Storage

Amir H. Payberah
amir@sics.se

Amirkabir University of Technology
(Tehran Polytechnic)

Motivation

Secondary Storage

- Main memory is usually too small.
- Computer systems must provide secondary storage to back up main memory.

File Systems and Secondary Storage

- The file system can be viewed logically as three parts:

File Systems and Secondary Storage

- The file system can be viewed logically as three parts:
  - The file systems at the lowest level: the structure of secondary storage.

File Systems and Secondary Storage

- The file system can be viewed logically as three parts:
  - The file systems at the lowest level: the structure of secondary storage.
  - The user and programmer interface to the file system.

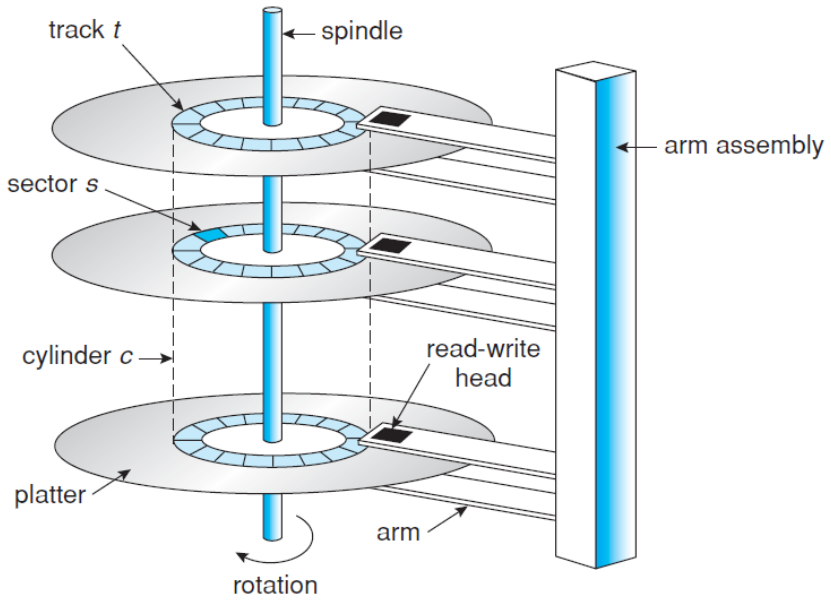
File Systems and Secondary Storage

- The file system can be viewed logically as three parts:
  - The file systems at the lowest level: the structure of secondary storage.
  - The user and programmer interface to the file system.
  - The internal data structures and algorithms used by the OS to implement this interface.

Overview of Mass-Storage Structure

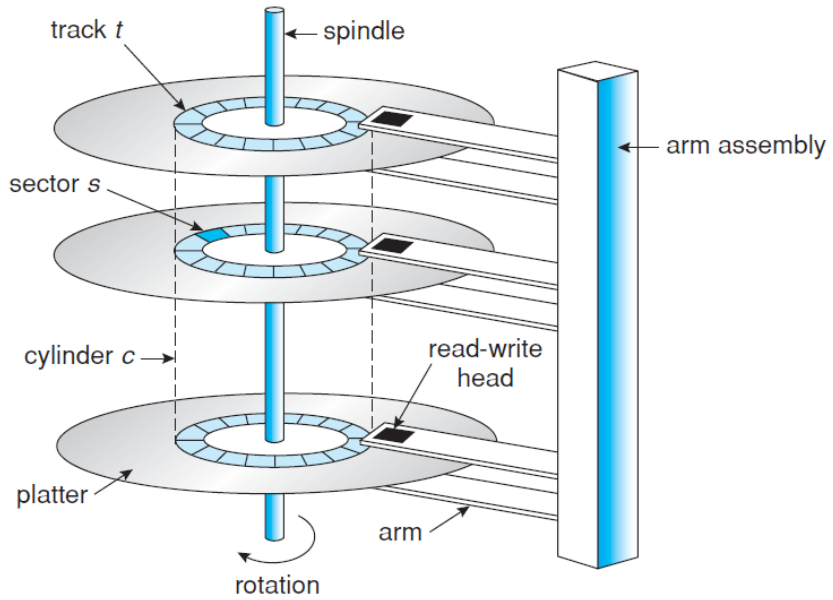
Mass Storage Structure (1/3)

- Magnetic disks: bulk of secondary storage



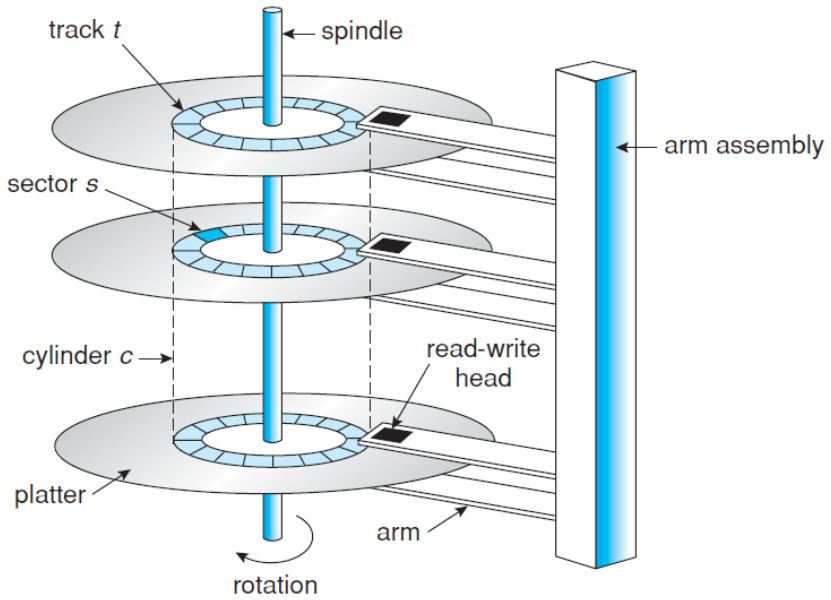
Mass Storage Structure (1/3)

- Magnetic disks: bulk of secondary storage
- Disk platter is a flat circular shape, covered with a magnetic material.



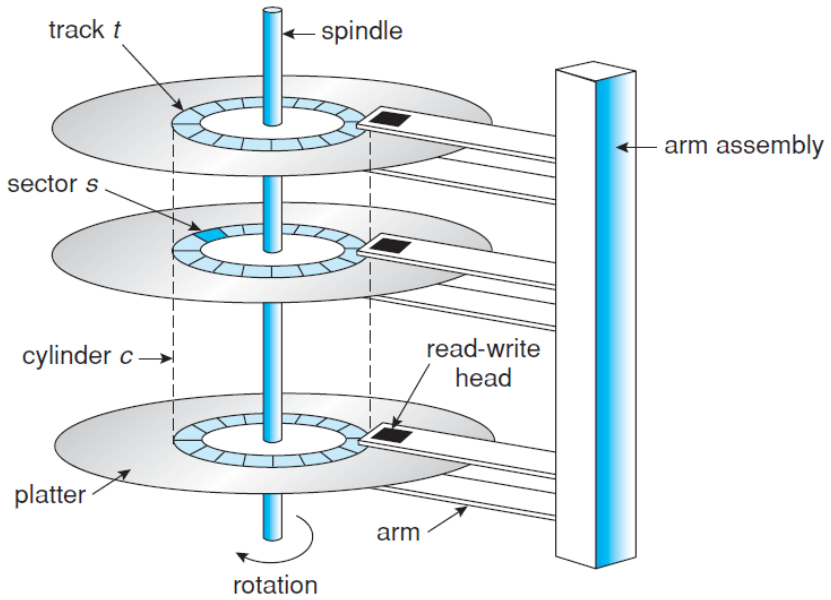
Mass Storage Structure (1/3)

- Magnetic disks: bulk of secondary storage
- Disk platter is a flat circular shape, covered with a magnetic material.
- Heads are attached to a disk arm.



