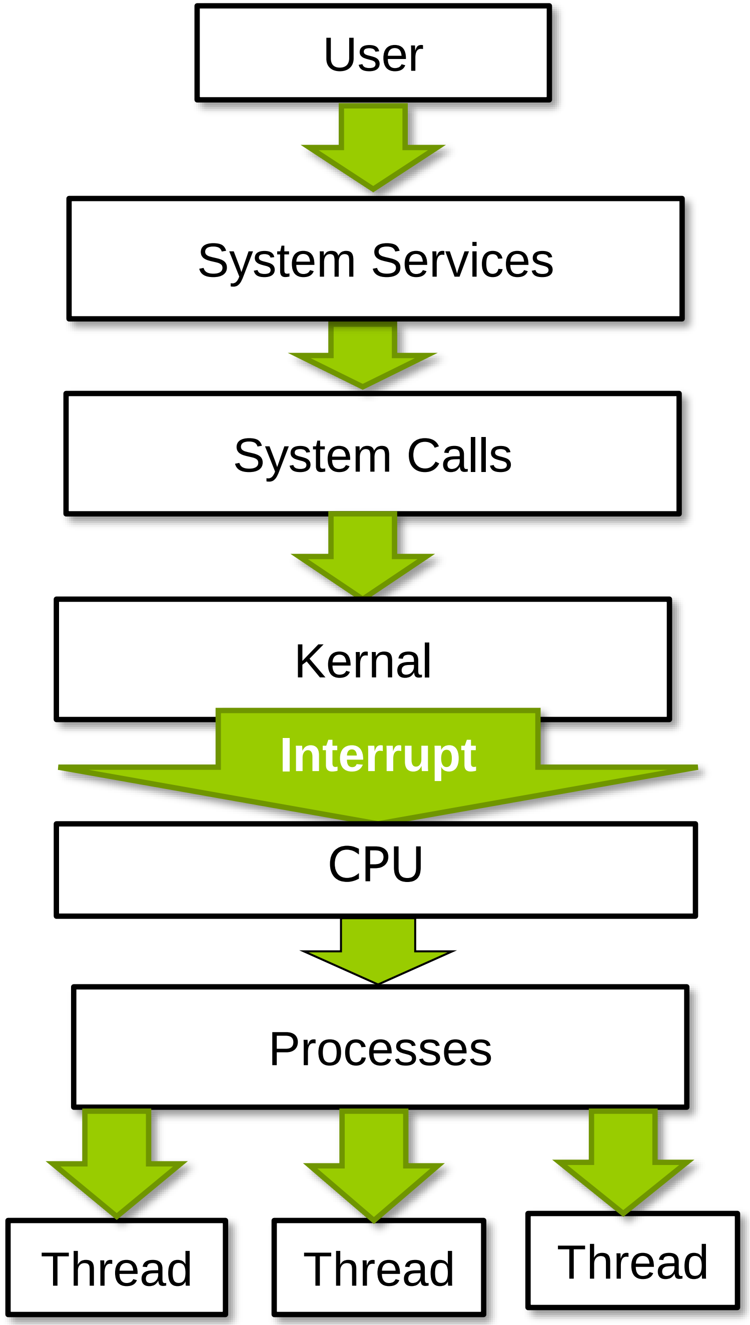
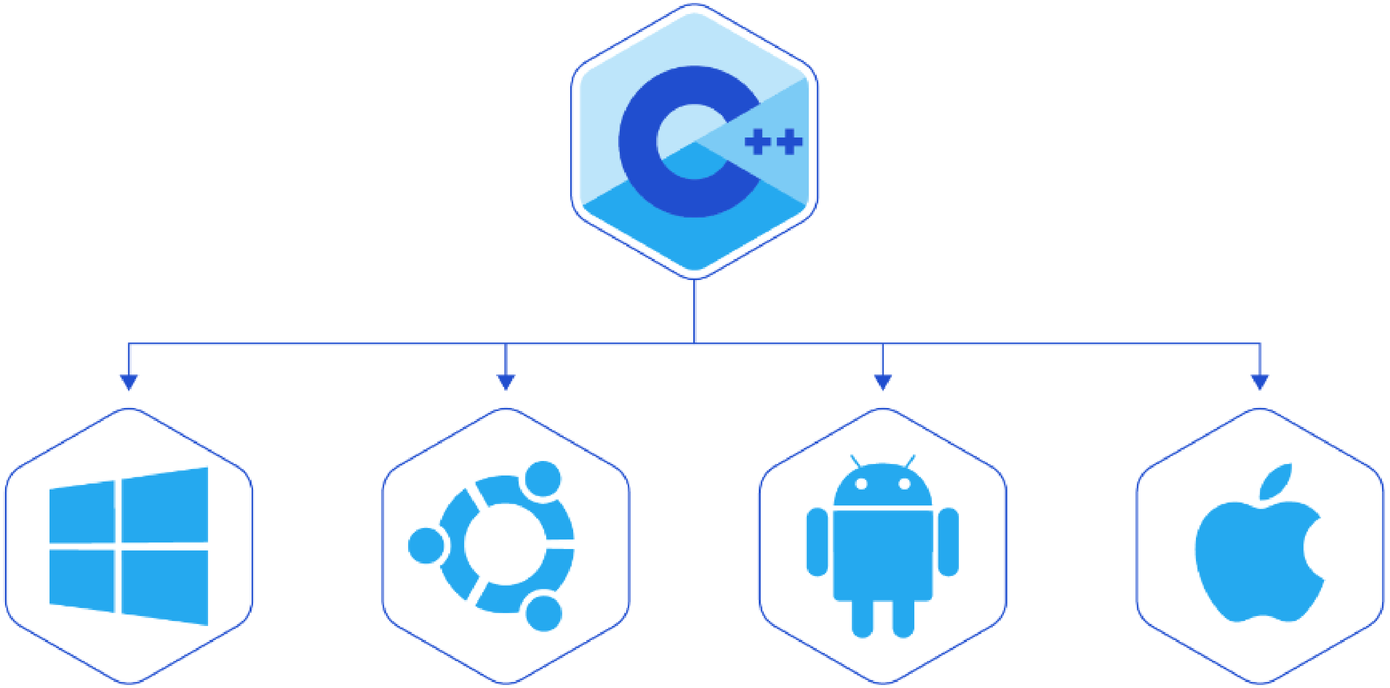
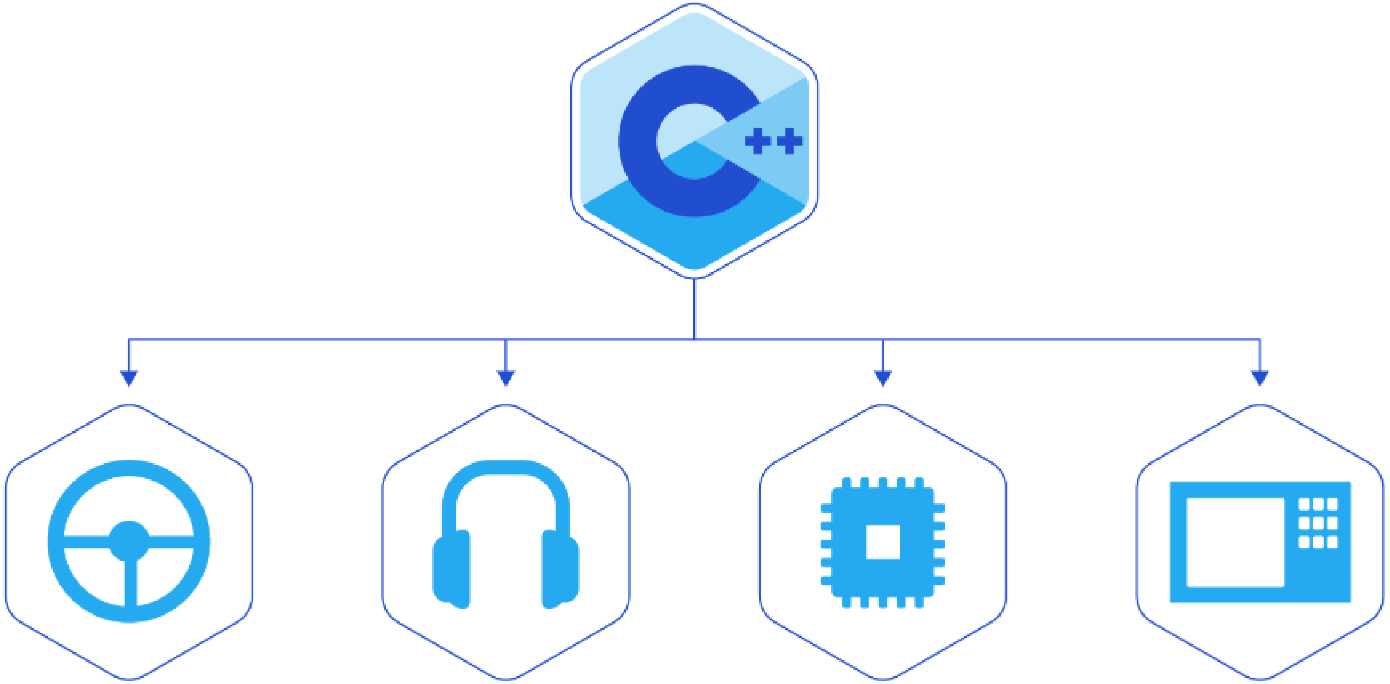
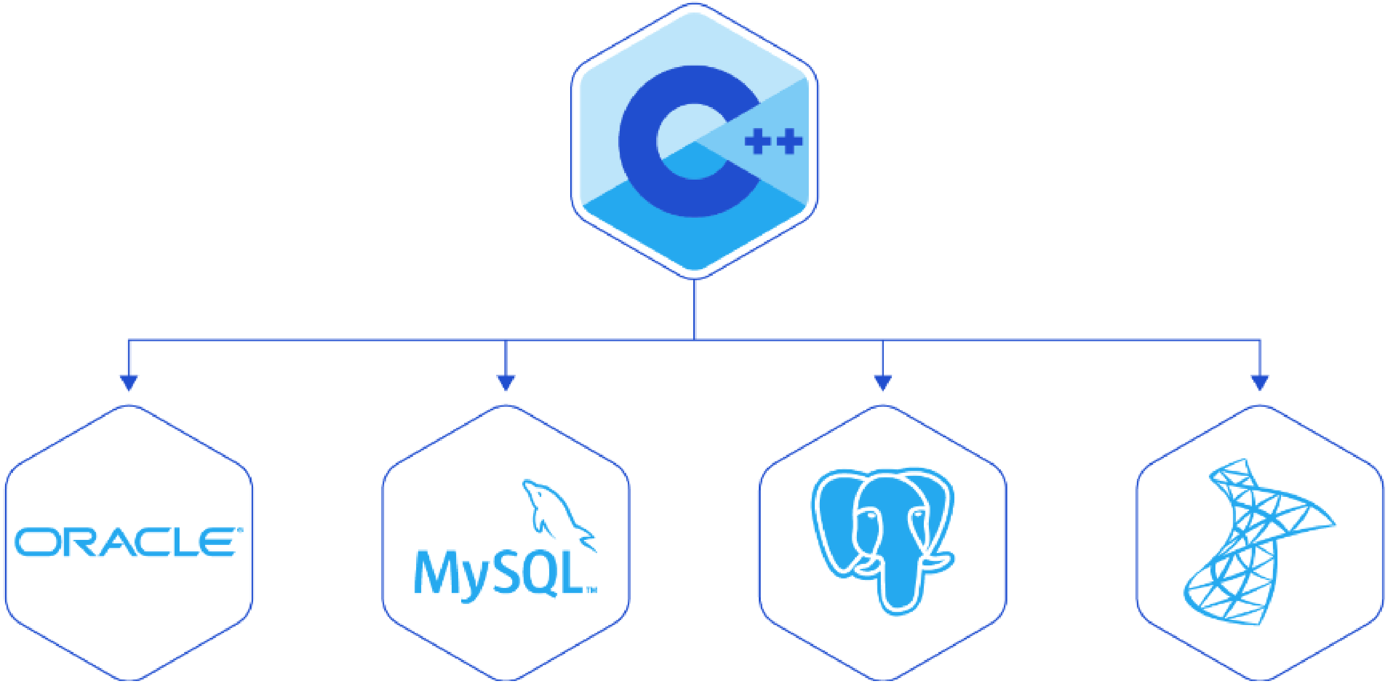
Part 4

The Big Pic of OS








ORACLE
MySQL

Lines of Code!

