CISC 3120

C18: Network Fundamentals and Reliable Sockets

Hui Chen

Department of Computer & Information Science

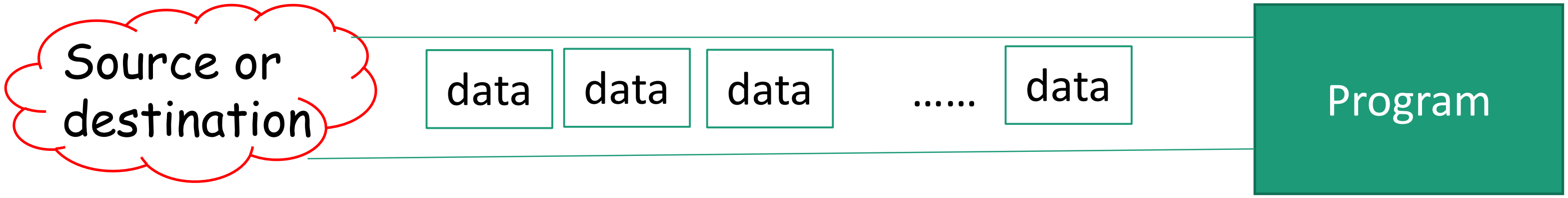
CUNY Brooklyn College

Outline

- Networking fundamentals
- Network interfaces
- Reliable sockets
- Reliable sockets and streams

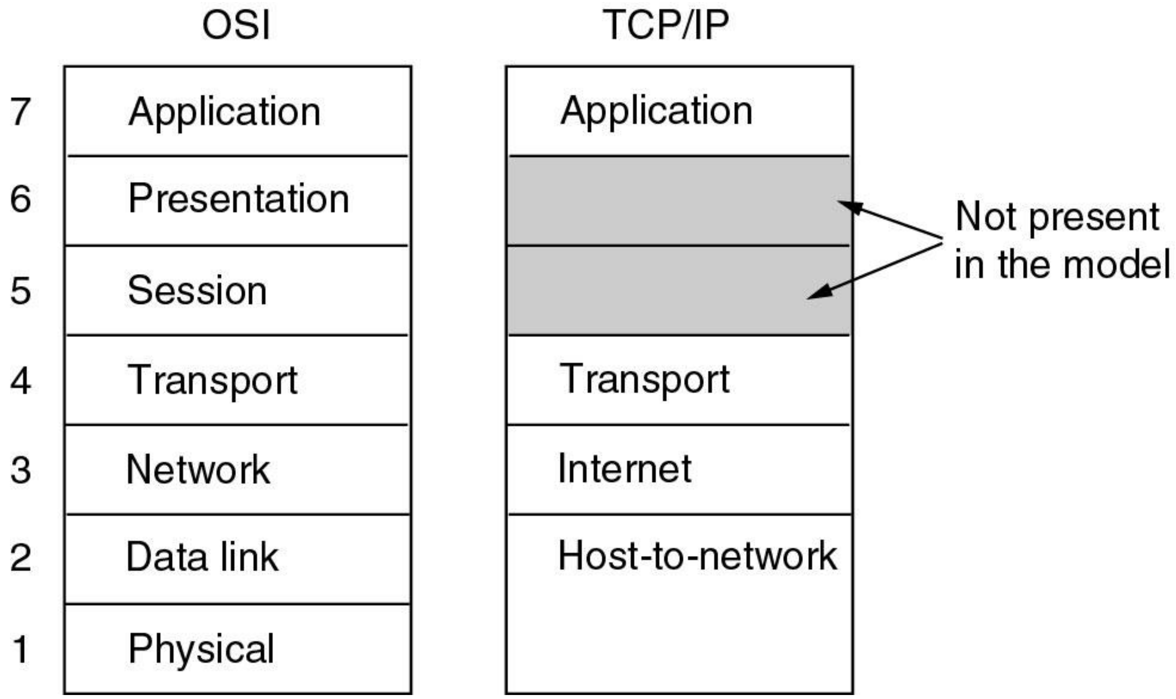
Network I/O

- A source or a destination of an I/O stream can be on a computer network
- How do we identify a source or a destination on the network?