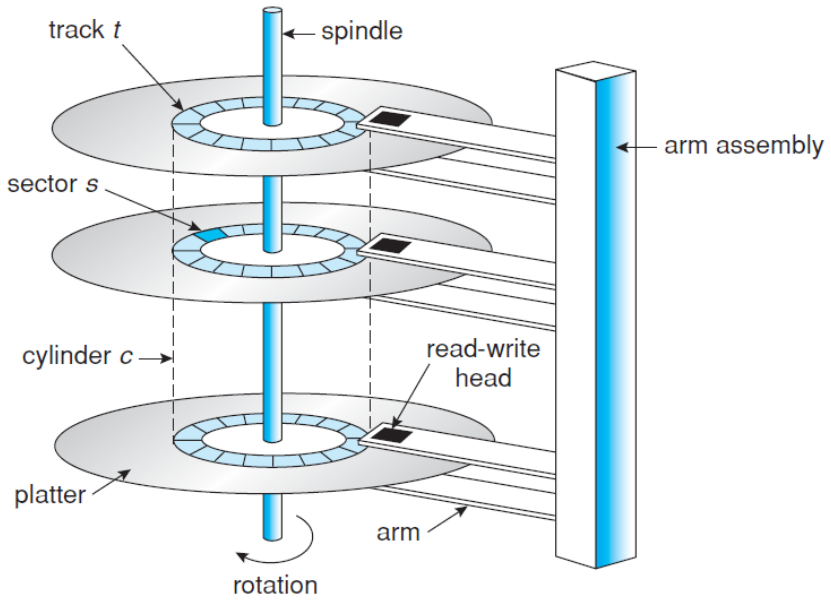
Mass Storage Structure (1/3)

- Magnetic disks: bulk of secondary storage
- Disk platter is a flat circular shape, covered with a magnetic material.
- Heads are attached to a disk arm.
- The surface of a platter is logically divided into circular tracks, which are subdivided into sectors.



Mass Storage Structure (1/3)

- Magnetic disks: bulk of secondary storage
- Disk platter is a flat circular shape, covered with a magnetic material.
- Heads are attached to a disk arm.
- The surface of a platter is logically divided into circular tracks, which are subdivided into sectors.
- The set of tracks that are at one arm position makes up a cylinder.



Mass Storage Structure (2/3)

- Drives rotate at 60 to 250 times per second.

Mass Storage Structure (2/3)

- Drives rotate at 60 to 250 times per second.
- Transfer rate: the rate at which data flow between drive and computer.

Mass Storage Structure (2/3)

- Drives rotate at 60 to 250 times per second.
- Transfer rate: the rate at which data flow between drive and computer.
- Positioning time (random-access time): the time to move disk arm to desired cylinder (seek time) and time for desired sector to rotate under the disk head (rotational latency).

Mass Storage Structure (2/3)

- Drives rotate at 60 to 250 times per second.
- Transfer rate: the rate at which data flow between drive and computer.
- Positioning time (random-access time): the 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.

Mass Storage Structure (3/3)

- Disks can be removable.
- Drive attached to computer via I/O bus.
  - ATA, SATA, USB, Fibre Channel, SCSI, SAS, Firewire

Mass Storage Structure (3/3)

- Disks can be removable.
- Drive attached to computer via I/O bus.
  - ATA, SATA, USB, Fibre Channel, SCSI, SAS, Firewire
- The host controller is the controller at the computer end of the bus.

Mass Storage Structure (3/3)

- Disks can be removable.
- Drive attached to computer via I/O bus.
  - ATA, SATA, USB, Fibre Channel, SCSI, SAS, Firewire
- The host controller is the controller at the computer end of the bus.
- A disk controller is built into each disk drive.

Mass Storage Structure (3/3)

- Disks can be removable.
- Drive attached to computer via I/O bus.
  - ATA, SATA, USB, Fibre Channel, SCSI, SAS, Firewire
- The host controller is the controller at the computer end of the bus.
- A disk controller is built into each disk drive.
- Host controller in computer uses bus to talk to disk controller built into drive or storage array.

Hard Disks

- Platters range from 0.85" to 14" (historically).
- Commonly 3.5", 2.5", and 1.8".
- Range from 30GB to 3TB per drive.

The First Commercial Disk Drive

- IBM, 1956
- 350 disk storage system
- 5M
- Access time $\leq$ 1 second

Solid-State Disks (SSDs)

- Non-volatile memory used like a hard drive.
- More expensive per MB.
- Maybe have shorter life span.
- Less capacity, but much faster.
- No moving parts, so no seek time or rotational latency.

Magnetic Tape

- Early secondary-storage medium.
- Relatively permanent and holds large quantities of data.
- Access time slow.
- Random access $\sim$ 1000 times slower than disk.
- Mainly used for backup, storage of infrequently-used data.
- Once data under head, transfer rates comparable to disk.

Disk Structure

Disk Structure (1/2)

- Disk drives are addressed as large 1-dimensional arrays of logical blocks.

Disk Structure (1/2)

- Disk drives are addressed as large 1-dimensional arrays of logical blocks.
- The logical block is the smallest unit of transfer.

Disk Structure (1/2)

- Disk drives are addressed as large 1-dimensional arrays of logical blocks.
- The logical block is the smallest unit of transfer.
- Low-level formatting creates logical blocks on physical media.

Disk Structure (2/2)

- The array of logical blocks is mapped into the sectors of the disk sequentially.

Disk Structure (2/2)

- The 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.

Disk Structure (2/2)

- The 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.

Disk Structure (2/2)

- The 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 num. of sectors per track via constant angular velocity.

Disk Attachment

Disk Attachment

- Host-attached storage
- Network-attached storage (NAS)
- Storage-area network (SAN)

Host-Attached Storage

- Host-attached storage accessed throughI/O ports talking to I/O buses.

Host-Attached Storage

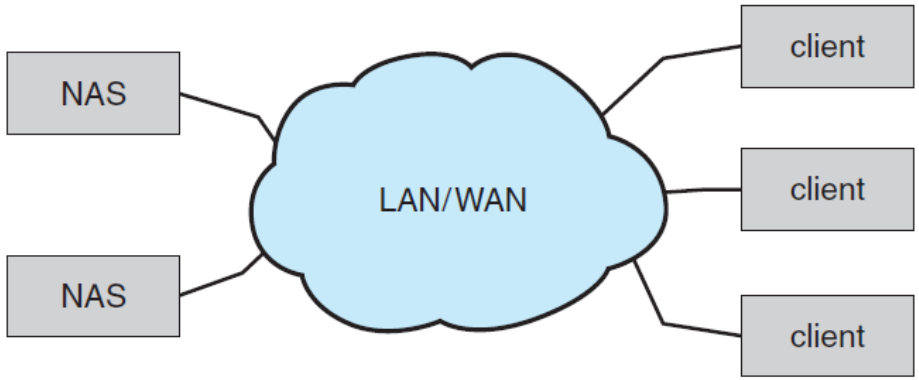
- Host-attached storage accessed throughI/O ports talking to I/O buses.
- The typical desktop PC uses an I/O bus architecture, called IDE or ATA.
  - Maximum of two drives per I/O bus.
  - A newer, similar protocol that has simplified cabling is SATA.