Number of Lines of Code

Windows XP: 40 million
Windows Vista : 50 million
Windows 7: 40 million
Windows 8: 50 million

Windows 10 : 50 million

Android : 12 Million

Facebook : 100 Million

Mac OS 10.4 : 86 Million

New Car: 100 Million

Google : 2 Billion

- Boeing 787 : 14 Million
- F35 Fighter Jet : 24 Million

Operating-System Structures

- Operating System **Services**
- User and Operating System-**Interface**
- System **Calls**
- **System Services**
- **Linkers and Loaders**
- Why Applications are **Operating System Specific**
- Operating-System **Design and Implementation**
- **Operating System Structure**
- **Building and Booting** an Operating System
- Operating System **Debugging**

Operating System Services

- Operating systems provide an environment **for execution of programs and services to programs and users**
- **One set** of operating-system services provides **functions that are helpful to the user:**
  - **User interface** - Almost all operating systems have a user interface (**UI**).
    - Varies between **Command-Line (CLI)**, **Graphics User Interface (GUI)**, **touch-screen, Batch**
  - **Program execution** - The system must be able to load a program into memory and to run that program, end execution, either normally or abnormally (**indicating error**)
  - **I/O operations** - A running program may require I/O, which may involve a file or an I/O device

- **Command-line interfaces**, where the user provides the input by typing a command string with the computer keyboard and the system provide output by printing text on the computer monitor. Used for system administration tasks etc.
- **Batch interfaces** are non-interactive user interfaces, where the user specifies all the details of the *batch job* in advance to batch processing, and receives the output when all the processing is done. The computer does not prompt for further input after the processing has started.
- **Graphical user interfaces** (GUI), which accept input via devices such as computer keyboard and mouse and provide articulated [graphical] output on the [computer monitor]. There are at least two different principles widely used in GUI design: object-oriented interfaces and application]] oriented interfaces.
- **Touch-screen interface**

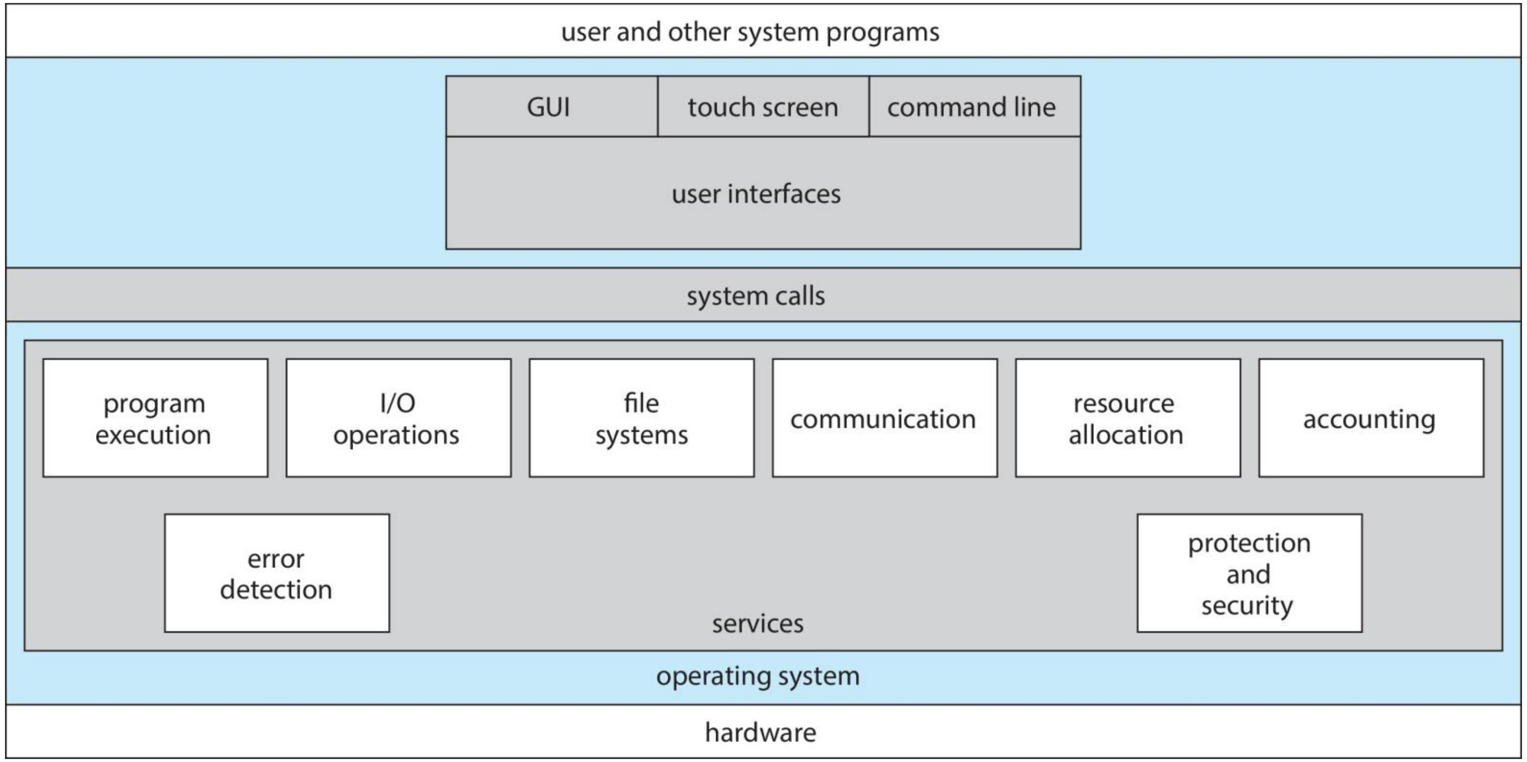
Operating System Services (Cont.)

- One set of operating-system services **provides functions** that are helpful to the user (Cont.):
  - **File-system manipulation** - The file system is of particular interest. Programs need to *read and write files and directories, create and delete them, search them, list file Information, permission management*.
  - **Communications** – Processes may exchange information, on the same computer or between computers over a network
    - Communications may be via shared memory or through message passing (packets moved by the OS)
  - **Error detection** – OS needs to be **constantly aware of possible errors**
    - May occur in the CPU and memory hardware, in I/O devices, in user program
    - For each type of error, OS should take the appropriate action to ensure correct and consistent computing
    - Debugging facilities can greatly enhance the user's and programmer's abilities to efficiently use the system

Operating System Services (Cont.)

- Another set of OS functions exists for ensuring the **efficient operation** of the system itself via **resource sharing**
  - **Resource allocation -** When **multiple users or multiple jobs running concurrently, resources must be allocated to each of them**
    - Many types of resources - CPU cycles, main memory, file storage, I/O devices.
  - **Logging -** To keep track of which users use how much and what kinds of computer resources
  - **Protection and security -** The owners of information stored in a multiuser or networked computer system may want to control use of that information, concurrent processes should not interfere with each other
    - **Protection** involves ensuring that all access to system resources is controlled
    - **Security** of the system from outsiders requires user authentication, extends to defending external I/O devices from invalid access attempts

A View of Operating System Services

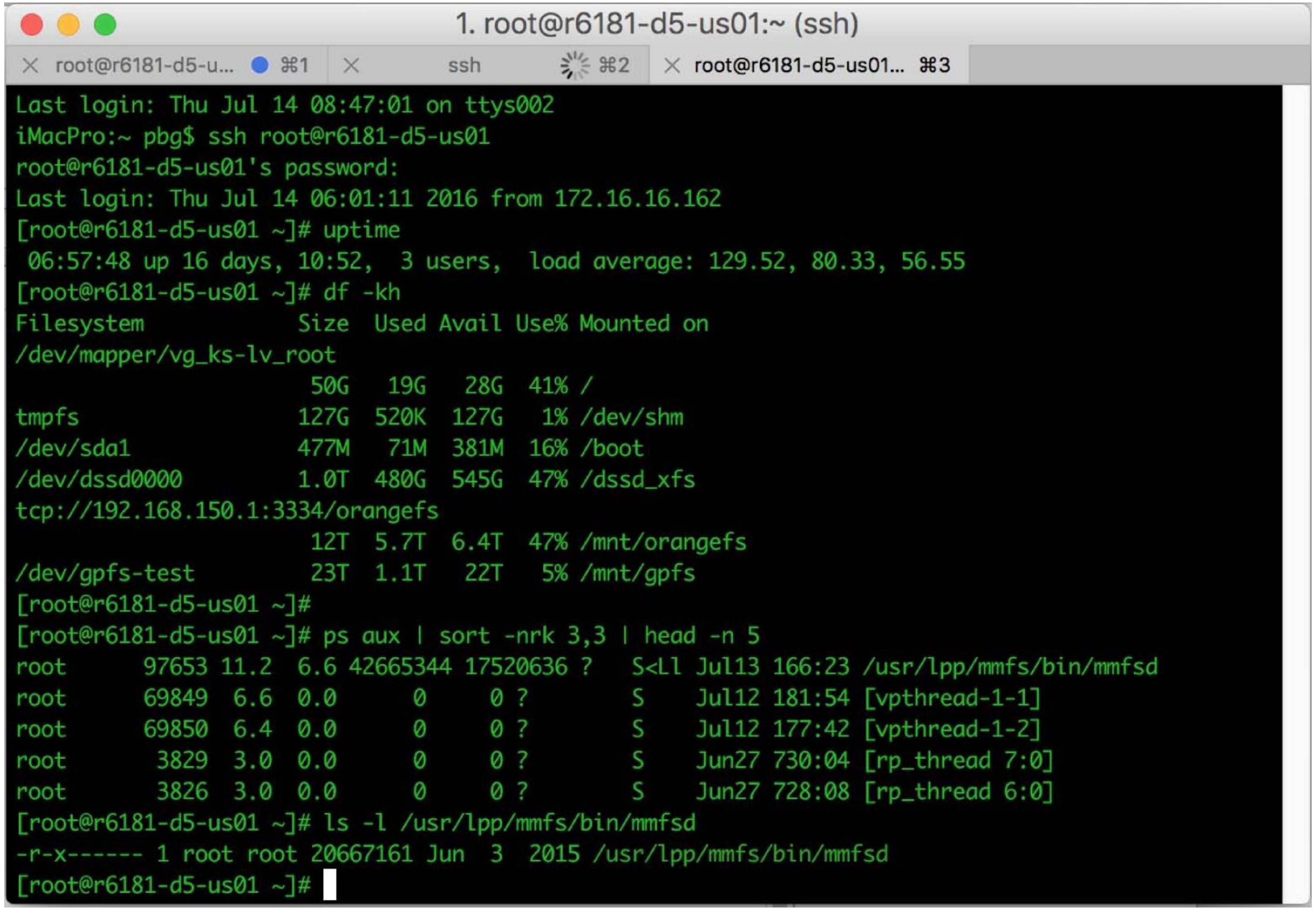


User Operating System Interface - CLI

CLI or **command interpreter** allows direct command entry

- Sometimes **implemented in kernel, sometimes by systems program**
- Sometimes multiple flavors implemented – **shells**
- Primarily fetches a command from user and executes it
- Sometimes commands built-in, sometimes just names of programs
  - If the latter, adding new features doesn’t require shell modification

Shell Command Interpreter



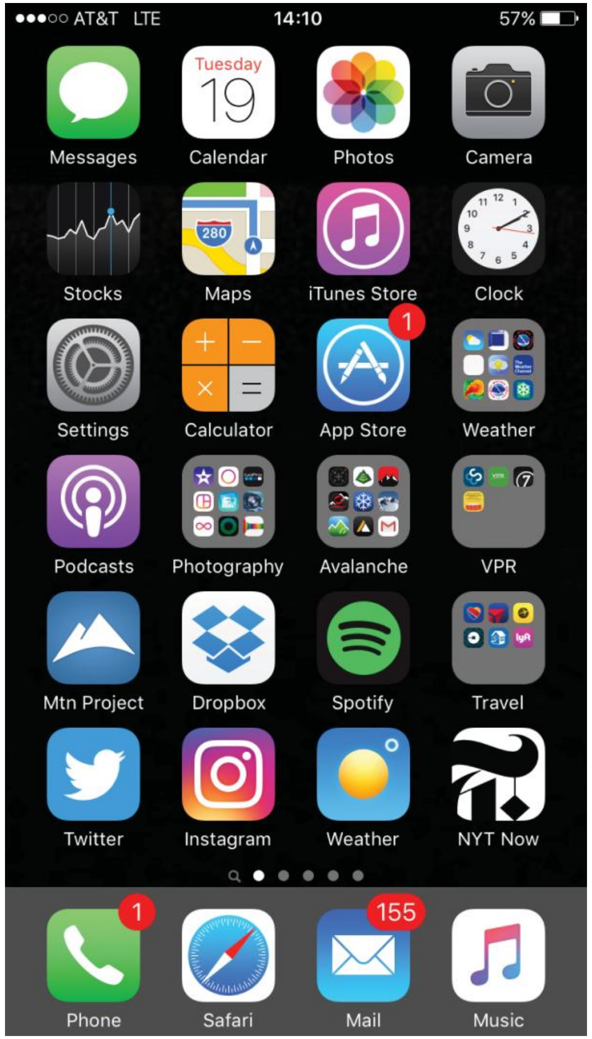
```
Last login: Thu Jul 14 08:47:01 on ttys002
iMacPro:~ pbg$ ssh root@r6181-d5-us01
root@r6181-d5-us01's password:
Last login: Thu Jul 14 06:01:11 2016 from 172.16.16.162
[root@r6181-d5-us01 ~]# uptime
 06:57:48 up 16 days, 10:52,  3 users,  load average: 129.52, 80.33, 56.55
[root@r6181-d5-us01 ~]# df -kh
Filesystem            Size  Used Avail Use% Mounted on
/dev/mapper/vg_ks-lv_root
                       50G   19G   28G  41% /
tmpfs                 127G  520K  127G   1% /dev/shm
/dev/sda1             477M   71M  381M  16% /boot
/dev/dssd0000         1.0T  480G  545G  47% /dssd_xfs
tcp://192.168.150.1:3334/orangefs
                       12T  5.7T  6.4T  47% /mnt/orangefs
/dev/gpfs-test         23T  1.1T   22T   5% /mnt/gpfs
[root@r6181-d5-us01 ~]#
[root@r6181-d5-us01 ~]# ps aux | sort -nrk 3,3 | head -n 5
root     97653 11.2  6.6 42665344 17520636 ?   S<Ll Jul13 166:23 /usr/lpp/mmfs/bin/mmfsd
root     69849  6.6  0.0      0     0 ?        S    Jul12 181:54 [vpthread-1-1]
root     69850  6.4  0.0      0     0 ?        S    Jul12 177:42 [vpthread-1-2]
root      3829  3.0  0.0      0     0 ?        S    Jun27 730:04 [rp_thread 7:0]
root      3826  3.0  0.0      0     0 ?        S    Jun27 728:08 [rp_thread 6:0]
[root@r6181-d5-us01 ~]# ls -l /usr/lpp/mmfs/bin/mmfsd
-r-x------ 1 root root 20667161 Jun  3  2015 /usr/lpp/mmfs/bin/mmfsd
[root@r6181-d5-us01 ~]# 
```

