Network : Protocol

Jae Hyeon Kim

Reference

William Stalling, Data and Computer Communications 10/E, Prentice Hall

Protocol Architecture

- Need for protocol architecture to transfer data several tasks must be performed
  1. The source must activate communications path or inform network of the identity of the desired destination
  2. The source must make sure that dest. Is prepared to receive data
  3. The file transfer application on source must confirm file mgmt program at destination is prepared to accept and store the file
  4. A format translation function may need to be performed if the formats used on the two systems are different

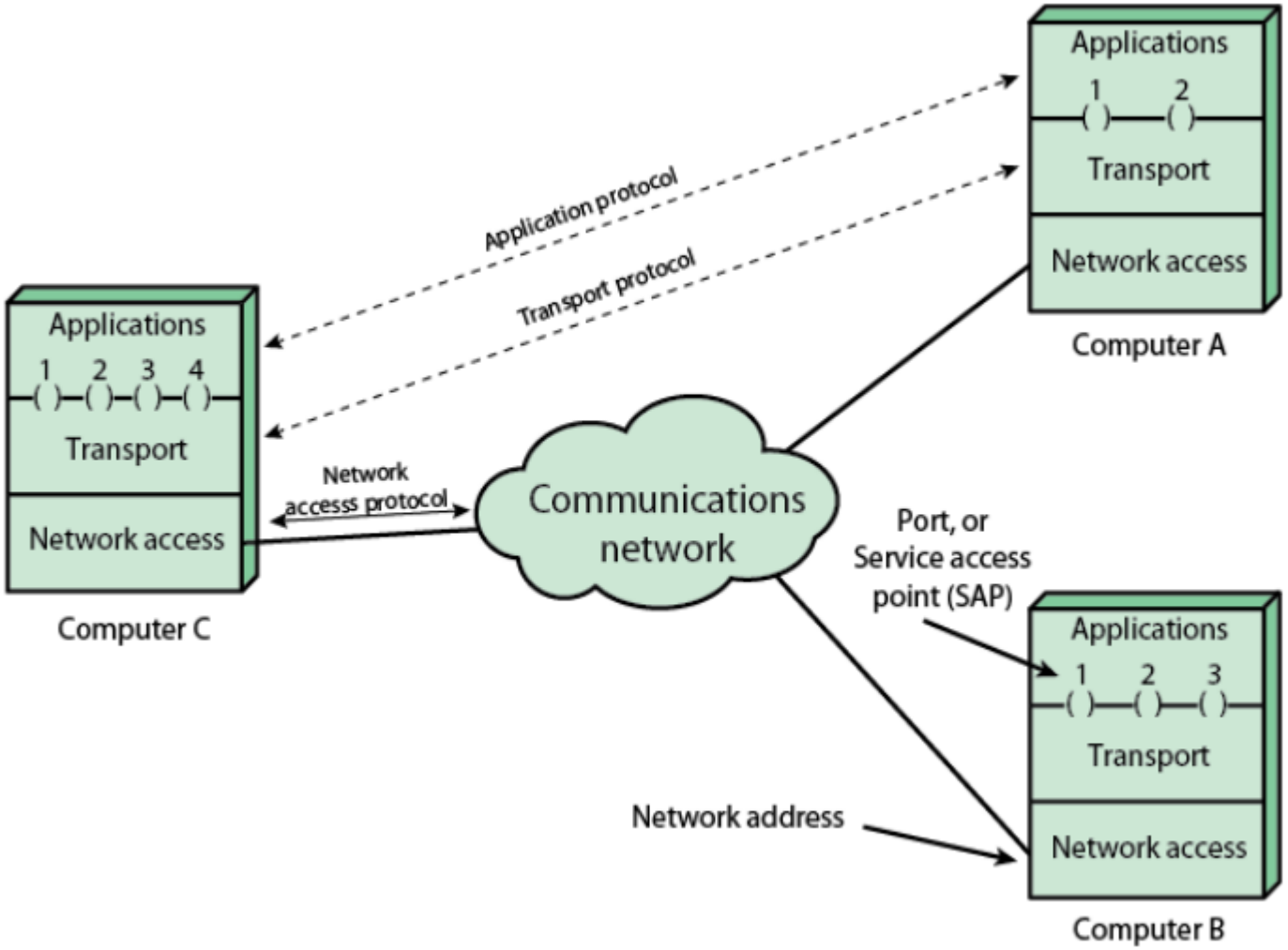
Function of Protocol Architecture

- Breaks logic into subtask modules
- Modules are arranged in a vertical stack, so layering
  - Each layer in the stack performs a subset of functions
  - Relies on next lower layer for primitive functions
  - Provides services to the next higher layer
  - Changes in one layer should not require changes in other layers
- A protocol is a set of rules or conventions that allow peer layers to communicate, the key features of a protocol are:
  - Syntax : format of data blocks
  - Semantics : control information for coordination
  - Timing : speed matching and sequencing

