common for desktop drives
  - Average seek time measured or calculated based on 1/3 of tracks
  - Latency based on spindle speed
    - 1 / (RPM / 60) = 60 / RPM
  - Average latency = ½ latency
- From Wikipedia

Spindle [rpm]	Average latency [ms]
4200	7.14
5400	5.56
7200	4.17
10000	3
15000	2

Performance of Magnetic Disks (Cont.)

- **Access Latency** = **Average access time** = average seek time + average latency
  - For fastest disk 3ms + 2ms = 5ms
  - For slow disk 9ms + 5.56ms = 14.56ms
- **Average I/O time** = average access time + (amount to transfer / transfer rate) + controller overhead
- For example to transfer a 4KB block on a 7200 RPM disk with a 5ms average seek time, 1Gb/sec transfer rate with a .1ms controller overhead =
  - 5ms + 4.17ms + 0.1ms + transfer time
  - Transfer time = 4KB / 1Gb/s * 8Gb / GB * 1GB / 1024^2KB = 32 / (1024^2) = 0.031 ms
  - Average I/O time for 4KB block = 9.27ms + .031ms = 9.301ms

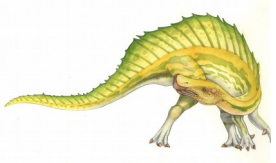
Disk Structure

- Disk drives are addressed as large 1-dimensional arrays of **logical blocks**, where the logical block is the smallest unit of transfer
  - Low-level formatting creates **logical blocks** on physical media
- The 1-dimensional array of logical blocks is mapped into the sectors of the disk sequentially
  - Sector 0 is the first sector of the first track on the outermost cylinder
  - Mapping proceeds in order through that track, then the rest of the tracks in that cylinder, and then through the rest of the cylinders from outermost to innermost
  - Logical to physical address should be easy
    - Except for bad sectors
    - Non-constant # of sectors per track via constant angular velocity
  - Each sector 512B or 4KB – smallest I/O the drive can do

Nonvolatile Memory (NVM)

- **NVM** is electrical vs. mechanical (HDD)
- Commonly composed of a controller and flash NAND die semiconductor chips (A **die** is a small block of semiconducting material, on which a given functional circuit is fabricated
  - Other forms include DRAM with battery backup, non-NAND die like 3D XPoint
- Flash-memory-based NVM is frequently used in disk-drive like container -> **solid-state disk (SSD)**

- Also can be in other formats like **USB drive**



NVM (Cont.)

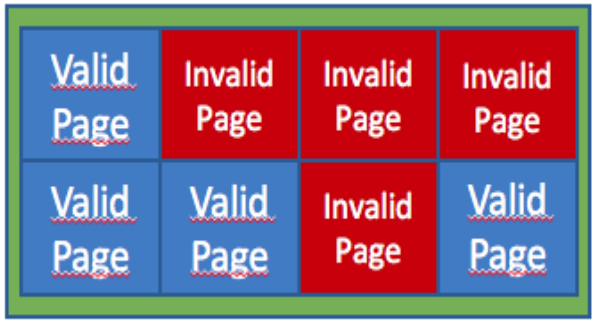
- In all forms, acts and treated the same way
- More reliable than HDD (no moving parts), can be faster (no seek time or latency), consumes less power
- More expensive per MB, lower capacity, may have shorter lifespan (writes wear it out)
- Higher speed means new connection methods
  - Direct to PCIe bus (called NVMe)
- Can be used in variety of ways – replacement for disk, caching tier, etc

NVM Details

- Read and written in “page” increment
  - Page size varies based on device
- Cannot overwrite data, must erase it first
- Erase occurs in ”block” increment composed of several pages
- Erase operation takes much longer than read or write
- NAND wears out a bit with every erase
  - ~100,000 program-erase cycles until cells no longer retain data

NVM Device Controller

- Several algorithms, usually implemented in NVM device controller
  - So operating system blissfully just reads and writes blocks and device deals with the physics
  - But can impact performance, so worth knowing about
- NAND block with valid and invalid pages



NVM Controller Algorithms

- NAND cannot be overwritten, therefore:
  - There are usually pages containing invalid data.
  - To track which logical blocks contain the valid data, the controller maintains a Flash Translation Layer (FTL).
  - This table maintains the mapping of which physical page contains the currently valid logical block.
  - This table It is also used to track physical block state (does the block only contain invalid pages and therefore can be erased).
  - Each logical block can have many versions, each stored in a physical page, one valid the others invalid.

Full SSD Controller Algorithms

- Full SSD (all pages written to, some hold valid data and others invalid)
  - Logically space available, physically no where to write data
  - **Garbage collection** copies good data from blocks with mix of valid and invalid pages to other locations, freeing up blocks to erase
    - But if device full, no where to copy good pages
    - **Over-provisioning** (20% of media set aside) as target for GC writes
    - As blocks invalid or made invalid by GC are erased, placed into overprovision pool if device full or returned to free pool

NVM Controller Algorithms

- Overprovision helps with wear leveling
  - Want media to wear out at the same time, not have a hot spot (repeated writes) wear out early
  - Track # of writes per block, GC and over provisioning space used to write to less written to blocks
- Data protected with NVM via ECC
  - Too many errors and mark block as bad to not use it
  - Uncorrectable needs to be recovered via RAID