Host-Attached Storage

- Host-attached storage accessed throughI/O ports talking to I/O buses.
- The typical desktop PC uses an I/O bus architecture, called IDE or ATA.
  - Maximum of two drives per I/O bus.
  - A newer, similar protocol that has simplified cabling is SATA.
- SCSI is a bus, up to 16 devices on one cable.
- Fiber Channel (FC) is high-speed serial architecture.

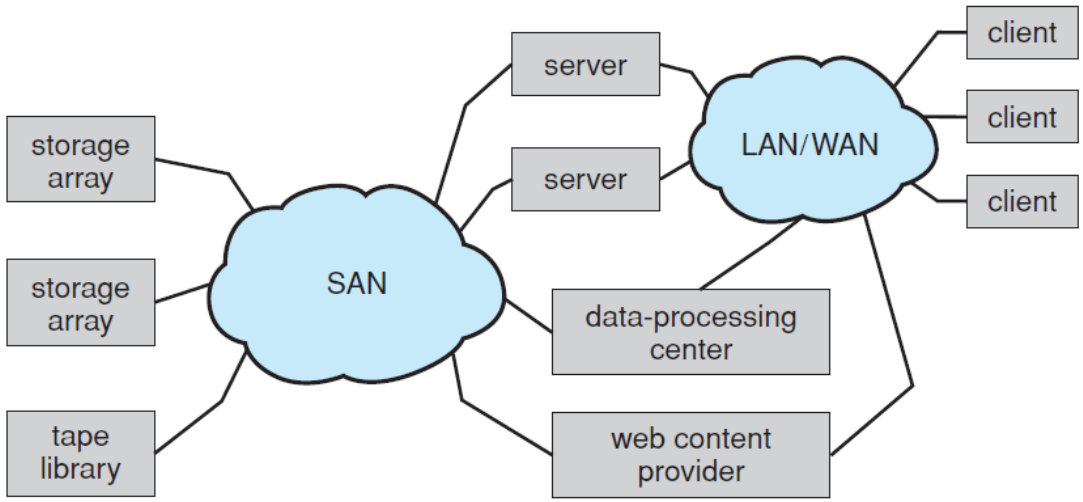
Network-Attached Storage

- Network-attached storage (NAS) is storage made available over a network rather than over a local connection.
- 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.



Storage-Area Network

- Storage-area network (SAN) is common in large storage environments.
- Multiple hosts attached to multiple storage arrays.
- Storage arrays and Hosts are connected to one or more Fibre Channel switches.



Disk Scheduling

Disk Scheduling (1/2)

- Having a fast access time and disk bandwidth.
- Minimize seek time.
- Seek time $\approx$ 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 (2/2)

- There are many sources of disk I/O request, e.g., OS, system processes, users processes

Disk Scheduling (2/2)

- There are many sources of disk I/O request, e.g., OS, system processes, users processes
- OS maintains queue of requests, per disk or device.

Disk Scheduling (2/2)

- There are many sources of disk I/O request, e.g., OS, system processes, users processes
- OS maintains queue of requests, per disk or device.
- Idle disk can immediately work on I/O request, busy disk means work must queue.

Disk Scheduling (2/2)

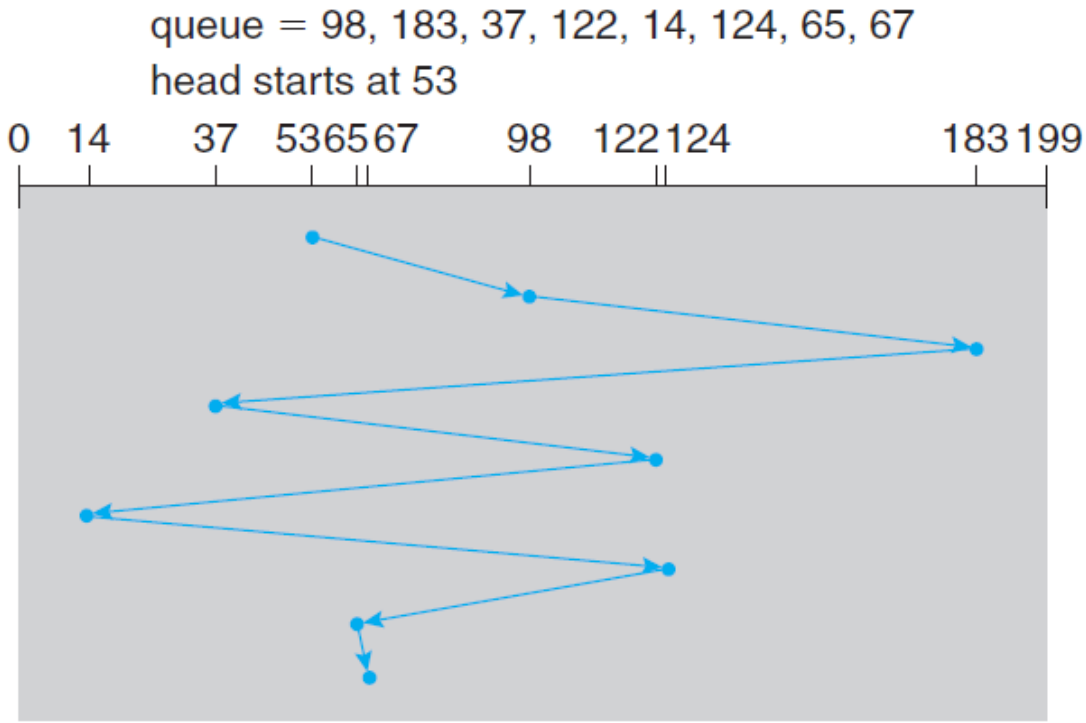
- There are many sources of disk I/O request, e.g., OS, system processes, users processes
- OS maintains queue of requests, per disk or device.
- Idle disk can immediately work on I/O request, busy disk means work must queue.
- Note that drive controllers have small buffers and can manage a queue of I/O requests.

Disk Scheduling Algorithms

- First Come First Serve (FCFS)
- Shortest Seek Time First (SSTF)
- SCAN
- C-SCAN
- C-Look

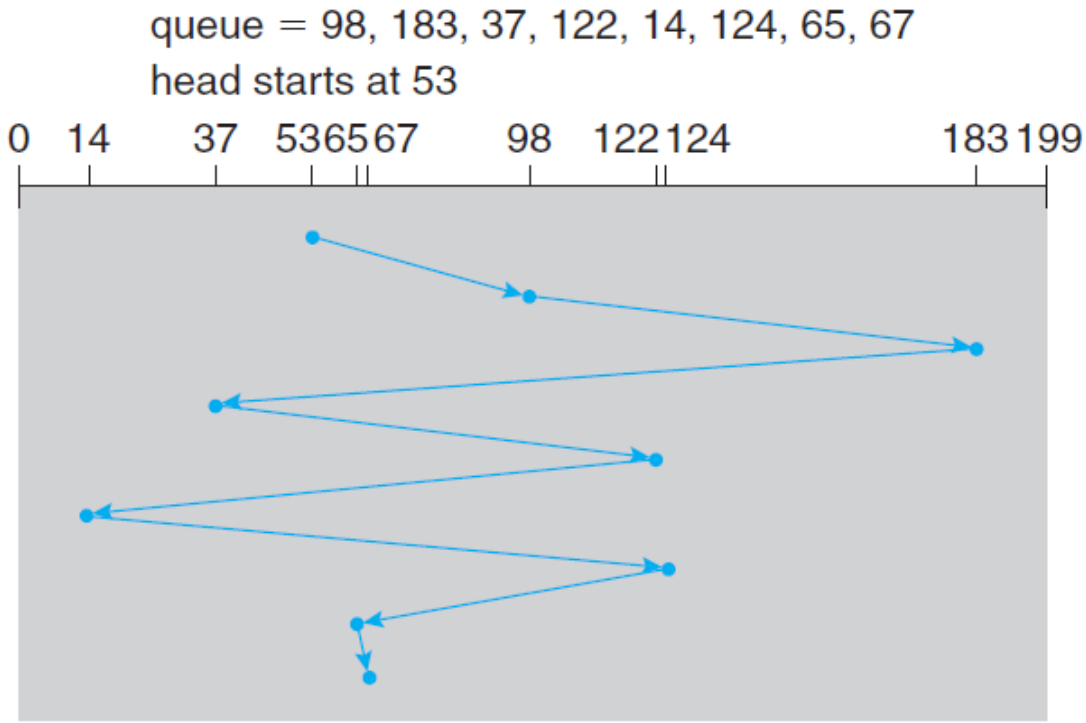
FCFS

- Request queue (0-199): 98, 183, 37, 122, 14, 124, 65, 67
- Head pointer 53



FCFS

- Request queue (0-199): 98, 183, 37, 122, 14, 124, 65, 67
- Head pointer 53
- Total head movement: 640 cylinders



