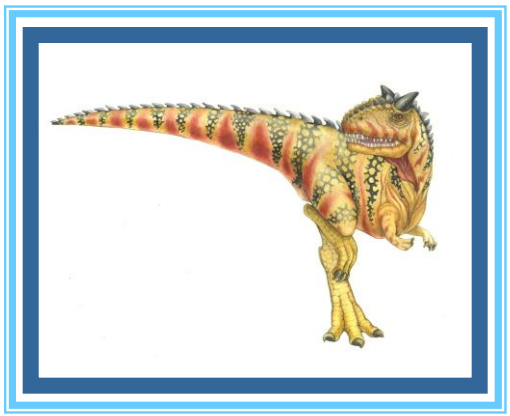
Chapter 11: Mass-Storage Systems

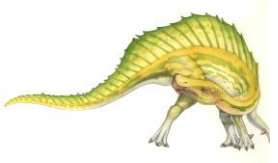


Silberschatz, Galvin and Gagne ©2018



Chapter 11: Mass-Storage Systems

- Overview of Mass Storage Structure
- HDD Scheduling
- NVM Scheduling
- Error Detection and Correction
- Storage Device Management
- Swap-Space Management
- Storage Attachment
- RAID Structure and Stable Storage





Objectives

- Describe the physical structure of secondary storage devices and the effect of a device's structure on its uses
- Explain the performance characteristics of mass-storage devices
- Evaluate I/O scheduling algorithms
- Discuss operating-system services provided for mass storage, including RAID

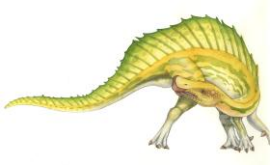


Troy University Resources



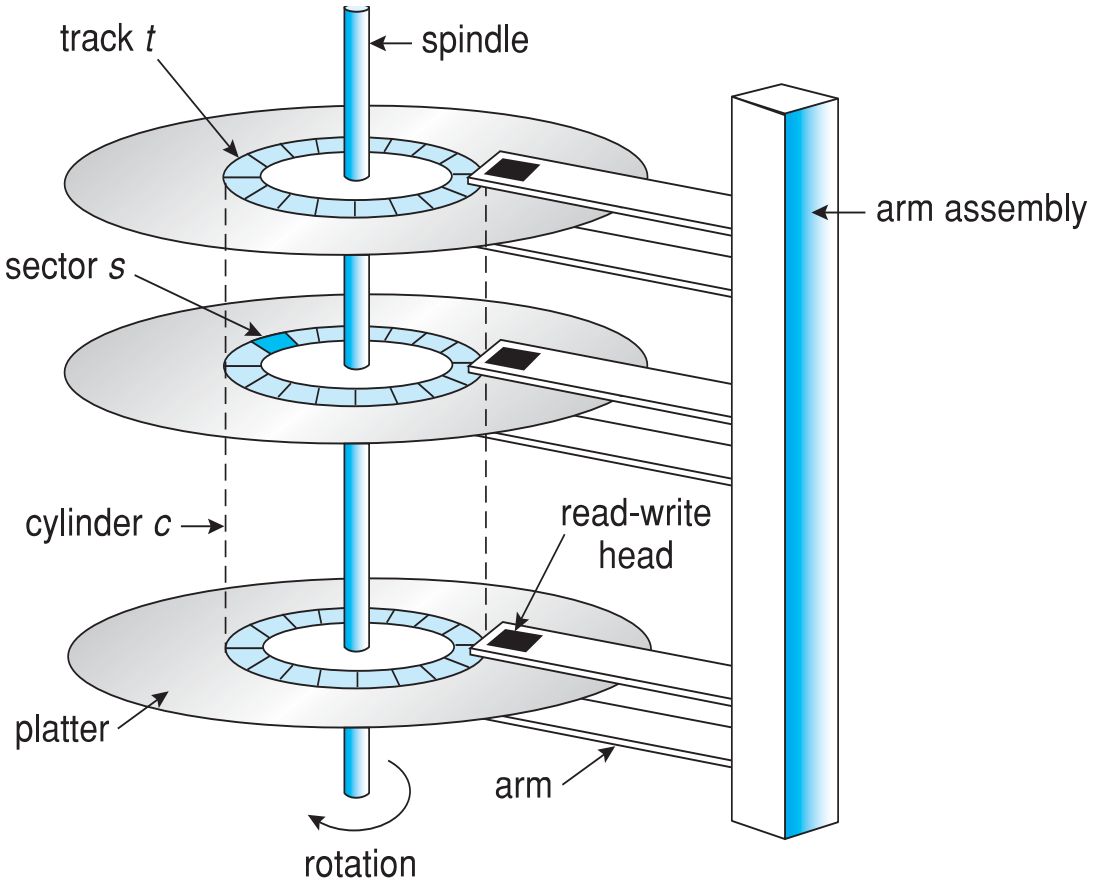
Overview of Mass Storage Structure

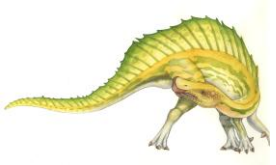
- Bulk of secondary storage for modern computers is **hard disk drives** (**HDDs**) and **nonvolatile memory** (**NVM**) devices
- **HDDs** spin platters of magnetically-coated material under moving read-write heads
  - Drives rotate at 60 to 250 times per second
  - **Transfer rate** is rate at which 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





Moving-head Disk Mechanism

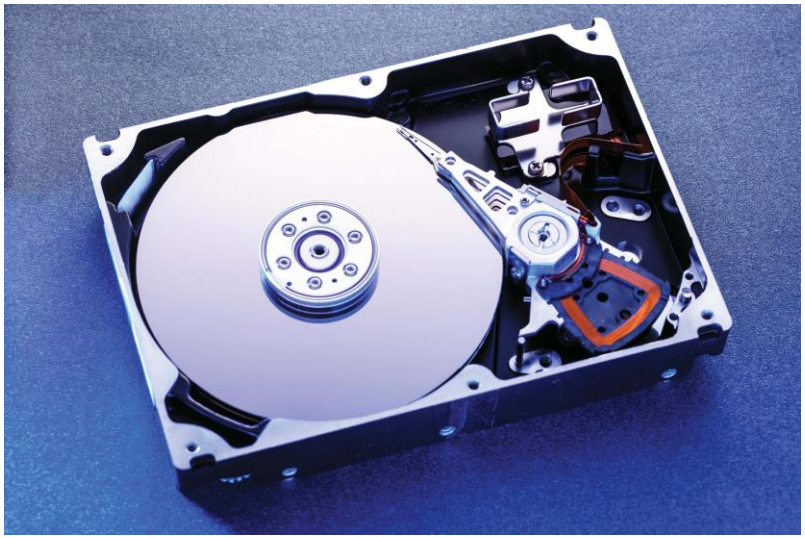






Hard Disk Drives

- Platters range from .85” to 14” (historically)
  - Commonly 3.5”, 2.5”, and 1.8”
- Historical capacity examples: 30 GB to 3 TB per drive
- Performance
  - SATA interface rate (example): 6 Gb/s
  - Illustrative HDD transfer rate: 1 Gb/s
  - 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

