

# Guide to Operating Systems, 6<sup>th</sup> Edition

## Module 4: File Systems

# Learning Objectives

By the end of this module, you should be able to:

- List the basic functions common to modern file systems
- Use and describe the file systems used by Windows OSs
- Use and describe the file systems used by Linux systems, including ufs and ext
- Use and describe the macOS file system



# File System Functions (1 of 2)

- All information stored on a computer's hard disk is managed, stored, and retrieved through a **file system**
  - The file system allocates locations on a disk for storage and keeps a record of where specific information is kept
- Some file systems also implement recovery procedures when a disk area is damaged or when the OS goes down

# File System Functions (2 of 2)

- File systems used by operating systems perform the following general tasks:
  - Provide a convenient interface for users and applications to open and save files
  - Provide a hierarchical structure to organize files
  - Store file metadata to provide detailed information about files
  - Organize space on a storage device

# User Interface

- When a user double-clicks a file to open it, the user interface calls the file system with a request to open the file
- The file type determines exactly how the file is opened
  - If the file is an application, the application is loaded into memory and run by the CPU
  - If the file is a document, the application associated with the document type is loaded into memory and opens the file

# Hierarchical Structure (1 of 5)

- The overall purpose of a file system is to create a structure for filing data
- A **file** is a set of data that is grouped in some logical manner, assigned a name, and stored on the disk
  - Data contained in files can be text, images, music and sounds, video, or Web pages
- OSs typically organize files in a hierarchy of folders or directories
  - The top of the hierarchy is called the “root” of the file system
  - The root of the file system often represents a disk drive or other mass storage device

