

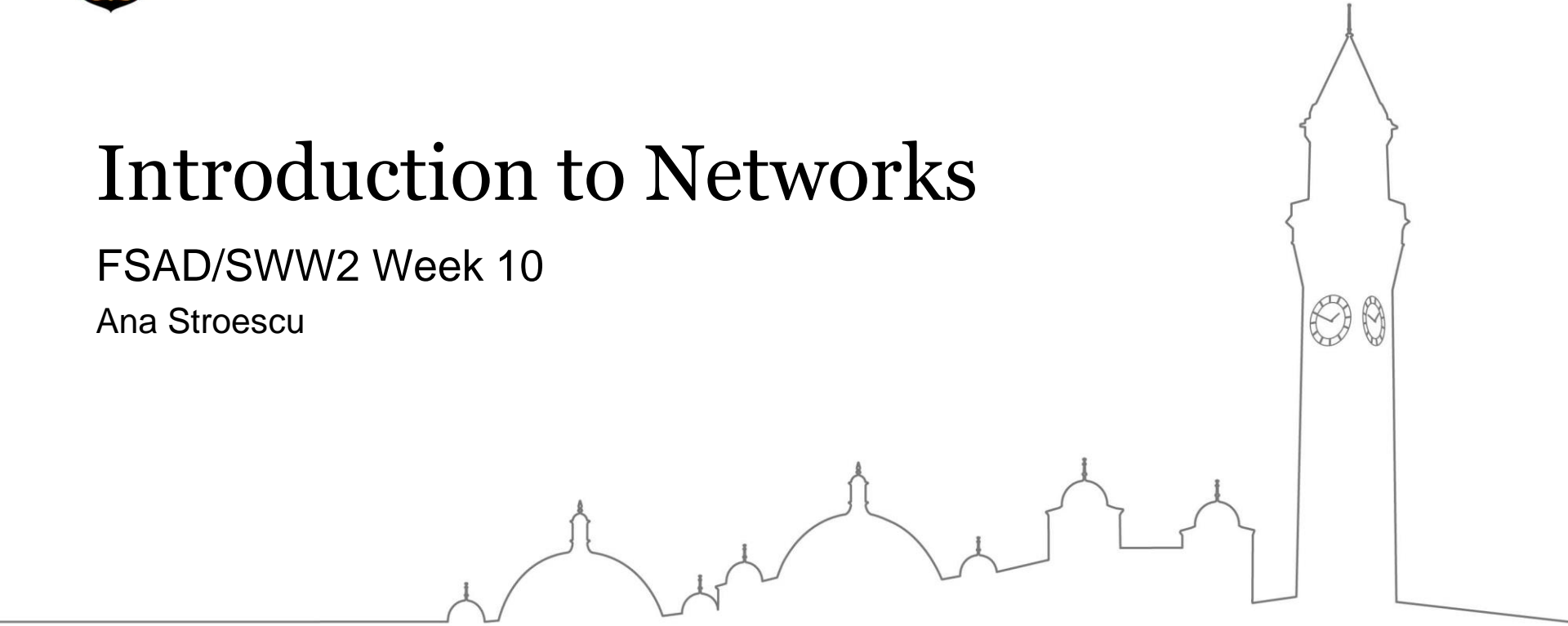


UNIVERSITY OF  
BIRMINGHAM

# Introduction to Networks

FSAD/SWW2 Week 10

Ana Stroescu



# Contents

- Objectives
- Networking Basics
- Communication Protocols
- TCP
- UDP
- TCP vs UDP
- Sockets
- Additional reading



# Objectives

- To understand client/server computing;
- To comprehend socket-based communication in Java;
- To implement java networking programs using server sockets;
- To develop servers for multiple clients;
- To send and receive objects on the network;
- To implement Java networking programs using datagram sockets.