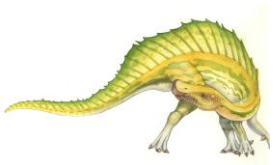






Hard Disk Performance

- **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 4 KiB block on a 7200 RPM disk with a 5ms average seek time, 1Gb/sec transfer rate with a .1ms controller overhead =
  - 5 ms + 4.1667 ms + 0.1 ms + transfer time =
  - Transfer time = (4096 bytes × 8) / (10^9 bits/s) = 0.032768 ms
  - Average I/O time = 5 + 4.1667 + 0.1 + 0.032768 ≈ 9.299 ms



Troy University Resources



The First Commercial Disk Drive



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

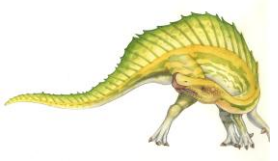
5M (7 bit) characters
50 x 24” platters
Access time = < 1 second





Nonvolatile Memory Devices

- If disk-drive like, then called **solid-state disks** (**SSDs**)
- Other forms include **USB** flash drives, motherboard modules, and storage in devices like smartphones
- Can be more reliable than HDDs
- More expensive per MB
- Maybe have shorter life span – need careful management
- Less capacity
- But much faster
- Busses can be too slow -> connect directly to PCI for example
- No moving parts, so no seek time or rotational latency





NAND Flash Devices

- Have characteristics that present challenges
- Read and written in “page” increments (think sector) but can’t overwrite in place
  - Must first be erased, and erases happen in larger ”block” increments
  - Erase-cycle endurance is finite and depends on the NAND technology
  - Life span measured in **drive writes per day** (**DWPD**)
    - A 1TB NAND drive with rating of 5DWPD is expected to have 5TB per day written within warranty period without failing

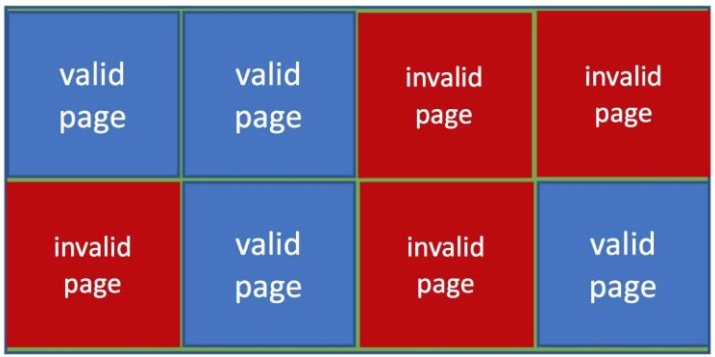




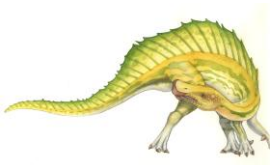


NAND Flash Controller Algorithms

- With no overwrite, pages end up with mix of valid and invalid data
- To track which logical blocks are valid, controller maintains **flash translation layer** (**FTL**) table
- Also implements **garbage collection** to free invalid page space
- Allocates **overprovisioning** to provide working space for GC
- Each cell has lifespan, so **wear leveling** needed to write equally to all cells



NAND block with valid and invalid pages





Volatile Memory

- DRAM frequently used as mass-storage device
  - Not technically secondary storage because volatile, but can have file systems, be used like very fast secondary storage
- **RAM drives** (with many names, including RAM disks) present as raw block devices, commonly file system formatted
- Computers have buffering, caching via RAM, so why RAM drives?
  - Caches / buffers allocated / managed by programmer, operating system, hardware
  - RAM drives under user control
  - Found in all major operating systems
    - Linux `/dev/ram`, macOS `diskutil`, and `tmpfs mounts (RAM with optional swap)`
- Used as high speed temporary storage
  - Programs could share bulk data, quickly, by reading/writing to RAM drive





Magnetic Tape

Magnetic tape was used as an early secondary-storage medium. Although it is nonvolatile and can hold large quantities of data, its access time is slow compared with that of main memory and drives. In addition, random access to magnetic tape is about a thousand times slower than random access to HDDs and about a hundred thousand times slower than random access to SSDs so tapes are not very useful for secondary storage. Tapes are used mainly for backup, for storage of infrequently used information, and as a medium for transferring information from one system to another.

A tape is kept in a spool and is wound or rewound past a read–write head. Moving to the correct spot on a tape can take minutes, but once positioned, tape drives can read and write data at speeds comparable to HDDs. Tape capacities vary greatly, depending on the particular kind of tape drive, with current capacities exceeding several terabytes. Some tapes have built-in compression that can more than double the effective storage. Tapes and their drivers are usually categorized by width, including 4, 8, and 19 millimeters and 1/4 and 1/2 inch. Some are named according to technology, such as LTO-6 (Figure 11.5) and SDLT.



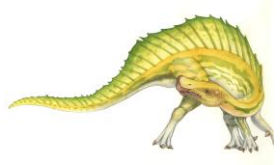
Figure 11.5 An LTO-6 Tape drive with tape cartridge inserted.





Disk Structure

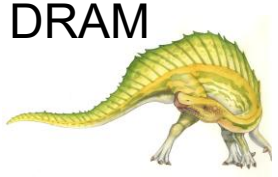
- Disk drives are addressed as large 1-dimensional arrays of **logical blocks**, where the logical block is the smallest unit of transfer
  - Low-level formatting creates **logical blocks** on physical media
- In the traditional CHS model, logical blocks map to sectors sequentially
  - Sector 0 is the 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
  - Modern drives hide physical mapping and remap bad sectors
    - Except for bad sectors
    - Non-constant # of sectors per track via constant angular velocity





Disk Attachment

- Host-attached storage accessed through I/O ports talking to **I/O busses**
- Several busses available, including **advanced technology attachment** (**ATA**), **serial ATA** (**SATA**), **eSATA**, **serial attached SCSI** (**SAS**), **universal serial bus** (**USB**), and **fibre channel** (**FC**).
- Most common is SATA
- Because NVM much faster than HDD, new fast interface for NVM called **NVM express** (**NVMe**), connecting directly to PCI bus
- Data transfers on a bus carried out by special electronic processors called **controllers** (or **host-bus adapters**, **HBAs**)
  - Host controller on the computer end of the bus, device controller on device end
  - Computer places command on host controller, using memory-mapped I/O ports
    - Host controller sends messages to device controller
    - Data transferred via DMA between device and computer DRAM

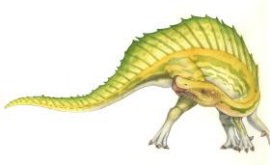


Troy University Resources



Address Mapping

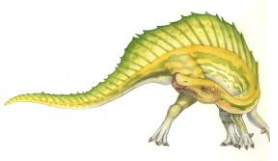
- Disk drives are addressed as large 1-dimensional arrays of **logical blocks**, where the logical block is the smallest unit of transfer
  - Low-level formatting creates **logical blocks** on physical media
- In the traditional CHS model, logical blocks map to sectors sequentially
  - Sector 0 is the 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
  - Modern drives hide physical mapping and remap bad sectors
    - Except for bad sectors
    - Non-constant # of sectors per track via constant angular velocity