Layered Architecture

- OSI model and TCP/IP

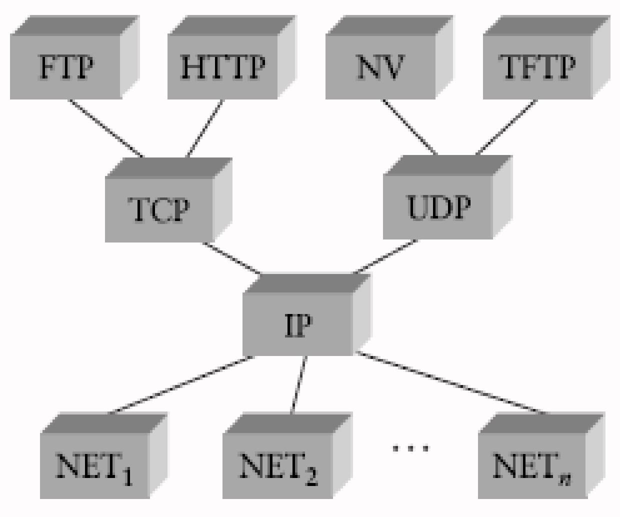


	OSI	TCP/IP
7	Application	Application
6	Presentation	
5	Session	
4	Transport	Transport
3	Network	Internet
2	Data link	Host-to-network
1	Physical	

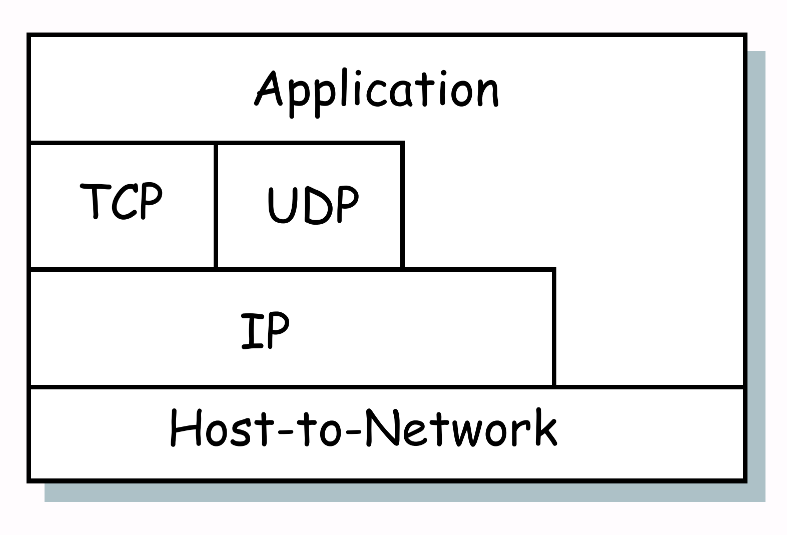
Not present in the model

The Internet Architecture (TCP/IP)

- Layering is not strict, hourglass design, representative implementation



Internet protocol graph.



Internet architecture.