Magnetic Tape

- Was early secondary-storage medium
  - Evolved from open spools to cartridges
- Relatively permanent and holds large quantities of data
- Access time slow
- Random access ~1000 times slower than disk
- Mainly used for backup, storage of infrequently-used data, transfer medium between systems

Magnetic Tape

- Kept in spool and wound or rewound past read-write head
- Once data under head, transfer rates comparable to disk
  - 140MB/sec and greater
- GB to TB typical storage
- Common technologies are LTO-{5,6}, SDLT, 4, 8, and 19mm, and ¼ and ½" widths

Secondary Storage Connection Methods

- Host-attached storage accessed through I/O ports talking to I/O busses
- Several bus technologies including **advanced technology attachment** (ATA), **serial ATA** (SATA), **eSATA**, **universal serial bus** (USB), **fibre channel** (FC), **serial attached SCSI** (SAS)
- Data transfers are carried out by special electronic processors: **controllers**
  - Host controller is in computer, device controller built into storage device
  - Talk to each other, usually via memory-mapped I/O ports
- Device controllers have built in caches
  - Data transfer from media into cache, then over bus to host (and vice versa)

Address Mapping

- Storage devices addressed as one-dimensional array of logical blocks
  - Logical block is smallest unit of transfer
  - Each logical block maps to physical sectors or pages on device
    - For example, logical block 0 might be sector 0 on cylinder 0 on platter 0 of an HDD
  - Easier to use logical address <0> - <N> than address containing <sector, cylinder, head> or <chip, block, page>
  - Also, defective sectors / blocks can be mapped out of use by logical to physical mapping and logical addresses still sequential
  - Number of sectors per track is not constant on some drives, also requiring logical addressing to hide complexity

Address Mapping

- Some media / devices use **constant linear velocity** (CLV) (CD, DVD)
  - Density of bits per track uniform
  - Farther a track from the center of the disk -> greater its length -> more sectors
  - Drive increases rotational speed as head moves outward to keep same rate of data under the read/write head
- Alternatively disk rotation speed can stay constant – **constant angular velocity** (CAV) (HDD)
  - Density of bits decreases from inner to outer tracks
  - (Blu-ray drives can do both CAV and CLV depending on media etc.)

Disk Scheduling

- The operating system is responsible for using hardware efficiently — for the disk drives, this means having a fast access time and disk bandwidth
- Minimize seek time
- Seek time ≈ seek distance
- Disk **bandwidth** is the total number of bytes transferred, divided by the total time between the first request for service and the completion of the last transfer

Disk Scheduling (Cont.)

- There are many sources of disk I/O request
  - OS
  - System processes
  - Users processes
- I/O request includes input or output mode, disk address, memory address, number of sectors to transfer
- OS maintains queue of requests, per disk or device
- Idle disk can immediately work on I/O request, busy disk means work must be queued.
  - Optimization algorithms only make sense when a queue exists

Disk Scheduling (Cont.)

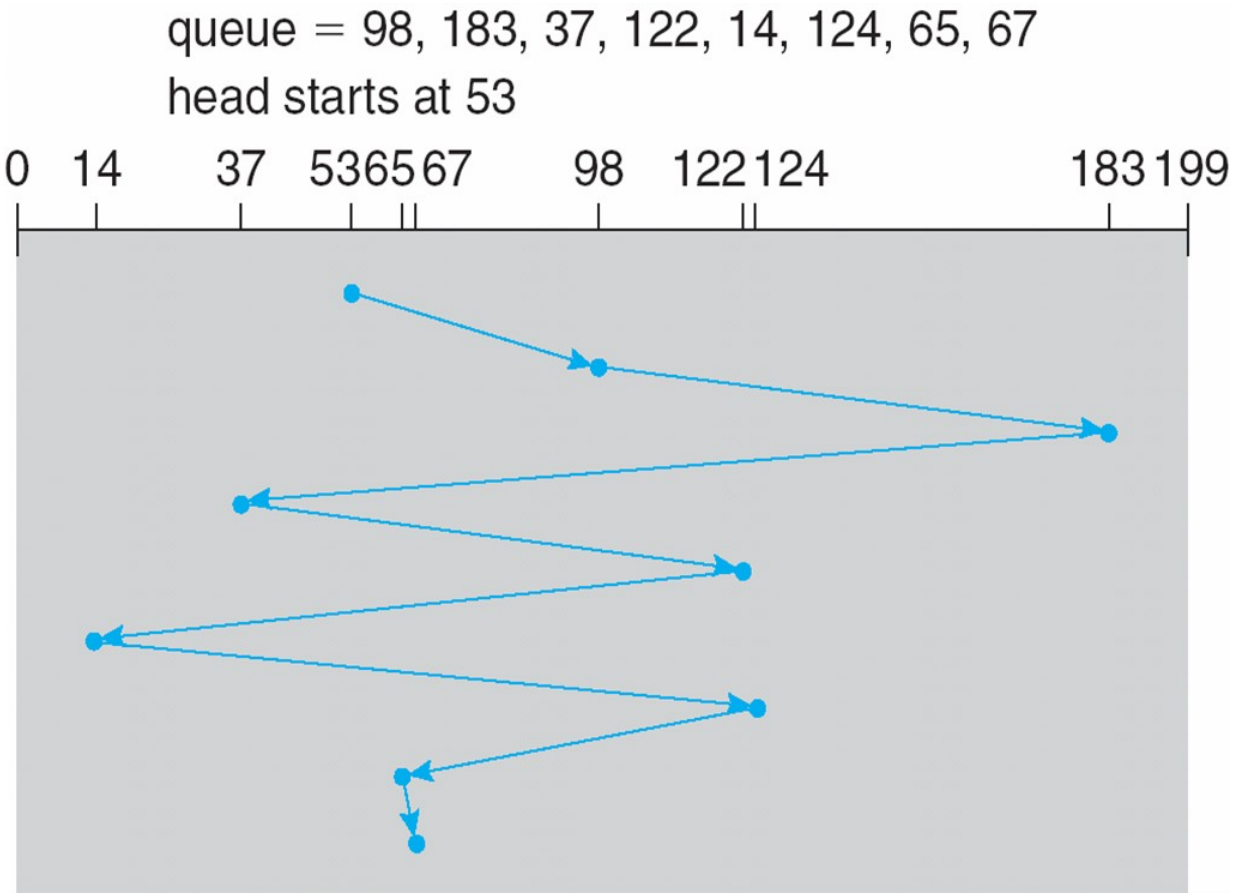
- Note that drive controllers have small buffers and can manage a queue of I/O requests (of varying “depth”)
- Several algorithms exist to schedule the servicing of disk I/O requests
- The analysis is true for one or many platters
- We illustrate scheduling algorithms with a request queue (0-199)