HDD 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

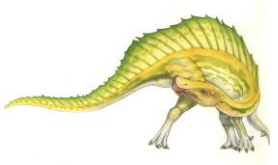


Troy University Resources



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 queue
  - Optimization algorithms only make sense when a queue exists
- In the past, operating system responsible for queue management, disk drive head scheduling
  - Now, built into the storage devices, controllers
  - Just provide LBAs, handle sorting of requests
    - Some of the algorithms they use described next





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

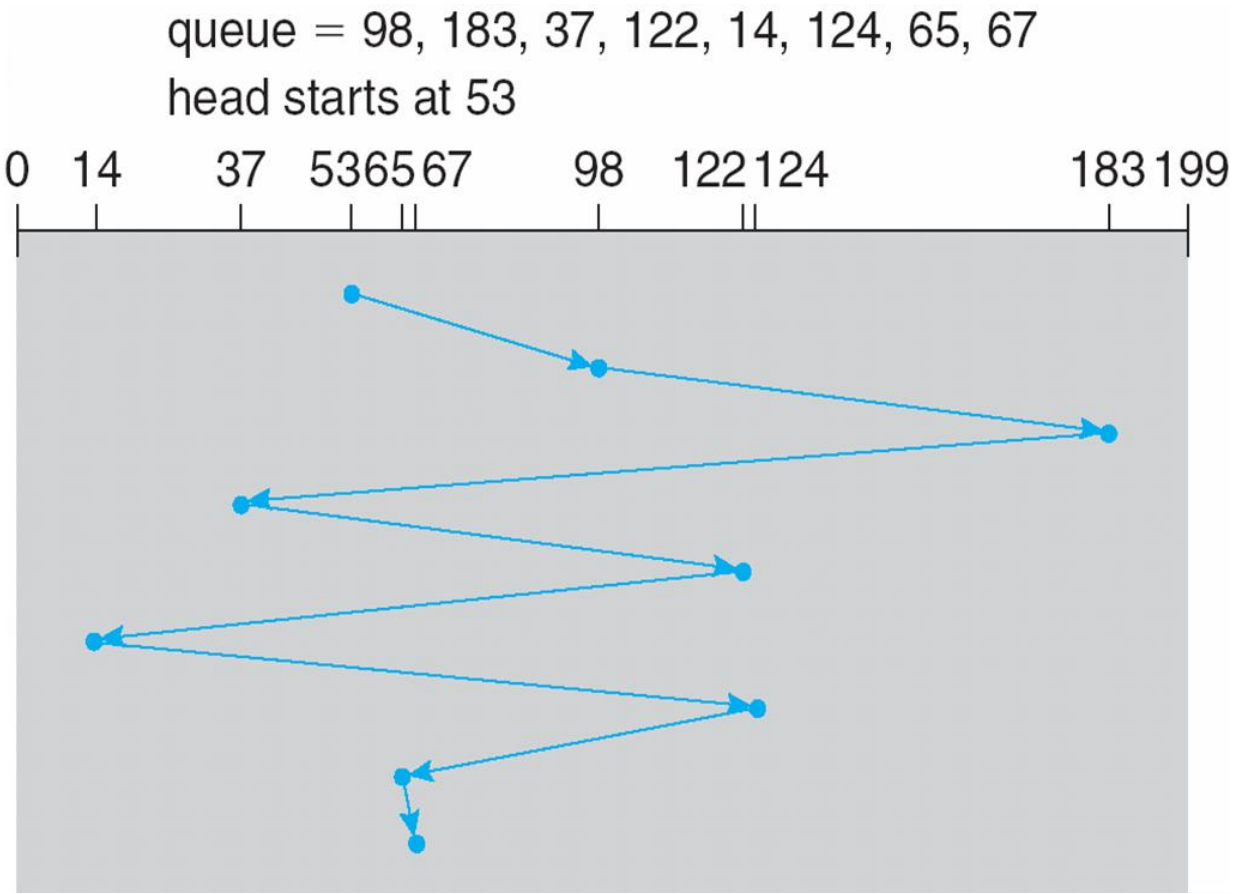


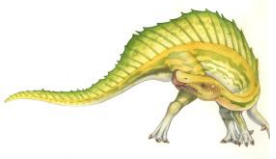
Troy University Resources



FCFS

Illustration shows total head movement of 640 cylinders

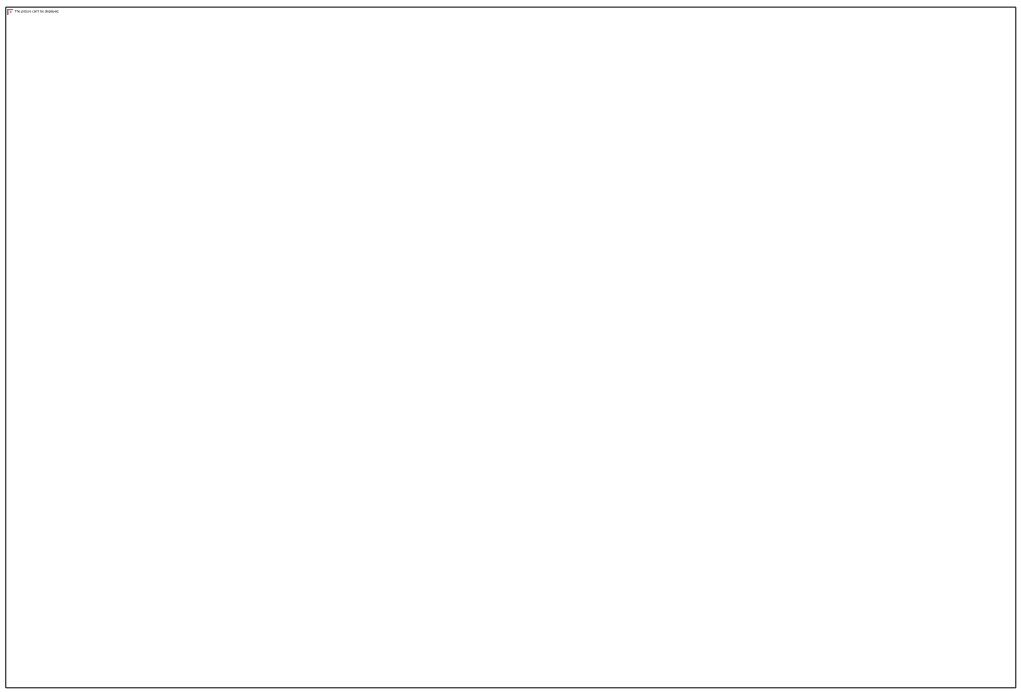






SSTF

- 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