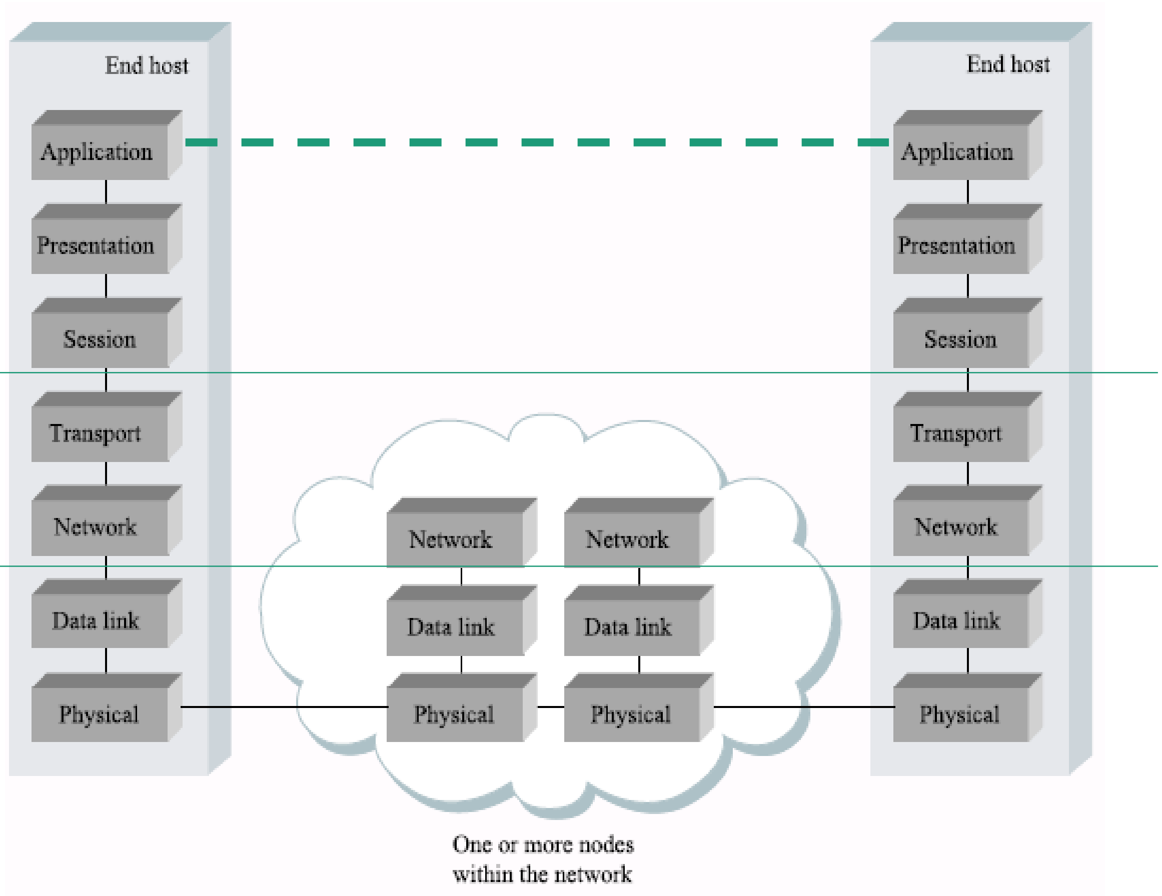
Two Hosts and a Router

- Typical delegation of responsibilities

Application and Application Libraries (e.g., JVM and Java applications)

Operating System

Network Interface Card/Adapter (NIC)



Network Protocol

- A distributed algorithm and associated data structures for data communication over a network
- Each layer may have many protocols

Host and Network Interface

- A host may have multiple network interface
- A network interface typically implements physical layer and link layer functionality (or the host-to-network layer)

Network Layer

- Example protocol
  - The Internet Protocol (IP)
  - Communication protocol for <u>hosts</u>
  - Transmit and receive IP packets
  - To identify a host on the Internet, use an IP address

IP Address

- Currently deployed Internet Protocols
  - IP version 4 (IPv4)
  - IP version 6 (IPv6)
  - The very first field in an IP packet indicates the version of IP protocol
  - Globally unique except local networks & private networks
  - Hierarchical (network number + host number)

IPv4 Address

- 32 bit integer
  - Divided into two parts
    - Network number and host number (using prefix or network mask)
- Human-readable form
  - IPv4 numbers-and-dots notation, each number corresponds to a byte in the address
  - Example: 146.245.201.50
- Facing exhaustion of address space, moving to IPv6

IPv4 Private Networks

- Private networks
  - Not routable in a public network
  - 24-bit block 10.0.0.0–10.255.255.255
  - 20-bit block 172.16.0.0–172.31.255.255
  - 16-bit block 192.168.0.0–192.168.255.255