# Hierarchical Structure (2 of 5)

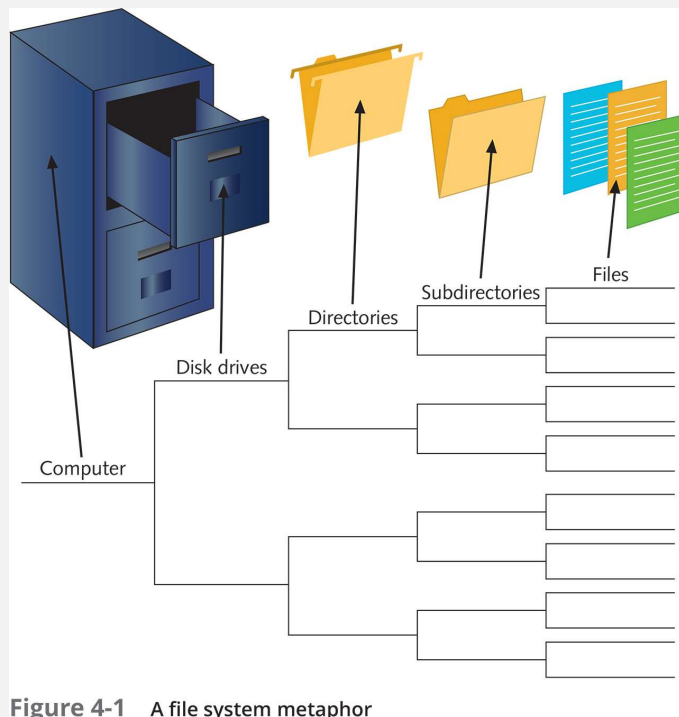


Figure 4-1 A file system metaphor

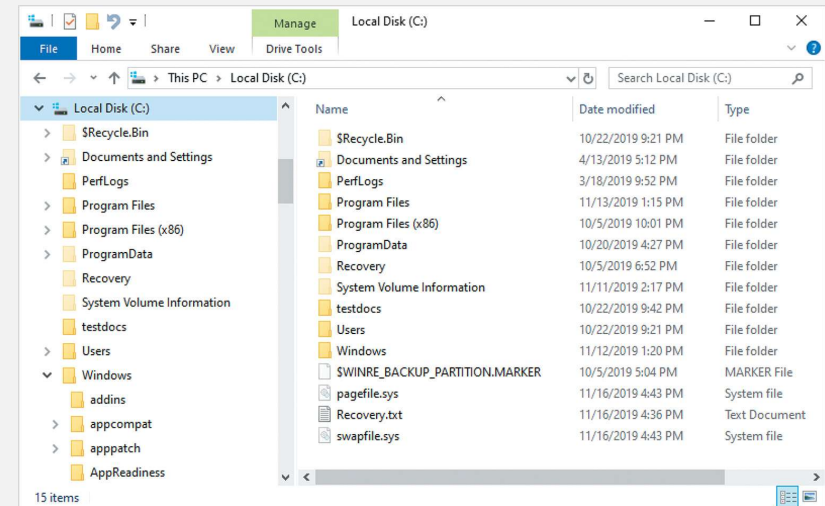


Figure 4-2 The root of the C: drive on a Windows system

# Hierarchical Structure (3 of 5)

- Folders can be organized in a hierarchy that is similar to a tree structure
- A default OS structure might consist of folders for the following:
  - OS files
  - Software applications
  - Individual user files
  - Public files that are available to all users
  - Other folders needed by applications and the administrator



# Hierarchical Structure (4 of 5)

- General best practices when designing a storage and file structure for a server:
  - The OS files should be on a separate hard disk from user files
  - Applications and system data files should be on a separate hard disk from the OS and user files
  - Folders should be organized with access controls in mind
  - Folders should be named to reflect the purpose of the files they contain
- The Linux folder structure is different from the Windows folder structure
  - Figure 4-3 shows the root of the Fedora file system in the Files tool

# Hierarchical Structure (5 of 5)

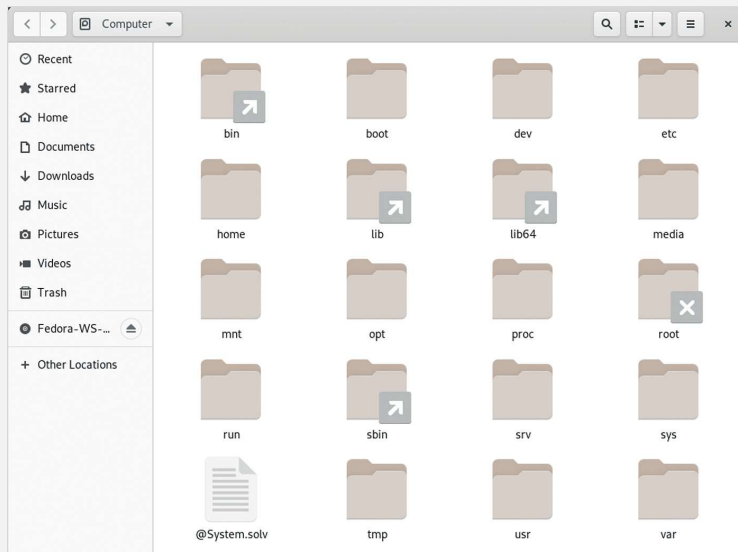


Figure 4-3 The Fedora Linux root folder

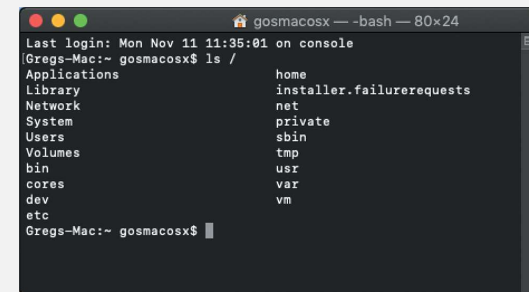


Figure 4-4 The macOS root folder

Source: Apple Inc.

# File Metadata

- Metadata is information that describes the file and its contents but is not the actual data itself
- Information found in metadata:
  - Filename
  - Date and time the file was created
  - Date and time the file was last modified
  - Date and time the file was last accessed
  - Size of the file
  - Folder or file attributes, such as permissions, whether the file is read-only, etc.