User Operating System Interface - GUI

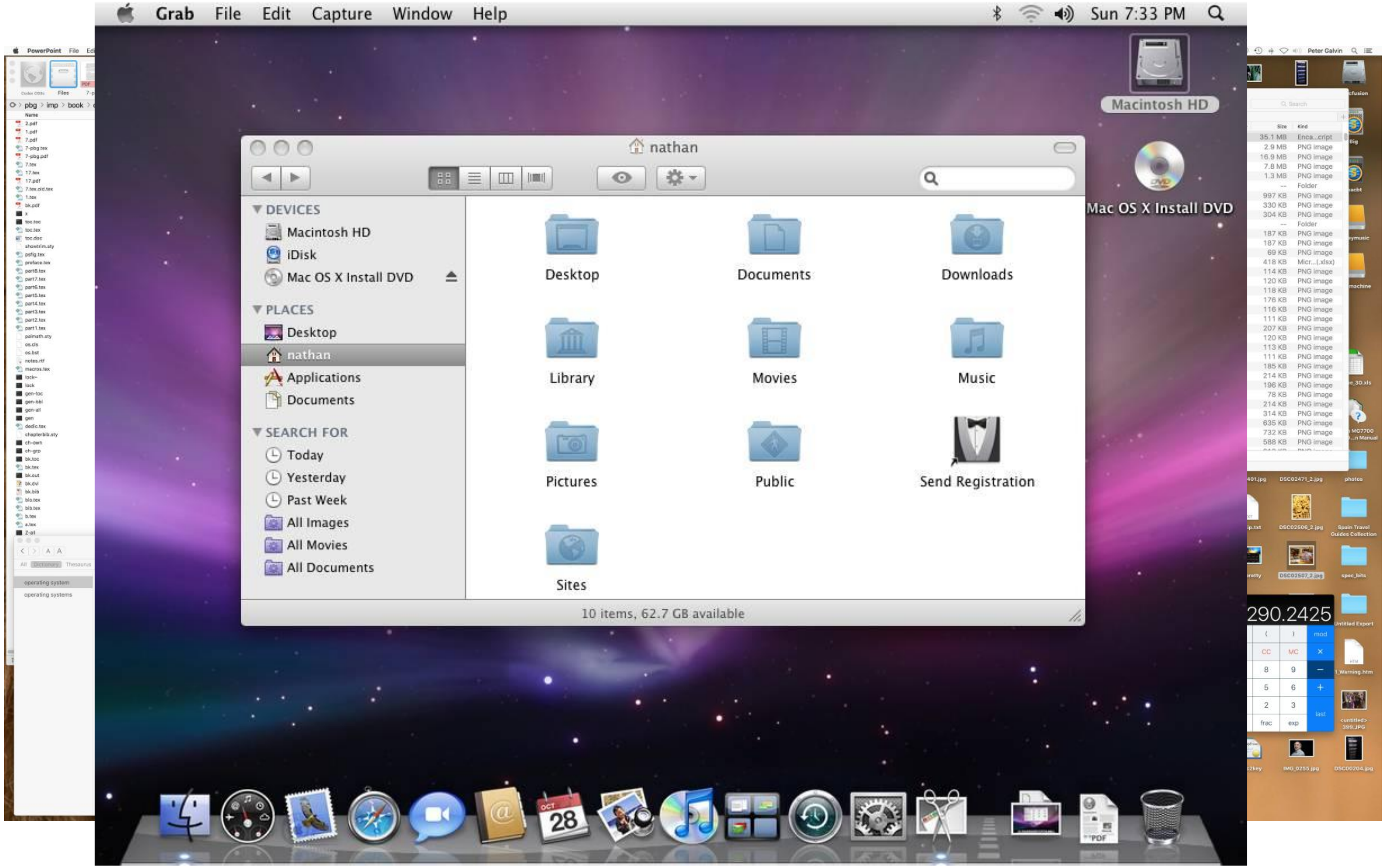
- User-friendly **desktop** metaphor interface
  - Usually mouse, keyboard, and monitor
  - **Icons** represent files, programs, actions, etc
  - Various mouse buttons over objects in the interface cause various actions (provide information, options, execute function, open directory (known as a **folder**)
  - **Invented at Xerox PARC**
- Many systems now include **both CLI and GUI** interfaces
  - Microsoft Windows is GUI with CLI “command” shell
  - Apple Mac OS GUI interface with UNIX kernel underneath and shells available
  - Unix and Linux have CLI with optional GUI interfaces (CDE, KDE, GNOME)

Touchscreen Interfaces

- Touchscreen devices require new interfaces
  - Mouse not possible or not desired
  - Actions and selection based on gestures
  - Virtual keyboard for text entry
- Voice commands



The Mac OS X GUI



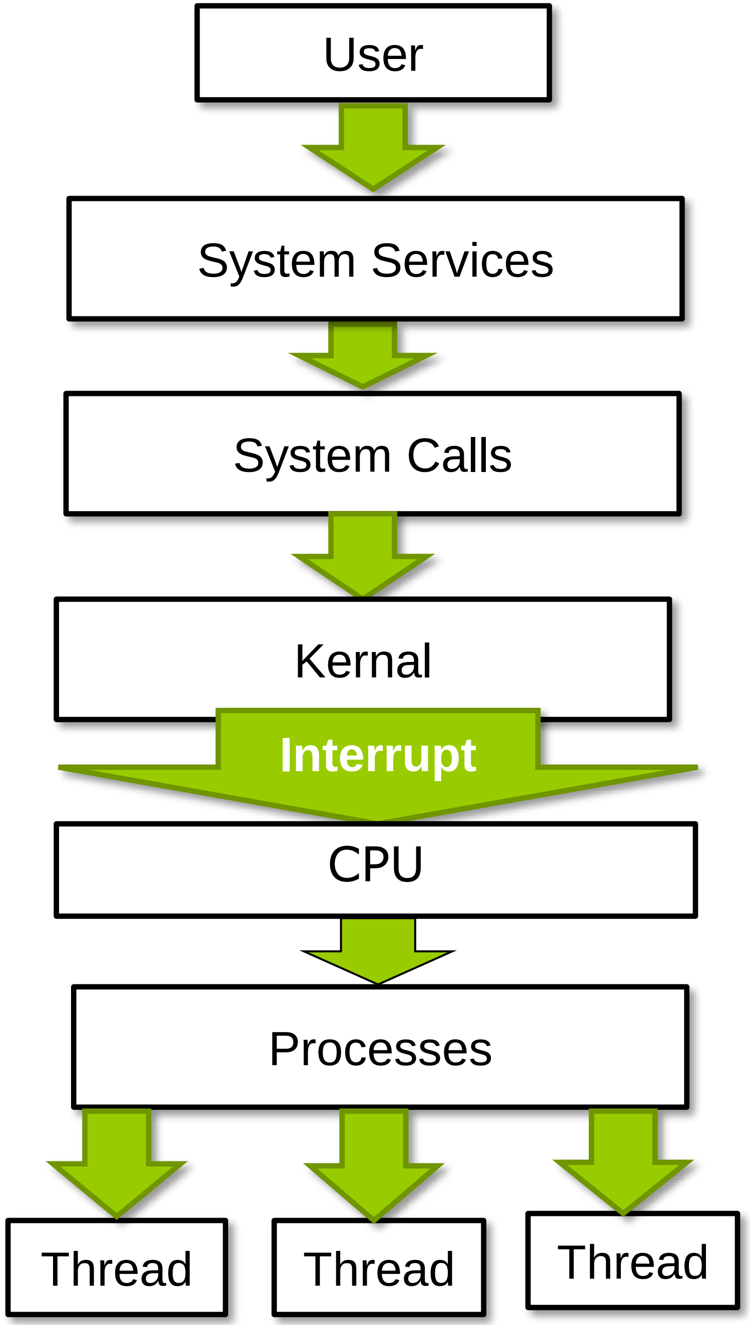
Important notes

- Please **mute** your microphone
- If you have any questions please use the **Chat feature of zoom** , **no microphones.**
- Please make sure zoom is showing **your full name**
- **No Recording** of my lectures
- **No make up** of Quizzes or classwork
- You need to be **in class** for quiz or classwork
- **Tests/Exams/Quizzes** answers from PPT and class notes, no outside sources.
- For **Homework and Research Paper you can** use **outside sources**

Netiquette

- Keep messages short and to the point.
- Never post a message that is in all capital letters — it comes across to the reader as SHOUTING
- Keep in mind that chat messages are meant to be constructive
- Be respectful and treat everyone as you would want to be treated yourself.
- **Be on Time**
- If you came late don't disturb the class , join us quietly, *no need to apologize*
- I don't read emails during the lecture

The Big Pic of OS



User

System Services

System Calls

Kernal

Interrupt

CPU

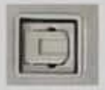
Processes

Thread

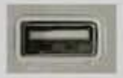
Thread

Thread

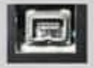
Ports



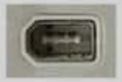
Optical Audio “Toslink”



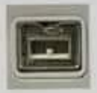
USB A 1.0/1.1/2.0



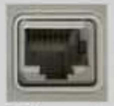
Firewire 4 pin iLink



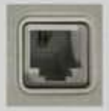
Firewire 400 1394a



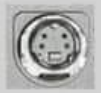
Firewire 800/3200 1394b/c



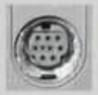
Ethernet 8P8C common:RJ-45



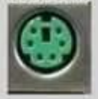
Modem RJ-11



Apple Desktop Bus - ADB



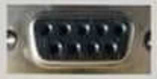
Mac Serial



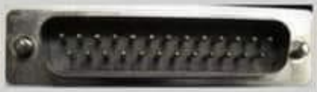
PS/2

Future

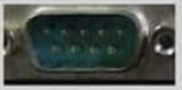
USB A 3.0



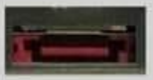
DE-9F



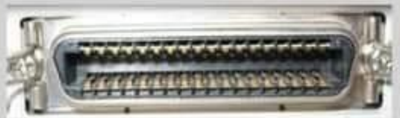
DB-25 Serial/Com Port



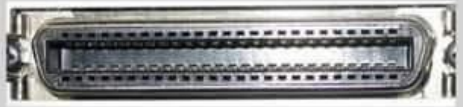
DE-9 Serial RS232



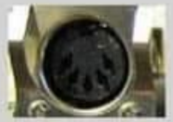
e-SATA



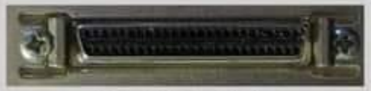
Centronics Parallel 36pin



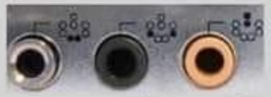
Centronics SCSI 50pin



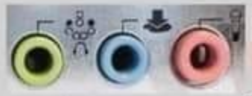
AT Keyboard



50 pin SCSI 2



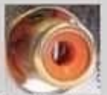
Surround sound



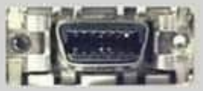
stereo/ Headphones

Line In

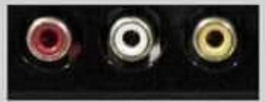
Mic



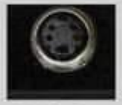
Digital Audio RCA plug style



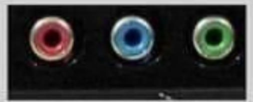
AAUI



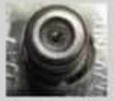
Composite Audio/Video



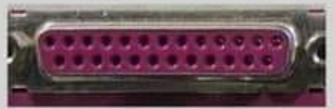
S-Video



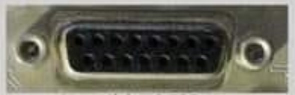
Component Video



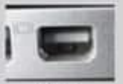
F-Connector RF/COAX



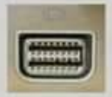
Parallel Port/SCSI 1/DB-25F



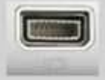
Mac Video/MIDI /gameport/AUI/DA-15



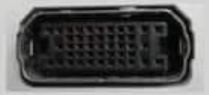
Mini DisplayPort



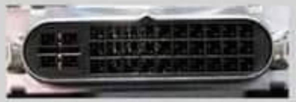
Mini-DVI



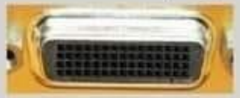
Mini-VGA



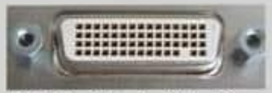
Apple Hi-Density Video HDI-45



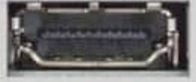
Apple Display Connector - ADC



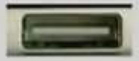
LFH60 (dual DVI-D)



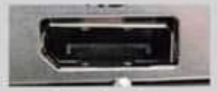
DMS59 (dual DVI-D)



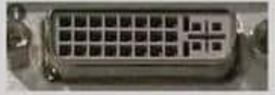
HDMI



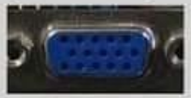
Micro-DVI



DisplayPort



DVI Video



DE-15/HD-15 VGA/SVGA

I/O Subsystem

- One purpose of OS is to hide peculiarities of hardware devices from the user
- I/O subsystem responsible for
  - Memory management of I/O including **buffering** (storing data temporarily while it is being transferred), **caching** (storing parts of data in faster storage for performance), **spooling** (the overlapping of output of one job with input of other jobs)
  - General device-driver interface
  - Drivers for specific hardware devices

Protection and Security

- **Protection** – any mechanism for controlling access of processes or users to resources defined by the OS
- **Security** – defense of the system against internal and external attacks
  - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service
- Systems generally first distinguish among users, to determine who can do what
  - User identities (**user IDs**, security IDs) include name and associated number, one per user
  - User ID then associated with all files, processes of that user to determine access control
  - Group identifier (**group ID**) allows set of users to be defined and controls managed, then also associated with each process, file
  - **Privilege escalation** allows user to change to effective ID with more rights