SSTF

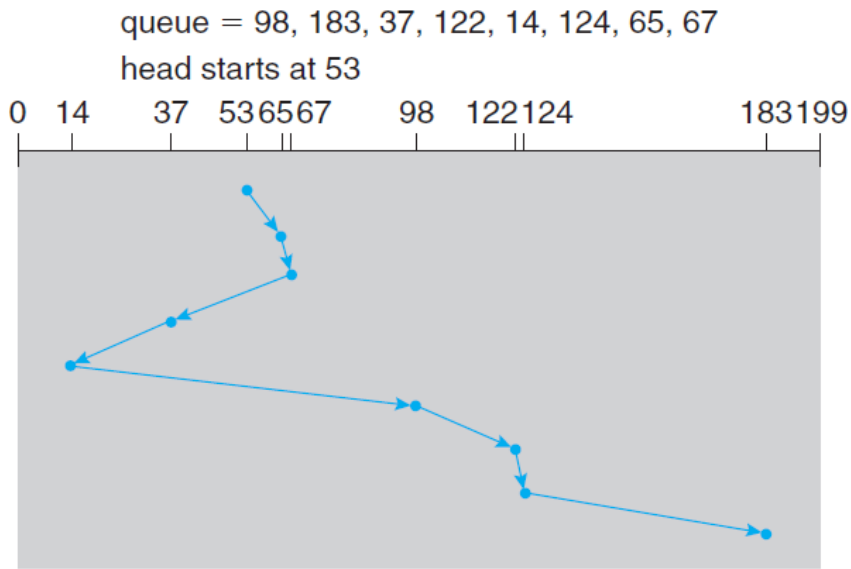
- Selects the request with the minimum seek time from the current head position.

SSTF

- 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.

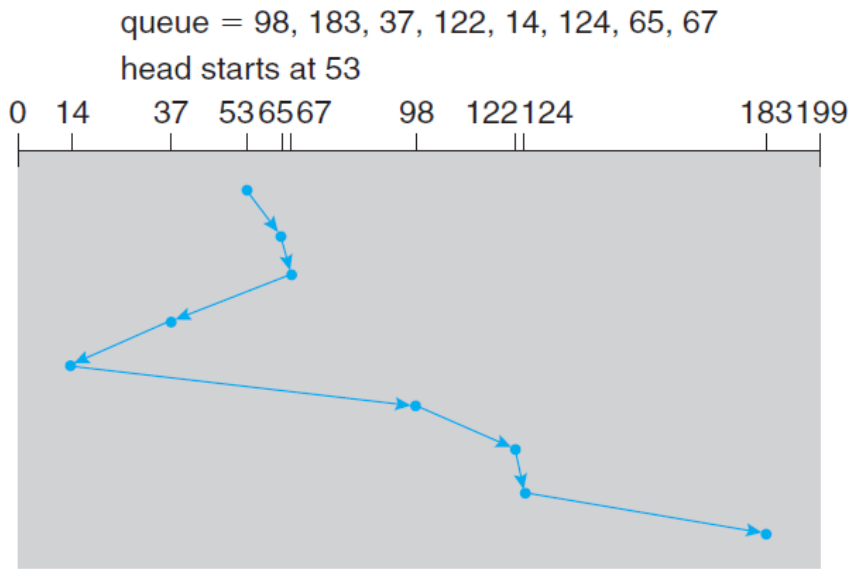
SSTF

- 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.



SSTF

- 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.
- Total head movement: 236 cylinders.

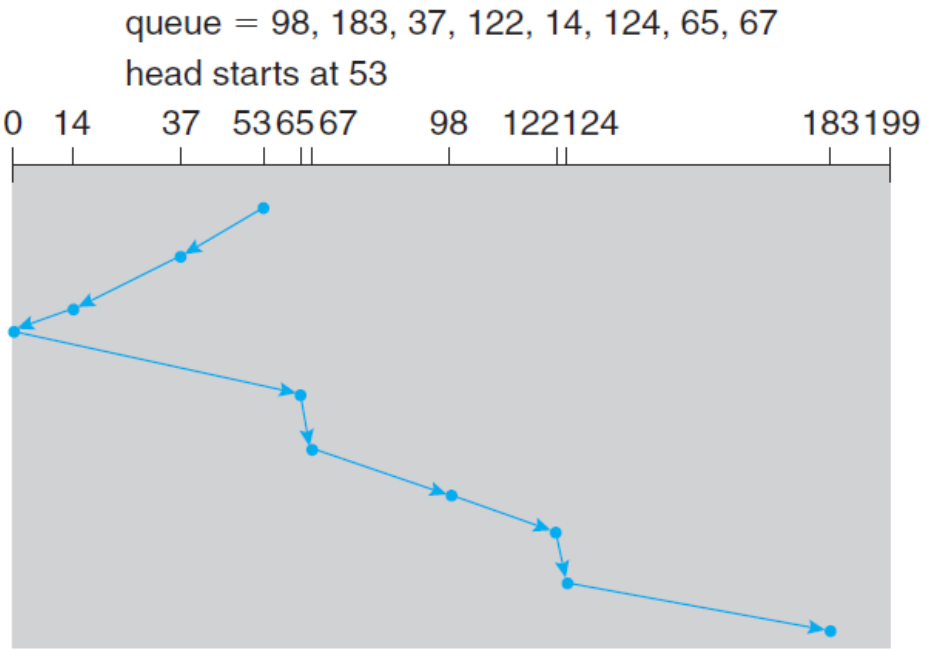


SCAN

- Starts from one end of the disk, and moves toward the other end.
  - Servicing requests until it gets to the other end of the disk.
  - At the end of the dist, the head movement is reversed and servicing continues.
- SCAN algorithm sometimes is called the elevator algorithm.