# Storage Device Space Organization (1 of 13)

- Hard disks arrive from the manufacturer with low-level formatting
  - A **low-level** format is a software process that marks the location of disk tracks and sectors
  - **Tracks** are like several circles around a disk; each track is divided into sections of equal size called **sectors**
- **Block allocation** divides the disk into logical blocks called **clusters**, which correlate to sectors, heads, and tracks on the disk
  - Called allocation units in Windows systems

# Storage Device Space Organization (2 of 13)

- **Block allocation** is stored using one of two techniques:
  - The file allocation table (FAT) uses a fixed portion of the disk to store this data
  - The New Technology File System (NTFS) and Linux file systems use various locations on the disk to store a special type of file called the Master File Table (MFT) that is used for directory and file allocation information
  - All OSs have tools that let you check, and sometimes repair, common file system and disk problems

root has its own partition

## Storage Device Space Organization (3 of 13)

- **Partitioning** is the process of reserving some or all of a disk to be used by a particular file system, such as FAT or NTFS
- After partitioning, the disk must be **high-level formatted** in order for the OS to store files
- A single disk can contain one partition or more
  - Each formatted partition is usually referred to as a **volume** in most OSs
  - You might want multiple file systems to allow for a dual-boot system where you can boot to Windows 10 or Linux

# Storage Device Space Organization (4 of 13)

- When a partition is created, information about that partition is stored in a special area of the disk known as the **partition table** (in MS-DOS, Mac OS, and Windows) or **disk label** (in UNIX/Linux)
- A traditional BIOS uses the Master Boot Record (MBR) partitioning method
  - It contains a small block of program code used to locate and boot the OS
  - This code is called the bootstrap code or master boot code

# Storage Device Space Organization (5 of 13)

- Most MBRs contain the following elements:
  - The bootstrap code examines the partition table to determine the partition from which to boot (the active partition)
  - An optional timestamp indicates when the MBR was created
  - An optional disk signature identifies the disk
  - The partition table contains up to four entries that describe the partitions
  - A boot signature is set to hexadecimal 55AA and marks the end of the MBR



# Storage Device Space Organization (6 of 13)

- MBR supports two partition types: primary and extended
- A primary partition can be formatted and assigned a drive letter
  - It can also be an **active partition**, which can contain boot code to start an OS
- An **extended partition** cannot be formatted
  - One or more logical drives must be created in the extended partition
  - A **logical drive** can then be formatted and assigned a drive letter, but cannot be active

# Storage Device Space Organization (7 of 13)

- Due to some limitations of MBR, a newer partitioning scheme called **GUID Partition Table (GPT)** was introduced in the late 1990s
- GPT is more reliable than MBR because multiple copies of the partition data are stored in various places on the disk
- GPT also makes use of a **cyclic redundancy check (CRC)** to ensure that no unauthorized changes have been made to the table
  - A CRC is a formula that uses the data bytes of a file or other data object to calculate a 32-bit value that verifies the integrity of the data
- GPT disks are typically used with UEFI firmware

parity drive (recovery in case of drive failure) = status bits to reflect data stored on other drives  
if lost, it's ok (if lost one drive, no data is lost since they know what's on there),  
lost performance

# Storage Device Space Organization (8 of 13)

- Windows supports two types of drive partitioning: basic and dynamic
- **Basic disks** use either the MBR or GPT disk partitioning scheme
  - They do not support disk spanning, disk striping, or RAID configurations
- **Dynamic disks** do not use traditional fixed partitioning schemes such as MBR and GPT
  - Volumes can be extended into free space on any disk drive
  - Spanned volumes can be created
  - Striped (RAID 0), mirrored (RAID 1), and striped with parity (RAID 5) volumes can be created

file gets split (striping across the disk)  
lose a drive = lose data  
good for making a big volume

file gets copied to two disks  
performance (push data faster)  
writes don't suffer  
less storage

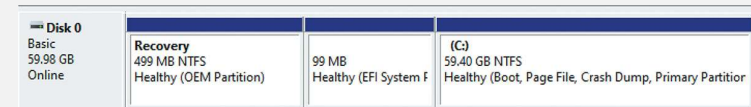
# Storage Device Space Organization (9 of 13)

- Since Windows XP, you can access a volume as a folder on another NTFS volume
- When using the Windows Disk Management utility to format a disk:
  - You have the option to **mount** the disk instead of assigning a drive letter
  - The empty folder into which a volume is mounted is called a **volume mount point**

ex. C:\data is not actually the C drive (no impact)

# Storage Device Space Organization (10 of 13)

- In versions of Windows starting with Windows 2000, you typically partition disks during installation of the OS
  - You can add or reconfigure partitions after the OS is installed



The screenshot shows the Windows Disk Management console. On the left, 'Disk 0' is listed as 'Basic', '59.98 GB', and 'Online'. The main area shows three partitions: a 'Recovery' partition (499 MB NTFS, Healthy (OEM Partition)), an 'EFI System F' partition (99 MB, Healthy (EFI System F)), and a '(C:)' partition (59.40 GB NTFS, Healthy (Boot, Page File, Crash Dump, Primary Partition)).