98, 183, 37, 122, 14, 124, 65, 67

Head pointer 53

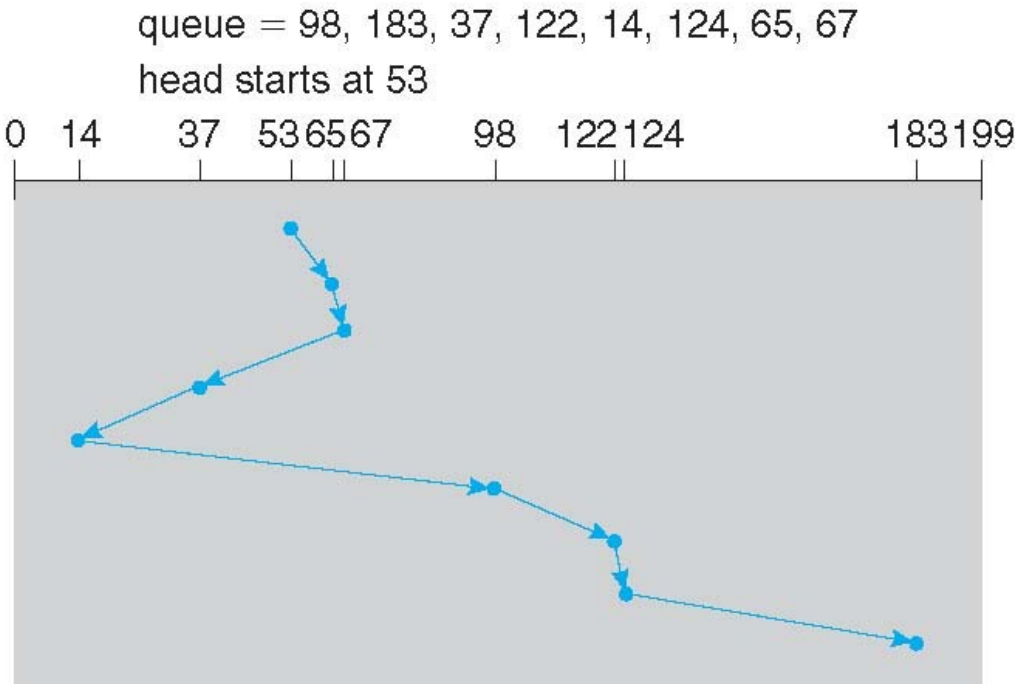
FCFS

Illustration shows total head movement of 640 cylinders



SSTF

- Shortest Seek Time First -- selects the request with the minimum seek time from the current head position
- SSTF scheduling is a form of SJF scheduling; may cause starvation of some requests
- Illustration shows total head movement of 236 cylinders