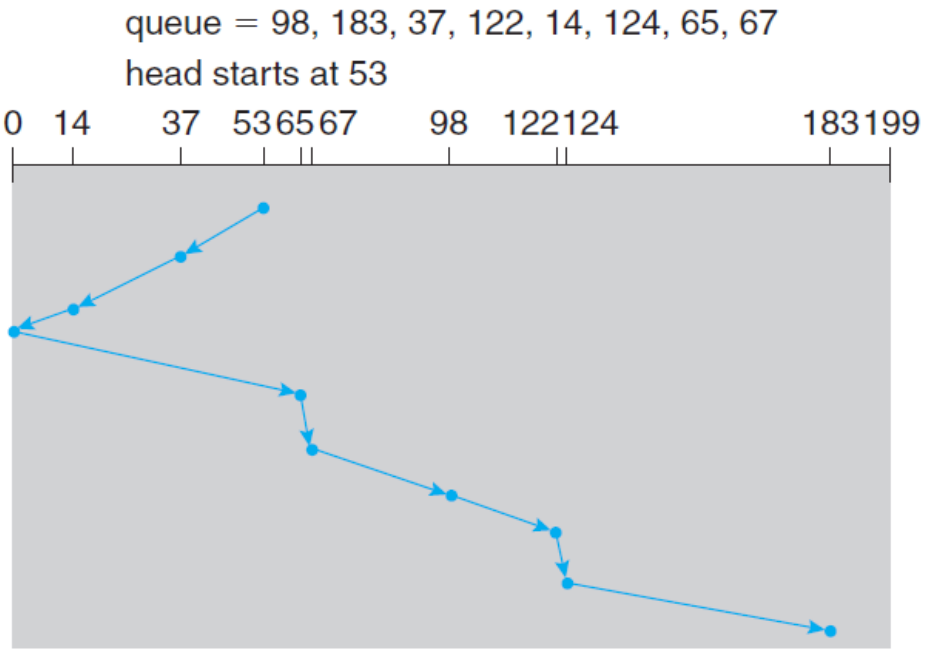
SCAN

- Starts from one end of the disk, and moves toward the other end.
  - Servicing requests until it gets to the other end of the disk.
  - At the end of the dist, the head movement is reversed and servicing continues.
- SCAN algorithm sometimes is called the elevator algorithm.



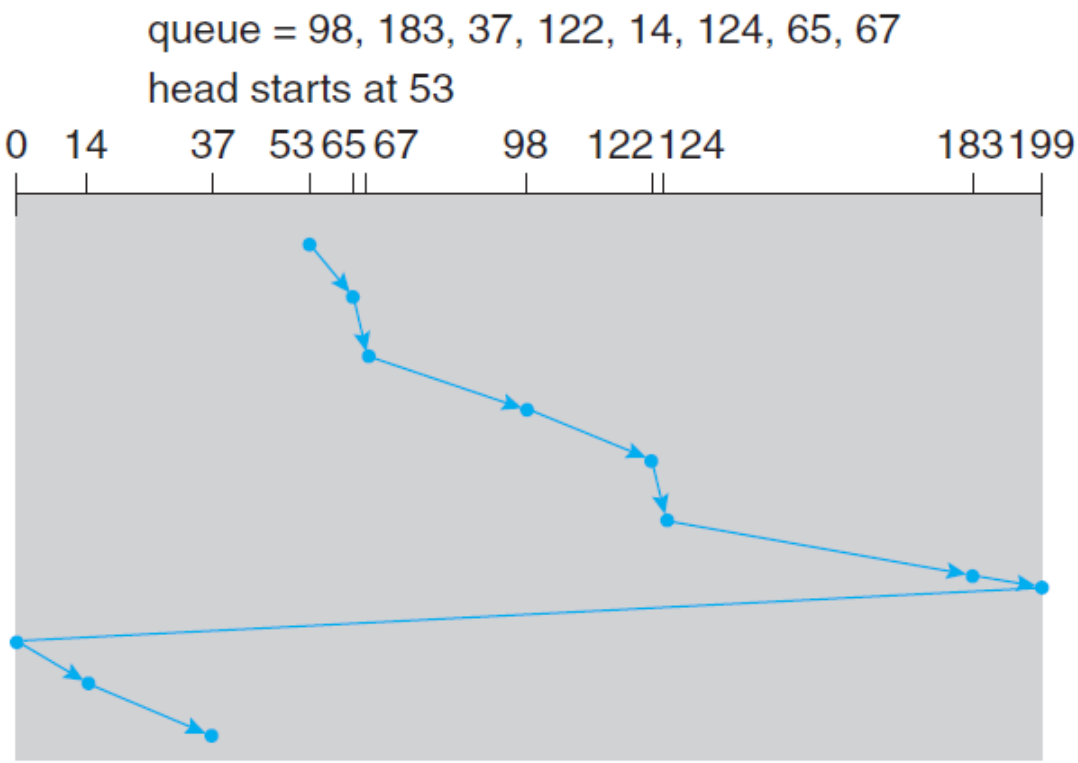
SCAN

- Starts from one end of the disk, and moves toward the other end.
  - Servicing requests until it gets to the other end of the disk.
  - At the end of the dist, the head movement is reversed and servicing continues.
- SCAN algorithm sometimes is called the elevator algorithm.
- Total head movement: 236 cylinders



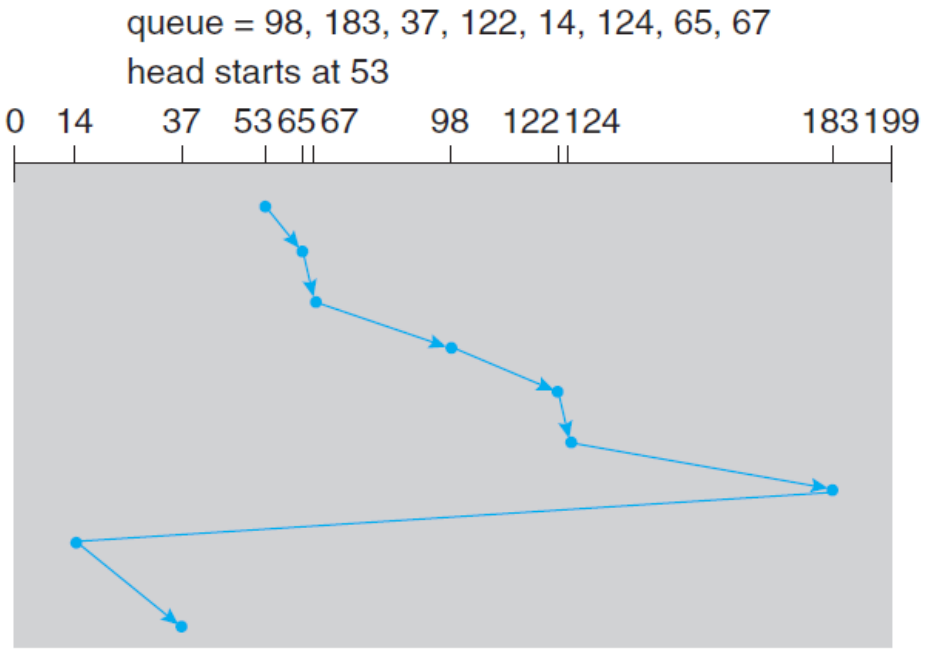
C-SCAN

- Provides a more uniform wait time than SCAN.
- When it reaches the end, it immediately returns to the beginning of the disk without servicing any requests on the return trip.



Look

- LOOK is a version of SCAN, C-LOOK is a version of C-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.



Selecting a Disk-Scheduling Algorithm

- SSTF is common and has a natural appeal: good performance
- SCAN and C-SCAN perform better for systems that place a heavy load on the disk: less starvation

Selecting a Disk-Scheduling Algorithm

- SSTF is common and has a natural appeal: good performance
- 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.

Disk Management

Disk Management

- Disk formatting
- Boot block
- Bad blocks

Disk Formatting (1/2)

- 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, data, and error correction code (ECC).
- Usually 512 bytes of data but can be selectable.

Disk Formatting (2/2)

- To use a disk to hold files, the OS needs to record its own data structures on the disk.

Disk Formatting (2/2)

- To use a disk to hold files, the OS 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.

Disk Formatting (2/2)

- To use a disk to hold files, the OS 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.