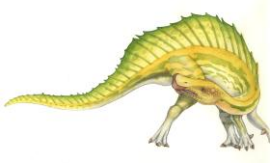


Troy University Resources



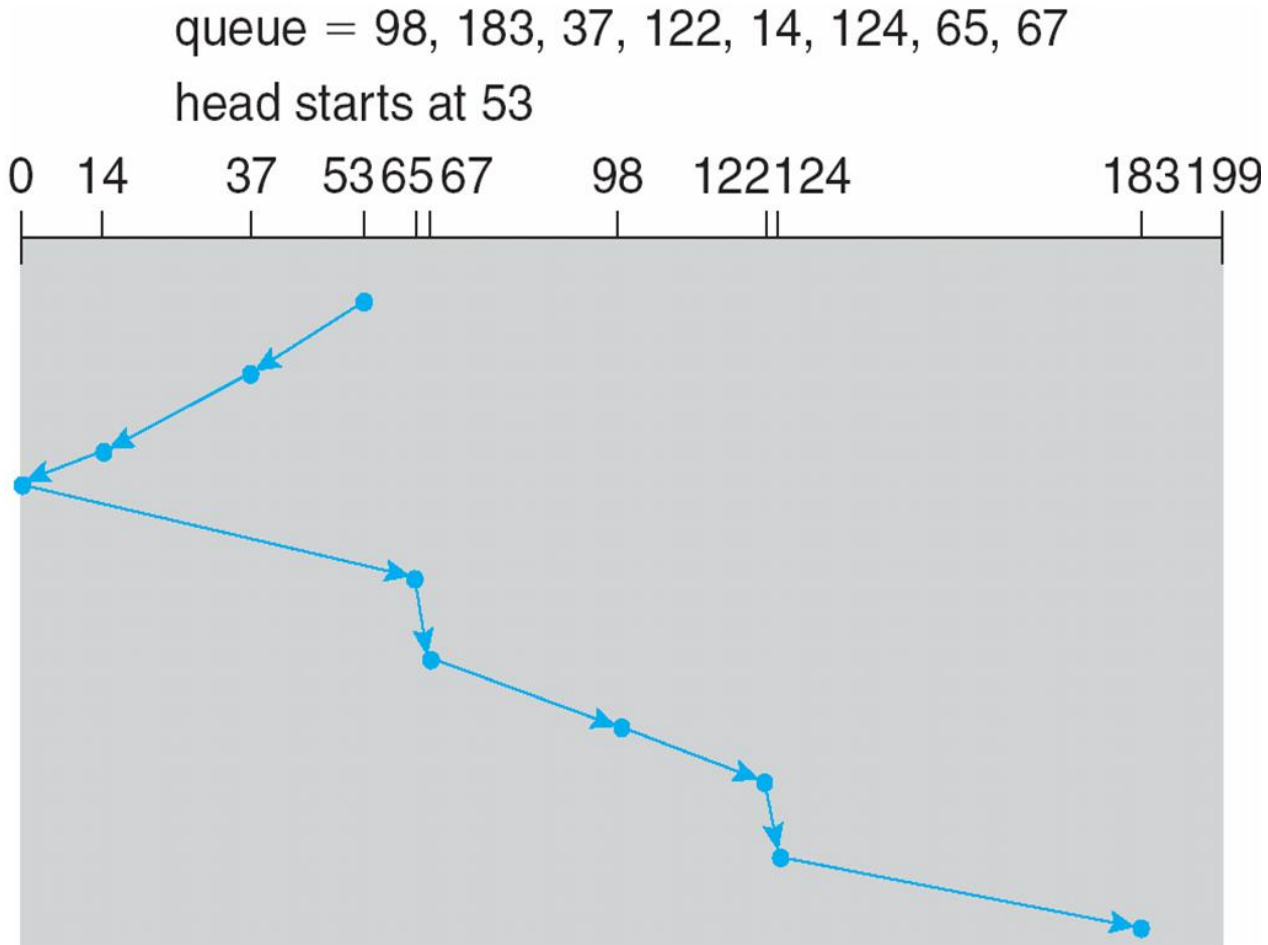
SCAN

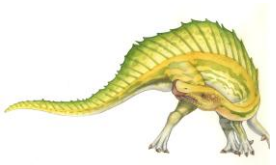
- The disk arm starts at one end of the disk, and moves toward the other end, servicing requests until it gets to the other end of the disk, where the head movement is reversed and servicing continues.
- **SCAN algorithm** Sometimes called the **elevator algorithm**
- For the illustrated initial sweep toward 0: 53 + 183 = 236 cylinders
- Requests just missed by the head wait until the next sweep





SCAN (Cont.)



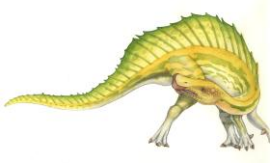


Troy University Resources



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
- Total number of cylinders?

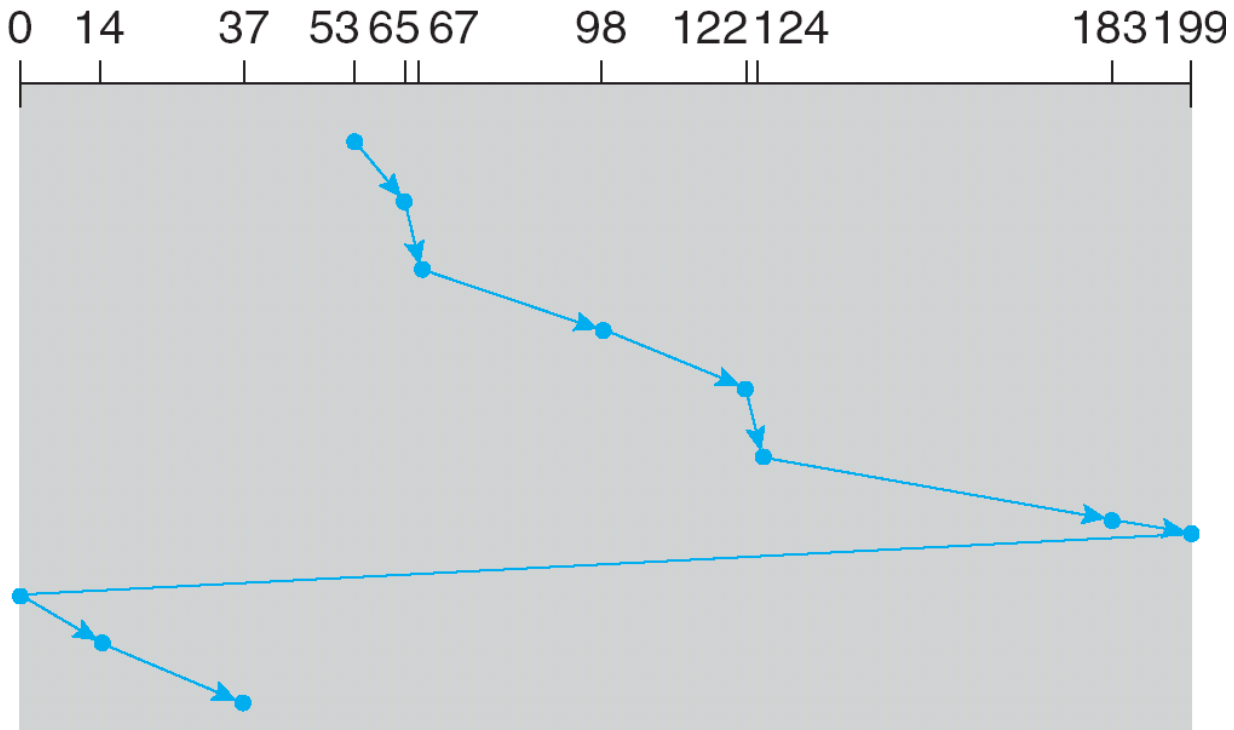


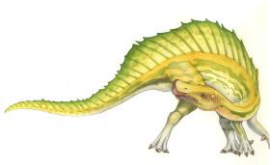


C-SCAN (Cont.)