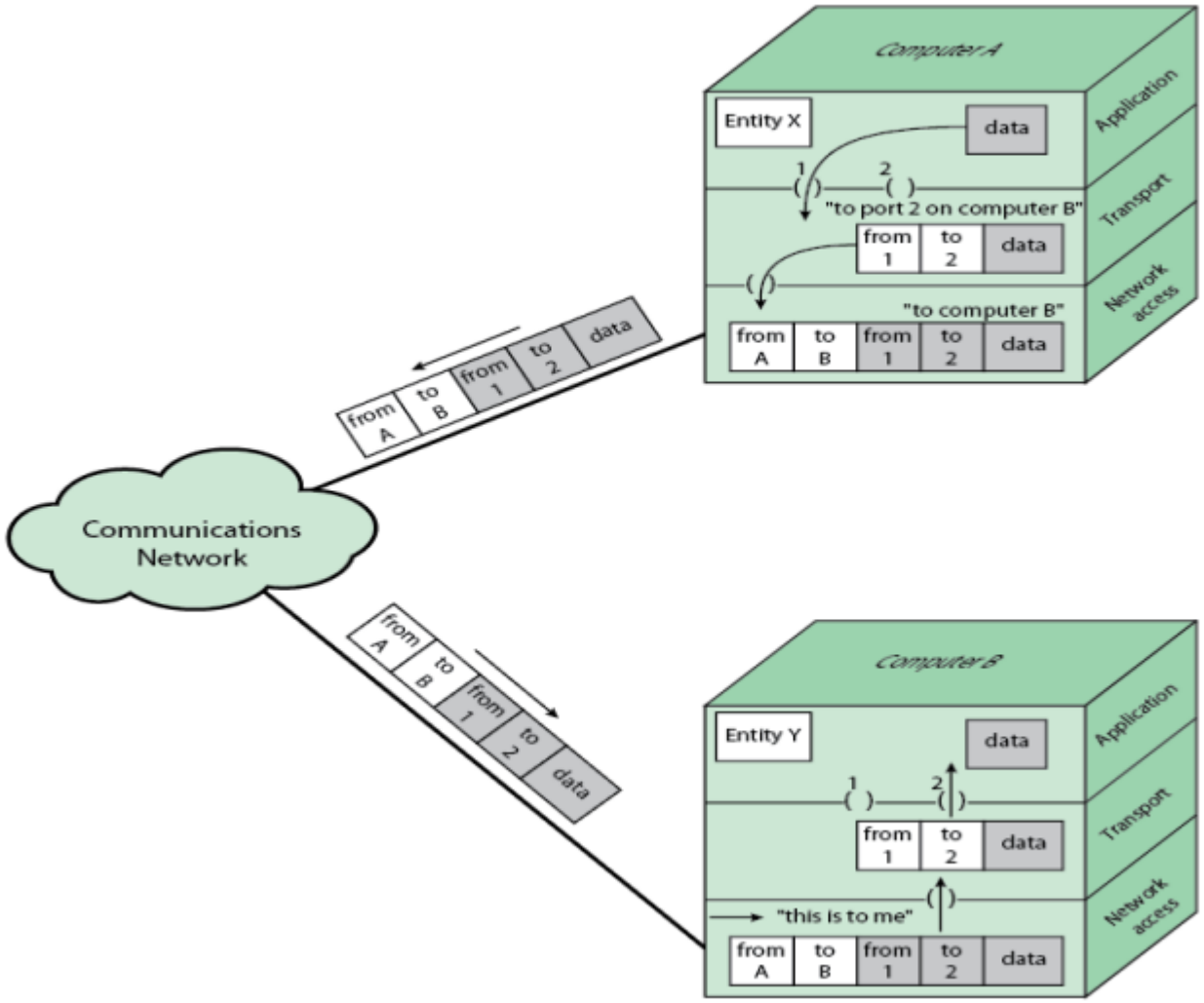
Communication Layer

- Communication tasks are organized into three relatively independent layers:
  - Application layer : contains logic to support applications
  - Transport layer : provides reliable data transfer
  - Network access layer : concerned with the exchange of data between a computer and the network to which it is attached