Partition	File System	Size	Health / Description
Recovery	NTFS	499 MB	Healthy (OEM Partition)
EFI System F	NTFS	99 MB	Healthy (EFI System F)
(C:)	NTFS	59.40 GB	Healthy (Boot, Page File, Crash Dump, Primary Partition)

Figure 4-10 A default Windows 10 volume configuration

# Storage Device Space Organization (11 of 13)

- **Formatting** is the process of placing the file system on a partition
  - When installing a Windows OS, Windows will automatically create and format the necessary partitions on the first hard disk
  - A formatted partition is referred to as a volume in Windows systems
  - You can manually format a hard disk using the *format* command from the Command Prompt window
  - The *format* command includes several switches
    - To view a list of switches, type *format /?*

# Storage Device Space Organization (12 of 13)

- A **disk cluster** is a group of one or more sectors used to store files
- When a file is stored to disk:
  - Data is written to clusters on the disk
  - The filename is stored in the folder, along with the number of the first cluster the data is stored in
  - When the OS fills the first cluster, data is written to the next free cluster and the FAT entry corresponding with the first cluster points to the number of the second cluster used
  - When the second cluster is full, the OS continues with the next free cluster and the FAT entry for the second cluster points to the number of the third cluster used, and so on...

# Storage Device Space Organization (13 of 13)

- When a file is stored to disk (continued):
  - When a file is completely written to the disk, the FAT entry for the final cluster is filled with all 1s (which means the end of the file)
  - This is commonly referred to as the **linked-list** method
- Unusable spots are marked in the FAT as **bad clusters** (never used for file storage)
- Cluster sizes can vary between 512 bytes up to 2 MB
  - A typical cluster size is 4 KB



# Knowledge Check 1

- Each formatted partition on a disk is known as which of the following?
  - A) disk label
  - B) partition table
  - C) sector
  - D) volume

**QUESTION**



# Knowledge Check 1: Answer

- Each formatted partition on a disk is known as which of the following?

- D) volume

**ANSWER**



# Windows File Systems

- Windows OSs support the following file systems:
  - FAT16, extended FAT16, FAT32, and exFAT
  - NTFS
  - ReFS
- All OSs that have a file system support FAT16, but it is mostly used for formatting removable media
- This section focuses on FAT32, exFAT, NTFS, and ReFS

# FAT32 and exFAT (1 of 2)

- Support for FAT32 started with Windows 95 Release 2
  - FAT32 is designed to accommodate larger-capacity disks
- FAT32:
  - Can use disk space more efficiently than FAT16 because it uses smaller cluster sizes
  - The largest volume that can be formatted on Windows systems is 32 GB
  - The maximum file size is 4 GB

# FAT32 and exFAT (2 of 2)

- The exFAT file system, also known as FAT64, is a proprietary file system introduced by Microsoft for mobile personal storage
- exFAT supports volumes up to 128 PB and files up to 16 EB
  - A good choice for high-capacity flash devices
- Support is available for Linux from a third party

# NTFS (1 of 5)

- NTFS is the primary Windows file system for all Windows OSs starting with Windows NT 3.1
- NTFS uses a **Master File Table (MFT)** instead of FAT tables
- The MFT and related files take up about 1 MB of disk space
- When a file is created, a record for that file is added to the MFT
  - The record contains additional attributes such as security settings, ownership, and permissions

## NTFS (2 of 5)

- The MFT record reflects the sequence of clusters that a file uses
- It is possible to have multiple filenames that refer to the same file
  - A technique known as **hard linking**
  - This feature is also available in UNIX/Linux file systems