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

head starts at 53

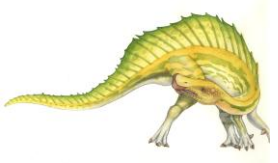






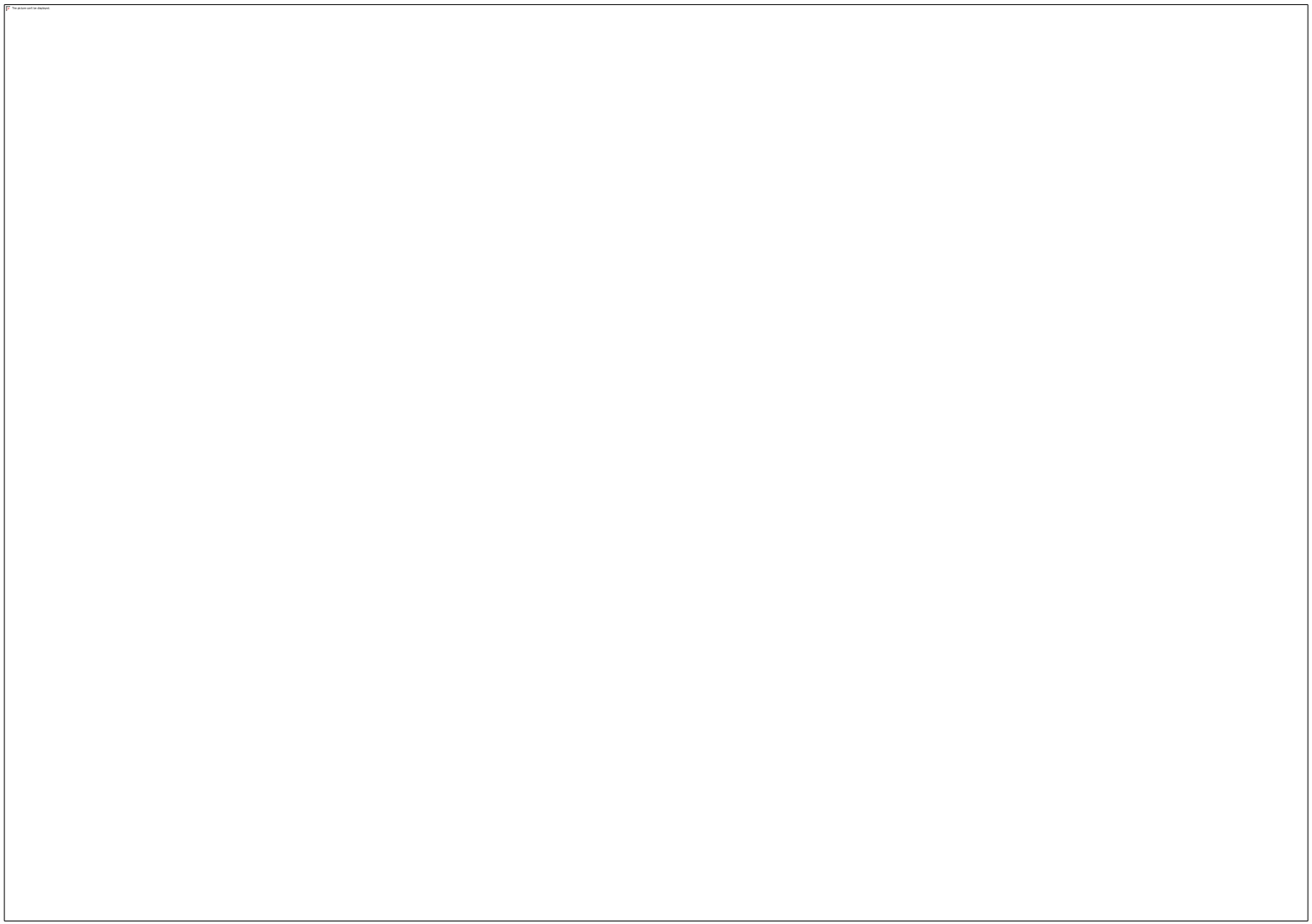
LOOK and C-LOOK

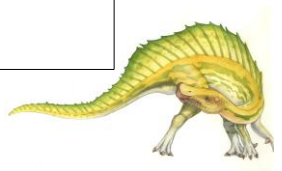
- LOOK and C-LOOK stop at the last pending request rather than the disk boundary
- LOOK reverses direction. C-LOOK jumps to the first pending request at the other end, then continues in the same direction
- Total number of cylinders?





C-LOOK (Cont.)





Troy University Resources



Selecting a Disk-Scheduling Algorithm

- Scheduling depends on workload, file allocation and metadata layout
- SCAN and C-SCAN perform better for systems that place a heavy load on the disk
  - Less starvation, but still possible
- To avoid starvation Linux implements **deadline** scheduler
  - Maintains separate read and write queues, gives read priority
    - Because processes more likely to block on read than write
  - Implements four queues: 2 x read and 2 x write
    - 1 read and 1 write queue sorted in LBA order, essentially implementing C-SCAN
    - 1 read and 1 write queue sorted in FCFS order
    - All I/O requests sent in batch sorted in that queue's order
    - After each batch, checks expired requests (defaults: reads 500 ms, writes 5 s)
      - If so, LBA queue containing that request is selected for next batch of I/O
- In RHEL 7 also **NOOP** and **completely fair queueing** scheduler (**CFQ**) also available, defaults vary by storage device





NVM Scheduling

- No disk heads or rotational latency but still room for optimization
- In RHEL 7 **NOOP** (no scheduling) is used but adjacent LBA requests are combined
  - NVM handles random I/O well. HDDs benefit strongly from sequential access
  - Sequential throughput depends on the device and interface
  - **Input/Output operations per second** (**IOPS**) much higher with NVM (hundreds of thousands vs hundreds)
  - But **write amplification** (one write, causing garbage collection and many read/writes) can decrease the performance advantage

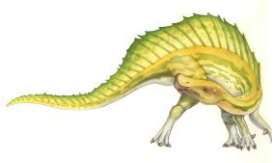


Troy University Resources



Error Detection and Correction

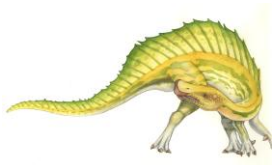
- Fundamental aspect of many parts of computing (memory, networking, storage)
- **Error detection** determines if there a problem has occurred (for example a bit flipping)
  - If detected, can halt the operation
  - Detection frequently done via parity bit
- Parity one form of **checksum** – uses modular arithmetic to compute, store, compare values of fixed-length words
  - Another error-detection method common in networking is **cyclic redundancy check** (**CRC**) which uses hash function to detect multiple-bit errors
- **Error-correction code** (**ECC**) not only detects, but can correct some errors
  - Correctability depends on the error pattern and the code's capability





Storage Device Management

- **Low-level formatting**, or **physical formatting** — Dividing a disk into sectors that the disk controller can read and write
  - Each sector can hold header information, plus data, plus error correction code (**ECC**)
  - Usually 512 bytes of data but can be selectable
- To use a disk to hold files, the operating system still needs to record its own data structures on the disk
  - **Partition** the disk into one or more groups of cylinders, each treated as a logical disk
  - **Logical formatting** or "making a file system"
  - To increase efficiency most file systems group blocks into **clusters**
    - Disk I/O done in blocks
    - File I/O done in clusters

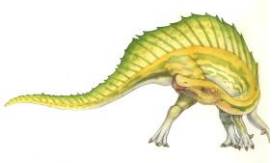


Troy University Resources



Storage Device Management (cont.)