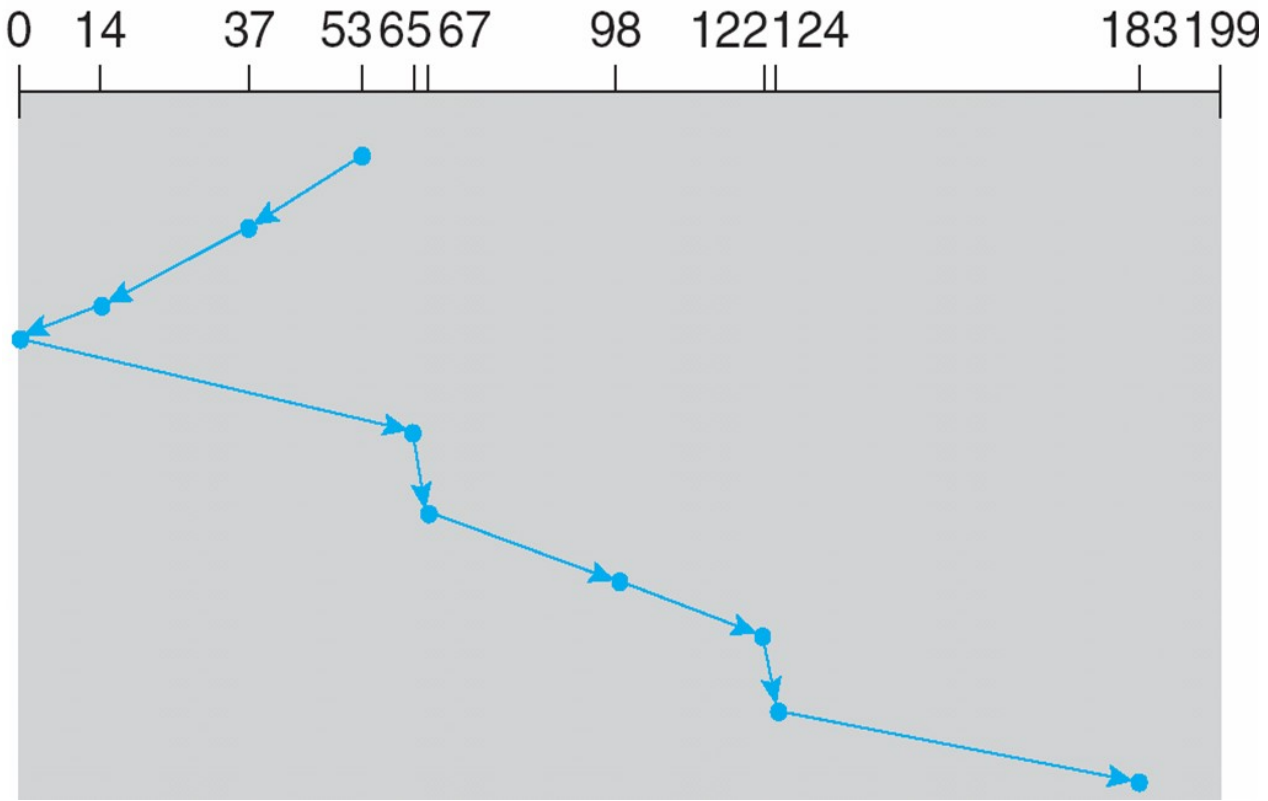
SCAN

- **SCAN algorithm**. The disk arm starts at one end of the disk, and moves toward the other end, servicing requests until it gets to the other end of the disk, where the head movement is reversed and servicing continues.
- Sometimes it is called the **elevator algorithm**
- Illustration shows total head movement of 208 cylinders
- But note that if requests are uniformly dense, largest density at other end of disk and those wait the longest

SCAN (Cont.)

queue = 98, 183, 37, 122, 14, 124, 65, 67

head starts at 53



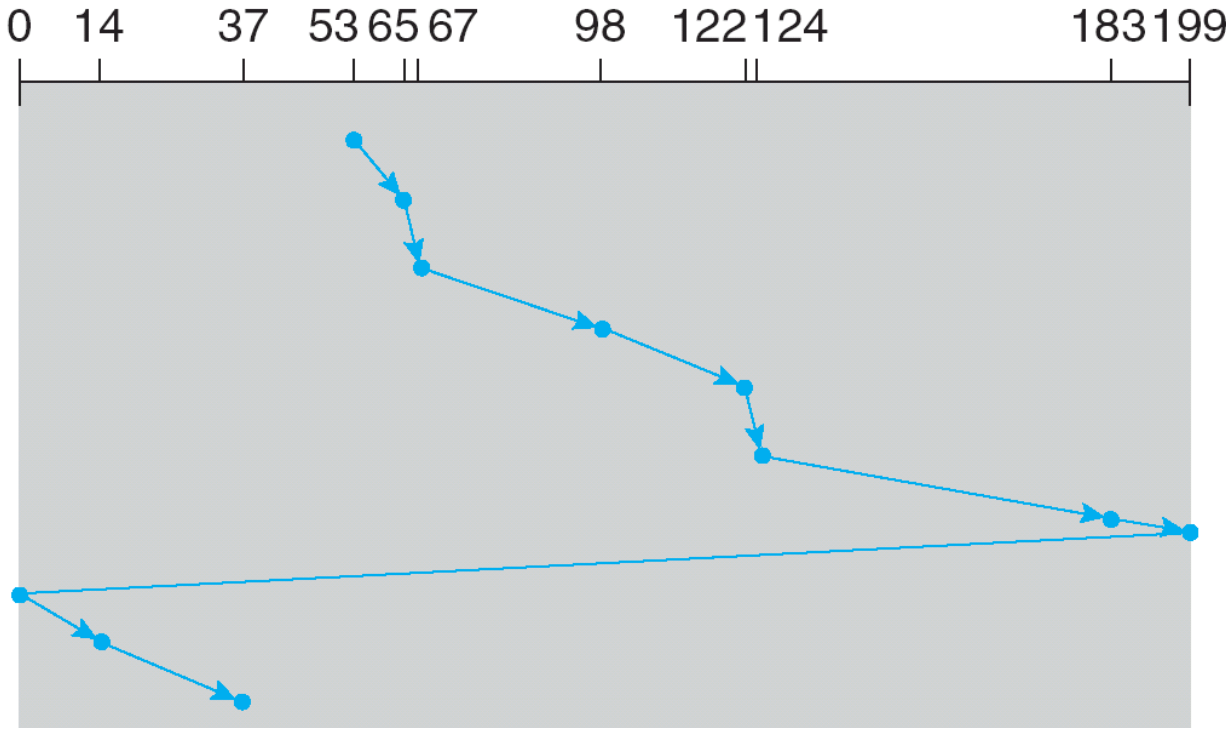
C-SCAN

- Provides a more uniform wait time than SCAN
- The head moves from one end of the disk to the other, servicing requests as it goes
  - When it reaches the other end, however, it immediately returns to the beginning of the disk, without servicing any requests on the return trip
- Treats the cylinders as a circular list that wraps around from the last cylinder to the first one