Protocol Architecture and Networks



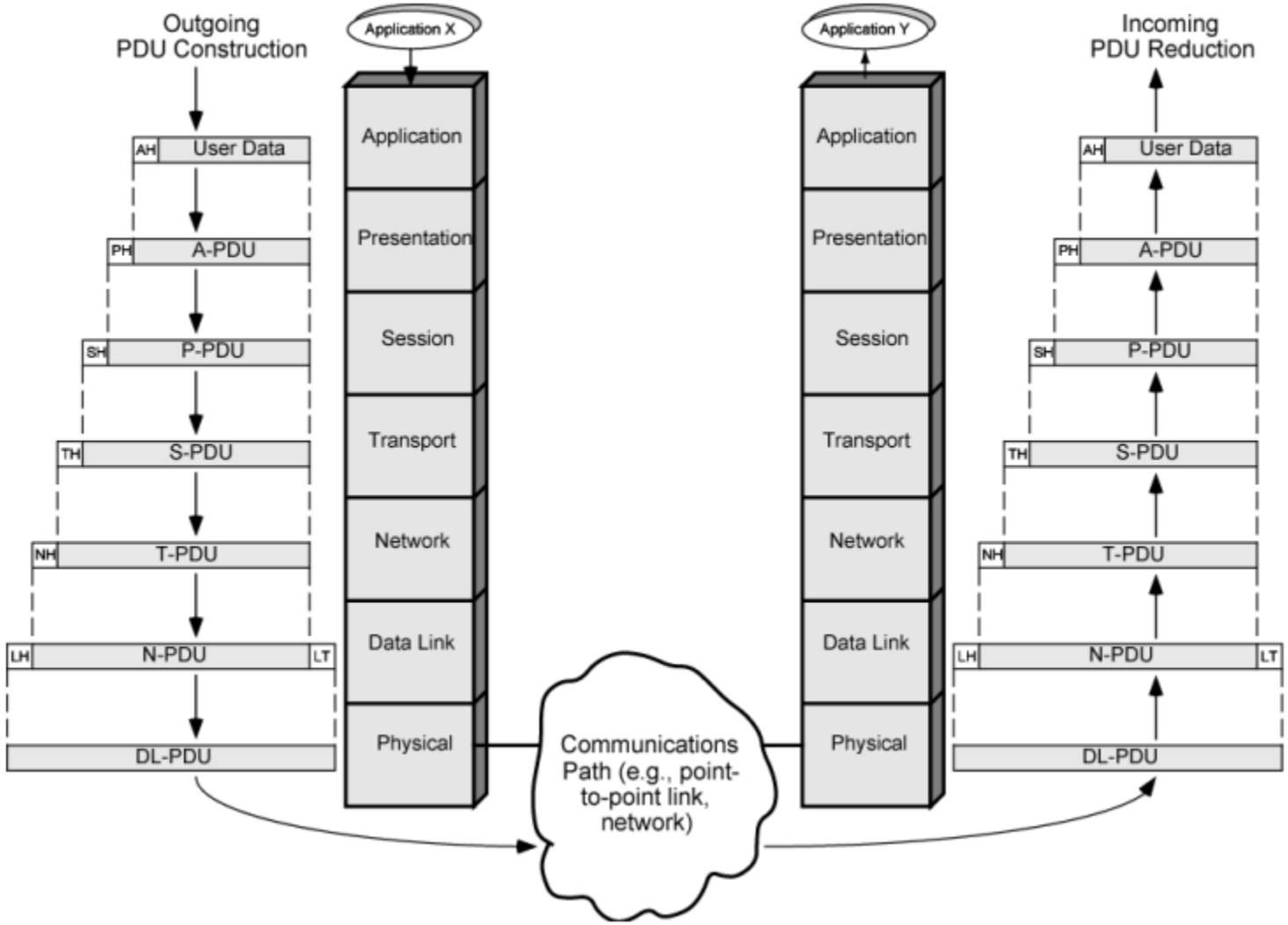
Protocol in a Simplified Architecture



Standardized Protocol Architectures

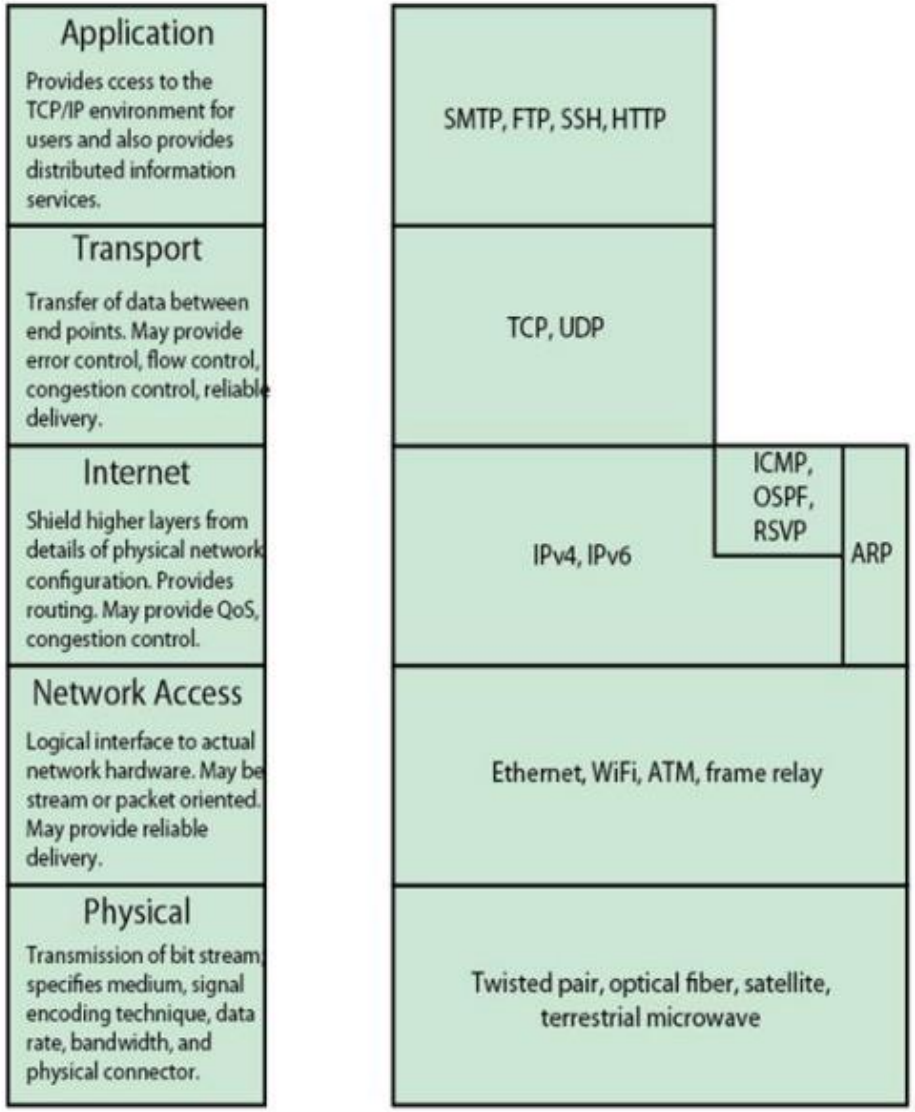
- Required for devices to communicate
- Vendors have more marketable products
- Customers can insist on standards based equipment
- Two standards:
  - OSI reference model : never lived up to early promises
  - TCP/IP protocol suite : most widely used
- Also : IBM System Network Architecture(SNA)

