

Networked Computers Interaction

- Timesharing = Centralized Computing
- Distributed processing
- Peer-to-peer networking
 - ❖ Computer communication in which all computers equal
- Client/Server
 - ❖ Program that offers a service is a *server*
 - ❖ Program that contacts a service is a *client*

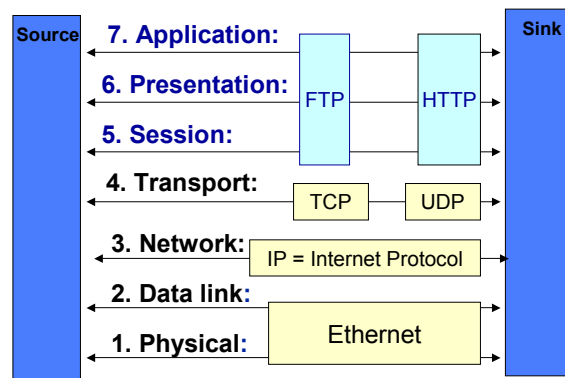
1

Client Server Applications

- Application-level protocols provide high-level services
 - ❖ FTP
 - ❖ World Wide Web
 - ❖ DNS
 - ❖ Electronic mail
 - ❖ Remote Log In
- All of these applications use client-server architecture

2

OSI Model: Client Server



3

Internet protocols and network applications

- Internet protocols provide
 - ❖ General-purpose facility for reliable data transfer TCP
 - ❖ Mechanism for contacting hosts IP
- Application programs
 - ❖ Use internet protocols to contact other applications TCP
 - ❖ Provide *user-level services*

4

Client-server paradigm

- Server application is "listener"
 - ❖ Waits for incoming message
 - ❖ Performs service
 - ❖ Returns results
- Client application establishes connection
 - ❖ Sends message to server
 - ❖ Waits for return message

5

Characteristics of client

- Application program
 - ❖ Becomes client when network service needed
 - ❖ Also performs other computations
- Invoked directly by user
- Runs locally on user's computer
- Initiates contact with server
- Can access multiple services
- Does not require special hardware or sophisticated operating system

6

Characteristics of server

- Special purpose application dedicated to providing network service
- Starts at system initialization time
- Runs on a remote computer
- Waits for service requests from clients
- Provides services to each clients
- Requires powerful hardware and sophisticated operating system

7

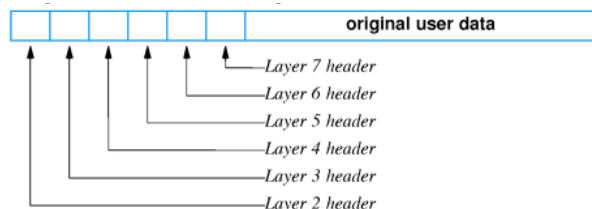
Transport protocols and client-server paradigm

- Clients and servers exchange messages through transport protocols: TCP or UDP



Protocol headers

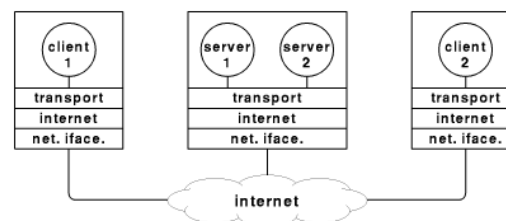
- The software at each layer communicates through information stored in headers
- Each layer adds its header to the front



9

Multiple services on one computer

- Sufficiently powerful computer - multi-tasking OS - may run multiple servers
- Servers run as independent processes and can manage clients simultaneously



10

Identifying a service

- Each service gets a unique identifier; both client and server use that identifier
 - ❖ Server registers with local protocol software under the identifier
 - ❖ Client contacts protocol software for session under that identifier
- Example - TCP uses protocol port numbers as identifiers

11

Connection-oriented and connectionless transport

- TCP - connection-oriented
 - ❖ Client establishes connection to server
 - ❖ Exchange multiple messages of arbitrary size
 - ❖ Client terminates connection
- UDP - connectionless
 - ❖ Client constructs message
 - ❖ Client sends message to server
 - ❖ Server responds
 - ❖ Message must fit in one UDP datagram

12

Client-server interactions

- Clients can access multiple services sequentially
- Clients may access different servers for one service
- Servers may become clients of other servers
- Circular dependencies may arise...

13

Socket Interface

- The socket is one form of interface between application programs and protocol software
- Provides OSI Layer 5 (Session) Functionality
- Widely available - program portability
- Used by both clients and servers
- Extension to UNIX *file I/O* paradigm
- WinSock for MS Windows

14

The Socket API

- Interface to protocol is call *Application Program Interface (API)*
 - ❖ Created for use with a programming language for a specific operating system
 - ❖ Includes collection of procedures for application program
- Application interactions with protocol software
- *Socket API* is a specific protocol API

15

Common socket system calls

- socket - create a new socket
- close - terminate use of a socket
- bind - attach a network address to a socket
- listen - wait for incoming messages
- accept - begin using incoming connection
- connect - make connection to remote host
- send - transmit data through connection
- recv - receive data through active connection

16