

What This Chapter Will Do For You

This chapter explains how data is organized and stored along with classification of data and storage access. Some of the items you will learn about in this chapter include:

- Understanding how data is stored as the foundation for subsequent chapters
 - Data has different value yet it is being managed uniformly, adding to expense
 - How to leverage various storage techniques to reduce storage management costs
-

Data Storage Fundamentals

2.1 Overview

There are two primary components for storage networking: storage and networking. This chapter looks at how information and data are organized and stored in order to be accessed via storage I/O interfaces, the networks in storage networking. For those already familiar with storage fundamentals, you may choose to skip over this chapter; however, you may want to review the material on storage and access classification.

2.2 How data is accessed and organized

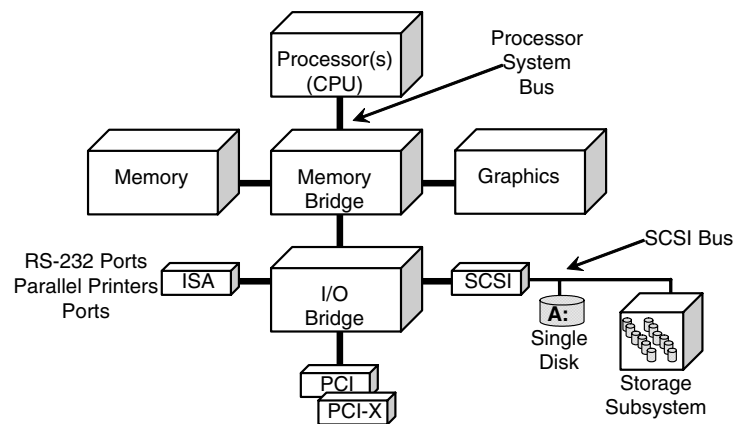
Storage is often taken for granted and is to some perhaps a mystery, yet it is an essential component to supporting information processing. Data can exist in a computer's random access memory (RAM) or saved to some type of recordable storage medium. Storage is an extension of memory, and cache is a convergence of memory and external recordable storage media. Digital data at a low level is stored as 1s and 0s (binary) bits that are implemented using different physical techniques depending on the physical storage medium (disk, tape, optical, solid-state memory—RAM). The data bits are generally grouped into bytes (8 bits) and subsequently organized into larger groups of bytes, words, and other data structures for different purposes. Data is defined into a usable format (data structures) understood by the application and written to and read from files or database objects. In a traditional paper environment you write on, type, or in some other manner transfer data to paper in some organized way (structured data). You might then group related documents and material into a file and store them in a file cabinet. An example of this is shown in Figure 2.1, using a file cabinet example on the right with disk storage shown on the left. In the middle of

and organize data via filesystems and, optionally, volume managers, providing a layer of abstraction, also known as a form of storage virtualization. A volume manager is storage software that is used to organize, group, and manage storage devices. The data flows (3) from the server over a storage interface in a storage format as blocks of data that represent files organized into directories (4) and stored in storage device volumes (5).

In Figure 2.1 the storage volumes represented as a file cabinet would be a storage volume accessible by the server. The folder contains the files and represents a directory structure to keep track of the files and their metadata information. Metadata is the information that describes the file, including location, size, creation and modification dates, security and access control policies, and other information that describes the file. The file contains data being stored, which could be text, graphics, video, audio, or a combination in a format known to the application. On the left in Figure 2.1 a companion view using a storage volume (6) is shown with a folder to represent a directory and files. The storage volume could be an individual disk device or part of a larger storage subsystem accessed via a storage interface.

In Figure 2.2 a typical computer server processor and I/O subsystem architecture is shown. In this example, a processor accesses RAM where data is stored and staged for processing. If data is not in the memory or no longer needed in memory, then an I/O (read or write) is performed to an external storage device. Data is moved in and out of memory or from I/O devices and includes application programs and data to be processed. Data that has been processed is then moved back to memory, moved to a storage device, or displayed using a graphics or video display adapter. The I/O bridge interface is used to move data via I/O data paths called busses, which in turn support devices such as RS-232 serial ports, parallel printer ports,

Figure 2.2
CPU, memory, and storage model.



network interfaces, and storage ports. One of the important things to note about this generic model is that memory is part of the I/O process and I/O is part of the memory subsystem.

2.2.1 Data organization (from bits to bytes)

Data is organized into various data structures that describe data for different purposes, with the smallest unit being a bit. A bit represents a binary value in that it is either set (on) or clear (off), which is represented as a one or a zero. Groups of bits organized into bytes, words, and other data structures are used to store and represent data. Since bits are either on (1) or off (0), multiple bits grouped together as a byte (8 bits) are used to describe characters. For example, 8 bits grouped together as a byte with individual values of 0 0 1 1 0 0 0 1 would represent the character 1 (hexadecimal—base 16, or decimal 49—base 10). This example uses little endian notation, which indicates the order that the sequence of bytes is stored in a computer's memory. IBM mainframes use big endian, while most computer systems utilize little endian (as in the previous example). Character sets include ASCII, which is used by most computer systems today, and EBCDIC for IBM mainframe environments. A Web search on the words ASCII and EBCDIC produces numerous Internet Web sites with various character tables and data conversion utilities.