Disk Formatting (2/2)

- To use a disk to hold files, the OS 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.
- Raw disk
  - Use a disk partition as a large sequential array of logical blocks, without any file-system data structures.

Boot Block

- The bootstrap program: initializes a computer when it is powered up and starts the OS.
  - E.g., CPU registers, device controllers, the contents of main memory

Boot Block

- The bootstrap program: initializes a computer when it is powered up and starts the OS.
  - E.g., CPU registers, device controllers, the contents of main memory
- Stored in read-only memory (ROM)

Boot Block

- The bootstrap program: initializes a computer when it is powered up and starts the OS.
  - E.g., CPU registers, device controllers, the contents of main memory
- Stored in read-only memory (ROM)
- A tiny bootstrap loader program in the boot ROM: brings in a full bootstrap program from disk.

Boot Block

- The bootstrap program: initializes a computer when it is powered up and starts the OS.
  - E.g., CPU registers, device controllers, the contents of main memory
- Stored in read-only memory (ROM)
- A tiny bootstrap loader program in the boot ROM: brings in a full bootstrap program from disk.
- The full bootstrap program is stored in the boot blocks at a fixed location on the disk.

Bad Blocks

- The controller maintains a list of bad blocks on the disk.
- The list is initialized during the low-level formatting at the factory and is updated over the life of the disk.

Bad Blocks

- The controller maintains a list of bad blocks on the disk.
- The list is initialized during the low-level formatting at the factory and is updated over the life of the disk.
- Sector sparing or forwarding: replacing each bad sector logically with one of the spare sectors.

Bad Blocks

- The controller maintains a list of bad blocks on the disk.
- The list is initialized during the low-level formatting at the factory and is updated over the life of the disk.
- Sector sparing or forwarding: replacing each bad sector logically with one of the spare sectors.
  - E.g., read logical block 87.

Bad Blocks

- The controller maintains a list of bad blocks on the disk.
- The list is initialized during the low-level formatting at the factory and is updated over the life of the disk.
- Sector sparing or forwarding: replacing each bad sector logically with one of the spare sectors.
  - E.g., read logical block 87.
  - The controller find block 87 is bad based on its ECC, and reports it to the OS.

Bad Blocks

- The controller maintains a list of bad blocks on the disk.
- The list is initialized during the low-level formatting at the factory and is updated over the life of the disk.
- Sector sparing or forwarding: replacing each bad sector logically with one of the spare sectors.
  - E.g., read logical block 87.
  - The controller find block 87 is bad based on its ECC, and reports it to the OS.
  - After rebooting, the controller replaces the bad sector with a spare.

Bad Blocks

- The controller maintains a list of bad blocks on the disk.
- The list is initialized during the low-level formatting at the factory and is updated over the life of the disk.
- Sector sparing or forwarding: replacing each bad sector logically with one of the spare sectors.
  - E.g., read logical block 87.
  - The controller find block 87 is bad based on its ECC, and reports it to the OS.
  - After rebooting, the controller replaces the bad sector with a spare.
  - Whenever the system requests logical block 87, the request is translated into the replacement sector's address by the controller.

Swap-Space Management

Swap-Space Management

- Swap-space: virtual memory uses disk space as an extension of main memory.

Swap-Space Management

- Swap-space: virtual memory uses disk space as an extension of main memory.
- Less common now due to memory capacity increases.

Swap-Space Management

- Swap-space: virtual memory uses disk space as an extension of main memory.
- Less common now due to memory capacity increases.
- It is safer to over-estimate than to under-estimate the amount of swap space

Swap-Space Location

- A swap space can reside in one of two places.

Swap-Space Location

- A swap space can reside in one of two places.
- It can be carved out of the normal file system.

Swap-Space Location

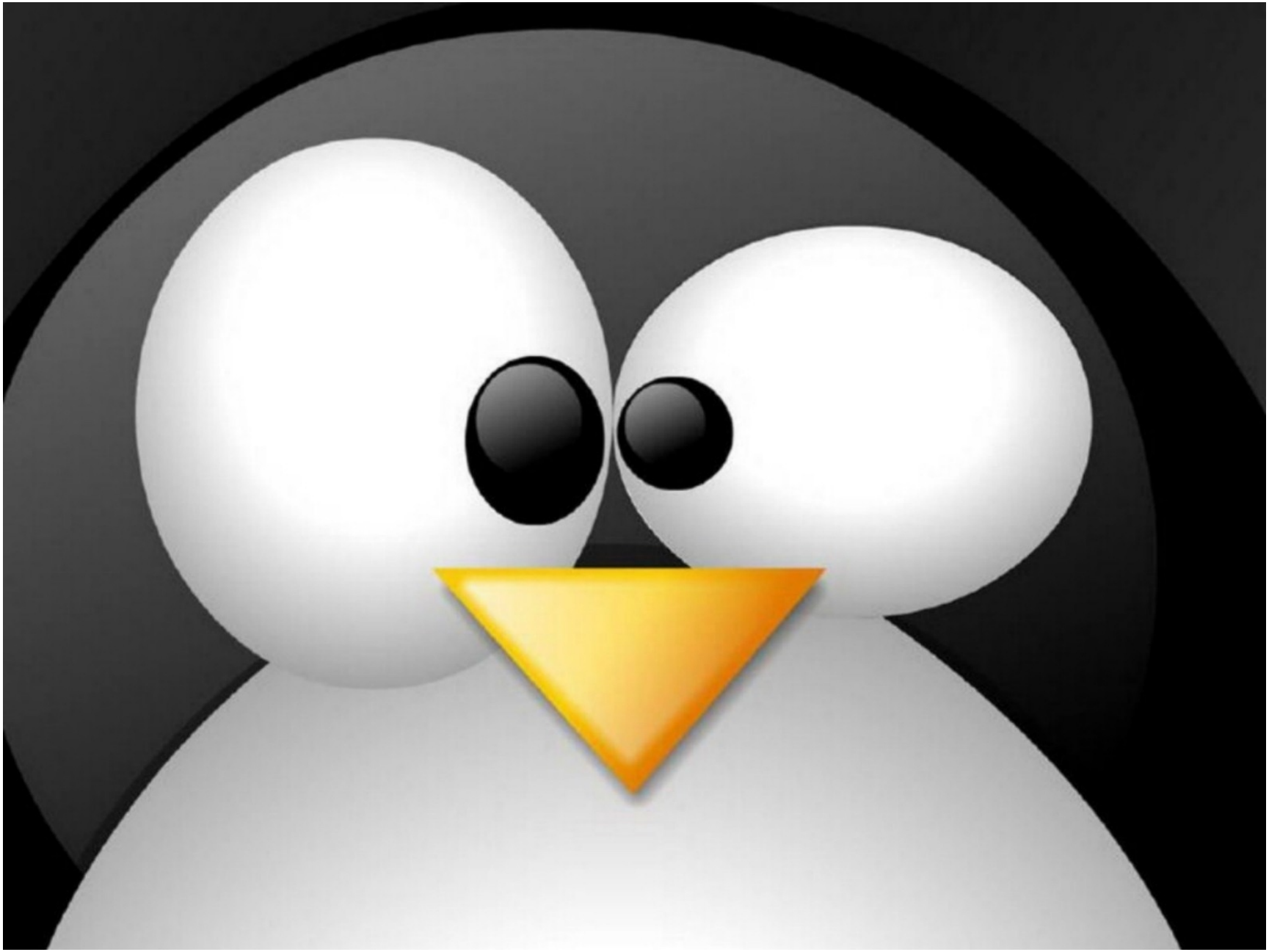
- A swap space can reside in one of two places.
- It can be carved out of the normal file system.
  - A large file within the file system.
  - Normal file-system routines can be used to create it, name it, and allocate its space.
  - Inefficient

Swap-Space Location

- A swap space can reside in one of two places.
- It can be carved out of the normal file system.
  - A large file within the file system.
  - Normal file-system routines can be used to create it, name it, and allocate its space.
  - Inefficient
- It can be in a separate disk partition.