# Networking Basics

## ■ IP address

- IP stands for “Internet Protocol”
- IP is a unique 32 bit address (IPv4) that identifies a device on the Internet or a local network
- Dotted decimal notation: 192.41.6.20
- Range: 0.0.0.0 – 255.255.255.255



## ■ Port

- Communication endpoint identified by a 16 bit number on your computer linked to your program
- Well known ports: 0..1023 are reserved ports, assigned and controlled  
Examples: FTP 21, TELNET 23, SMTP 25, HTTP 80, HTTPS 443, etc.
- Registered ports: 1024...49151 are not assigned or controlled, but can be registered
- Dynamic ports: 49152...65535 are not assigned, controlled or registered



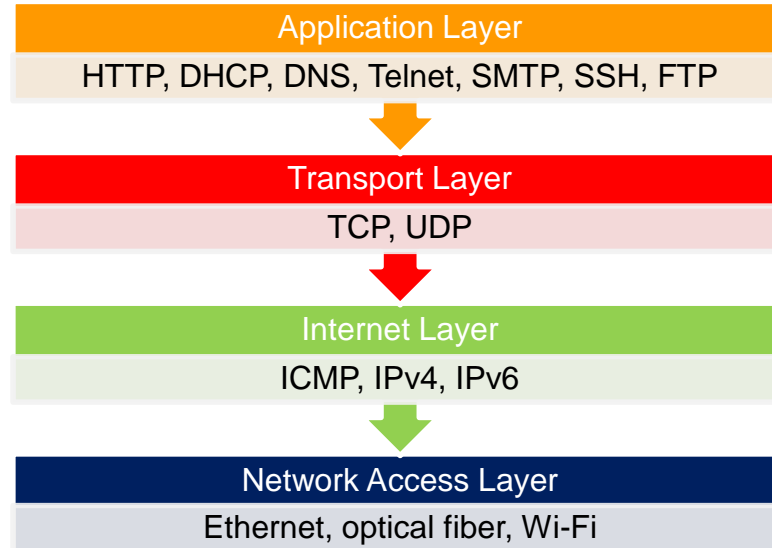
## ■ Packet

- Unit of data sent from one computer to another
- Data sent over the networks is divided into packets



- Internet Protocol (IP)

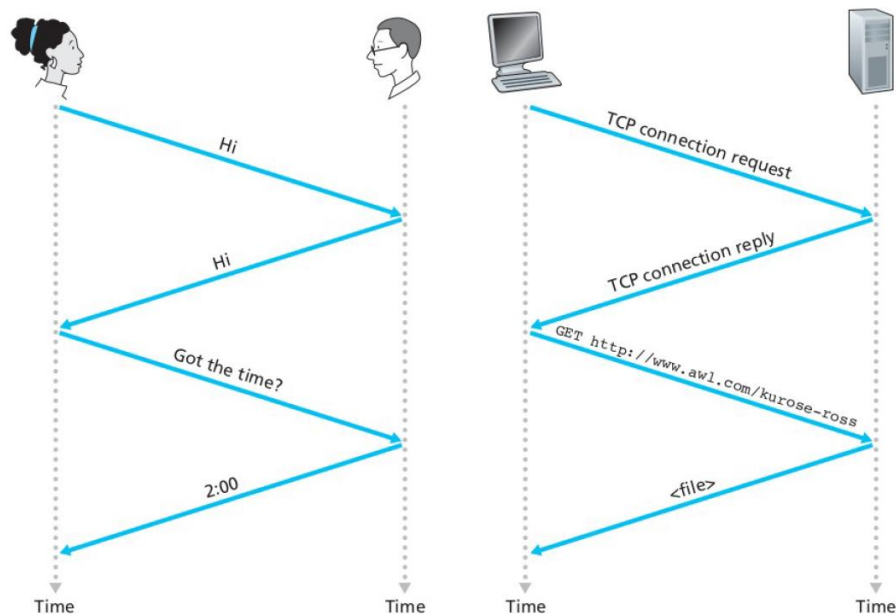
- The method by which data is sent from one computer (host) to another on the Internet.
- At the core of IP there are additional transport protocols that enable the actual communication between hosts. One of the main protocols is the TCP (Transmission Control Protocol), which is why IP is often referred to as TCP/IP.



TCP/IP model

# What is a communication protocol?

- Communication protocols describe the set of rules to be used in communication exchange.
- Protocols define format, order of messages sent and received among network entities and actions taken on message transmission and receipt.



A human protocol and a computer network protocol.