C-SCAN (Cont.)

queue = 98, 183, 37, 122, 14, 124, 65, 67

head starts at 53



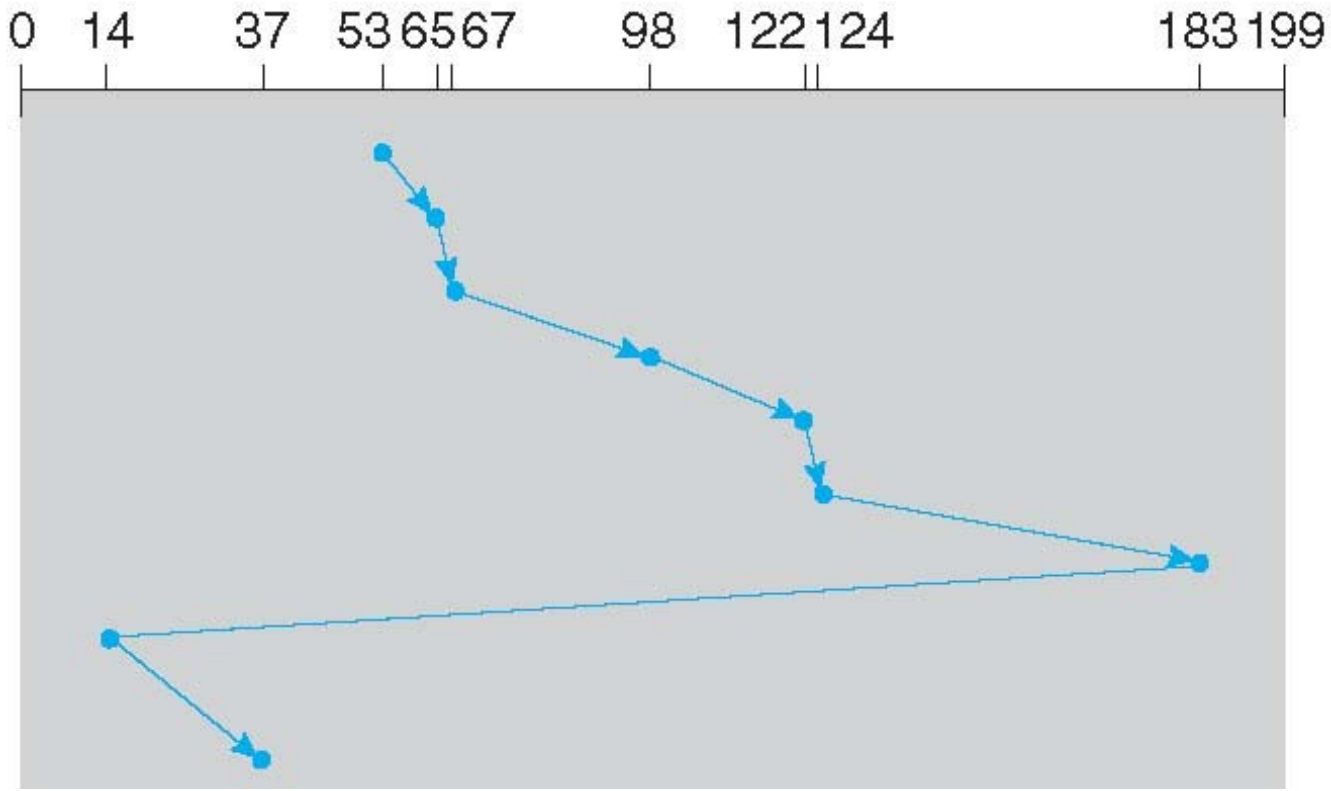
LOOK and C-LOOK

- **LOOK** is a version of SCAN. Arm only goes as far as the last request in each direction, then reverses direction immediately, without first going all the way to the end of the disk
- **C-LOOK** a version of C-SCAN. Arm only goes as far as the last request in one direction, then reverses direction immediately, without first going all the way to the end of the disk

C-LOOK (Cont.)

queue = 98, 183, 37, 122, 14, 124, 65, 67

head starts at 53



Selecting a Disk-Scheduling Algorithm

- SSTF is common and has a natural appeal
- SCAN and C-SCAN perform better for systems that place a heavy load on the disk
  - Less starvation
- Performance depends on the number and types of requests
- Requests for disk service can be influenced by the file-allocation method
  - And metadata layout

Disk-Scheduling Algorithm (Cont.)