OSI Reference Model

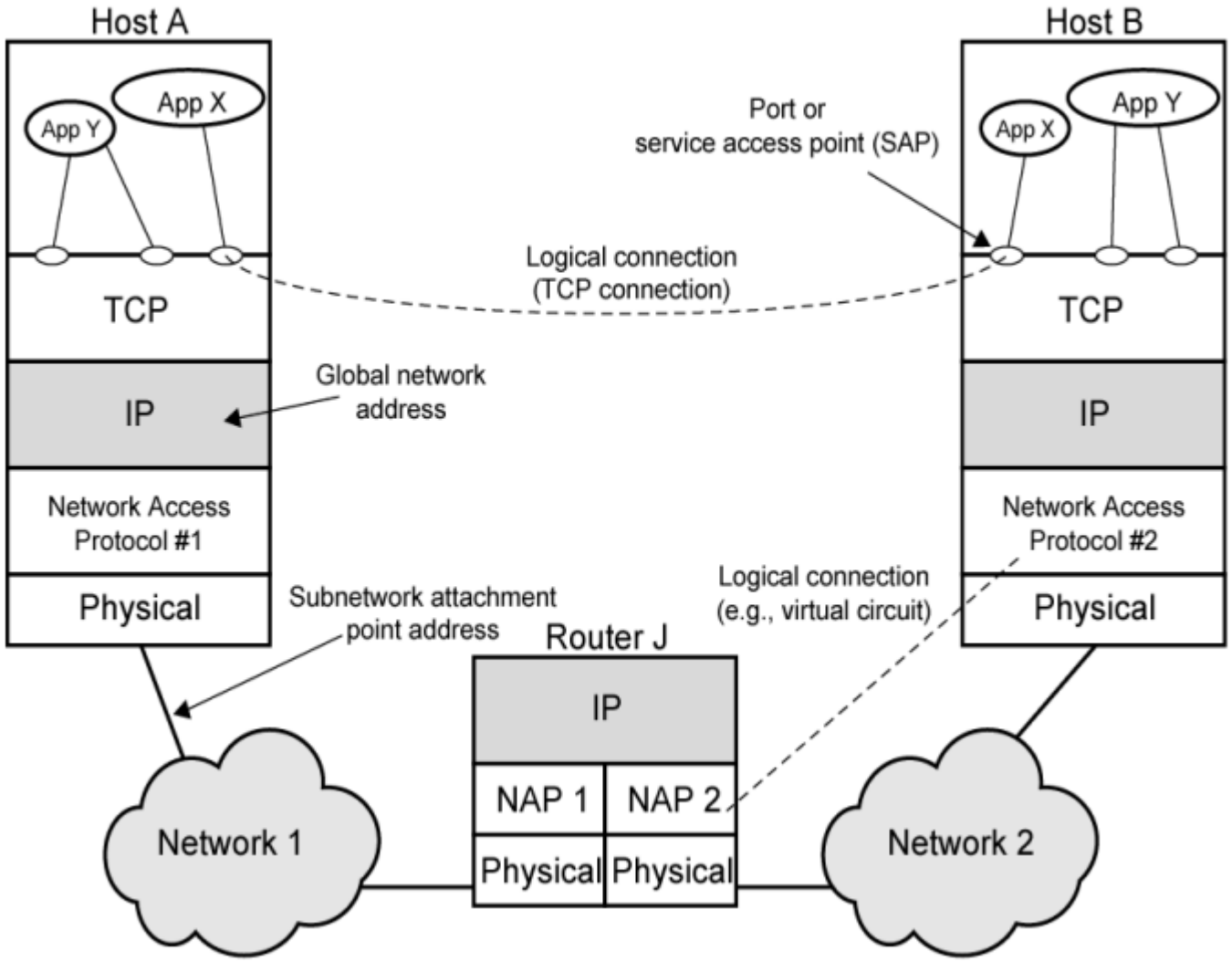


TCP/IP Layers and Example Protocols

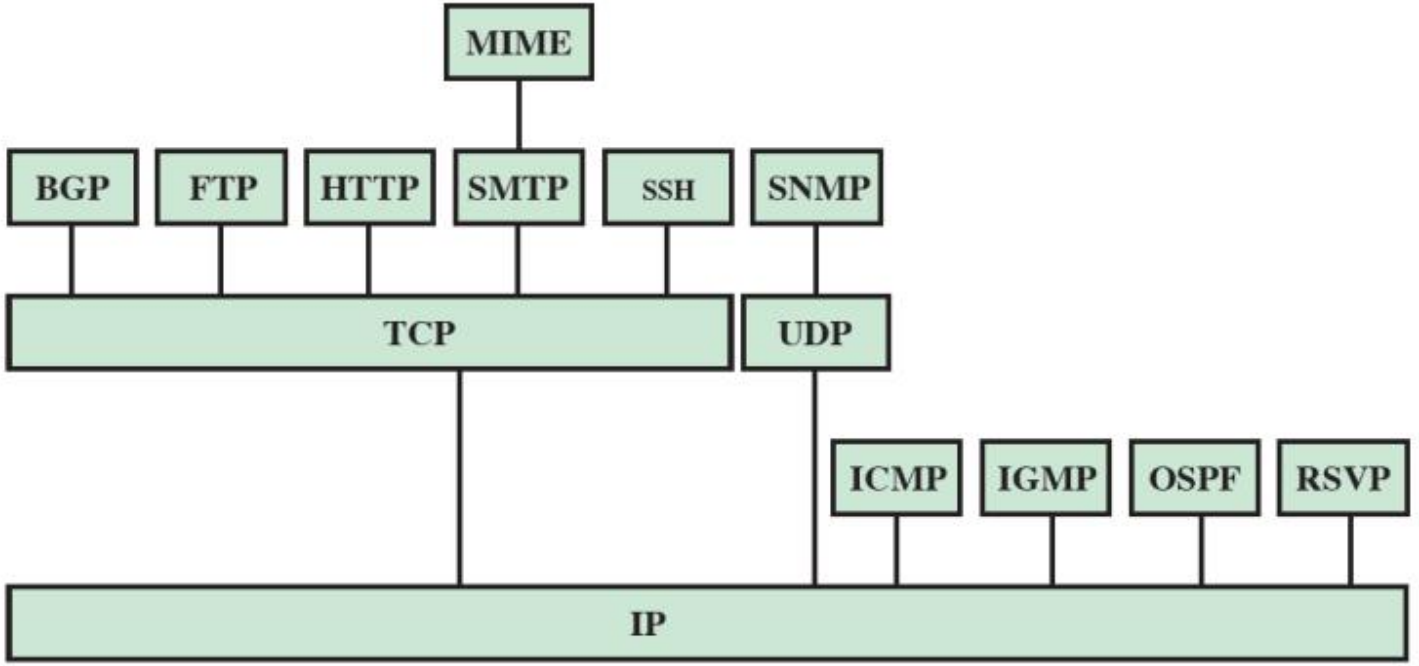
- Dominant commercial protocol architecture
  - Specified and extensively used before OSI
- No official model but a working one



Operation of TCP/IP



Some Protocols in TCP/IP Suite



BGP = Border Gateway Protocol
FTP = File Transfer Protocol
HTTP = Hypertext Transfer Protocol
ICMP = Internet Control Message Protocol
IGMP = Internet Group Management Protocol
IP = Internet Protocol
MIME = Multipurpose Internet Mail Extension
OSPF = Open Shortest Path First
RSVP = Resource ReSerVation Protocol
SMTP = Simple Mail Transfer Protocol
SNMP = Simple Network Management Protocol
SSH = Secure Shell
TCP = Transmission Control Protocol
UDP = User Datagram Protocol

Standards

- Required to allow for interoperability between equipment
  - De facto standards e.g.) UNIX, IBM PC, Windows
  - De jure standards e.g.) OSI 7 layers
- Advantages
  - Ensures a large market for equipment and software
  - Allows products from different vendors to communicate
- Disadvantages
  - Freeze technology
  - May be multiple standards for the same thing
- Web sites for IETF, IEEE, ITU-T, ISO

Socket Programming

- Concept was developed in the 1980s in the UNIX environment as the Berkeley sockets interface
  - De facto standard application programming interface (API)
  - Basis for Window Sockets (WinSock)
- Enables communications between a client and server process
- It may be connections oriented or connectionless
- TCP/UDP header includes source port and dest. Port
  - These port values identify the respective users(applications) while the IP addresses identify the respective host systems