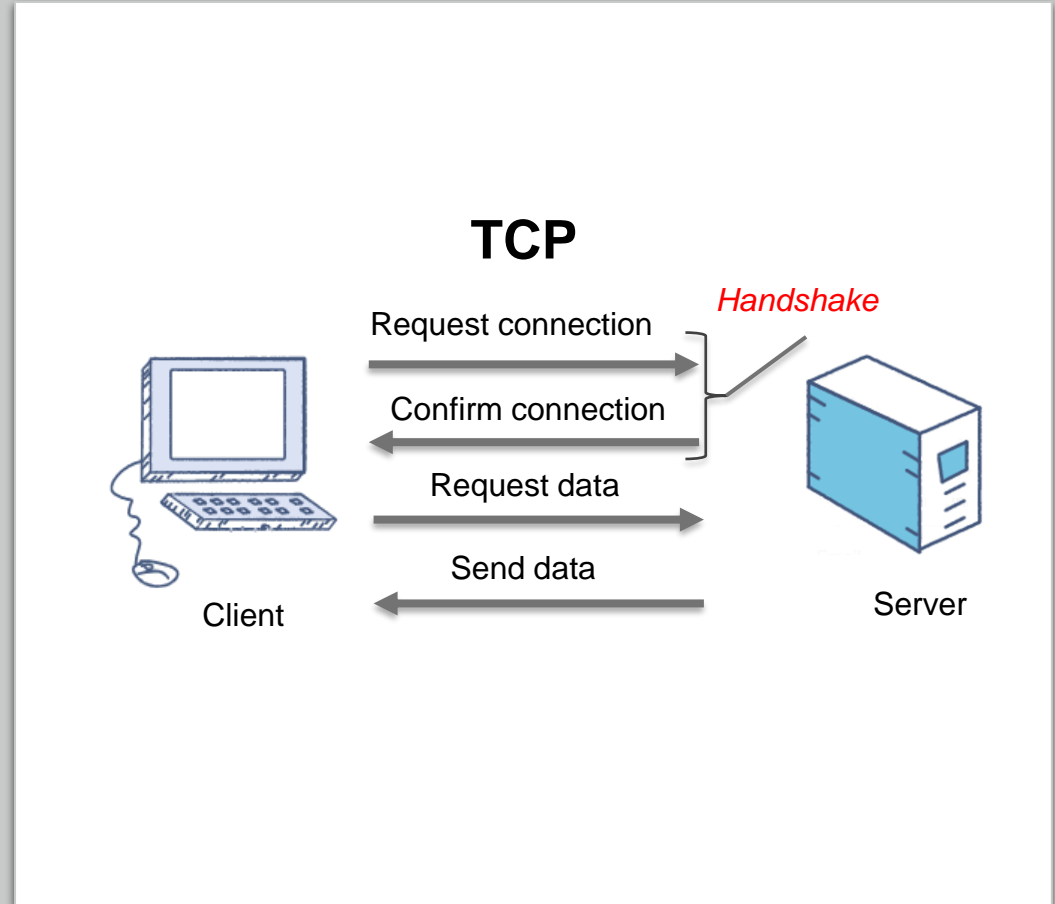
# Communication Protocols

- **TCP** (Transmission Control Protocol) – enables the flow of data across IP address connections.
  - **UDP** (User Datagram Protocol) – widely used for low-latency process communication such as DNS lookup and VoIP.
- 
- **FTP** (File Transfer Protocol) – built for accessing and managing files across connected IP hosts.
  - **HTTP** (Hypertext Transfer Protocol) – enables websites and web browsers to view content. It typically runs over port 80.
  - **HTTPS** – HTTP that runs with encryption, typically served over port 443.
  - **SMTP** (Simple Mail Transfer Protocol) – used by mail servers to send, receive, and/or relay outgoing mail between email senders and receivers.
  - .....

# TCP

**Transmission Control Protocol (TCP) – connection oriented, reliable.**

- Used when two applications want to communicate reliably.
- It is connection based.
- Packets are re-assembled in the order that they were sent.
- Transmission guaranteed, otherwise error is reported.
- Examples: HTTP, FTP, SMTP, Telnet.