Another example is the word “byte”, which would require 4 bytes (6 including quotes), and the phrase “bits and bytes”, which would require 14 bytes without quotes. Assuming no formatting characters (including line breaks), a file with “bits and bytes” would only have 14 bytes of data. However, the smallest unit of storage is typically a block or page, which is at least 512 bytes or larger. Consequently, there would be a lot of wasted space if you were to create hundreds of files each with only the phrase “bits and bytes.” This is an oversimplified example of a real problem today, which is that there is a lot of storage that is being allocated but not fully utilized. Many database systems pre-allocate storage to optimize future use. So, while a database may not be full, it still utilizes physical space on a storage volume. Files and data with large amounts of empty or blank space are referred to as being sparse; these files and data can also be compressed to utilize less space—for example, during backups. Typically a file will be made up of many blocks of data, with larger files being made up of hundreds if not thousands of blocks of data or more. Thin provisioning is a new technique that can address traditional storage over provisioning. Thin provisioning enables applications to think they have storage pre-allocated and only require storage capacity for what is actually used.

Table 2.1 *Data Units of Measures*

Unit of Measure	Acronym	Size
Byte	Byte	8 Bits
Block or Page	Blk	512 Bytes
Kilobyte	KB	1,024 Bytes
Megabyte	MB	1 Million Bytes
Gigabyte	GB	1,000 MB
Terabyte	TB	1,000 GB
Petabyte	PB	1,000 TB
Exabyte	EB	1 Million TB
Zetabyte	ZB	1,000 EB

Table 2.1 shows some data units of measures, ranging from a single bit to a byte on up to some storage capacities that may seem unimaginably large. Keep in mind that ten years ago a 1Gbyte disk drive spinning at 5,400 revolutions per minute (RPM) was considered big and fast. Today disk drives are in the 140 to 200Gbyte and larger range spinning at up to 15,000 RPM. So while a petabyte, exabyte, or even zetabyte may seem big today, wait a while and watch how data continues to grow and how your storage gets used.

A closer look at data storage and formats is shown in Figure 2.3, where a file is read or written (1) to a storage device volume (2) as an I/O stream of bits and bytes. The file is stored on disk at a location determined by the file-system as blocks of data that are usually 512 bytes or a multiple of 512 bytes. The file format is determined by the application that created the file, and it could contain text data, graphics, video, or audio. To the storage subsystem, the file is simply a collection of bits and bytes grouped together as blocks of data by the filesystem and application that created the file.

Also shown in Figure 2.3 is a representation of a spinning disk storage medium (3) with platters attached to a spindle powered by a motor to rotate the platters. The physical platters are logically divided and allocated into cylinders, tracks, and sectors as part of the disk subsystem's formatting. Each sector contains one or more blocks of data depending on the blocking factor that is used to format the disk. For example, a block size of 512 bytes would map one to one a 512-byte sector to a 512-byte block of data, which