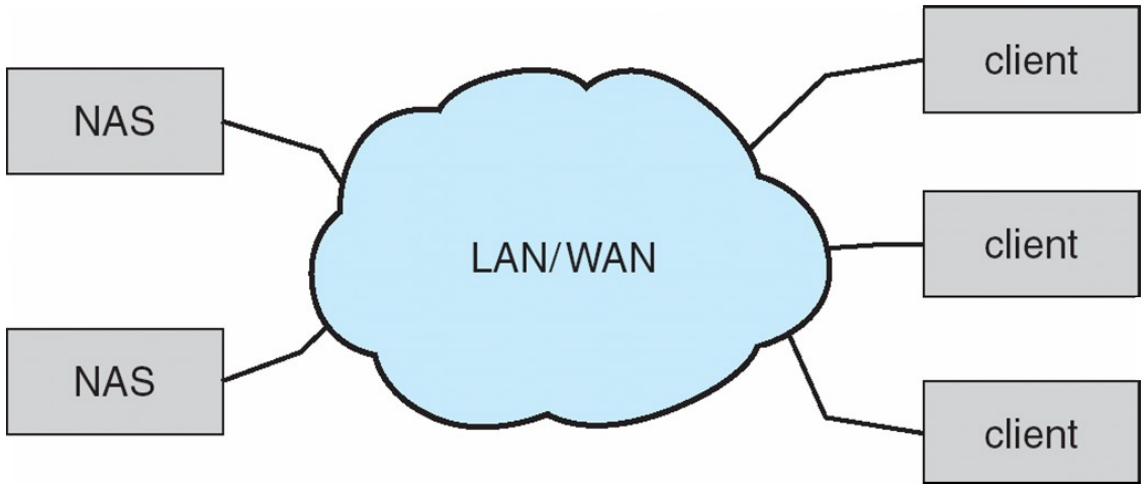
- The disk-scheduling algorithm should be written as a separate module of the operating system, allowing it to be replaced with a different algorithm if necessary
- Either SSTF or LOOK is a reasonable choice for the default algorithm
- What about rotational latency?
  - Difficult for OS to calculate
- How does disk-based queuing effect OS queue ordering efforts?

Network-Attached Storage

- Network-attached storage (**NAS**) is storage made available over a network rather than over a local connection (such as a bus)
  - Remotely attaching to file systems
  - Manages storage, provides data management, has network interfaces, and talks network storage protocols
- NFS and CIFS are common protocols
- Implemented via remote procedure calls (RPCs) between host and storage over typically TCP or UDP on IP network

Network-Attached Storage (Cont.)

- **iSCSI** protocol uses IP network to carry the SCSI protocol
  - Remotely attaching to devices (blocks)

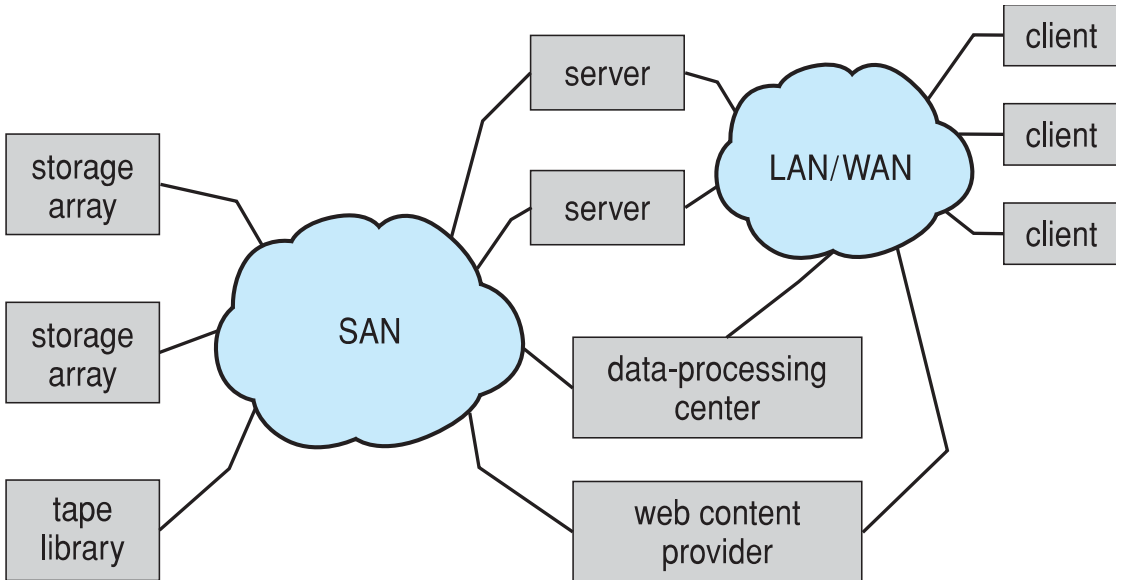


Cloud Storage

- Similar to NAS
  - Provides storage across the network
  - Usually not owned by the company or user, but provided for a fee (based on time, storage capacity used, I/O done, etc.)
  - But across a WAN rather than a LAN
  - Frequently too slow, more prone to connection interruption than NAS so CIFS, NFS, iSCSI possibly but less used
  - Frequently use their own APIs, and apps that use those APIs to do I/O
    - Dropbox, Microsoft OneDrive, Apple iCloud, etc.