Virtualization

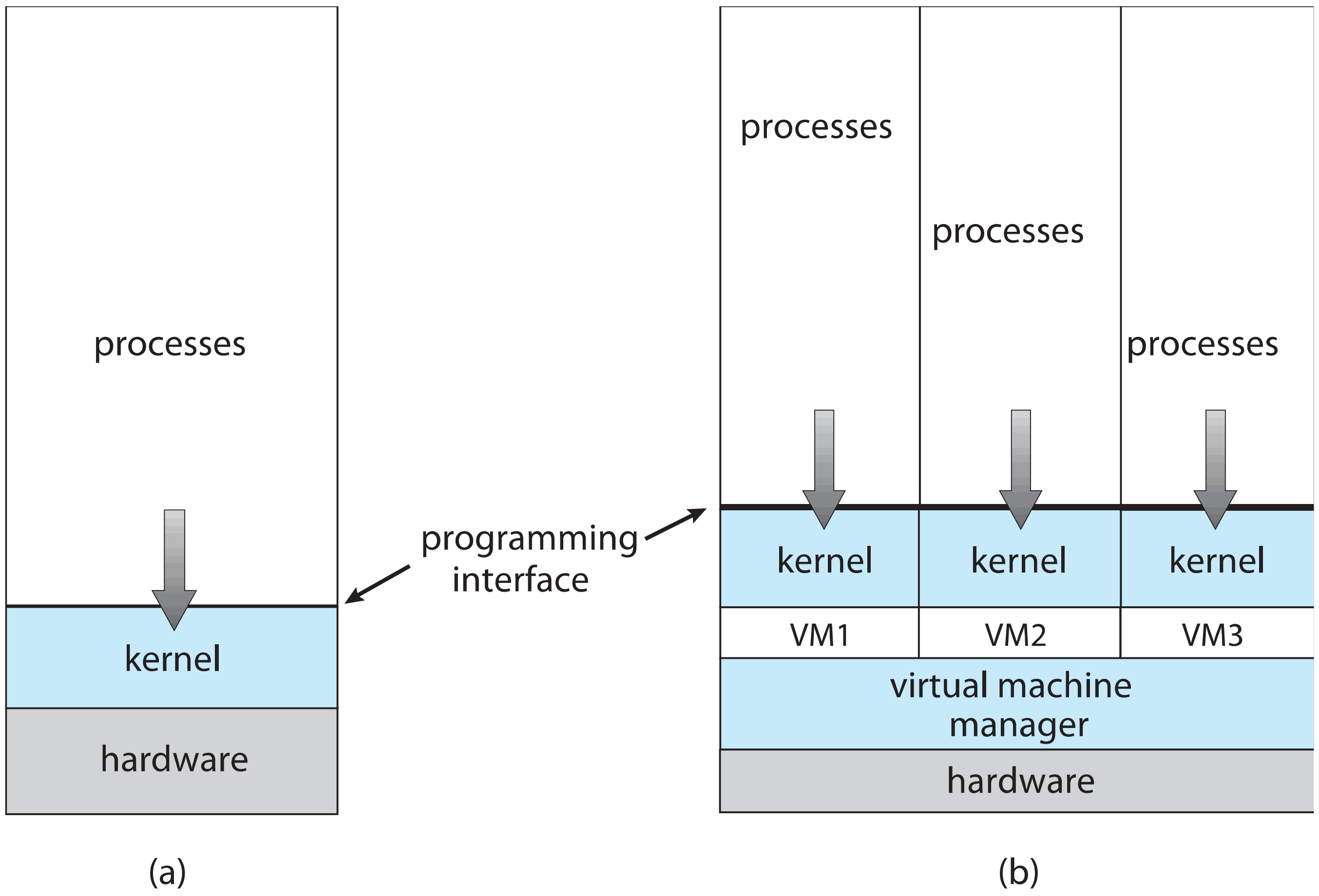
- Allows operating systems to run applications within other OSes
  - Vast and growing industry
- **Emulation** used when source CPU type different from target type (i.e. PowerPC to Intel x86)
  - Generally slowest method
  - When computer language not compiled to native code – **Interpretation**
- **Virtualization** – OS natively compiled for CPU, running **guest** OSes also natively compiled
  - Consider VMware running WinXP guests, each running applications, all on native WinXP **host** OS
  - **VMM** (virtual machine Manager) provides virtualization services

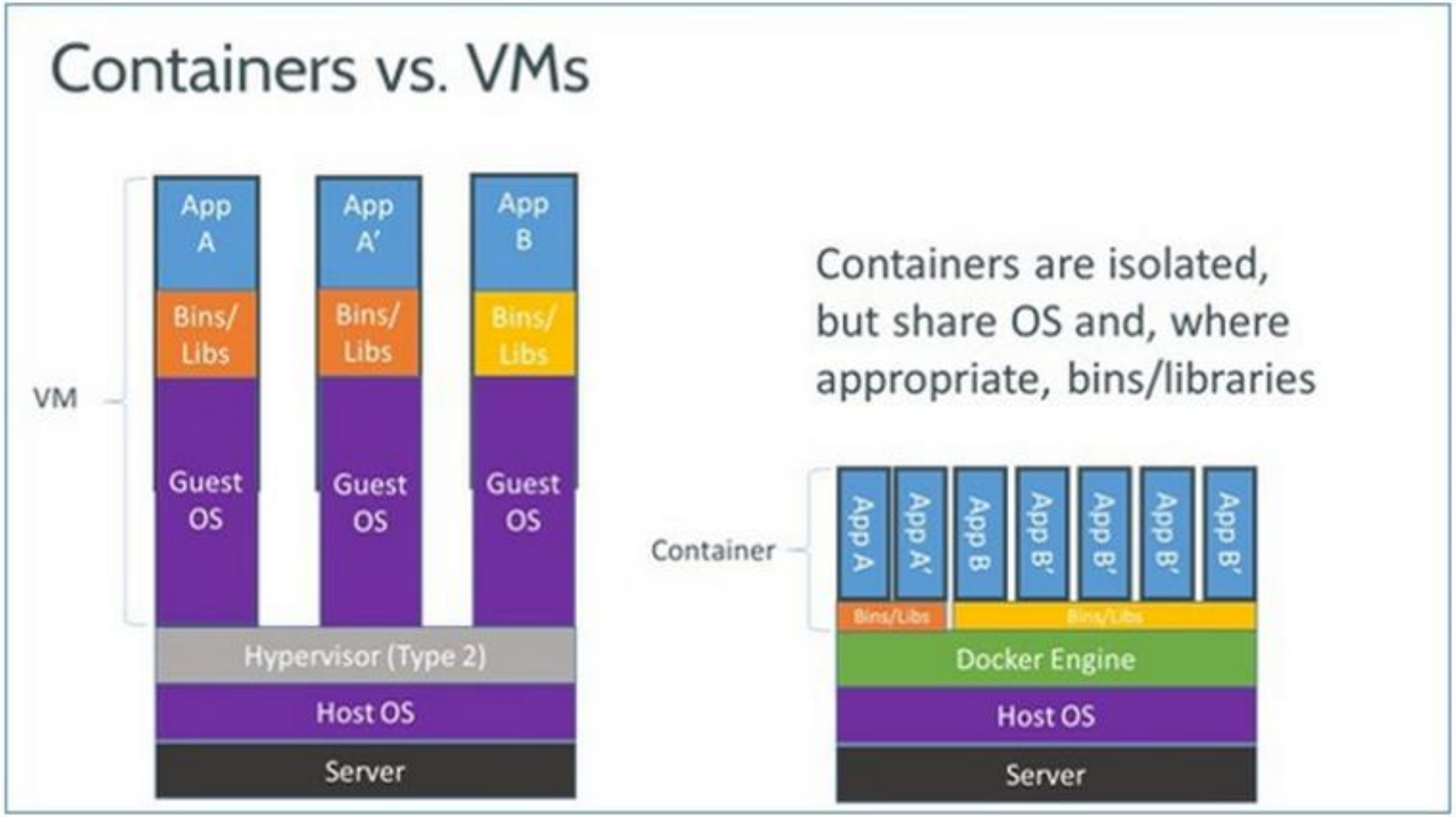
Virtualization (cont.)

- Use cases involve laptops and desktops running multiple OSes for exploration or compatibility
  - Apple laptop running Mac OS X host, Windows as a guest
  - Developing apps for multiple OSes without having multiple systems
  - QA testing applications without having multiple systems
  - Executing and managing compute environments within data centers
- VMM can run natively, in which case they are also the host
  - There is no general purpose host then (VMware ESX and Citrix XenServer)

Explaining Virtual Machines

Computing Environments - Virtualization





Containers vs. VMs

Containers are isolated, but share OS and, where appropriate, bins/libraries

- Containers let you freeze and restart an exact copy of a system that you plan to deploy, including the operating system and configuration files. This makes debugging easy and testing a snap, and it even changes the way that deploys and rollbacks happen in IT operations.

- Container packages are not only complete, but they are also small and efficient enough to download and run in seconds. Cluster managers provide the load balancing and scale to ensure uptime even during a rollout.

Docker

- The first and still most popular container technology, Docker's open-source containerization engine works with most of the products that follow, as well as many open-source tools.

Kubernetes

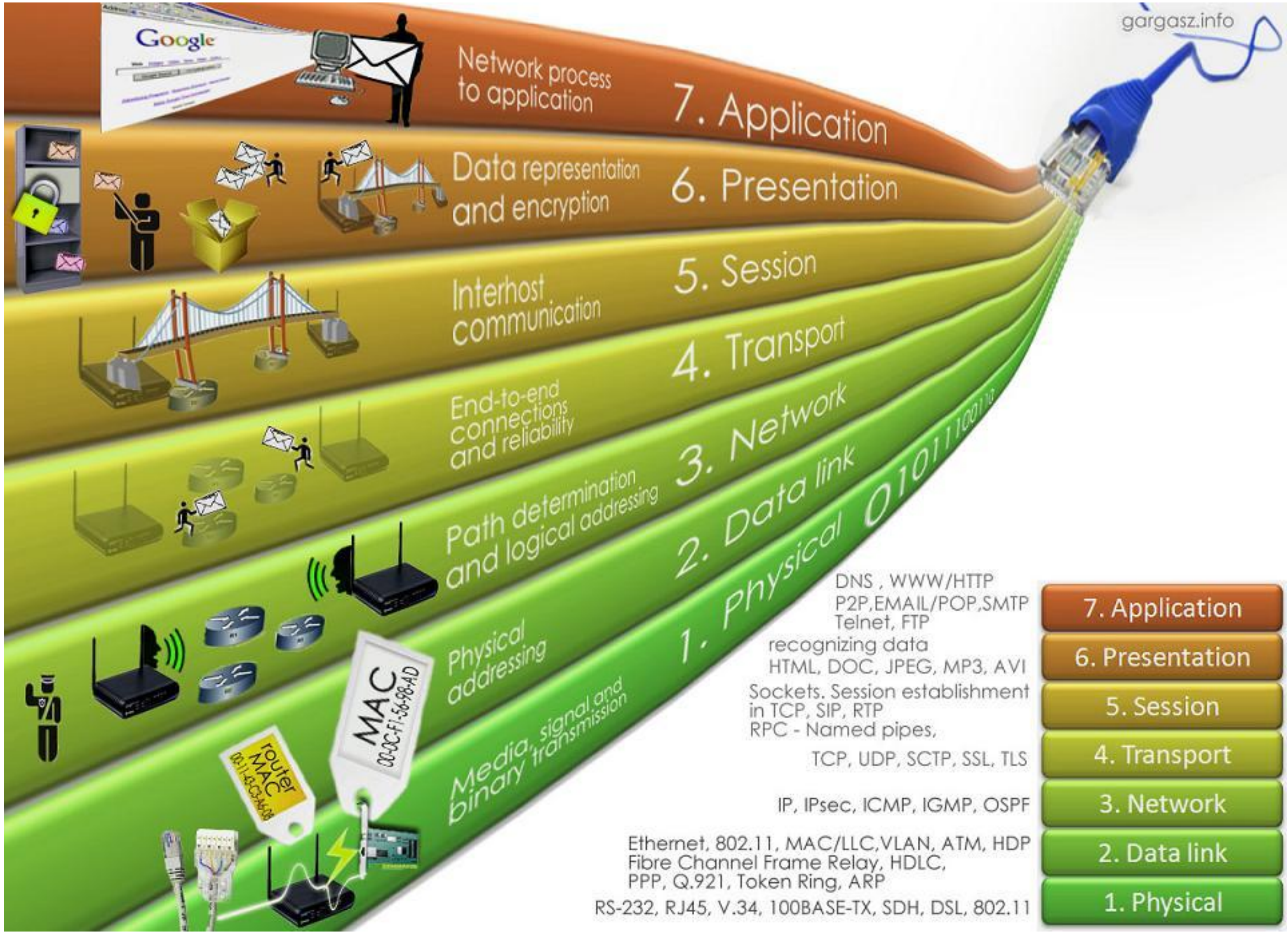
- While there is no standard for cluster management, the Kubernetes open-source cluster manager, originally developed by Google, is far and away the most popular. Supported by Amazon's AWS, Google's Cloud Engine (GCE) and Microsoft's Azure Container service, Kubernetes is relatively portable, which helps prevent vendor lock-in.