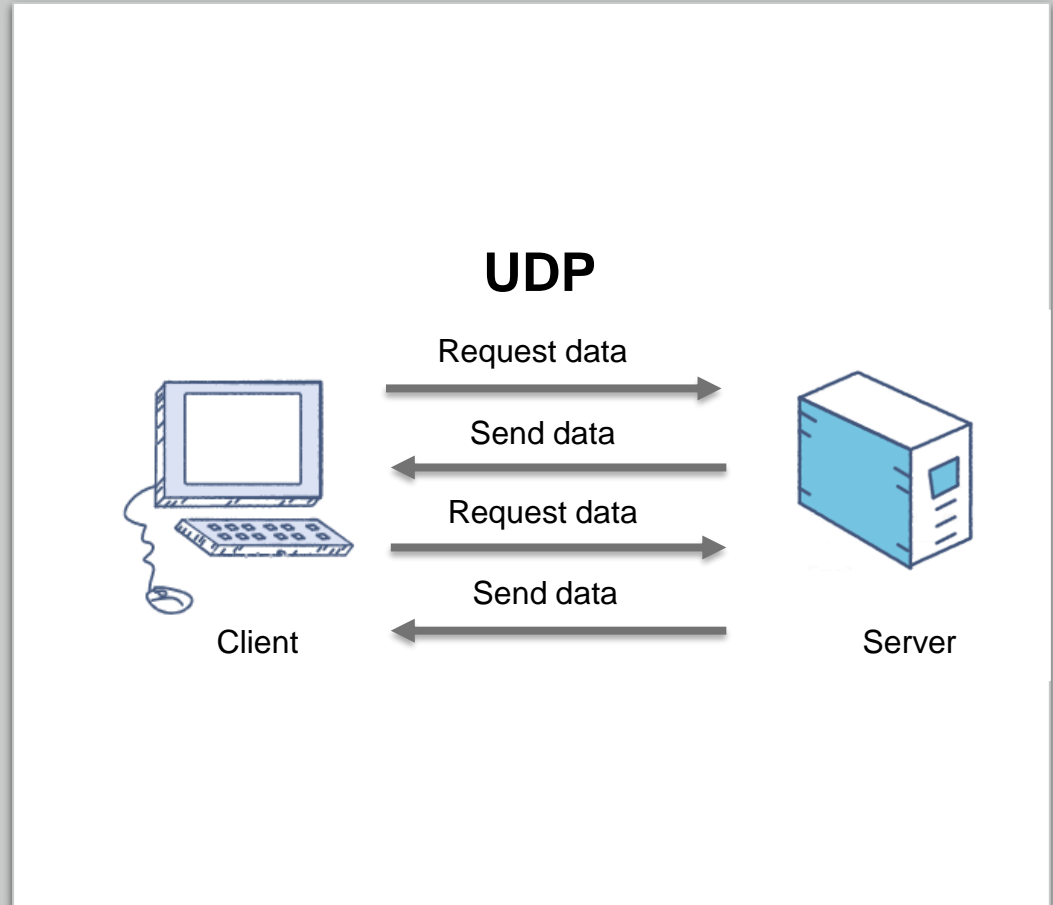




# UDP

## User Datagram Protocol (UDP) – connectionless, unreliable.

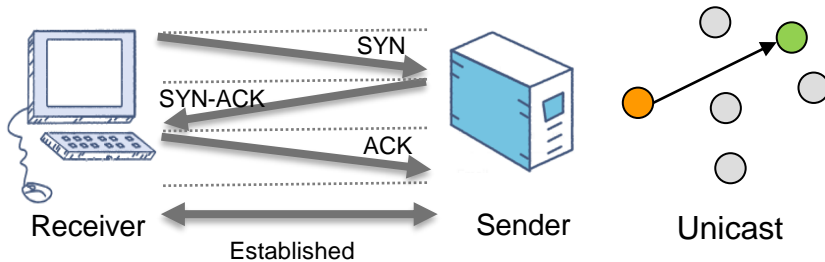
- Sends independent packets of data, called datagrams, from one computer to another, with no guarantees about arrival.
- Not connection based.
- Communication is not guaranteed.
- Datagram: a packet sent by UDP.
- The order of datagrams are not guaranteed.
- Example: VoIP (voice over IP), video streaming, online games.



# TCP vs UDP

## TCP

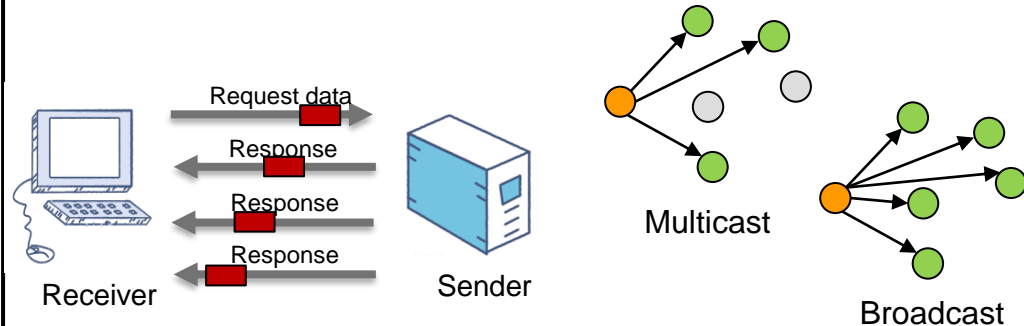
- Reliable delivery, all data is acknowledged.
- Source and destination are implied by the connection.
- Server and Client code look quite different.
- Lower transmission speed.
- Point to point transmission: unicast