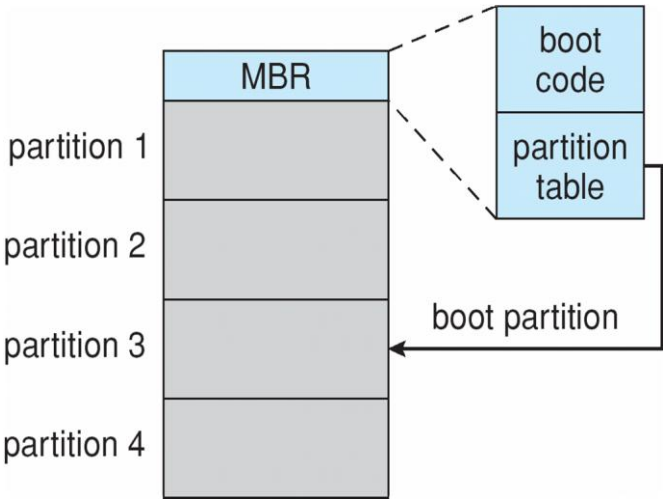
- **Root partition** contains the OS, other partitions can hold other Oses, other file systems, or be raw
  - **Mounted** at boot time
  - Other partitions can mount automatically or manually
- At mount time, check file-system state and recover or check consistency as needed
  - Is all metadata correct?
    - If not, fix it, try again
    - If yes, add to mount table, allow access
- Boot block can point to boot volume or boot loader set of blocks that contain enough code to know how to load the kernel from the file system
  - Or a boot management program for multi-os booting





Device Storage Management (Cont.)

- Raw disk access for apps that want to do their own block management, keep OS out of the way (databases for example)
- Boot block initializes system
  - The bootstrap is stored in ROM, firmware
  - **Bootstrap loader** program stored in boot blocks of boot partition
- Methods such as **sector sparing** used to handle bad blocks



Booting from secondary storage in Windows

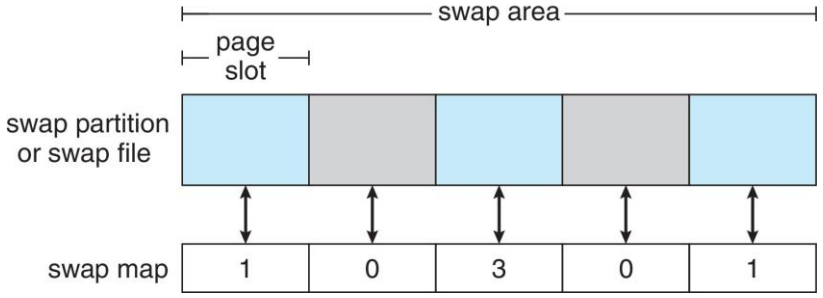


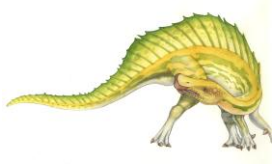
Troy University Resources



Swap-Space Management

- Used for moving entire processes (swapping), or pages (paging), from DRAM to secondary storage when DRAM not large enough for all processes
- Operating system provides **swap space management**
  - Secondary storage slower than DRAM, so important to optimize performance
  - Usually multiple swap spaces possible – decreasing I/O load on any given device
  - Best to have dedicated devices
  - Can be in raw partition or a file within a file system (for convenience of adding)
  - Data structures for swapping on Linux systems:







Storage Attachment

- Computers access storage in three ways
  - host-attached
  - network-attached
  - cloud
- Host attached access through local I/O ports, using one of several technologies
  - To attach many devices, use storage busses such as USB, firewire, thunderbolt
  - High-end systems use **fibre channel** (**FC**)
    - High-speed serial architecture using fibre or copper cables
    - Multiple hosts and storage devices can connect to the FC fabric

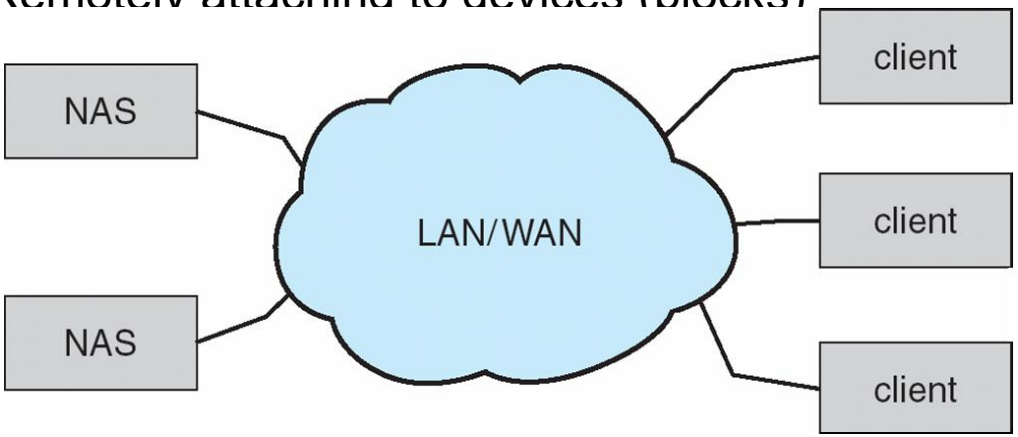


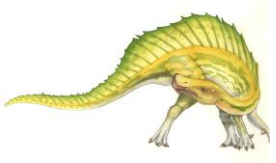
Troy University Resources



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
- NFS and CIFS are common protocols
- Implemented via remote procedure calls (RPCs) between host and storage over typically TCP or UDP on IP network
- **iSCSI** carries SCSI over IP for block access (SAN), rather than NAS file access
  - Remotely attaching to devices (blocks)

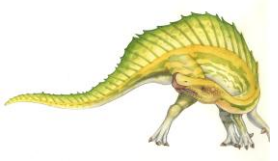






Cloud Storage

- Similar to NAS, provides access to storage across a network
  - Often accessed over the Internet or a WAN to a remote data center
- Cloud storage can expose object APIs, file protocols or block devices
  - Examples include Dropbox, Amazon S3, Microsoft OneDrive, Apple iCloud
  - Object APIs support remote access with explicit request and failure handling