Distributed Systems

- Distributed computiing
  - Collection of separate, possibly heterogeneous, systems networked together
    - **Network** is a communications path, **TCP/IP** most common
      - **Local Area Network** (**LAN**)
      - **Wide Area Network** (**WAN**)
      - **Metropolitan Area Network** (**MAN**)
      - **Personal Area Network** (**PAN**)
  - **Network Operating System** provides features between systems across network
    - Communication scheme allows systems to exchange messages
    - Illusion of a single system

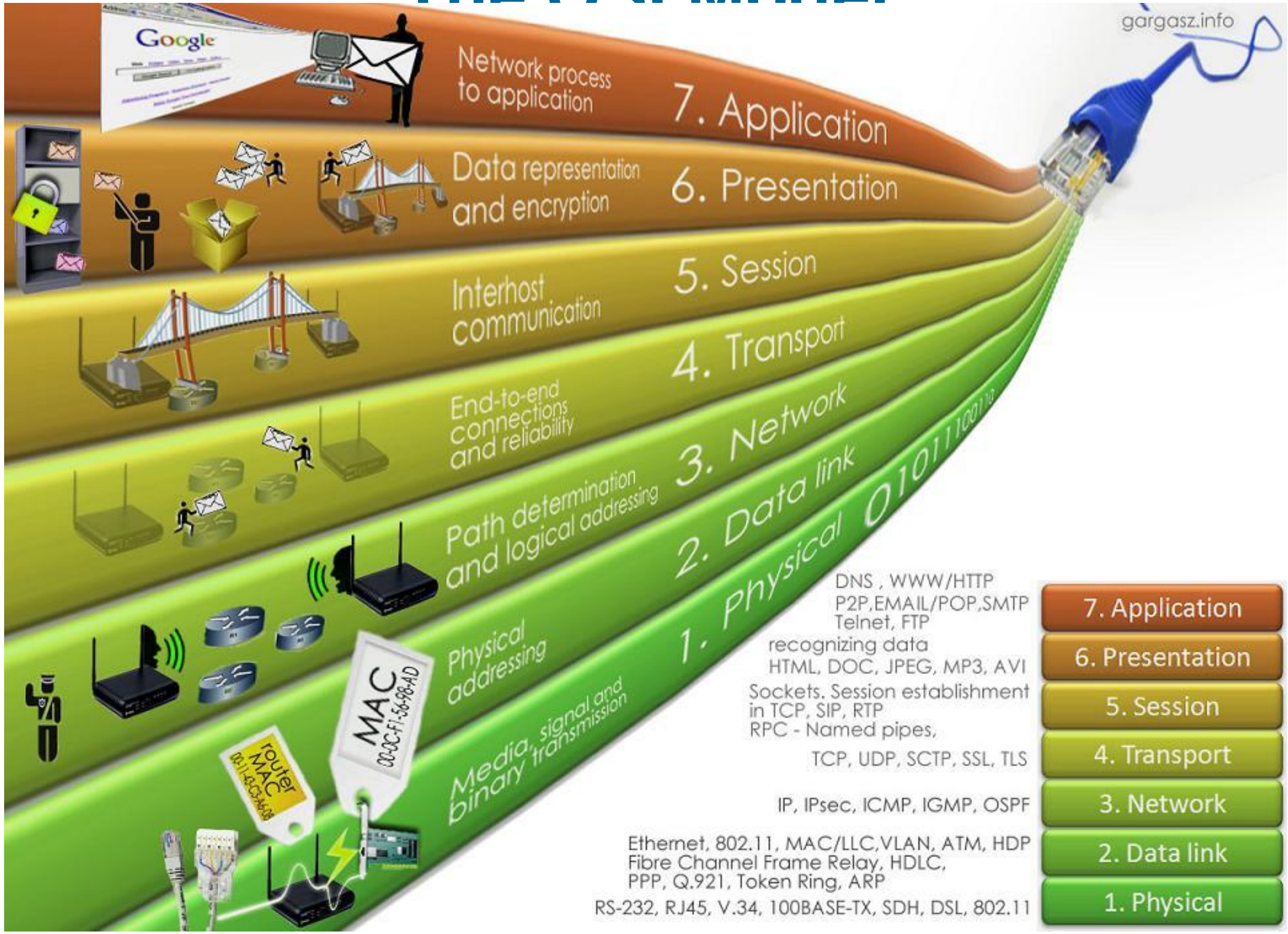
OSI Model

The OSI Model

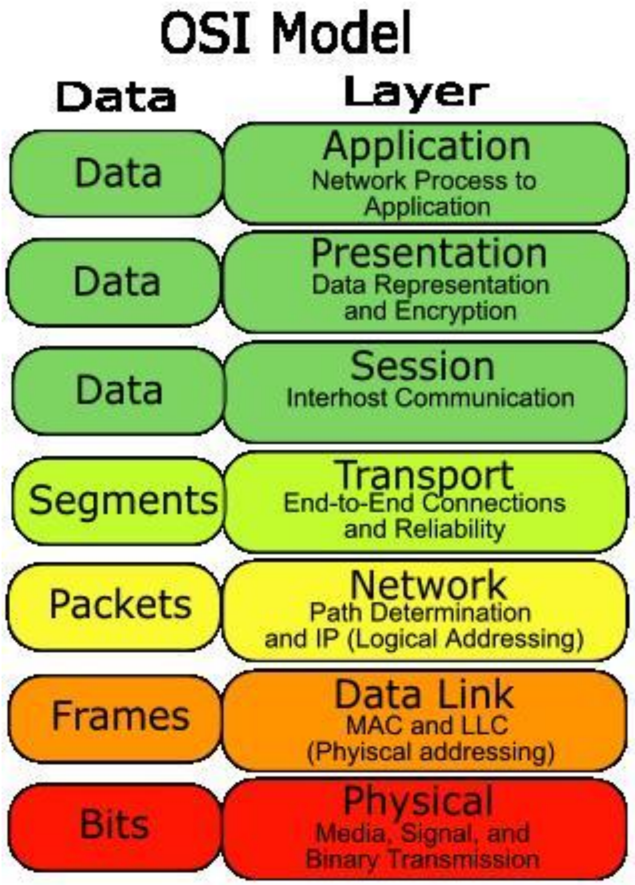

gargasz.info
Network process to application
7. Application
Data representation and encryption
6. Presentation
Interhost communication
5. Session
End-to-end connections and reliability
4. Transport
Path determination and logical addressing
3. Network
Physical addressing
2. Data link
Media, signal and binary transmission
1. Physical
0101110011
MAC
00-0C-F1-56-98-AD
router MAC
00-11-43-C5-A6-08
DNS , WWW/HTTP
P2P,EMAIL/POP,SMTP
Telnet, FTP
recognizing data
HTML, DOC, JPEG, MP3, AVI
Sockets, Session establishment
in TCP, SIP, RTP
RPC - Named pipes,
TCP, UDP, SCTP, SSL, TLS
IP, IPsec, ICMP, IGMP, OSPF
Ethernet, 802.11, MAC/LLC,VLAN, ATM, HDP
Fibre Channel Frame Relay, HDLC,
PPP, Q.921, Token Ring, ARP
RS-232, RJ45, V.34, 100BASE-TX, SDH, DSL, 802.11
7. Application
6. Presentation
5. Session
4. Transport
3. Network
2. Data link
1. Physical

- All People Seem To Need Data Processing

The OSI Model

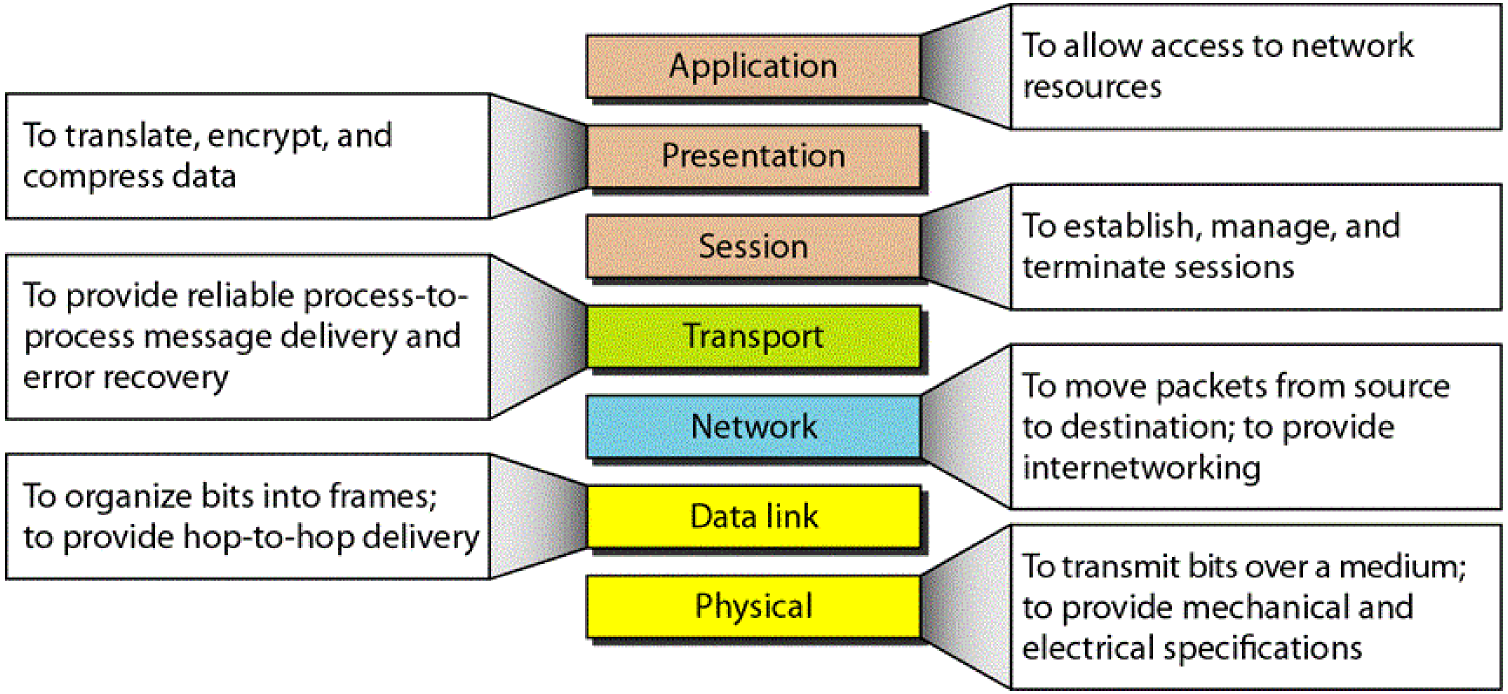

gargasz.info
Google
Network process to application
7. Application
Data representation and encryption
6. Presentation
Interhost communication
5. Session
End-to-end connections and reliability
4. Transport
Path determination and logical addressing
3. Network
Physical addressing
2. Data link
Media, signal and binary transmission
1. Physical 0101110011010
MAC 00-0C-F1-56-98-AD
router MAC 00-11-43-C3-A6-08

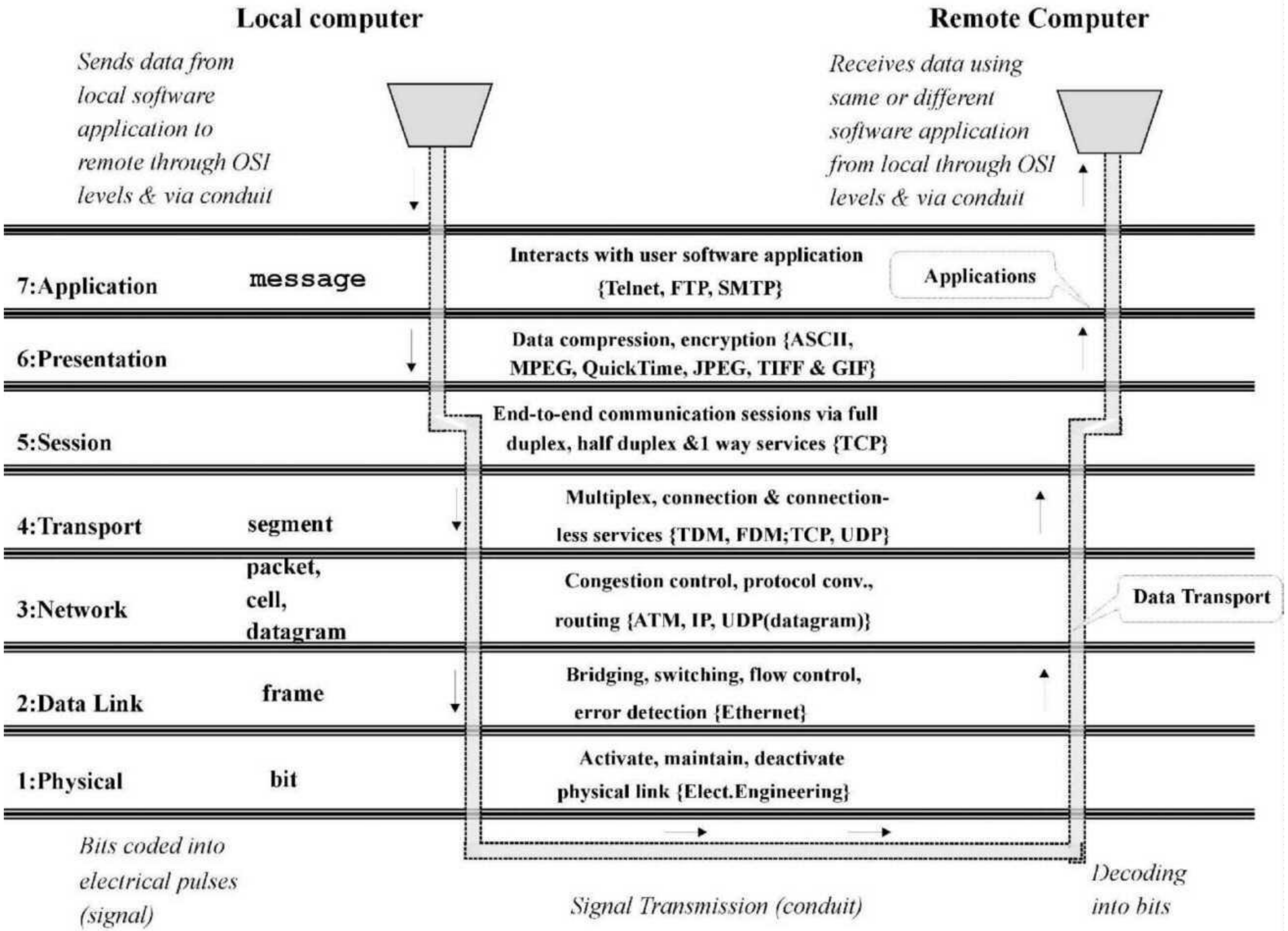
DNS , WWW/HTTP P2P,EMAIL/POP,SMTP Telnet, FTP	7. Application
recognizing data HTML, DOC, JPEG, MP3, AVI	6. Presentation
Sockets. Session establishment in TCP, SIP, RTP RPC - Named pipes,	5. Session
TCP, UDP, SCTP, SSL, TLS	4. Transport
IP, IPsec, ICMP, IGMP, OSPF	3. Network
Ethernet, 802.11, MAC/LLC,VLAN, ATM, HDP Fibre Channel Frame Relay, HDLC, PPP, Q.921, Token Ring, ARP	2. Data link
RS-232, RJ45, V.34, 100BASE-TX, SDH, DSL, 802.11	1. Physical



OSI Model

Data	Layer
Data	Application Network Process to Application
Data	Presentation Data Representation and Encryption
Data	Session Interhost Communication
Segments	Transport End-to-End Connections and Reliability
Packets	Network Path Determination and IP (Logical Addressing)
Frames	Data Link MAC and LLC (Phyiscal addressing)
Bits	Physical Media, Signal, and Binary Transmission


Application
To allow access to network resources
To translate, encrypt, and compress data
Presentation
Session
To establish, manage, and terminate sessions
To provide reliable process-to-process message delivery and error recovery
Transport
Network
To move packets from source to destination; to provide internetworking
To organize bits into frames; to provide hop-to-hop delivery
Data link
Physical
To transmit bits over a medium; to provide mechanical and electrical specifications



Local computer

Sends data from local software application to remote through OSI levels & via conduit