# NTFS (3 of 5)

- Features incorporated into NTFS:
  - NTFS File and Folder Permissions – NTFS is equipped with security features that meet the US government's C2 security specifications
    - Refers to high-level, “top-secret” standards for data protection, system auditing, and system access
  - NTFS File Compression and Encryption are implemented as file attributes
    - On NTFS volumes, you can enable file compression on the entire volume, a folder and its contents, or a file
    - NTFS volumes use Encrypting File System (EFS) for encryption



# NTFS (4 of 5)

- Features incorporated into NTFS (continued):
  - **Disk quotas** can be used to put a hard limit on the amount of storage a user's files can occupy
  - Volume mount points enable you to access a volume as a folder in another volume instead of using a drive letter
  - Shadow copies are a feature where users can access previous versions of files in shared folders and restore files that have been deleted or corrupted
  - Journaling is the ability to keep a log or journal of file system activity

# NTFS (5 of 5)

- Features incorporated into NTFS (continued):
  - The **hot fix** capability allows NTFS to automatically copy information from one bad area of a disk to another disk area that is not damaged
  - Self-healing NTFS is a utility that runs in the background to correct hard disk problems, making downtime less frequent

# Resilient File System (ReFS)

- ReFS became available starting with Windows Server 2012
  - The main use is in large file-sharing applications where volumes are managed by Storage Spaces
  - ReFS is mostly backward-compatible with NTFS
    - Doesn't support file compression, disk quotas, and EFS (Encrypting File System)
  - Windows can't be booted from an ReFS volume
  - ReFS can correct some types of data corruption automatically

# CDFS and UDF

- Windows OSs since Windows 2000 recognize some additional file systems used by peripheral storage technologies
- **CD-ROM File System (CDFS)** is supported so that OSs can read and write files to DVD/CD-ROM drives
- **Universal Disk Format (UDF)** is used on DVD/CD-ROMs, which in turn are used for large file storage to accommodate movies and games

# Knowledge Check 2

- What NTFS feature enables the user to access a volume as a folder in another volume instead of using a drive letter?
  - A) volume mount points
  - B) shadow copies
  - C) journaling
  - D) self-healing NTFS

**QUESTION**



# Knowledge Check 2: Answer

- What NTFS feature enables the user to access a volume as a folder in another volume instead of using a drive letter?

- A) volume mount points

**ANSWER**



# The Linux File System (1 of 9)

- There are many different file systems that can be used with Linux
  - Some file systems are more “native” to specific Linux operating systems than others
- Most versions of Linux support the UNIX file system (ufs), which is the original native UNIX file system
  - Ufs is a hierarchical file system that is expandable, supports large storage, provides excellent security, and is reliable

# The Linux File System (2 of 9)

- In Linux, the native file system is called the **extended file system (ext or ext fs)**
- Ext is modeled after ufs and enables the use of the full range of built-in Linux commands, file manipulation, and security
- The first ext version had bugs
  - ext2 – reliable file system that handles large disk storage
  - ext3 – added journaling capabilities
  - ext4 – supports file sizes up to 16 TB

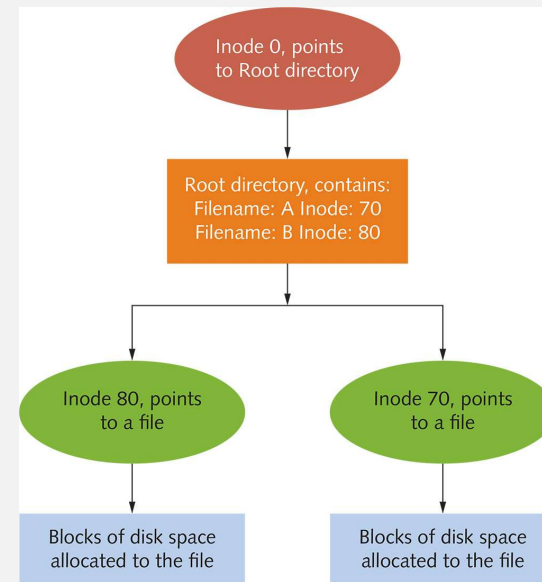


# The Linux File System (3 of 9)

- Both ufs and ext use the same structure, which is built on the concept of information nodes (or **inodes**)
  - Each file has an inode and is identified by an inode number
  - An inode contains general information about the file, such as:
    - User and group ownership, permissions, size and type of file, date the file was created, and the date the file was last modified and read
- Each disk is divided into logical blocks
  - The **superblock** contains information about the layout of blocks, sectors, and cylinder groups on the file system