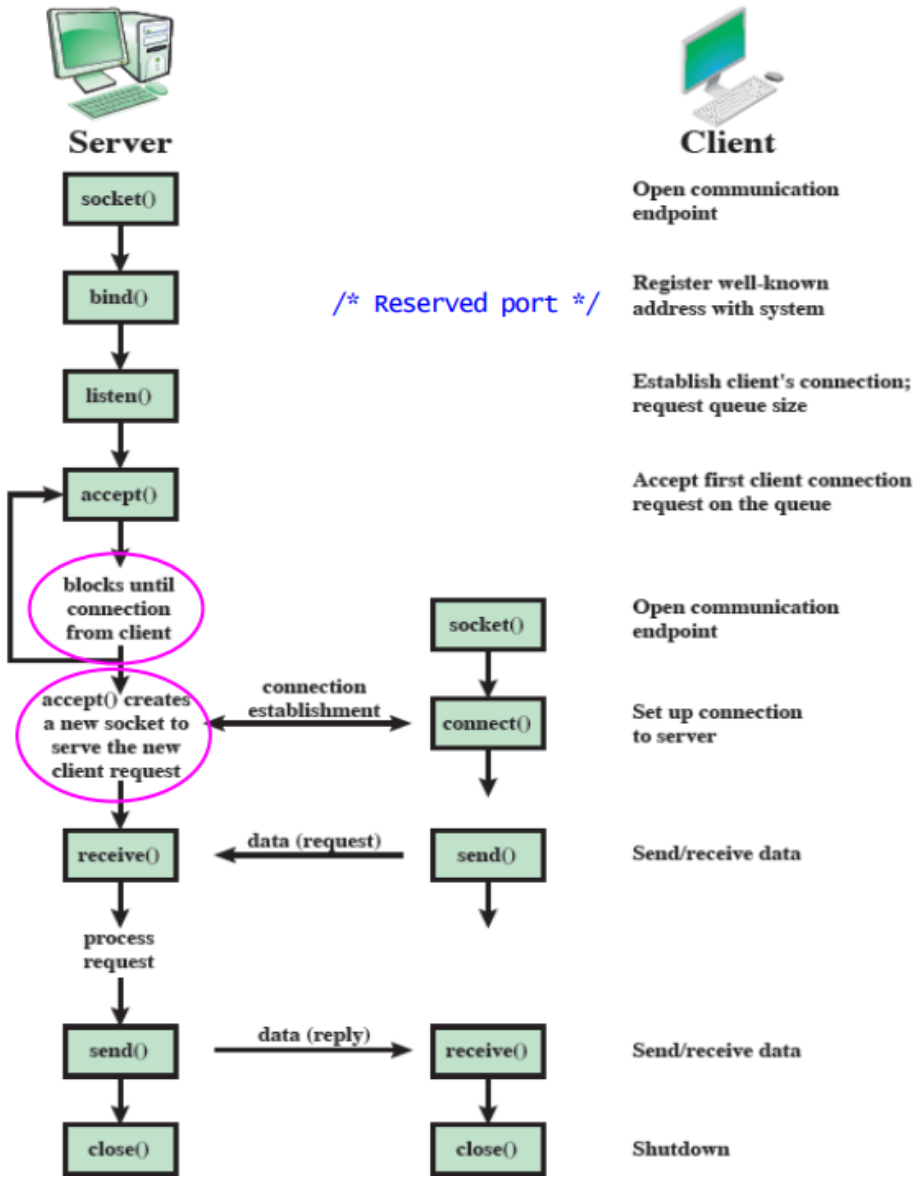
The Socket

- Formed by concatenating a port value and an IP address
  - Unique throughout the Internet
- Used to define a communication API
  - Generic interface for writing programs that use TCP or UDP
- Socket parameters
  - domain {AFUNIX, AF_INET, AF_INET6}
  - Type {stream, datagram, raw}
  - Protocol {TCP, UDP, ICMP}
- Socket data structure depends on the implementation
- Cf) reserved port ≈ well-known port ≈ server port