Swap-Space Location

- A swap space can reside in one of two places.
- It can be carved out of the normal file system.
  - A large file within the file system.
  - Normal file-system routines can be used to create it, name it, and allocate its space.
  - Inefficient
- It can be in a separate disk partition.
  - A separate raw partition.
  - Optimized for speed rather than for storage efficiency.

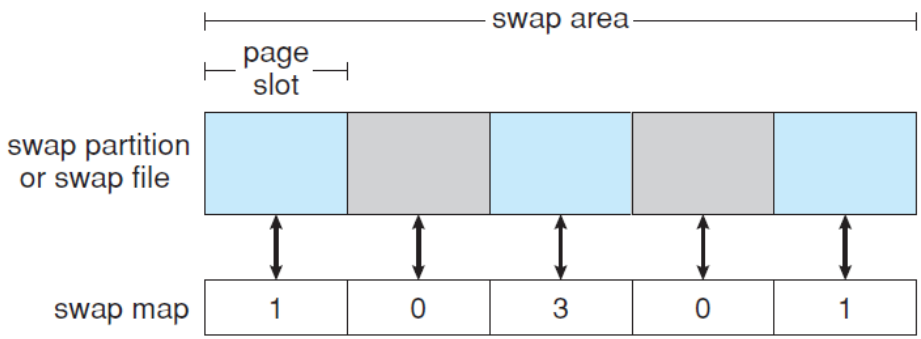


Swap-Space On Linux (1/2)

- Either a swap file on a regular file system or a dedicated swap partition.
- The swap space is used only for anonymous memory.
  - It is more efficient to reread a page from the file system than to write it to swap space and then reread it from there.

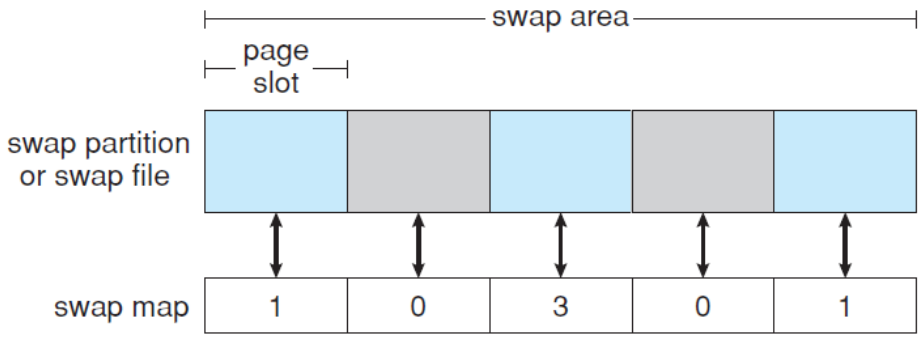
Swap-Space On Linux (2/2)

- Each swap area consists of a series of 4KB page slots, which are used to hold swapped pages.



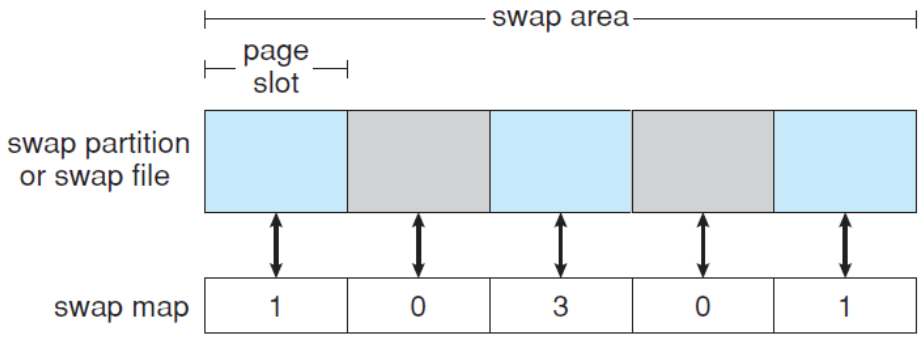
Swap-Space On Linux (2/2)

- Each swap area consists of a series of 4KB page slots, which are used to hold swapped pages.
- Swap map: an array of integer counters associated with each swap area.
  - 0: the page slot is available.
  - > 0: the page slot is occupied.



Swap-Space On Linux (2/2)

- Each swap area consists of a series of 4KB page slots, which are used to hold swapped pages.
- Swap map: an array of integer counters associated with each swap area.
  - 0: the page slot is available.
  - > 0: the page slot is occupied.
- The value indicates the number of mappings to the swapped page.
  - E.g., a value of 3 indicates that the swapped page is mapped to three different processes.



RAID Structure

RAID Structure

- RAID: redundant array of inexpensive disks

RAID Structure

- RAID: redundant array of inexpensive disks
- Multiple disk drives provides reliability via redundancy.

RAID Structure

- RAID: redundant array of inexpensive disks
- Multiple disk drives provides reliability via redundancy.
- Increases the mean time to failure.

RAID Structure

- RAID: redundant array of inexpensive disks
- Multiple disk drives provides reliability via redundancy.
- Increases the mean time to failure.
  - E.g., if the mean time to failure of a single disk is 100,000 hours.

RAID Structure

- RAID: redundant array of inexpensive disks
- Multiple disk drives provides reliability via redundancy.
- Increases the mean time to failure.
  - E.g., if the mean time to failure of a single disk is 100,000 hours.
  - The mean time to failure of some disk in an array of 100 disks will be 100,000/100 = 1,000 hours, or 41.66 days

RAID Structure

- RAID: redundant array of inexpensive disks
- Multiple disk drives provides reliability via redundancy.
- Increases the mean time to failure.
  - E.g., if the mean time to failure of a single disk is 100,000 hours.
  - The mean time to failure of some disk in an array of 100 disks will be 100,000/100 = 1,000 hours, or 41.66 days
  - It is not long at all.

Mirroring

- The simplest approach to introducing redundancy is to duplicate every disk, called mirroring.

Mirroring

- The simplest approach to introducing redundancy is to duplicate every disk, called mirroring.
- A logical disk consists of two physical disks, and every write is carried out on both disks.

Mirroring

- The simplest approach to introducing redundancy is to duplicate every disk, called mirroring.
- A logical disk consists of two physical disks, and every write is carried out on both disks.
- If one of the disks in the volume fails, the data can be read from the other.

Mean Time to Failure of A Mirrored Volume

- Depends on two factors:
  - Mean time to failure of the individual disks.
  - Mean time to repair: to replace a failed disk and to restore the data on it.

Mean Time to Failure of A Mirrored Volume

- Depends on two factors:
  - Mean time to failure of the individual disks.
  - Mean time to repair: to replace a failed disk and to restore the data on it.
- Example
  - The mean time to failure of a single disk: 100,000 hours
  - The mean time to repair: 10 hours
  - The mean time to data loss of a mirrored disk system: $100,000^2/(2 \times 10) = 500 \times 10^6$ hours, or 57,000 years!

Improvement in Performance via Parallelism

- Disk striping uses a group of disks as one storage unit.

Improvement in Performance via Parallelism

- Disk striping uses a group of disks as one storage unit.
- Data striping consists of splitting the bits of each byte across multiple disks.
  - It is called bit-level striping.
  - E.g., if we have an array of eight disks, we write bit i of each byte to disk i.