IPv4 Link Local and Loopback Address

- Link local address
  - Not routable
  - For configuration purpose
  - 169.254.0.0/16 (16 bit block: 169.254.0.0 – 169.254.255.255)
- Loopback address
  - Only stay within the host
  - 127.0.0.0/8 (24 bit block: 127.0.0.0 – 127.255.255.255)

Broadcast, Multicast, and Unicast

- The addresses are divided into broadcast, multicast, and unicast address
  - Broadcast address: all 1's in the host number for the network
  - IPv4 Multicast: 224.0.0.0/4 (224.0.0.0 – 239.255.255.255)

A Few IPv4 Address Types

Address Type	Binary Prefix	IPv4 CIDR Notation
Private Network	1100 0000 1010 1000	192.168.0.0/16
	1010 1100 0001	172.16.0.0/12
	1010 0000	10.0.0.0/8
Loopback	0111 1111	127.0.0.0/8
Link-local Unicast	1111 1110 10	169.254.0.0/16
Documentation (TEST-NET-1)	1100 0000 0000 0000 0000 0010	192.0.2.0/24
Documentation (TEST-NET-2)	1100 0110 0011 0011 0110 0100	198.51.100.0/24
Documentation (TEST-NET-3)	1100 1011 0000 0000 0111 0001	203.0.113.0/24
Multicast	1110	224.0.0.0/4
Global Unicast	Everything else (with exceptions)	

IPv6 Address

- 128 bits/16 bytes in length
- IPv6 Notation: a human friendly text representation
- x:x:x:x:x:x:x:x where x is a 16-bit (or 2-byte) hexadecimal number, e.g.,
    - `47CD:1234:4422:AC02:0022:0022:1234:A456`
- Contiguous 0s can be compressed, e.g.,
    - `47CD:0000:0000:0000:0000:0000:A456:0124`
    - can be written as
    - `47CD::A456:0124`

A Few IPv6 Address Types

Address Type	Binary Prefix	IPv6 Notation
Unspecified	00…0 (128 bits)	::/128
Loopback	00…1 (128 bits)	::1/128
Multicast	1111 1111	FF00::/8
Link-local Unicast	1111 1110 10	FE80::/10
Private Network	1111 110	FC00::/7
Documentation	0010 0000 0000 0001 0000 1101 1011 1000	2001:0DB8::/32
Global Unicast	Everything else (with exceptions)	

Host Name

- A host may be identified by its name
  - Example: the Domain Name Service (DNS)
- Domain Name Service (DNS)
  - A global name database, and an application on the Internet that does the translation
    - (host name/DNS resolution) Host name → IP address
    - (reverse host name/DNS resolution) IP address → host name
  - Example
    - www.brooklyn.cuny.edu
    - www.google.com
  - Communications are done using IP addresses
    - DNS provides the translation

Look Up Host IP Address

- While on a host, you can look up its IP addresses
- Be aware that a host may have multiple IP addresses
  - an IP address is assigned to a network interface on a host, and a host can have multiple network interfaces
  - a network interface can be assigned multiple IP addresses
- Windows
  - ipconfig
- Mac OS X
  - ifconfig
- Linux
  - ip address or ifconfig

Look Up IP addresses for Host Names

- Use nslookup, available on many operating systems (Windows, Mac OS X, Linux …)
- Use dig on Linux
- Example
  - nslookup www.google.com
  - nslookup www.brooklyn.cuny.edu
  - dig www.google.com