Remote Computer

Receives data using same or different software application from local through OSI levels & via conduit

7:Application	message	Interacts with user software application {Telnet, FTP, SMTP}
6:Presentation		Data compression, encryption {ASCII, MPEG, QuickTime, JPEG, TIFF & GIF}
5:Session		End-to-end communication sessions via full duplex, half duplex &1 way services {TCP}
4:Transport	segment	Multiplex, connection & connection-less services {TDM, FDM;TCP, UDP}
3:Network	packet, cell, datagram	Congestion control, protocol conv., routing {ATM, IP, UDP(datagram)}
2:Data Link	frame	Bridging, switching, flow control, error detection {Ethernet}
1:Physical	bit	Activate, maintain, deactivate physical link {Elect.Engineering}

Bits coded into electrical pulses (signal)

Signal Transmission (conduit)

Decoding into bits

Application Layer

Example of Application layer protocol: **HTTP**

Software

About the Library

READ.GOV JOHN CARTER "A PRINCESS OF MARS"

The Library's Mission
The History of the Library
Awards & Honors
Jobs & Fellowships
Frequently Asked Questions
Support the Library

Collection Highlights

American Memory
Prints & Photographs
Historic Newspapers
Performing Arts
Veterans History
Sound Recordings
Film
Maps
Manuscripts

Topics
American History
Arts & Culture
Government, Politics & Law
Maps & Geography
News & Journalism
Religion & Philosophy
Science, Technology & Business
Sports & Leisure
World History & Cultures

Especially for
Kids & Families
Librarians
Publishers
Researchers
Teachers

Thomas.gov U.S. Legislative information from the Library of Congress

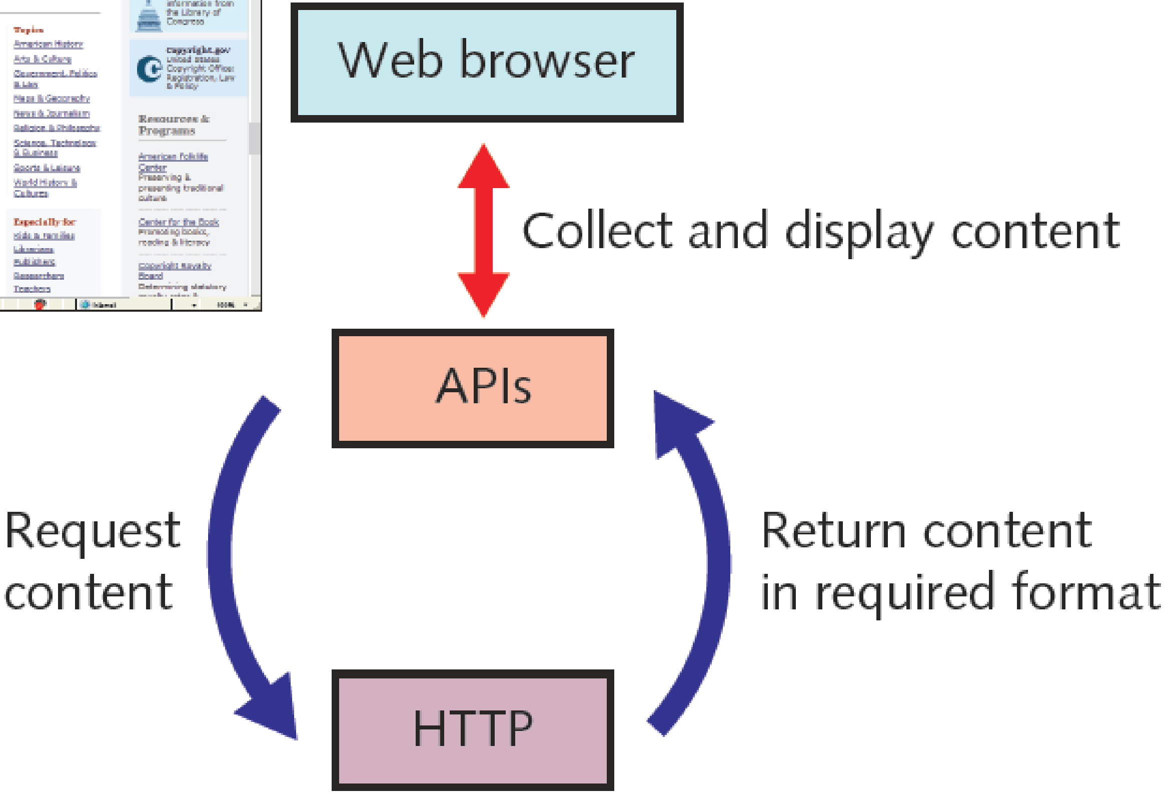
Copyright.gov United States Copyright Office: Registration, Law & Policy

Resources & Programs

American Folklife Center
Preserving & presenting traditional culture

Center for the Book
Promoting books, reading & literacy

Copyright Royalty Board
Determining statutory



Web browser

Collect and display content

APIs

Request content

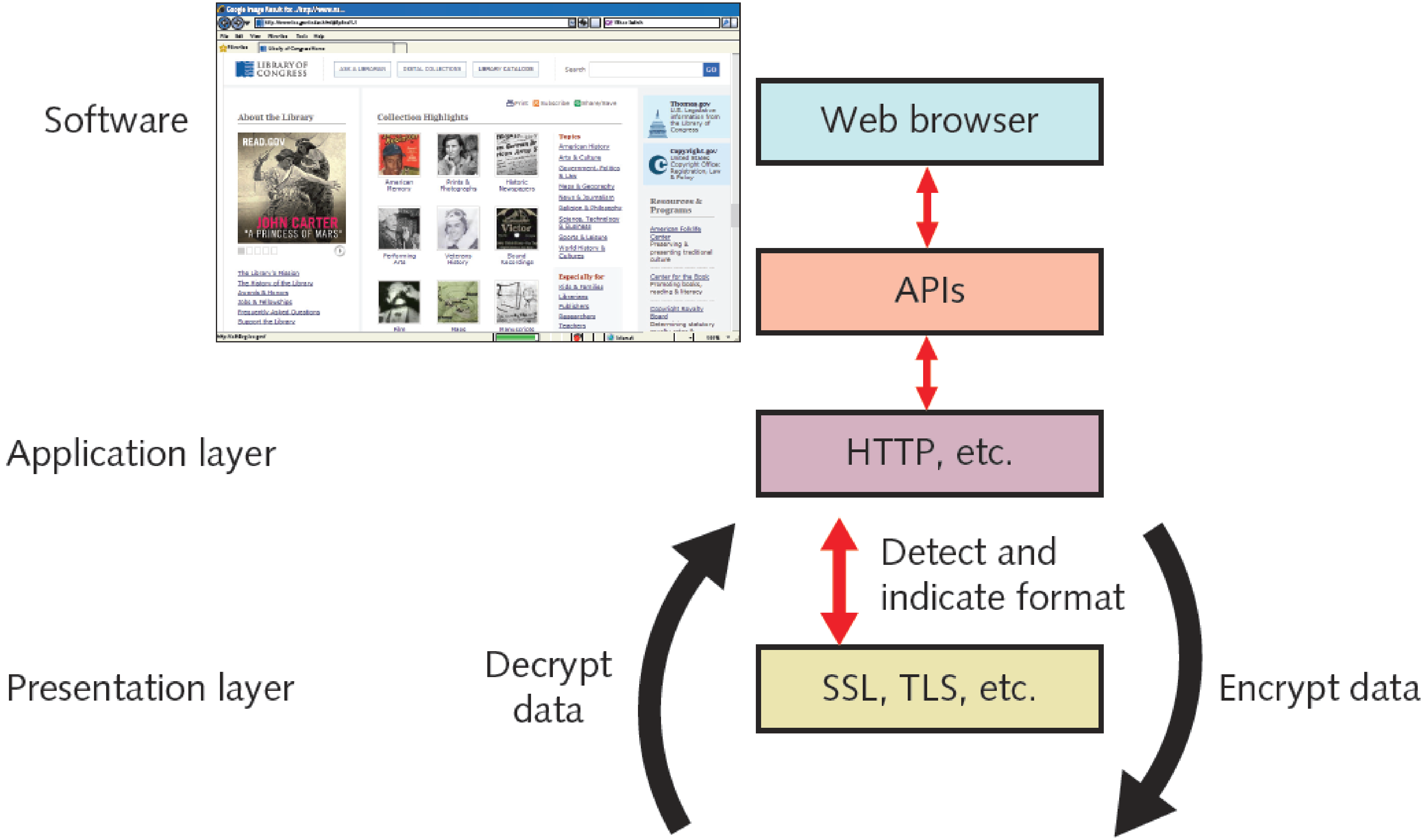
Return content in required format

HTTP

Application layer

Presentation Layer

- Example protocol: Secure Sockets Layer (**SSL**)



Software

Application layer

Presentation layer

Web browser

APIs

HTTP, etc.

Detect and indicate format

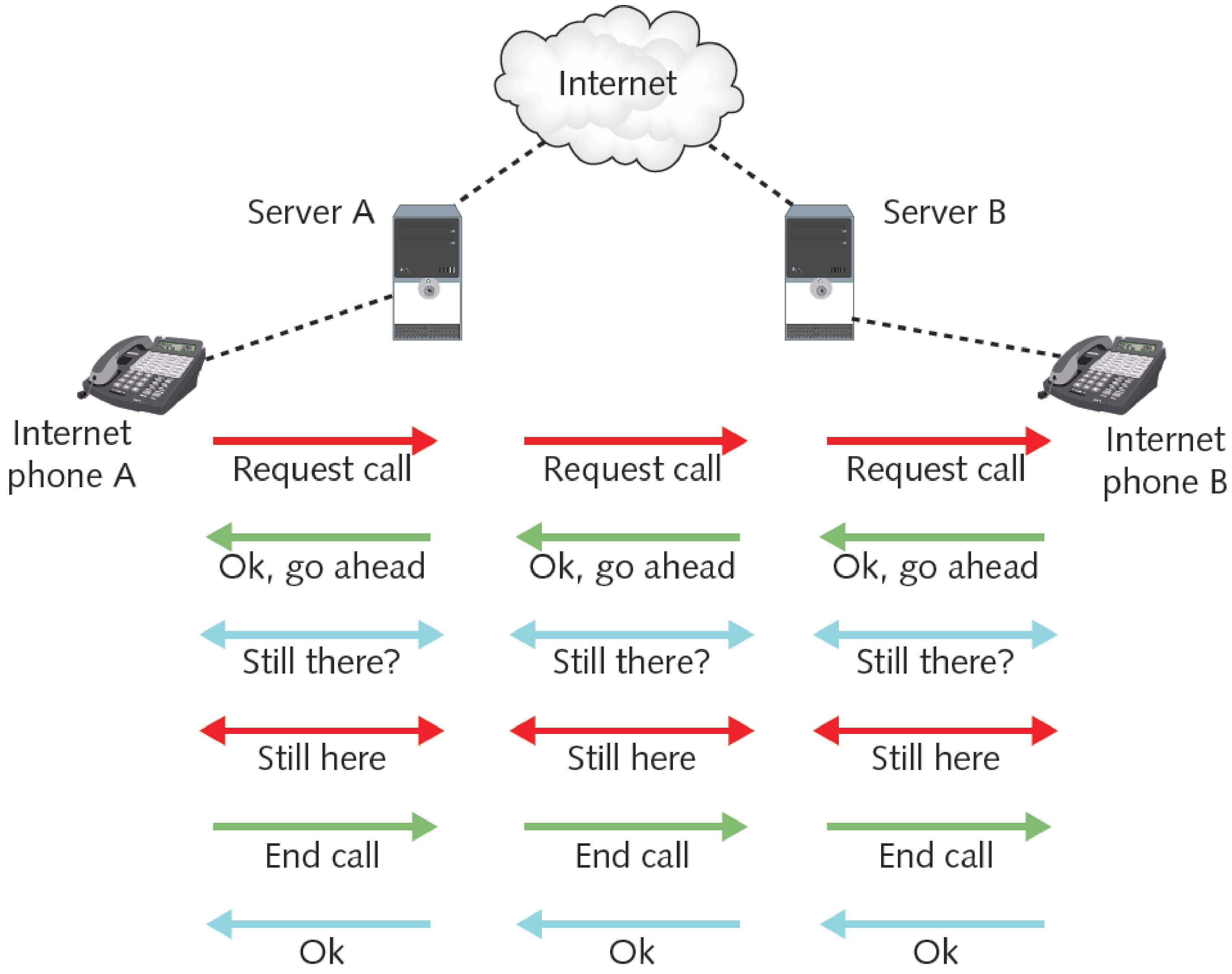
SSL, TLS, etc.