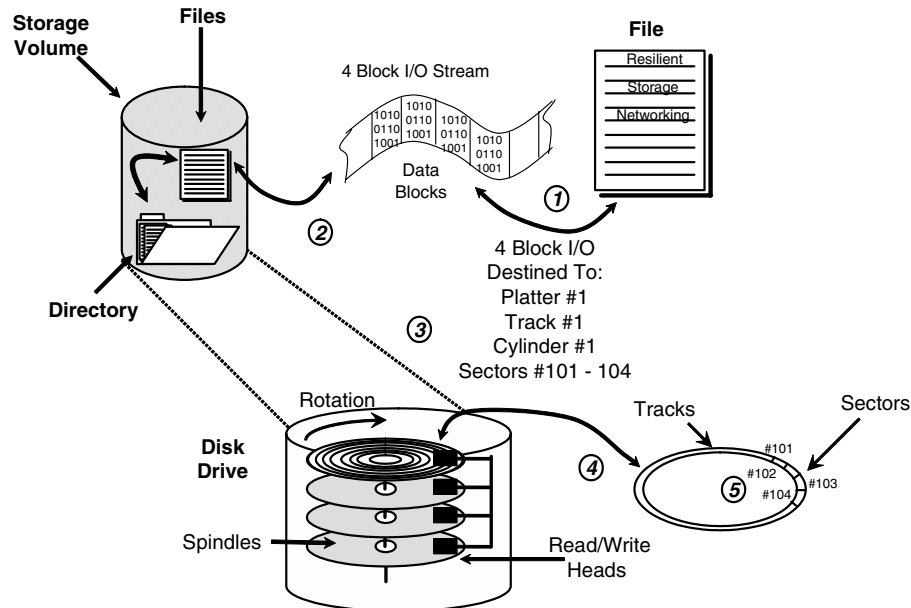


Figure 2.3 *Data storage format and organization.*

in turn maps to a 512-byte page of memory. The blocking factor could be a higher value—for example, 1,024, 2,048, 8,096, or higher for larger volumes (also known as the disk's cluster size). The cluster size is not to be confused with high-availability clustering of servers; rather, the cluster size of the disk determines how many blocks of data will be grouped together. For larger capacity storage devices, a larger disk cluster size is needed to keep track of all the data blocks. Data is accessed on the disk by a physical and logical address, sometimes known as Physical Block Number (PBN) or Logical Block Number (LBN). The filesystem or an application performing direct (raw) I/O keeps track of what storage is mapped to which logical blocks on which storage volumes. Within the storage controller and disk drive, a mapping table is maintained to associate logical blocks with physical block locations on the disk or other media—for example, tape.

Some operating systems also use different block sizes, while others use variable block sizes—for example, IBM mainframes, Count Key Data (CKD), and Extended Count Key Data (ECKD). These environments are referred to as variable-block architectures. Environments that have fixed block sizes are referred to as fixed-block architectures, which would apply to open systems and Windows platforms. For fixed-block architecture, the typical block size is 512 bytes; however, this can vary depending on the

operating system and other influences, including the size of the device. For example, a storage device might be formatted for 528-byte blocks (sectors) at a low level yet present as 512 bytes, with the extra bytes being used for parity protection and other things.

2.2.2 Storage access and classes of storage

Not all data is equal in value and importance, yet generally all information is treated equally in the way it is stored and managed. While the cost of storage is decreasing, more can be done to leverage the costs. In Chapter 1, life-cycle management and data classification were covered briefly, so let's expand on these a bit to help better understand where and how to store data. It is important to understand which is the best or applicable interface and access method to use. In Chapter 3, various access methods are discussed, including Network Attached Storage (NAS), also known as file sharing; block, also known as Storage Area Network (SAN); and Direct Attached Storage (DAS). Also covered in Chapter 3, is object-based storage, known as content addressable storage (CAS). Traditional storage has been accessed via an address, which can be a real or virtual address as to where the data is actually stored.

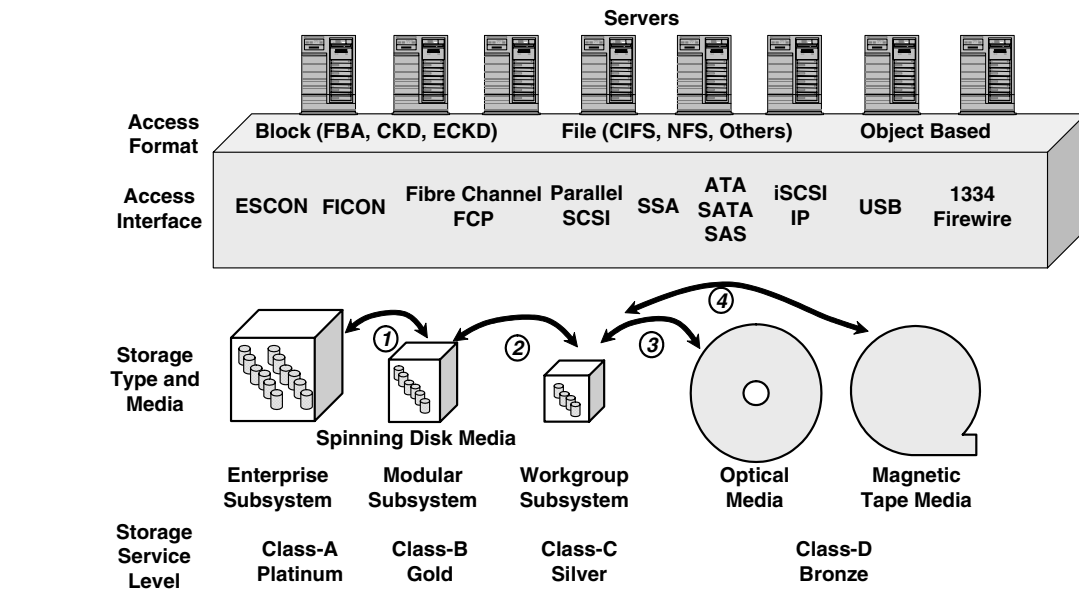


Figure 2.4 Types of storage and access classes.

