CISC 3120
C19: User Datagram and Multicast

Hui Chen

Department of Computer & Information Science

CUNY Brooklyn College

Outline

- Recap
    - Network fundamentals
        - IPv4, IPv6 addresses
        - TCP and UDP
        - Unicast, broadcast, and multicast
- User datagram and datagram socket
- Datagram unicast, broadcast, and multicast
- Datagram packet and streams

A Few IPv4 Address Types

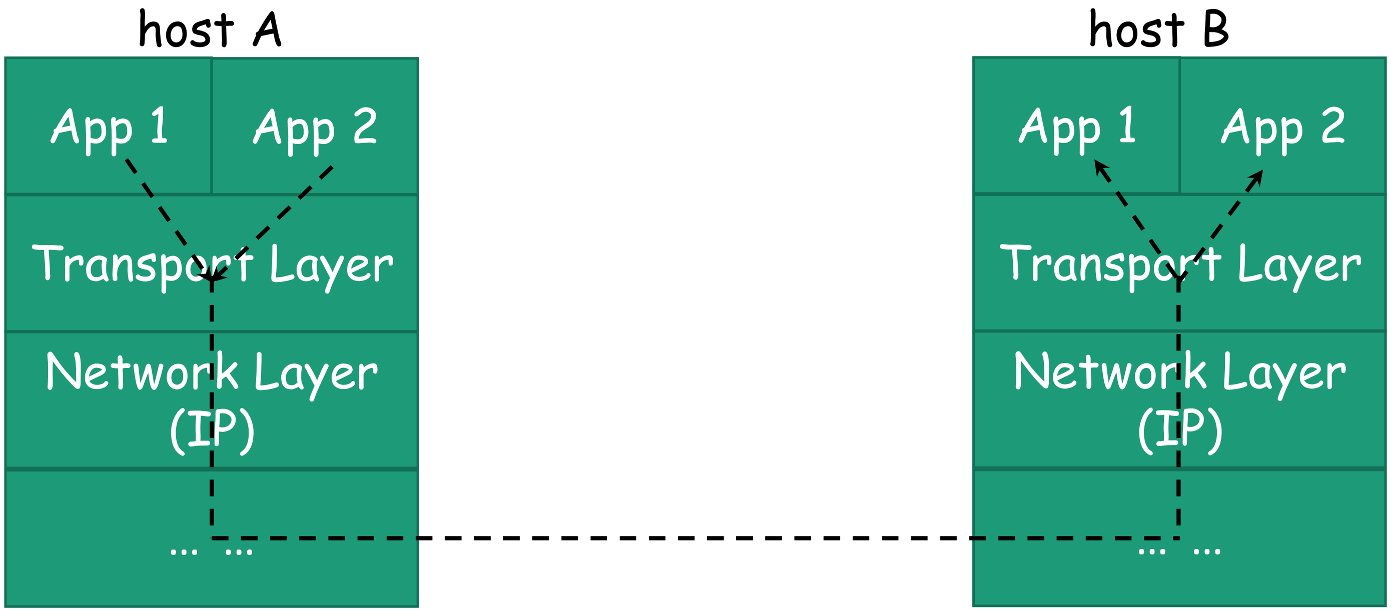
Address Type	Binary Prefix	IPv4 CIDR Notation
Private Network	1100 0000 1010 1000	192.168.0.0/16
	1010 1100 0001	172.16.0.0/12
	1010 0000	10.0.0.0/8
Loopback	0111 1111	127.0.0.0/8
Link-local Unicast	1111 1110 10	169.254.0.0/16
Documentation (TEST-NET-1)	1100 0000 0000 0000 0000 0010	192.0.2.0/24
Documentation (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)	

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)	

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 processes (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 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)

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

Questions

- Recap of relevant concepts

Use Datagram

- Datagram
  - Independent, self-contained message
  - Best effort, no connection establishment is required
  - Unreliable: there is no guarantee on arrival, arrival time, and order of arrival
  - Light weight (less resource)
- UDP in Java
  - The DatagramPacket, DatagramSocket, and MulticastSocket classes
- Unicast, broadcast, and multicast
  - Often use for broadcasting or multicasting

UDP Unicast: Example

- Echo a receive message (UDP datagram): knock, knock. Who is there?
- Essential classes: DatagramSocket, DatagramPacket
- KnockKnock1st: receive first
  - Create a DatagramSocket whose local end point is bound to an address and a port of the host
  - Receive a packet
  - Prepare and send a reply packet
- KnockKnock2nd: send first
  - Create a DatagramSocket, let JVM/OS determine the local port number
  - Prepare and send a packet, filled with destination address and port number
  - Receive a packet

Datagram Multicasting and Broadcasting

- One important use case of Datagram is to realize multicasting or Broadcasting
  - Multicasting: one-to-many
  - Broadcasting: one-to-all (a special case of multicast)
- Question
  - What kind of applications may multicast (or broadcast) benefit?
  - Why?

UDP Broadcast: Example

- Generate a list of random integers, and broadcast to any receivers
- Essential classes: DatagramSocket, DatagramPacket
- Broadcast Sender (one sender)
    - Create a DatagramSocket
    - Make sure SO_BROADCAST is enabled
        - Not all networks support broadcast
    - Send packets to broadcast address at a remote port
- Broadcast receivers
    - Receive packet at the designated port
        - Matching the port number of the remote end point given at the sender
- Use ByteArrayOutputStream, ByteArrayInputStream, DataInputStream, DataOutputStream to process packets

UDP Multicast: Example

- Generate a list of random integers, and multicast to a group of receivers
- Essential classes: DatagramSocket, MulticastSocket, DatagramPacket
- Multicast sender (one sender)
  - Create either a DatagramSocket or a MulticastSocket
  - Send packet to the two addresses representing two multicast groups
- Multicast receivers (a group of receivers)
  - Must create a MulticastSocket
  - Join one of the two multicast groups indicated by their respective mutilcast address
  - Receive packet
  - Leave the group when done
- Use ByteArrayOutputStream, ByteArrayInputStream, DataInputStream, DataOutputStream to process packets

IPv4 and IPv6

- Java made it transparent to use IPv4 or IPv6.
  - The code is essentially identical
  - The difference is to use an IPv4 or an IPv6 address, respectively

Questions

- User Datagram Protocol (UDP)
- UDP sockets in Java
- Connection-oriented (TCP) and connectionless (UDP)
- Unicast, broadcast, and multicast
- Use IPv4 or IPv6

Assignments

- Practice assignments
- Project 4