# The Linux File System (4 of 9)

- The inode does not contain a filename; the filename is stored in a folder
- Several folder entries can point to the same inode
  - Called a hard link



**Figure 4-14** Linux information nodes (inodes) design

# The Linux File System (5 of 9)

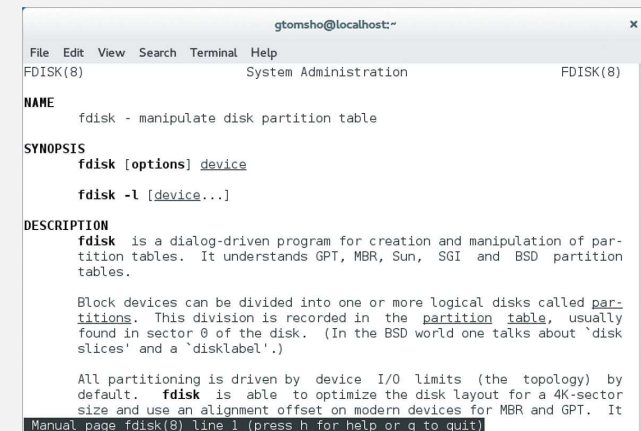
- A Linux system can have many file systems
- Linux uses only mount points, in which each file system is a subfolder of the root
  - All file systems are referred to by a path
  - The path starts out with root (/)
- The *mount* command has several options
  - Typing it without parameters results in a display of the disks currently mounted

# The Linux File System (6 of 9)

- Disks are referenced by a special inode called a device
- There are two types of devices:
  - A raw device has no logical division in blocks
  - A block device does have logical division in blocks
- Devices are normally kept in the /dev or /devices folder
- The **symbolic link** is used to link a directory entry to a file that is on a different partition
  - A symbolic link is merely a pointer to a file

# The Linux File System (7 of 9)

- You must first partition a disk in order to use the Linux file system
- The command to partition the disk differs slightly
  - Most Linux systems use either *fdisk* or *format*
  - Typing *man fdisk* or *man format* at the command prompt gives you an overview of available commands



```
gatomsho@localhost:~  
File Edit View Search Terminal Help  
FDISK(8) System Administration FDISK(8)  
  
NAME  
    fdisk - manipulate disk partition table  
  
SYNOPSIS  
    fdisk [options] device  
    fdisk -l [device...]  
  
DESCRIPTION  
    fdisk is a dialog-driven program for creation and manipulation of partition tables. It understands GPT, MBR, Sun, SGI and BSD partition tables.  
  
    Block devices can be divided into one or more logical disks called partitions. This division is recorded in the partition table, usually found in sector 0 of the disk. (In the BSD world one talks about 'disk slices' and a 'disklabel'.)  
  
    All partitioning is driven by device I/O limits (the topology) by default. fdisk is able to optimize the disk layout for a 4K-sector size and use an alignment offset on modern devices for MBR and GPT. It  
Manual page fdisk(8) line 1 (press h for help or q to quit)
```

Figure 4-17 The man page for the *fdisk* command

# The Linux File System (8 of 9)

- Once a partition is made, a file system can be created
  - You must know the device name of the partition on which you want to create a file system
  - Type *newfs*, followed by the name of the device
  - The *newfs* command is not available in all versions of Linux
    - Use the *mkfs* command instead

# The Linux File System (9 of 9)

- When a file is saved to disk, the system stores part of the data to memory until it has time to write to disk
  - If a computer is shut down prior to data being written to disk, you can end up with a damaged file system
  - Manually force a write of all data in memory by using the *sync* command
- *Fsck* is a utility that verifies the integrity of the superblock, the inodes, all cluster groups, and all directory entries

# MacOS File Systems

- MacOS supports the following file systems:
  - Mac OS Extended
  - Apple File System (APFS)
  - FAT and exFAT