Decrypt data

Encrypt data

Session Layer

- Establishing and keeping alive communications link
- Keeping communications secure
- Synchronizing dialogue between two nodes
- Identify session participants


Internet
Server A
Server B
Internet phone A
Internet phone B
Request call
Request call
Request call
Ok, go ahead
Ok, go ahead
Ok, go ahead
Still there?
Still there?
Still there?
Still here
Still here
Still here
End call
End call
End call
Ok
Ok
Ok

Transport Layer

TCP

- Accept data from Session layer
- Manage end-to-end data delivery
- Handle flow control
- Examples
  - Checksum
  - Segmentation
  - MTU (maximum transmission unit)

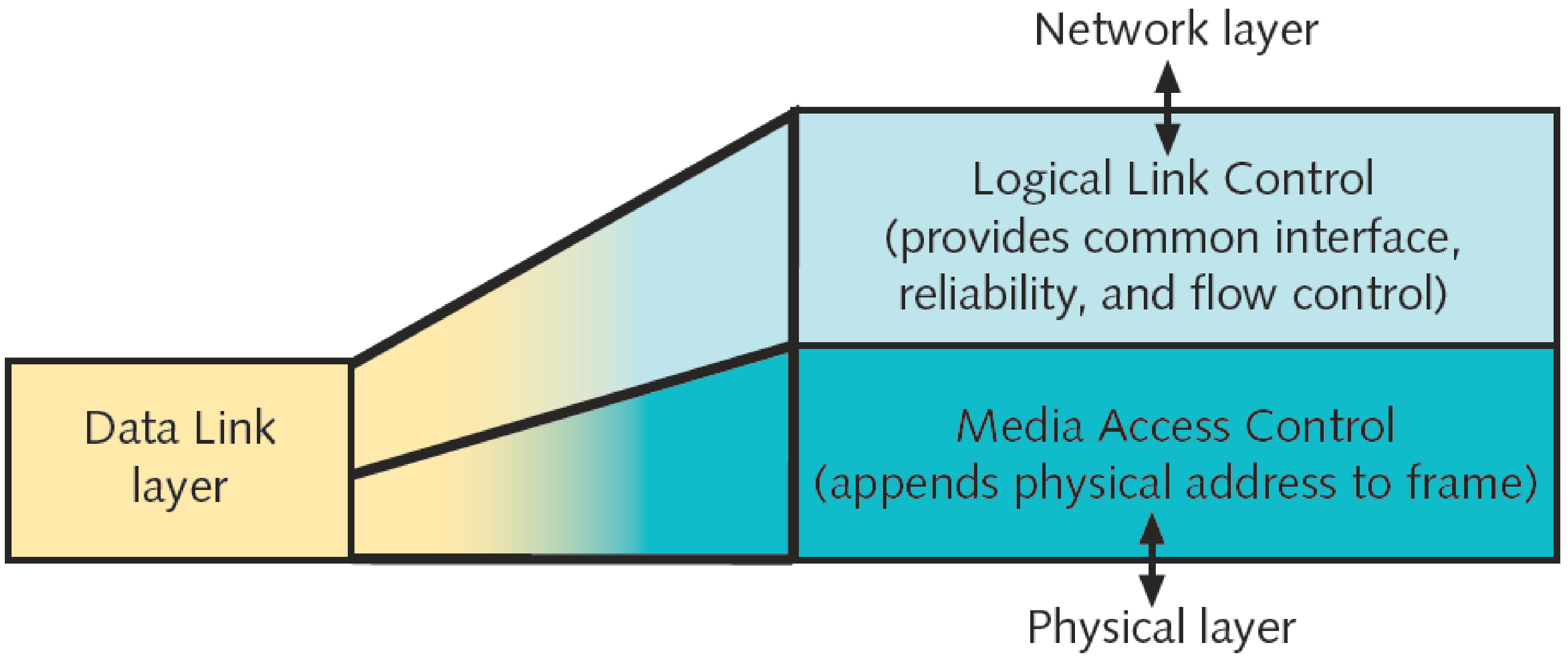
Network Layer

- Routers /Switches
- IP

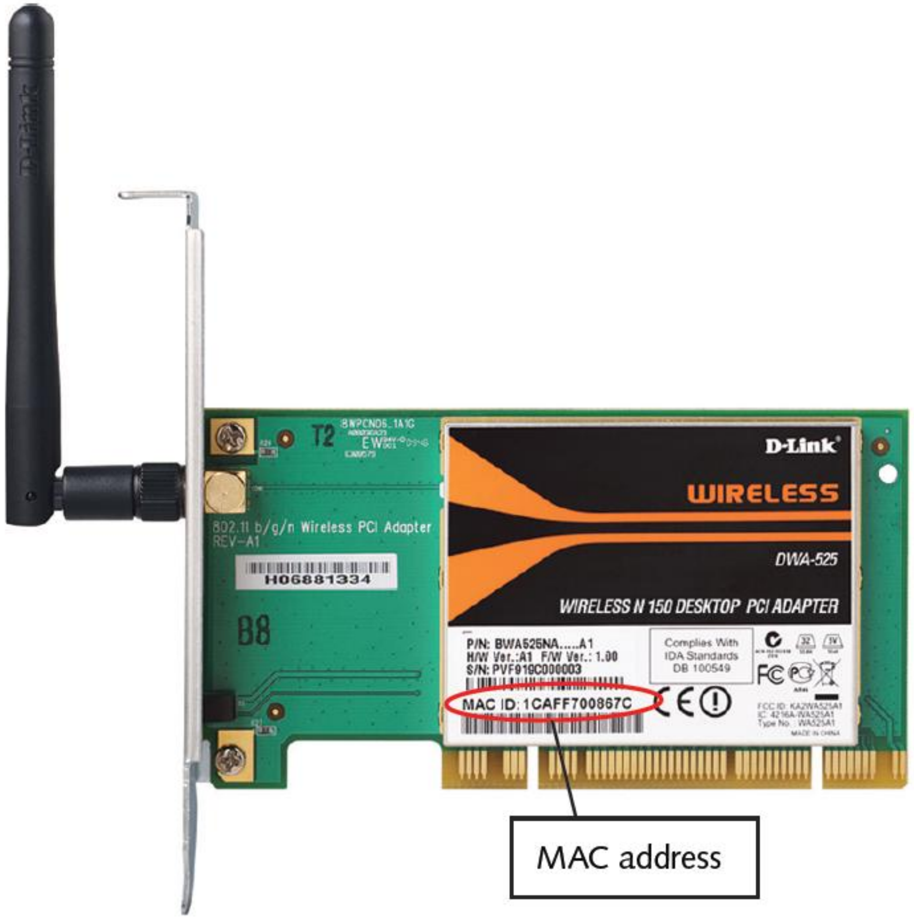
Data Link Layer

Two Data Link layer sublayers

1. LLC (Logical Link Control) sublayer (Error checking methods: CRC (cyclic redundancy check)
2. MAC (Media Access Control) sublayer : NIC



Network layer
Logical Link Control
(provides common interface,
reliability, and flow control)
Data Link
layer
Media Access Control
(appends physical address to frame)
Physical layer


D-Link
WIRELESS
DWA-525
WIRELESS N 150 DESKTOP PCI ADAPTER
802.11 b/g/n Wireless PCI Adapter
REV-A1
H06881334
P/N: BWA525NA.....A1
H/W Ver.:A1 F/W Ver.: 1.00
S/N: PVF919C000003
Complies With IDA Standards DB 100549
MAC ID: 1CAFF700867C
MADE IN CHINA
MAC address

Physical Layer

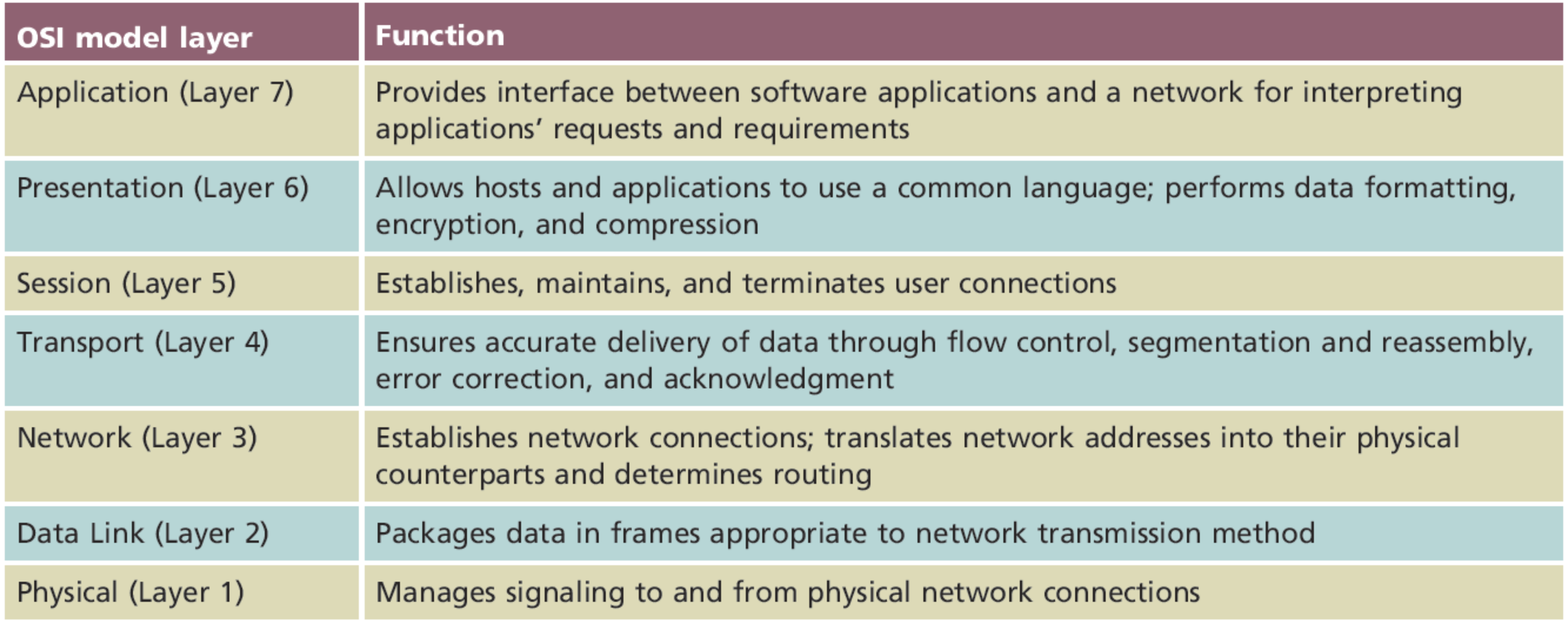
- Copper transmission medium

  Signals issued as voltage

- Fiber-optic cable transmission medium Signals issued as light pulses
- Wireless transmission medium

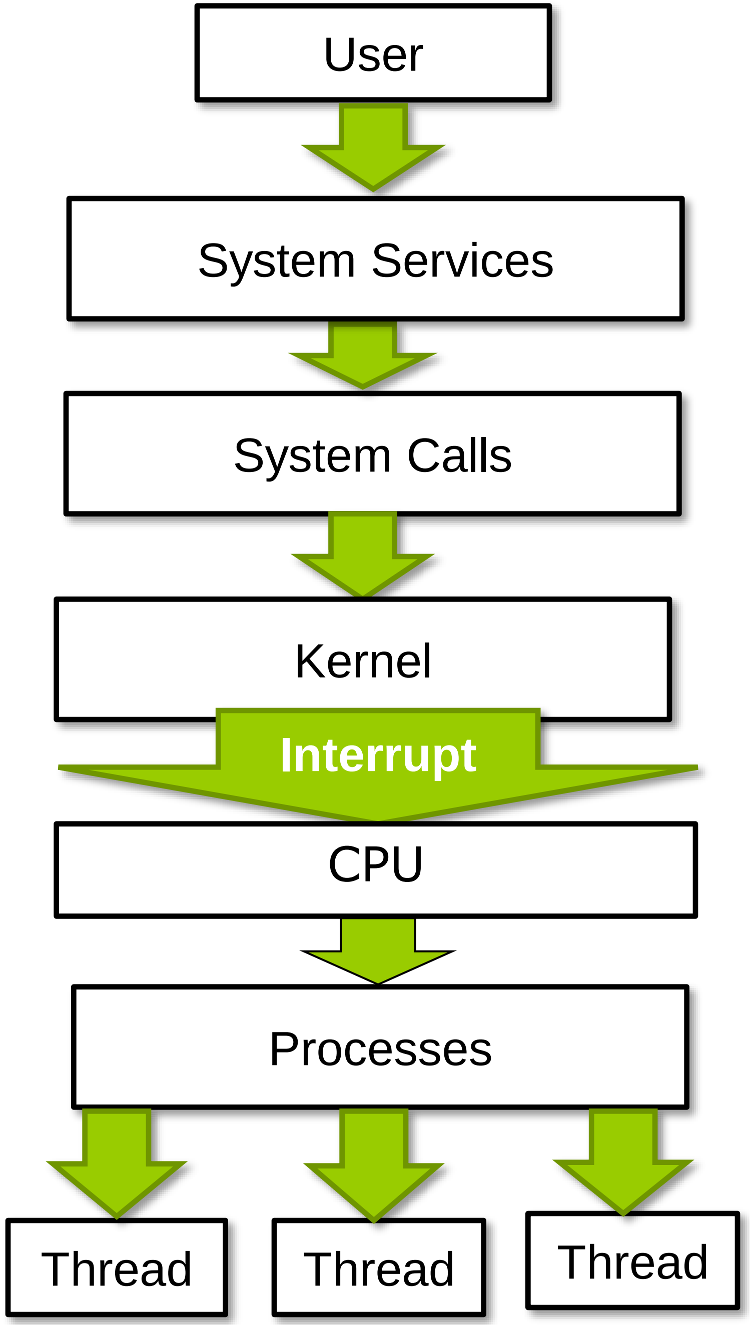
  Signals issued as electromagnetic waves

Functions of the OSI layers



OSI model layer	Function
Application (Layer 7)	Provides interface between software applications and a network for interpreting applications' requests and requirements
Presentation (Layer 6)	Allows hosts and applications to use a common language; performs data formatting, encryption, and compression
Session (Layer 5)	Establishes, maintains, and terminates user connections
Transport (Layer 4)	Ensures accurate delivery of data through flow control, segmentation and reassembly, error correction, and acknowledgment
Network (Layer 3)	Establishes network connections; translates network addresses into their physical counterparts and determines routing
Data Link (Layer 2)	Packages data in frames appropriate to network transmission method
Physical (Layer 1)	Manages signaling to and from physical network connections

The Big Pic of OS



System Calls ,Services and Processes

- A **system call** is the programmatic way in which a computer program requests **a service** from the kernel of the operating system it is executed on.
- A system call is a way for programs to **interact with the operating system**.
- A System call **provides** the services of the operating system to the user programs via **Application Program Interface(API).**
- It provides an interface between a process and operating system to allow user-level processes to request services of the operating system.
- System calls are the only **entry points** into the kernel system.
- All programs needing resources must use **system calls**.