Storage Area Network

- Common in large storage environments
- Multiple hosts attached to multiple storage arrays - flexible



Reliability and Redundancy

- **Mean time to failure**. The average time it takes a disk to fail.
- **Mean time to repair.** The time it takes (on average) to replace a failed disk and restore the data on it.
- **Mirroring.** Copy of a disk is duplicated on another disk.
  - Consider disk with 100,000 hours of mean time to failure and 10 hour mean time to repair
    - Mean time to data loss is $100{,}000^2 / (2 * 10) = 500 * 10^6$ hours, or 57,000 years If mirrored disks fail independently,
- Several improvements in disk-use techniques involve the use of multiple disks working cooperatively

RAID Structure

- RAID – redundant array of inexpensive disks
- Use of many disks Increases the **mean time to failure.**
  - 100 disks with MTF of 100,000 hours.
  - 100,000/100 = 1,000 hours or 41.66 days.
- Solution is to have data redundancy over the 100 disks.
- **Disk striping.** Splitting the bits (or blocks) across multiple disks
  - **bit-level striping**. The bits of a byte are split across multiple disks.
  - **block-level striping**. The blocks of a file byte are split across multiple disks.
- For example, if we have an array of eight disks, we write bit "*i*" of each byte to disk "*i*". The array of eight disks can be treated as a single disk with sectors that are eight times the normal size and, more important, that have eight times the access rate.

RAID Structure (Cont.)

- RAID schemes improve performance and improve the reliability of the storage system by storing redundant data
- RAID within a storage array can still fail if the array fails:
  - Replication of the data between arrays is common -- automatic duplication of writes between separate sites
    - For redundancy and disaster recovery
    - Can be synchronous or asynchronous
  - Frequently, a small number of hot-spare disks are left unallocated, automatically replacing a failed disk and having data rebuilt onto them
  - Hot spare disk is not used for storing data. It is configured to be used as a replacement in case of a disk failure.
- Snapshot is a view of file system before a set of changes take place (i.e., at a point in time). More in Ch 12

RAID Levels

(a) RAID 0: non-redundant striping.

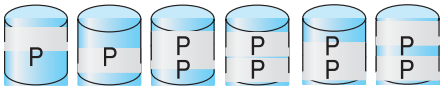
(b) RAID 1: mirrored disks.

(c) RAID 2: memory-style error-correcting codes.

(d) RAID 3: bit-interleaved parity.

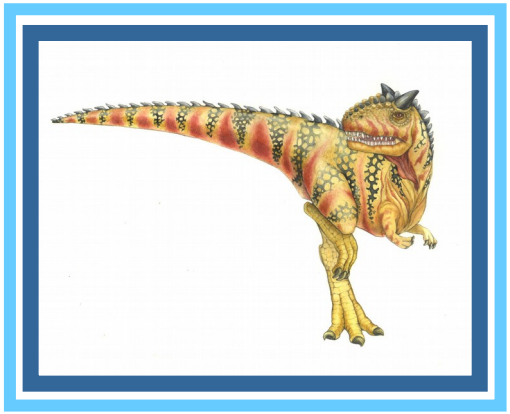
(e) RAID 4: block-interleaved parity.

(f) RAID 5: block-interleaved distributed parity.



(g) RAID 6: P + Q redundancy.

End of Chapter 11



RAID Structure

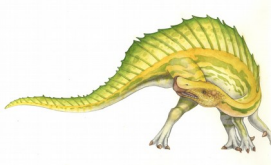
- RAID – redundant array of inexpensive drives
  - multiple drives provides reliability via **redundancy**
- Increases the **mean time to failure**
- **Mean time to repair –** exposure time when another failure could cause data loss
- **Mean time to data loss** based on above factors
- If mirrored disks fail independently, consider disk with 1300,000 mean time to failure and 10 hour mean time to repair
  - Mean time to data loss is $100,000^2 / (2 * 10) = 500 * 10^6$ hours, or 57,000 years!
- Frequently combined with **NVRAM** to improve write performance
- RAID is arranged into six different levels

RAID Structure

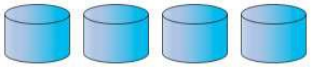
- RAID – redundant array of inexpensive drives
  - multiple drives provides reliability via **redundancy**
- Increases the **mean time to failure**
- **Mean time to repair –** exposure time when another failure could cause data loss
- **Mean time to data loss** based on above factors
- If mirrored disks fail independently, consider disk with 1300,000 mean time to failure and 10 hour mean time to repair
  - Mean time to data loss is $100,000^2 / (2 * 10) = 500 * 10^6$ hours, or 57,000 years!
- Frequently combined with **NVRAM** to improve write performance
- RAID is arranged into six different levels

RAID (Cont.)

- Several improvements in storage-use techniques involve the use of multiple drives working cooperatively
- **striping** uses a group of drives as one storage unit
- RAID schemes improve performance and improve the reliability of the storage system by storing redundant data
  - **Mirroring** or **shadowing** (**RAID 1**) keeps duplicate of each drive
  - Striped mirrors (**RAID 1+0**) or mirrored stripes (**RAID 0+1**) provides high performance and high reliability
  - **Block interleaved parity** (**RAID 4, 5, 6**) uses much less redundancy
- RAID within a storage array can still fail if the array fails, so automatic **replication** of the data between arrays is common
- Frequently, a small number of **hot-spare** drives are left unallocated, automatically replacing a failed drive and having data rebuilt onto them



RAID Levels



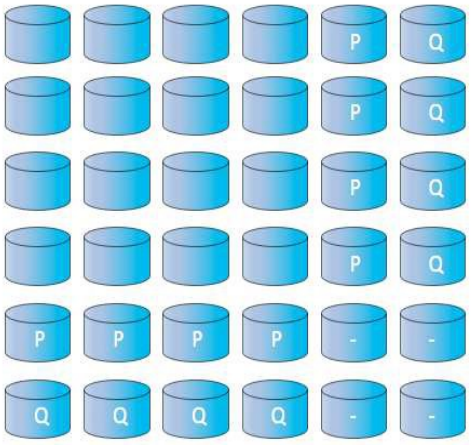
(a) RAID 0: non-redundant striping.