Work with Network Interface

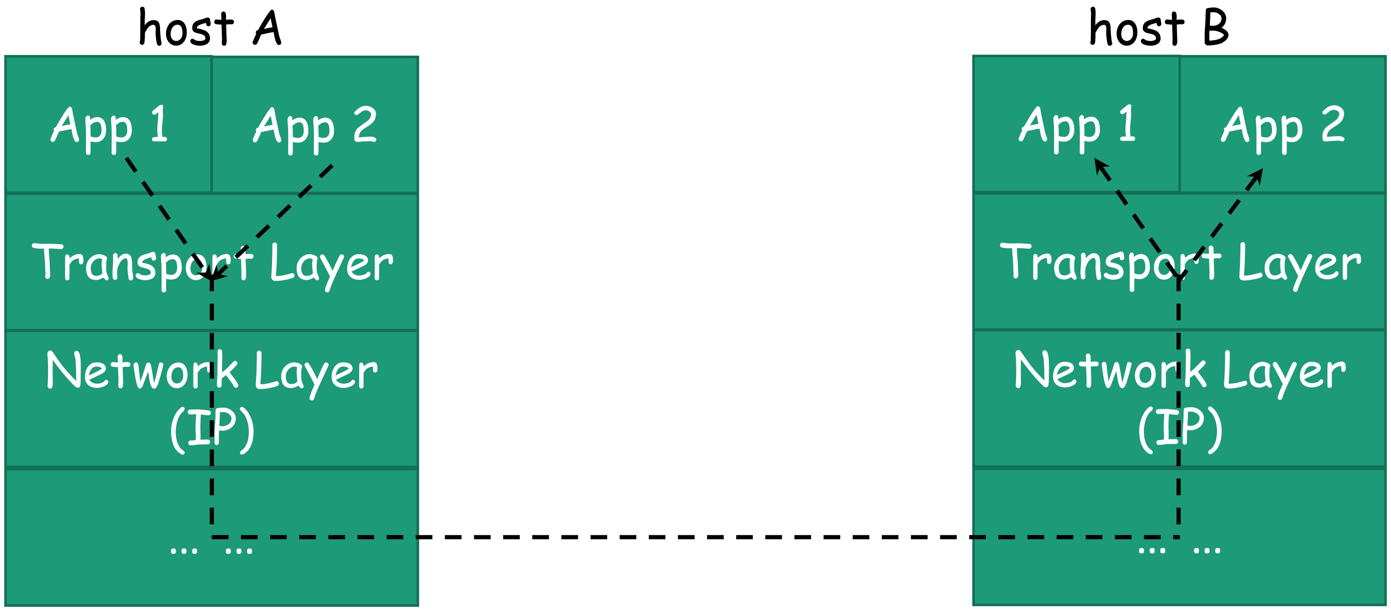
- In Java, use java.net.NetworkInterface to deal with network interfaces on a host
- Example application
    - What do you observe?
    - Link type, name, unicast address, broadcast address, network number …

Questions

- Network architecture and layered model
- Host, node, and network interface
- IP addresses
  - IPv4 and IPv6
- Practical operations
  - Look up hosts' IP addresses
  - Examine network interfaces

Multiplexing and Demultiplexing

- Network layer functionality belongs to a host
- How do applications share the network?
- Transport layer: multiplexing and demultiplexing



TCP and UDP

- Transport Control Protocol
- User Datagram protocol
- Communication protocol for <u>processes</u> (a process represents a running program)
- Multiplexing and demultiplexing over the network layer (the Internet protocol)

UDP

- User Datagram Protocol
- Implement solely multiplexing and demultiplexing over the network layer (the Internet protocol)
- Transmit independent datagram one at a time
- Communication is not reliable (called best effort)
  - No guarantee on the order of datagrams
  - No guarantee on the delivery of datagrams

TCP

- Transmission Control Protocol
- Besides multiplexing and demultiplexing, abstract a connection-oriented reliable byte stream
  - Create an abstraction data are transmitted or received one byte at a time, reliably
    - Maintain the order of the bytes
    - Guarantee delivery of data, or an error is reported
  - Must establish connection

TCP and UDP Port Numbers

- For multiplexing and demultiplexing, how do we differentiate multiple processes (running programs) on a host?
- UDP port numbers
  - 16 bit integer
  - Use them to differentiate different processes on a host
- TCP port numbers
  - 16 bit integer
  - Use them to differentiate different processes on a host

List TCP/UDP Port Statistics

- Use netstat , available on many operating systems (Windows, OS X, Linux …)
- Windows
    - Examples
        - netstat -n -o -p TCP; netstat -f -o -p TCP; netstat -n -o -p UDP; and netstat -f -o -p TCP
- Linux
    - Examples
        - netstat –n –p -a -t; netstat –p -a -t; netstat –n –p -a -u; and netstat–p -a -u
- OS X
    - Examples
        - netstat -n –a –p tcp; netstat –a –p tcp; netstat -n –a –p udp; and netstat –a –p udp;