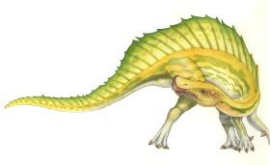


Troy University Resources



Storage Array

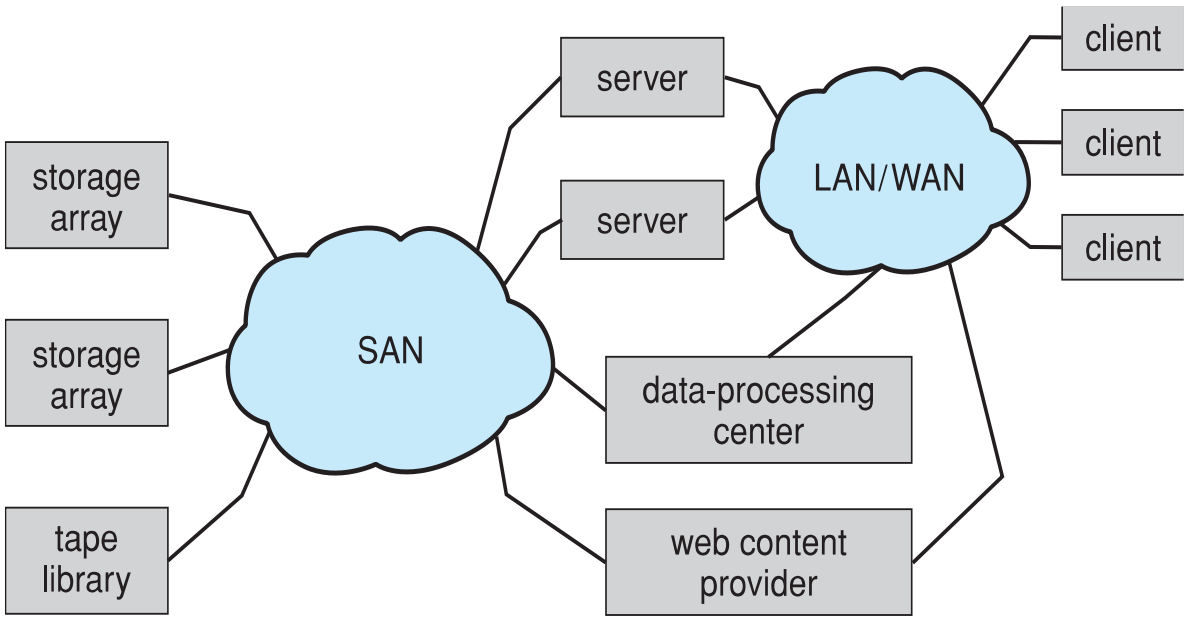
- Can just attach disks, or arrays of disks
- Arrays can attach directly or through a dedicated storage fabric
- Storage Array has controller(s), provides features to attached host(s)
  - Ports to connect hosts to array
  - Memory, controlling software (sometimes NVRAM, etc)
  - A few to thousands of disks
  - RAID, hot spares, hot swap (discussed later)
  - Shared storage -> more efficiency
  - Features found in some file systems
    - Snaphots, clones, thin provisioning, replication, deduplication, etc





Storage Area Network

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





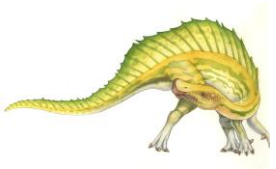


Storage Area Network (Cont.)

- SAN is one or more storage arrays
  - Connected to one or more Fibre Channel switches or **InfiniBand** (**IB**) network
- Hosts also attach to the switches
- Storage made available via **LUN Masking** from specific arrays to specific servers
- Easy to add or remove storage, add new host and allocate it storage
- Why have separate storage networks and communications networks?
  - Consider iSCSI, FCOE



A Storage Array





RAID Structure

- **RAID** – **redundant array of inexpensive disks**
  - multiple disk drives provides reliability via **redundancy**
- Redundancy can increase the **mean time to data loss**
- **Mean time to repair –** exposure time when another failure could cause data loss
- **Mean time to data loss** based on above factors
- For two independent mirrored disks, assume MTTF = 100,000 hours per disk and MTTR = 10 hours
  - Approximate mean time to data loss = $100{,}000^2 / (2 \times 10) = 500 \times 10^6$ hours (about 57,000 years in this model)
- Frequently combined with **NVRAM** to improve write performance
- Several improvements in disk-use techniques involve the use of multiple disks working cooperatively

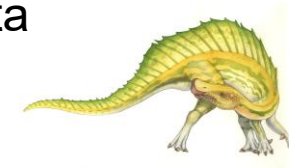


Troy University Resources



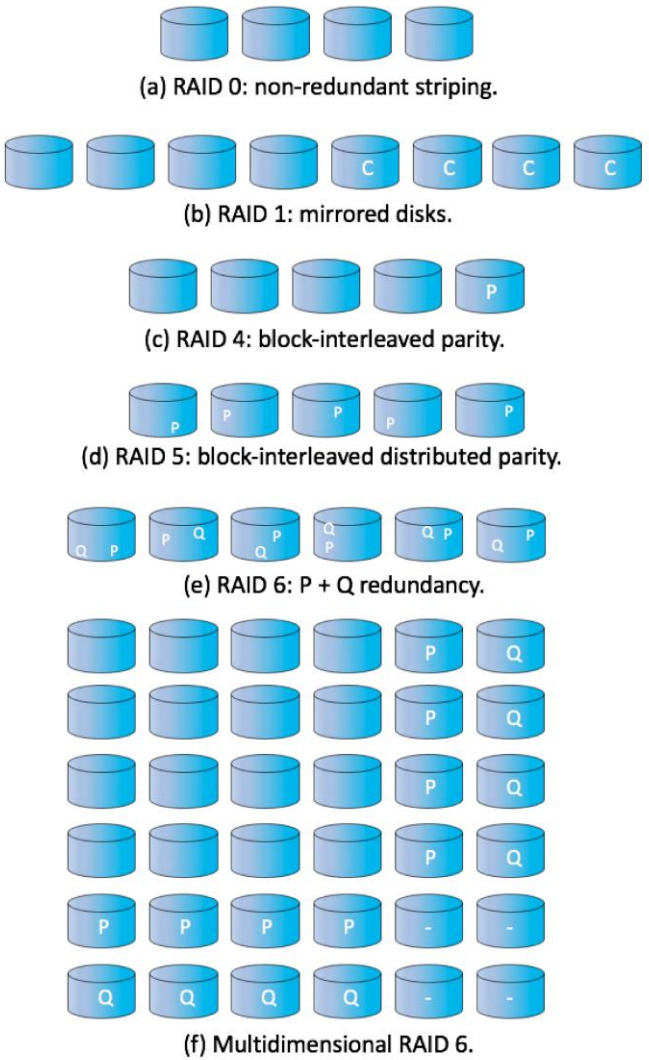
RAID (Cont.)

- Disk **striping** uses a group of disks as one storage unit
- Standard levels include RAID 0 through RAID 6, plus nested levels
- Redundant RAID levels improve fault tolerance. RAID 0 provides striping without redundancy
  - **Mirroring** or **shadowing** (**RAID 1**) keeps duplicate of each disk
  - 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** disks are left unallocated, automatically replacing a failed disk and having data rebuilt onto them