(b) RAID 1: mirrored disks.

(c) RAID 4: block-interleaved parity.

(d) RAID 5: block-interleaved distributed parity.

(e) RAID 6: P + Q redundancy.



(f) Multidimensional RAID 6.

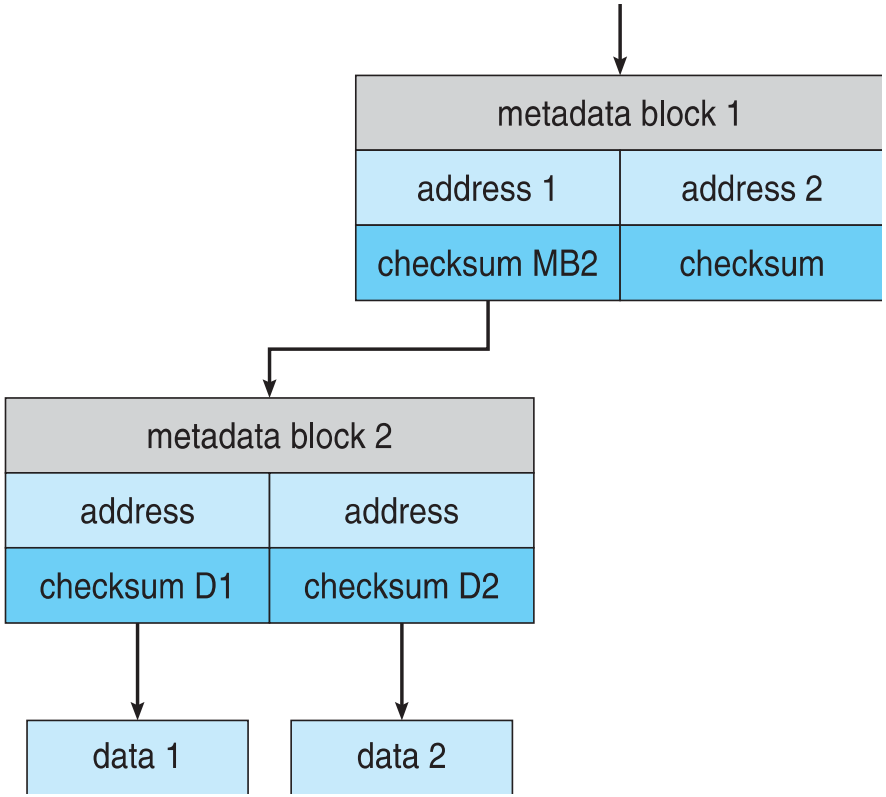
Other Features

- Regardless of where RAID implemented, other useful features can be added
- **Snapshot** is a view of file system before a set of changes take place (i.e. at a point in time)
  - More in Ch 12
- Replication is automatic duplication of writes between separate sites
  - For redundancy and disaster recovery
  - Can be synchronous or asynchronous
- Hot spare disk is unused, automatically used by RAID production if a disk fails to replace the failed disk and rebuild the RAID set if possible
  - Decreases mean time to repair

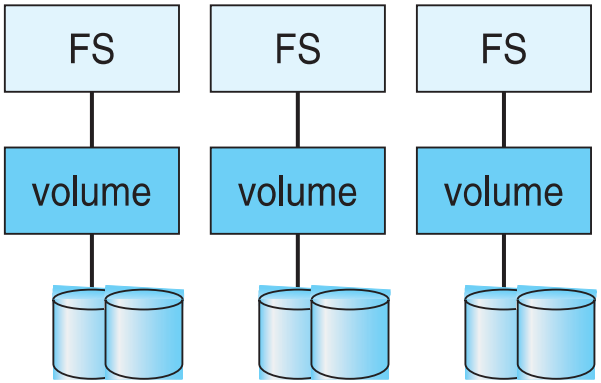
Extensions

- RAID alone does not prevent or detect data corruption or other errors, just disk failures
- Solaris ZFS adds **checksums** of all data and metadata
- Checksums kept with pointer to object, to detect if object is the right one and whether it changed
- Can detect and correct data and metadata corruption
- ZFS also removes volumes, partitions
  - Disks allocated in **pools**
  - Filesystems with a pool share that pool, use and release space like `malloc()` and `free()` memory allocate / release calls

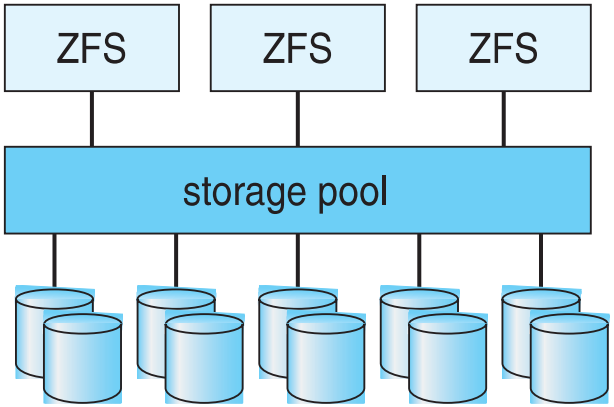
ZFS Checksums All Metadata and Data



Traditional and Pooled Storage



(a) Traditional volumes and file systems.



(b) ZFS and pooled storage.