# Mac OS Extended (1 of 3)

no questions on the versions

- Mac OS Extended is used on macOS 10.12 and earlier versions
  - It is a journaling file system, which allows data to be recovered from a journal file if a disk or system problem occurs while data is being updated or modified
- Mac OS Extended supports volume sizes of up to 16 TB
- The first two sectors of a Mac-formatted disk are boot sectors, or boot blocks
- Boot blocks are followed by a **volume information block**, which points to other important areas of information
  - The **catalog-b tree** is the list of all files on the volume and the **extents b-tree** keeps track of the location of file fragments

# Mac OS Extended (2 of 3)

- Mac OS has always supported medium-length filenames (up to 31 characters)
  - The current Mac OS Extended system supports filenames of up to 255 characters
  - Any character may be used in a filename except the colon (:)
  - Macintosh paths are written as colon-separated entities such as:
    - *Hard Drive:System Folder:Preferences:Finder Prefs*
- Mac uses **type codes** and **creator codes** instead of the filename extensions used in Windows

# Mac OS Extended (3 of 3)

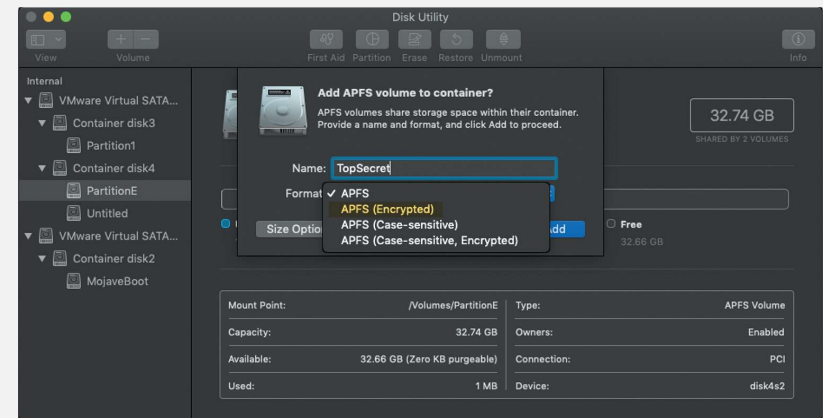
- Macintosh files can contain two parts, or forks:
  - A **data fork** contains frequently changing information (such as word-processing data)
  - A **resource fork** contains information that is fixed (such as a program's icons, menu resources, and splash screens)
- Apple's equivalent to a Windows shortcut is the **alias**
  - The system-level Alias Manager keeps track of the original
  - The word "*alias*" is added to the filename when the alias is created, and the filename is italicized

# Apple File System (APFS) (1 of 2)

- APFS was introduced with macOS 10.13 High Sierra
- Its underlying partitioning structure is GPT, but APFS uses a concept of containers and volumes
  - A **container** is a block of reserved space on a drive that contains one or more volumes
  - The volumes share the available space in a container and are dynamically sized according to the amount of actual space required by the files stored on them (a feature Apple calls **Space Sharing**)

# Apple File System (APFS) (2 of 2)

- APFS supports the following features:
  - Compatibility across devices
  - Encryption
  - Snapshots
    - keeps track of previous versions



**Figure 4-20** Creating an APFS volume on macOS

Source: Apple Inc.

# Summary (1 of 3)

- One of the basic functions of an OS is to enable you to store and access information on a computer or other digital device
- Tasks performed by a file system include providing a convenient user interface, providing a hierarchical structure, storing file metadata, and organizing space on a storage device
- The file system should offer the ability to defragment files, compress file contents, ensure file and data integrity, secure files, and control removable storage media
- Two disk partitioning schemes are MBR and GPT



## Summary (2 of 3)

- The main file systems used in Windows are extended FAT16, FAT32, and NTFS
- The exFAT file system is mainly used for personal mobile storage devices
- NTFS is the native file system for Windows 2000 and later versions
- ReFS became available starting with Windows Server 2012
- Linux supports many different file systems, but typically employs ufs or ext
- Ufs and ext use information nodes (inodes) to organize information about files



# Summary (3 of 3)

- Different varieties of Linux use various file system utilities, such as *fdisk* and *format*, to partition and format disks
- The macOS system supports the Mac OS Extended, APFS, and FAT/exFAT file systems