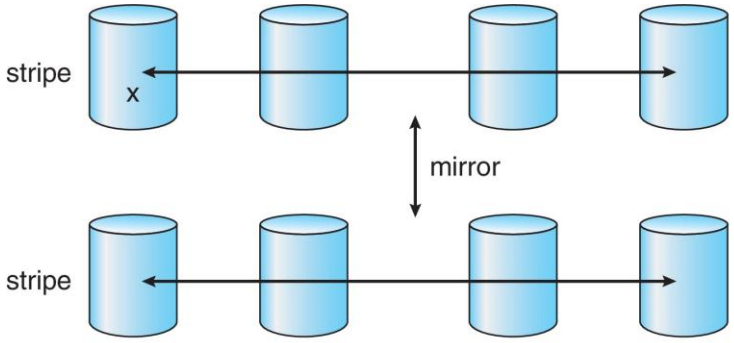


RAID Levels

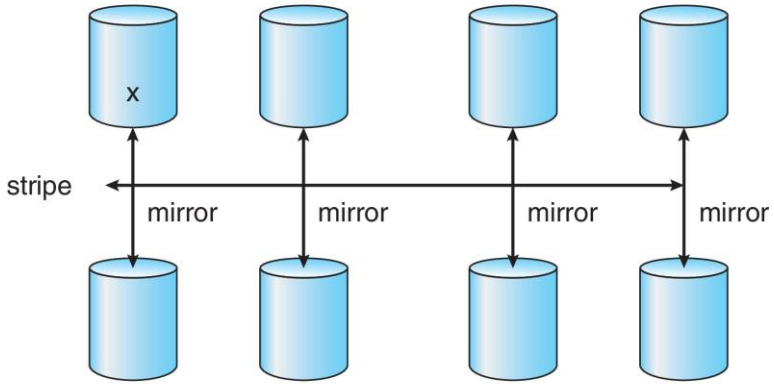

C
P
P
P
P
P
Q
P
Q
P
Q



RAID (0 + 1) and (1 + 0)



a) RAID 0 + 1 with a single disk failure.



b) RAID 1 + 0 with a single disk failure.





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 14
- 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

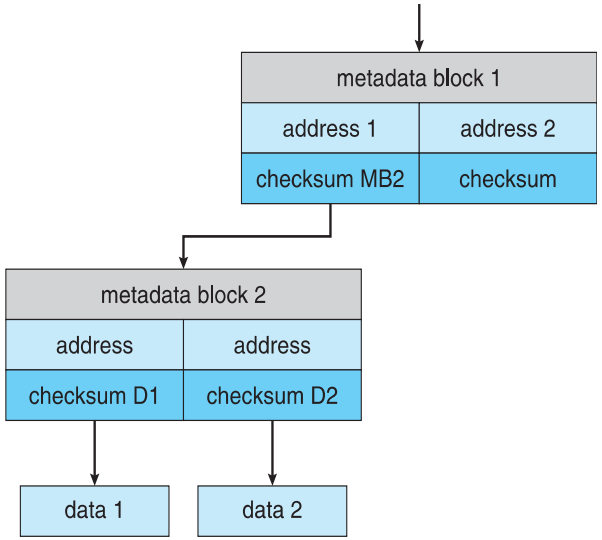


Troy University Resources

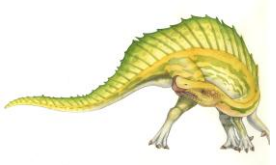


Extensions

- 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 corruption and repair it when a valid redundant copy is available
- ZFS integrates volume management and file systems through storage pools
  - Disks allocated in **pools**
  - Filesystems with a pool share that pool, use and release space like **`malloc()`** and **`free()`** memory allocate / release calls

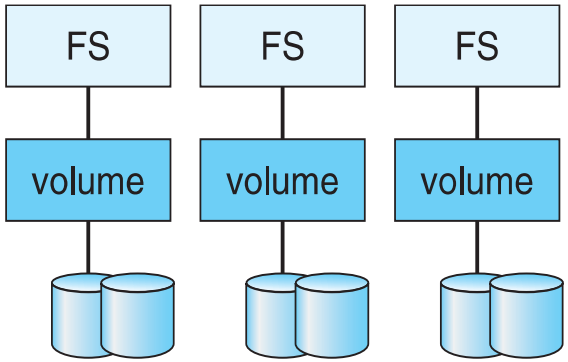


ZFS checksums all metadata and data

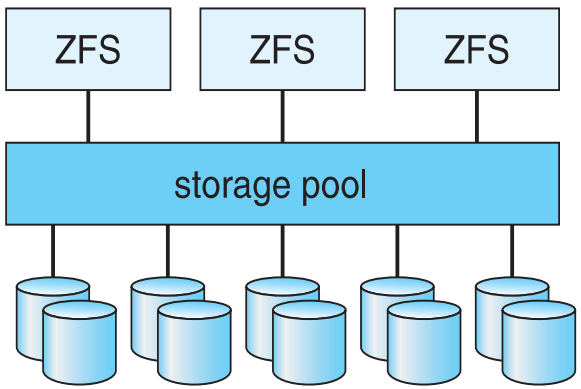




Traditional and Pooled Storage



(a) Traditional volumes and file systems.



(b) ZFS and pooled storage.



Troy University Resources



Stable-Storage Implementation

- Write-ahead log scheme requires stable storage
- Stable storage preserves committed data under an explicitly assumed failure model
- To implement stable storage:
  - Replicate information on more than one nonvolatile storage media with independent failure modes
  - Update information in a controlled manner to ensure that we can recover the stable data after a failure covered by that model during data transfer or recovery
- Disk write has 1 of 3 outcomes
  1. **Successful completion -** The data were written correctly on disk
  2. **Partial failure -** A failure occurred in the midst of transfer, so only some of the sectors were written with the new data, and the sector being written during the failure may have been corrupted
  3. **Total failure -** The failure occurred before the disk write started, so the previous data values on the disk remain intact

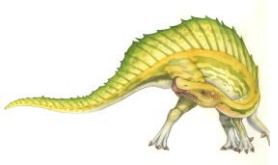




Stable-Storage Implementation (Cont.)

- Recovery: if the first copy is invalid, restore it from the second. Otherwise, make the second match the first
  - System maintains 2 physical blocks per logical block and does the following:
    1. Write to 1st physical
    2. When successful, write to 2nd physical
    3. Declare complete only after second write completes successfully

  Systems frequently use NVRAM as one physical to accelerate

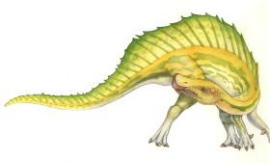


Troy University Resources



Object Storage

- General-purpose computing, file systems not sufficient for very large scale
- Another approach – start with a storage pool and place objects in it
  - Object just a container of **data**
  - A flat namespace uses object keys or IDs, with APIs for lookup and listing
  - Computer-oriented, not user-oriented
- Typical sequence
  - Create an object within the pool, receive an object ID
  - Access object via that ID
  - Delete object via that ID





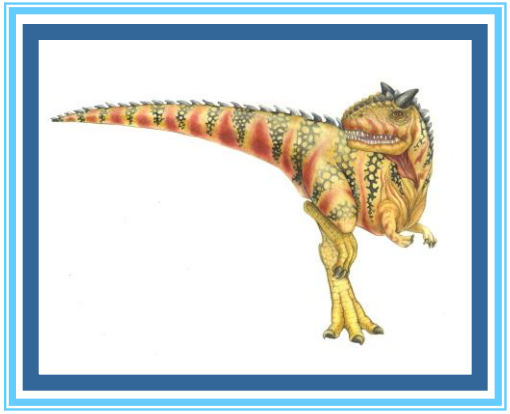
Object Storage (Cont.)

- Object stores such as Ceph RADOS manage object placement and protection. **HDFS is a distributed file system**
  - Typically by storing N copies, across N systems, in the object storage cluster
  - **Horizontally scalable**
  - **Objects contain data and metadata. Content-based addressing is optional**



Troy University Resources

End of Chapter 11



Silberschatz, Galvin and Gagne ©2018