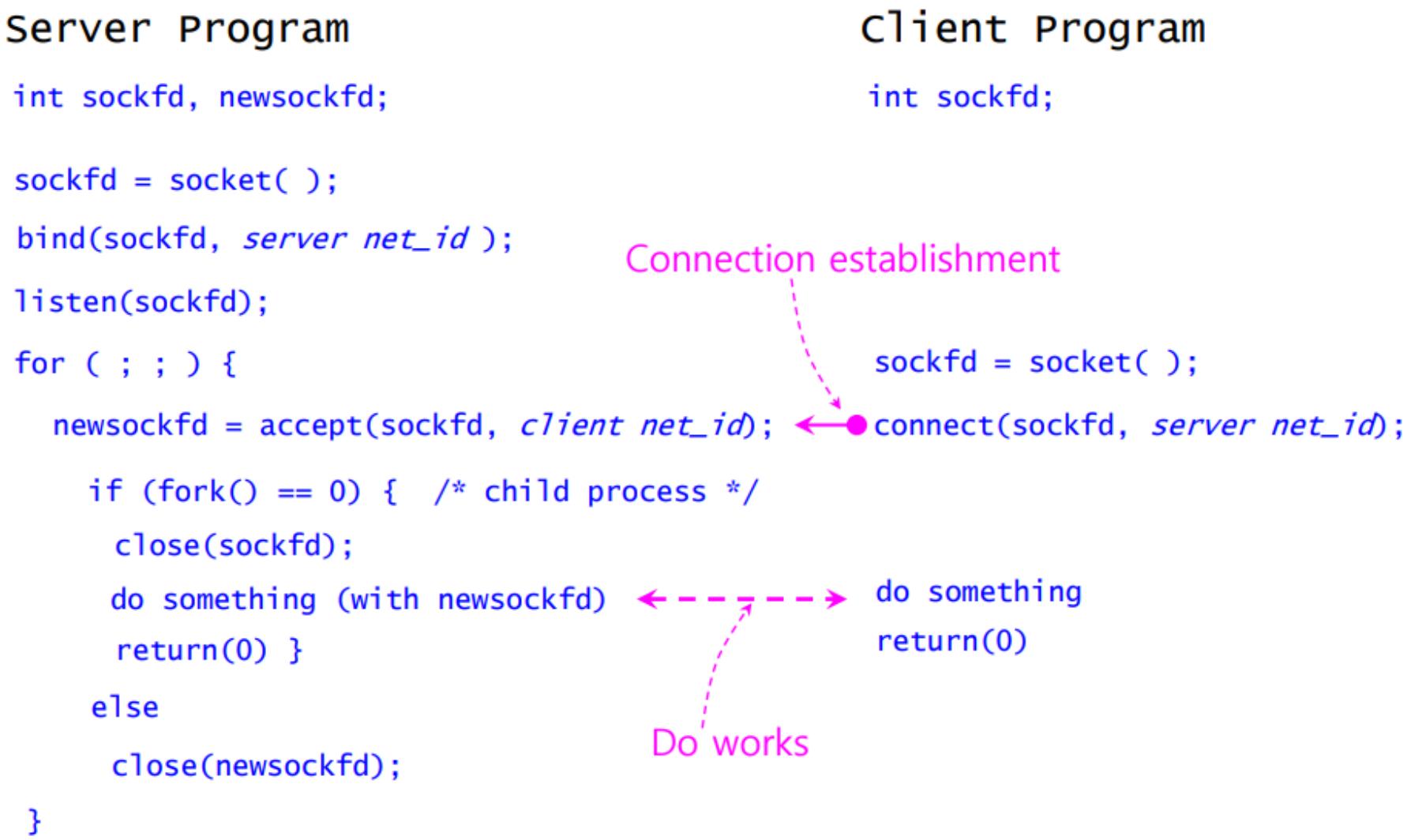
Core Socket Functions

Format	Function	Parameters
socket( )	Initialize a socket	**domain** Protocol family of the socket to be created (AF_UNIX, AF_INET, AF_INET6) **type** Type of socket to be opened (stream, datagram, raw) **protocol** Protocol to be used on socket (UDP, TCP, ICMP)
bind( ) /* Reserved port */	Bind a socket to a port address	**sockfd** Socket to be bound to the port address **localaddress** Socket address to which the socket is bound **addresslength** Length of the socket address structure
listen( )	Listen on a socket for inbound connections	**sockfd** Socket on which the application is to listen **queuesize** Number of inbound requests that can be queued at any time
accept( )	Accept an inbound connection	**Sockfd** Socket on which the connection is to be accepted **remoteaddress** Remote socket address from which the connection was initiated **addresslength** Length of the socket address structure
connect( )	Connect outbound to a server	**Sockfd** Socket on which the connection is to be opened **remoteaddress** Remote socket address to which the connection is to be opened **addresslength** Length of the socket address structure
send( ) recv( ) read( ) write( )	Send/receive data on a stream socket (either send/recv or read/write can be used)	**sockfd** Socket across which the data will be sent or read **data** Data to be sent, or buffer into which the read data will be placed **datalength** Length of the data to be written, or amount of data to be read
sendto( ) recvfrom( )	Send and receive data on a datagram socket	**sockfd** Socket across which the data will be sent or read **data** Data to be sent, or buffer into which the read data will be placed **datalength** Length of the data to be written, or amount of data to be read
close( )	Close a socket	**sockfd** Socket which is to be closed

Socket System Calls for Connection-oriented Protocol



Skeleton of Client/Server Programming



Client/Server Operational View