3-way handshake

## UDP

- Unreliable best-effort delivery without acknowledgements.
- Must specify destination for each datagram.
- Server and Client code vary mostly in who sends first.
- Very high transmission speed.
- Supports multicast and broadcast

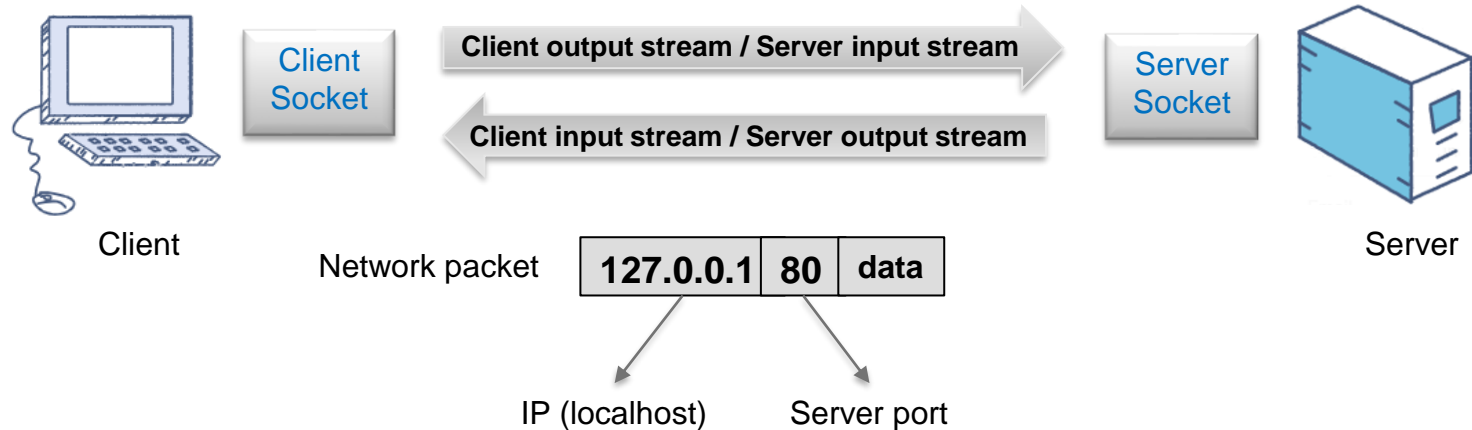



# Sockets

- What is a socket?

A socket is a one end-point of a two-way communication link between two programs running on the network.

A socket enables a Java program running on a client machine to open a connection with a web server.

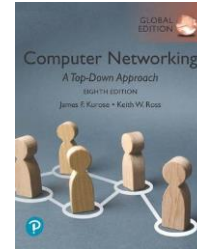


- ❑ A socket is a software abstraction to represent the “terminals” of a connection.
  - ❑ To an application, a socket is a file descriptor that lets the application read/write from/to the network.
  - ❑ Clients and servers communicate with each other by reading from and writing to socket descriptors.
  
  - ❑ There are three Socket classes in java:
    - ServerSocket: for server
      - On connection returns a new Socket
    - Socket: for client
      - Has `getInputStream()` and `getOutputStream()` functions.
    - DatagramSocket()
      - Used for UDP connections
- 
- Used for TCP connections

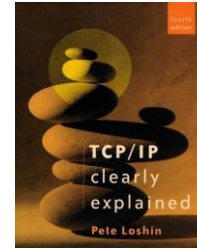
Note: A socket is a software element and it does not imply hardware.

# Additional reading

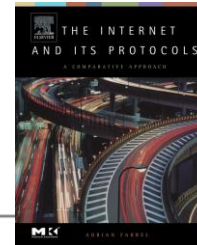
- Computer Networking: A Top-Down Approach (7<sup>th</sup> edition), by Kurose & Ross  
**Chapter #1**
- TCP/IP Clearly Explained, by Pete Loshin  
**Chapters #1 - #7, #15 - #17**
- The Internet and its Protocols, by Adrian Farrel  
**Chapters #2, #3, #5, #6**



[E-book](#)



[E-book](#)



[E-book](#)