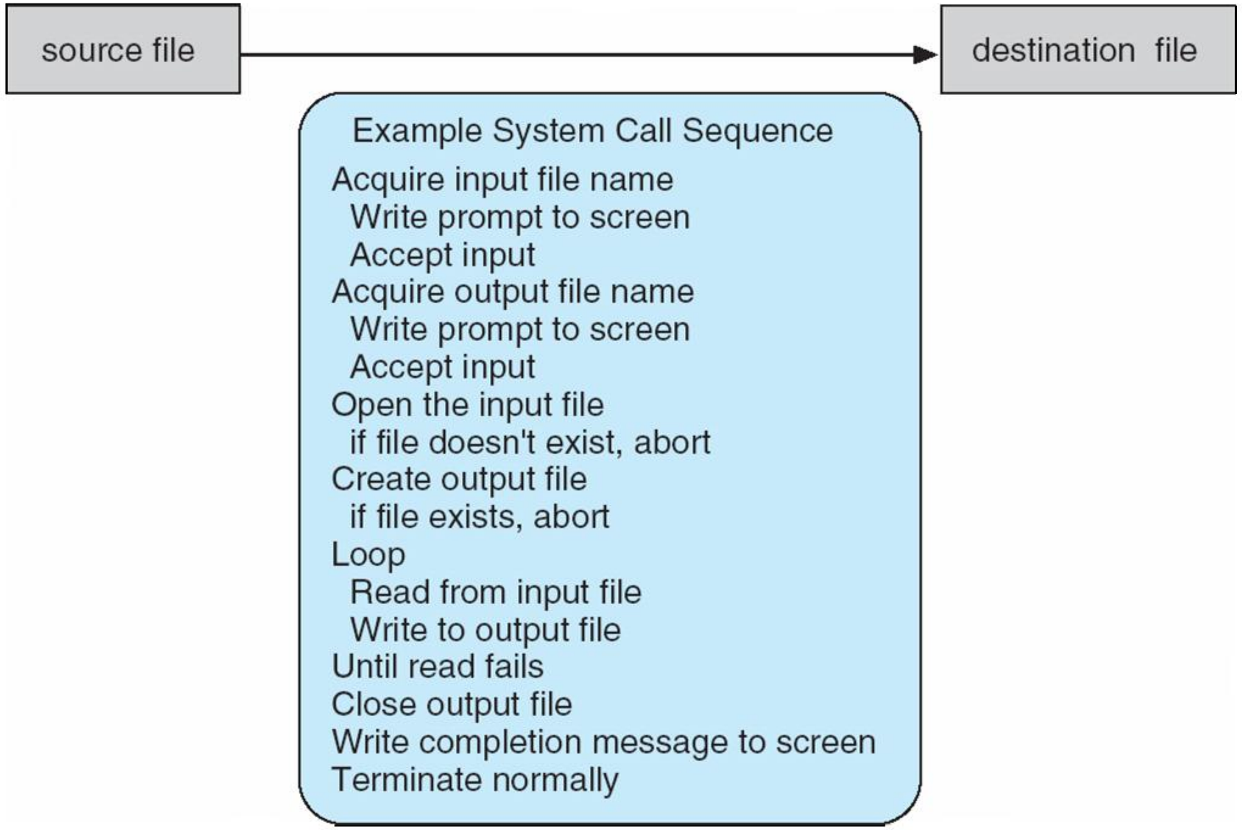
System Calls

- **Programming interface to the services provided by the OS**
- Typically written in a high-level language (C or C++)
- Mostly accessed by programs via a high-level **Application Programming Interface (API)** rather than direct system call use
- Three most common **APIs are Win32 API for Windows, POSIX API for POSIX-based systems (including virtually all versions of UNIX, Linux, and Mac OS X), and Java API for the Java virtual machine (JVM)**

Note that the system-call names used throughout this text are generic

Example of System Calls

- System call sequence to copy the contents of one file to another file



source file

destination file

Example System Call Sequence

```
Acquire input file name
  Write prompt to screen
  Accept input
Acquire output file name
  Write prompt to screen
  Accept input
Open the input file
  if file doesn't exist, abort
Create output file
  if file exists, abort
Loop
  Read from input file
  Write to output file
Until read fails
Close output file
Write completion message to screen
Terminate normally
```

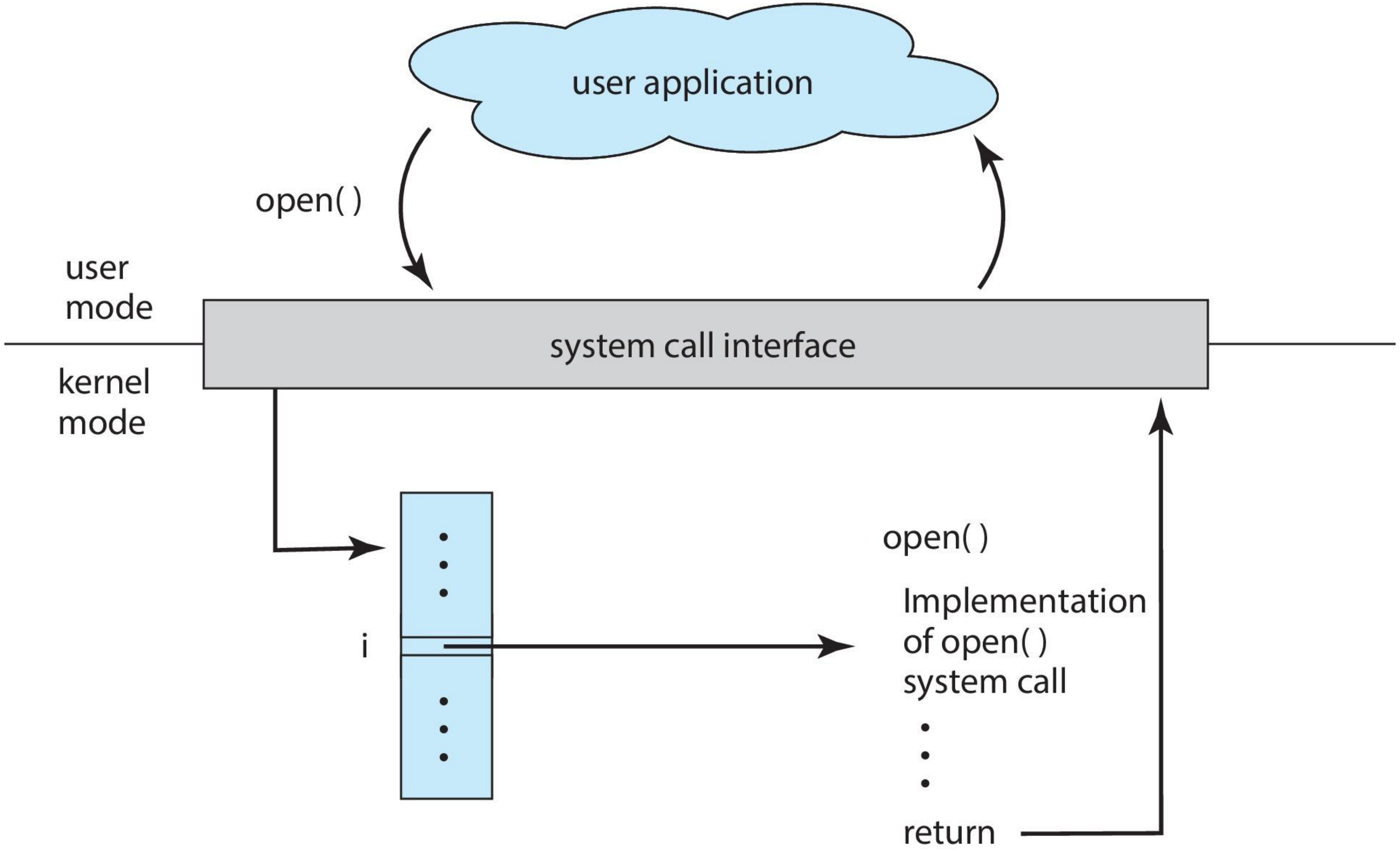
Example of a System Call

```c
#include <stdio.h>
#include <fcntl.h>    // For open() system call
#include <unistd.h>   // For read() and close() system calls

int main() {
    // Open the file
    int fd = open("example.txt", O_RDONLY);
    if (fd == -1) {
        perror("Error opening the file");
        return 1;
    }

    // Buffer to store the data
    char buffer[100];
    ssize_t bytesRead;

    // Read up to 100 bytes from the file
    bytesRead = read(fd, buffer, sizeof(buffer) - 1);
    if (bytesRead == -1) {
        perror("Error reading the file");
        close(fd);
        return 1;
    }

    // Null-terminate the buffer to print it as a string
    buffer[bytesRead] = '\0';

    // Print the contents read
    printf("Data read from the file:\n%s\n", buffer);

    // Close the file
    close(fd);

    return 0;
}
```

- **open()**: This system call opens a file and returns a file descriptor (fd). In this example, the file example.txt is opened in read-only mode (O_RDONLY).
- **read()**: This system call reads data from the file using the file descriptor returned by open(). The data is stored in the buffer and the function returns the number of bytes read.
- **close()**: This system call closes the file once you are done working with it.

This is a typical example of using system calls to interact with files in an operating system. The calls to open(), read(), and close() are all direct interactions with the OS kernel.

API – System Call – OS Relationship



Types of System Calls

- **Process control**
  - create process, terminate process
  - end, abort
  - load, execute
  - get process attributes, set process attributes
  - wait for time
  - wait event, signal event
  - allocate and free memory
  - Dump memory if error
  - **Debugger** for determining **bugs, single step** execution
  - **Locks** for managing access to shared data between processes

Types of System Calls (cont.)

- **File management**
  - create file, delete file
  - open, close file
  - read, write, reposition
  - get and set file attributes
- **Device management**
  - request device, release device
  - read, write, reposition
  - get device attributes, set device attributes
  - logically attach or detach devices

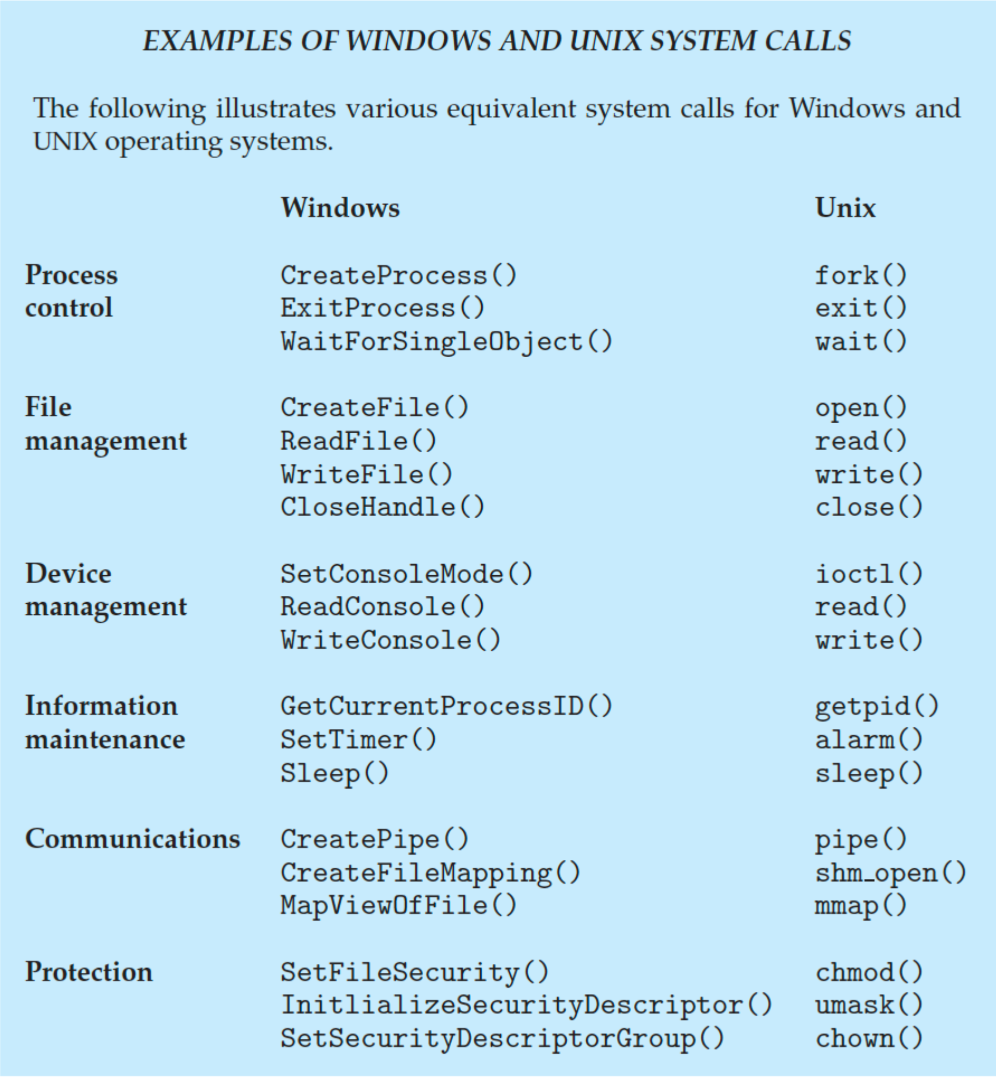
Types of System Calls (Cont.)

- **Information maintenance**
  - get time or date, set time or date
  - get system data, set system data
  - get and set process, file, or device attributes
- **Communications**
  - create, delete communication connection
  - send, receive messages if **message passing model** to **host name** or **process name**
    - From **client** to **server**
  - **Shared-memory model** create and gain access to memory regions
  - transfer status information
  - attach and detach remote devices

Types of System Calls (Cont.)

- **Protection**
  - Control access to resources
  - Get and set permissions
  - Allow and deny user access

Examples of Windows and Unix System Calls



EXAMPLES OF WINDOWS AND UNIX SYSTEM CALLS

The following illustrates various equivalent system calls for Windows and UNIX operating systems.

	Windows	**Unix**
Process control	`CreateProcess()`	`fork()`
	`ExitProcess()`	`exit()`
	`WaitForSingleObject()`	`wait()`
File management	`CreateFile()`	`open()`
	`ReadFile()`	`read()`
	`WriteFile()`	`write()`
	`CloseHandle()`	`close()`
Device management	`SetConsoleMode()`	`ioctl()`
	`ReadConsole()`	`read()`
	`WriteConsole()`	`write()`
Information maintenance	`GetCurrentProcessID()`	`getpid()`
	`SetTimer()`	`alarm()`
	`Sleep()`	`sleep()`
Communications	`CreatePipe()`	`pipe()`
	`CreateFileMapping()`	`shm_open()`
	`MapViewOfFile()`	`mmap()`
Protection	`SetFileSecurity()`	`chmod()`
	`InitlializeSecurityDescriptor()`	`umask()`
	`SetSecurityDescriptorGroup()`	`chown()`