Improvement in Performance via Parallelism

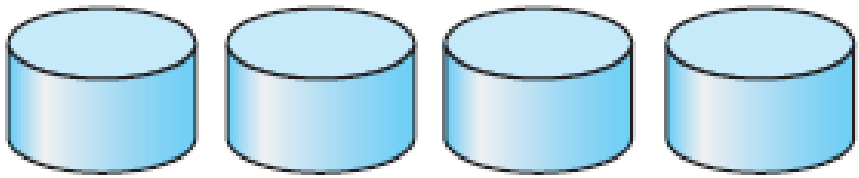
- Disk striping uses a group of disks as one storage unit.
- Data striping consists of splitting the bits of each byte across multiple disks.
  - It is called bit-level striping.
  - E.g., if we have an array of eight disks, we write bit i of each byte to disk i.
- Block-level striping: blocks of a file are striped across multiple disks.
  - E.g., with n disks, block i of a file goes to disk $(i \bmod n) + 1$.

RAID Levels

- RAID is arranged into six different levels.
- RAID schemes improve performance and improve the reliability of the storage system by storing redundant data.

RAID Level 0

- Disk arrays with striping at the level of blocks but without any redundancy.



RAID Level 1

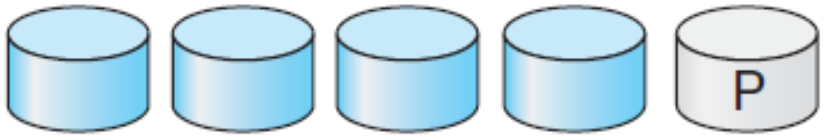
- Disk mirroring

RAID Level 2

- Error-correcting code (ECC)
- Each byte is associated with a parity bit
- It indicates whether the number of bits in the byte set to 1 is even (parity = 0) or odd (parity = 1).

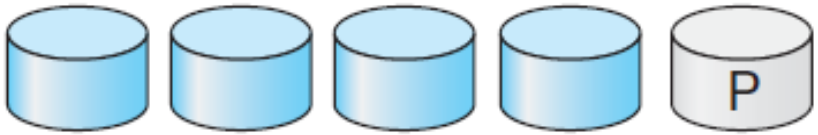
RAID Level 3

- A single parity bit can be used for error detection and correction.
- If one of the sectors is damaged, we know exactly which sector it is
- We can figure out whether any bit in the sector is a 1 or a 0 by computing the parity of the corresponding bits from sectors in the other disks.



RAID Level 4

- Block-level striping, as in RAID 0.
- Keeps a parity block on a separate disk for corresponding blocks from N other disks.



RAID Level 5

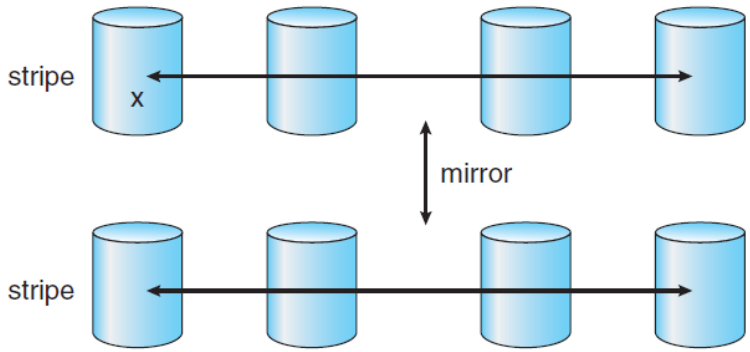
- Spreads data and parity among all N+1 disks, rather than storing data in N disks and parity in one disk.

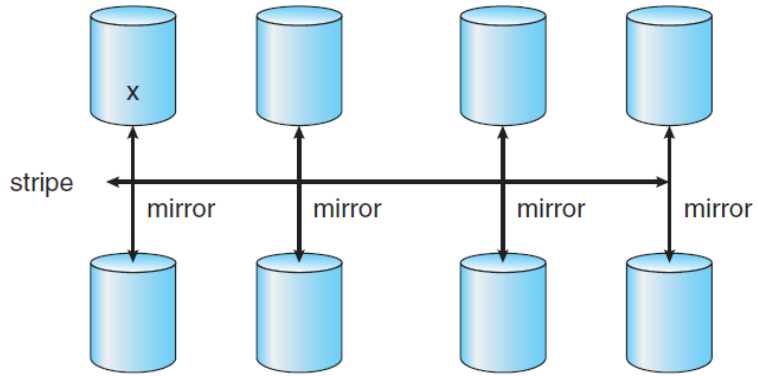
RAID Level 6

- Like RAID level 5 but stores extra redundant information to guard against multiple disk failures.

RAID Level 0 + 1 and 1 + 0

- 0 + 1: RAID 0 provides the performance, while RAID 1 provides the reliability.
- 1 + 0: disks are mirrored in pairs and then the resulting mirrored pairs are striped.





Stable-Storage Implementation

Stable-Storage Implementation (1/4)

- Stable storage: data is never lost (due to failure, etc)

Stable-Storage Implementation (1/4)

- Stable storage: data is never lost (due to failure, etc)
- Write-ahead log (WAL) scheme requires stable storage.

Stable-Storage Implementation (1/4)

- Stable storage: data is never lost (due to failure, etc)
- Write-ahead log (WAL) scheme requires stable storage.
- In a system using WAL, all modifications are written to a log before they are applied.

Stable-Storage Implementation (2/4)

- 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 any failure during data transfer or recovery.

Stable-Storage Implementation (3/4)

- Disk write has 1 of 3 outcomes:

Stable-Storage Implementation (3/4)

- Disk write has 1 of 3 outcomes:
- Successful completion: the data were written correctly on disk.

Stable-Storage Implementation (3/4)

- Disk write has 1 of 3 outcomes:
- Successful completion: the data were written correctly on disk.
- 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.

Stable-Storage Implementation (3/4)

- Disk write has 1 of 3 outcomes:
- Successful completion: the data were written correctly on disk.
- 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.
- Total failure: the failure occurred before the disk write started, so the previous data values on the disk remain intact.

Stable-Storage Implementation (4/4)

- If failure occurs during block write, recovery procedure restores block to consistent state.

Stable-Storage Implementation (4/4)

- If failure occurs during block write, recovery procedure restores block to consistent state.
- System maintains two 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 then second write completes successfully

Summary

Summary

- Mass storage structure: platter, track, sector, cylinder

Summary

- Mass storage structure: platter, track, sector, cylinder
- Disk attachment: host-attached, network-attached, storage-area-network

Summary

- Mass storage structure: platter, track, sector, cylinder
- Disk attachment: host-attached, network-attached, storage-area-network
- Disk scheduling: FCFS, SSTF, SCAN, C-SCAN, C-Look

Summary

- Mass storage structure: platter, track, sector, cylinder
- Disk attachment: host-attached, network-attached, storage-area-network
- Disk scheduling: FCFS, SSTF, SCAN, C-SCAN, C-Look
- Disk management: formatting, boot block, bad blocks

Summary

- Mass storage structure: platter, track, sector, cylinder
- Disk attachment: host-attached, network-attached, storage-area-network
- Disk scheduling: FCFS, SSTF, SCAN, C-SCAN, C-Look
- Disk management: formatting, boot block, bad blocks
- Swap management

Summary

- Mass storage structure: platter, track, sector, cylinder
- Disk attachment: host-attached, network-attached, storage-area-network
- Disk scheduling: FCFS, SSTF, SCAN, C-SCAN, C-Look
- Disk management: formatting, boot block, bad blocks
- Swap management
- RAID: RAID0-RAID6

Questions?

Acknowledgements

Some slides were derived from Avi Silberschatz slides.