Some Practical Considerations

- Is a port (TCP, UDP, or both) available to our own programs?
  - 1 – 1023 are privileged
  - Registered ports (with iana.org, sometimes called well-known or service ports)
    - See /etc/services on Mac OS X, or, Linux or Unix
    - See C:\Windows\system32\drivers\etc\services on Windows
  - A process may be running and assigned (called bound to) one or more ports
    - A port can only be assigned to a single process
- Does the host-based or network-based firewall get in your way (at home, at the college, or at the coffee shop …)?
  - A firewall is an application that filter out some IP packets/TCP segments/UDP datagrams
  - Commonly, an organization only allows traffics to a small number of registered ports (e.g., 80 for HTTP, 443 for HTTPS, 53 for DNS)

Questions?

- Multiplexing and demultiplexing over the Internet
- TCP and UDP
- TCP port and UDP port
- Query network statistics on a host
- Some practical consideration
    - What ports are available for us to use in our own programs?

Programming with TCP and UDP

- Java network applications typically use TCP or UDP to communicate
- Typically no need to concern with innerworkings of TCP or UDP
  - Use java.net package or other network related packages
  - TCP communications
    - The Socket, ServerSocket, URL, and URLConnection classes
  - UDP communications
    - The DatagramPacket, DatagramSocket, and MulticastSocket classes
- Need to understand the concept of Socket
  - Most lower-level networking APIs are modeled after the Berkeley Socket API

Socket

- A data structure (or an object) representing a two-way communication link between two programs running on the network
  - Two end points
    - Local and remote end points
    - Each is a combination of IP address and port number
    - IP address: identify a host
    - Port number: identify a process (running program) on the host

TCP Socket: Client and Server

- TCP requires to establish a connection
- Client and server
  - Client
    - The program that actively initiates the connection establishment
  - Server
    - The program that passively waits for the client to connect to it, and accepts the connection
  - It is always that a client connects to the server, and server accepts the connection request in this context

TCP Socket in Java

- Socket and ServerSocket classes
    - Represent the connection between a client program and a server program.
        - A connection has two end points, so a socket usually has two end points (local and remote)
    - Socket class
        - Represent the connection at the client side of the connection
    - ServerSocket class
        - Present the connection at the server side of the connection
    - Low-level communication directly using TCP

TCP Client-Server Application using Sockets

- A server programs runs on a host and has a socket that is bound to a port number and an IP address.
- The server just waits, listening to the socket for a client to make a connection request.
- The client attempts to connect to the server program by using the server's address and port to which the server is listening to (service address and port).
    - To identify itself to the server, the client binds to a local port number (usually assigned by the system, not done by the programmer) that it will use during this connection.
- When the server accepts the connection, the server does the following,
    - It creates a new socket bound to the same local port (local endpoint) and also has its remote endpoint set to the address and port of the client.
    - It can continue to listen to the original socket for connection requests while tending to the needs of the connected client using the newly created socket
- On the client side, the socket is ready if the connection is accepted at the server
- The client and server can now communicate by writing to or reading from their sockets.

I/O Streams and TCP Socket in Java

- A TCP socket represents a connection at either the client or the server
- Socket can be source or destination of I/O streams
  - We can obtain an InputStream or an OutputStream from a socket
  - High-level streams can be created by wrapping the InputStream or the OutputStream

Example: Download a File

- Use try-catch-finally to handle errors and release resources
- Can you use try-with-resources instead?
- Regardless which one to use, make sure all the resources are closed
- Communication: unicast (one-to-one)

Questions

- Concept of socket
- TCP socket in Java
    - Concept of client and server
    - TCP sockets and I/O streams
    - Communication: unicast
- Make sure all resources are released

Questions

- Networking fundamentals
- Network interfaces
- Sockets and network I/O
- Reliable socket and byte streams

Assignments

- Practice assignments
- Project 4