The example shown in Figure 2.4 represents servers with different applications and operating systems, including open systems and legacy platforms. These servers access storage resources via access methods (block, file, or object) using different access interfaces. Various storage networking interfaces are covered in more detail in Chapter 4. The servers access various classes of storage resources implemented with different storage media types. For example, Class-A in this example represents high-performance, enterprise-class storage. Less frequently accessed data may be migrated (1) to modular storage (Class-B), which may incorporate lower-cost ATA/SATA storage devices. To facilitate backup smaller storage subsystems (2) could be used to support disk to disk to tape (D2D2T) backup to reduce backup windows and expedite recovery times. Less frequently accessed data, along with backup data, could then be migrated (3) to magnetic tape or optical media for long-term retention. This process could be done via manual intervention, in-house developed scripts and software, or vendor-supplied backup and life-cycle management tools.

In Table 2.2 various storage attributes are shown, along with some common classification terminology, including tier, class, and level. Also shown are some general characteristics, along with features and functions for each type of storage and some representative applications. Another way to view the information is by using a pyramid with memory and cache at the top, then Solid-State Disk (SSD), followed by the various tiers (1–4) in decreasing order of performance and cost.

Table 2.2 *Some Storage Classifications and Characteristics*

Tier/Class	Level	Characteristics	Features and Functions
1/A	Platinum	<ul style="list-style-type: none"> On-line storage High performance High availability Advanced functionality Multiple interconnects Enterprise storage Storage sharing Traditionally higher cost 	<ul style="list-style-type: none"> Cache centric (read and write cache) Remote mirroring and replication Many ports for server attachment Monolithic or modular design Redundant components and software SCSI, SSA, and Fibre Channel disk Snapshots and point-in-time copies Some levels of RAID Volume management features Supports enterprise, including open systems and S/390 mainframe

Table 2.2 *Some Storage Classifications and Characteristics (continued)*

Tier/Class	Level	Characteristics	Features and Functions
2/B	Gold	On-line High performance Some advanced features Open interconnect interfaces Potentially lower cost General application usage	Active/Active for redundancy Active/Passive for failover Read cache, write back cache Fewer number of interconnect ports Modular design and packaging Open interconnect ports and protocols (Fibre Channel [FCP], SCSI, iSCSI/GbE) Point-in-time (PIT) copies and snapshots Remote mirroring and replication Redundant components SCSI, Fibre Channel, ATA, and SATA storage devices (back-end) Some levels of RAID Volume management features Supports open systems
3/C	Silver	On-line and near-line storage Alternative to tape media General applications for workgroup, SMB, and departmental environments Useful for long-term data retention and storage of reference data	ATA/SATA disk drives Block-based interface Possible tape (virtual tape) interface Can also include WORM technology Optical media and tape media Fewer number of interconnect ports Good to adequate performance Open systems interface ports Block and File (NAS) access Possible RAID support Random access characteristics Small to large storage capacity Some redundancy capabilities Some volume management features
4/D	Bronze	Off-line and near-line data Removable media Interchangeable media Lowest cost of storage Lower performance Ideal for backup and archive	Random capabilities Random access for optical media Semisequential access for tape Tape can be in standalone device Automated tape and media handling

Table 2.3 *Some Storage Interface and Access Classification Characteristics*

Tier/ Class	Level	Characteristics	Features and Functions
1/A	Platinum	High availability High performance Redundant paths Clustering and failover	Fibre Channel (FCP) and FICON as well as ESCON and SCSI for legacy storage interfaces Full bandwidth for performance Nonblocking and high-speed LAN, Metropolitan Area Network (MAN), WAN, SAN Higher costs
2/A	Gold	Good performance Good availability Single paths (attachment)	Block access NAS file access SCSI, Fibre Channel, iSCSI iSCSI adapters with TCP/IP Offload Engines (TOEs) to enhance performance Moderate-speed long-distance interfaces, including SONET/SDH and IP
3/C	Silver	Moderate performance Lowest cost SMB and workgroup Slower wide area interfaces	iSCSI with Gb Ethernet using standard adapters (NICs) SATA interfaces for disk storage Leverage Ethernet infrastructure
4/D	Bronze	Robotic or possibly manual handling of storage media Slower network interfaces Slowest wide area interfaces	Various interfaces and media types Various media and handling characteristics

Table 2.3 shows various characteristics and classifications for storage access interfaces for local, metropolitan, and wide area situations. Storage interfaces are covered in more detail in the following chapters, along with access methods.

2.3 Chapter summary

Not all data and information are the same in terms of frequency of access and retention, yet typically all data is treated the same. While the cost per unit of storage is decreasing, the amount of storage that can be managed per person is not increasing at a proportional rate, resulting in a storage management efficiency gap. Information data can be stored in different formats

on various types of media and use various interfaces to access it. In Chapter 3, we will take a closer look at how data is accessed using block, file, and object access methods. Storage I/O and networking access interfaces will be covered in subsequent chapters.

