Operating System Concepts

Lecture 9: Distributed Systems

Omid Ardakanian
oardakan@ualberta.ca
University of Alberta

Today's class

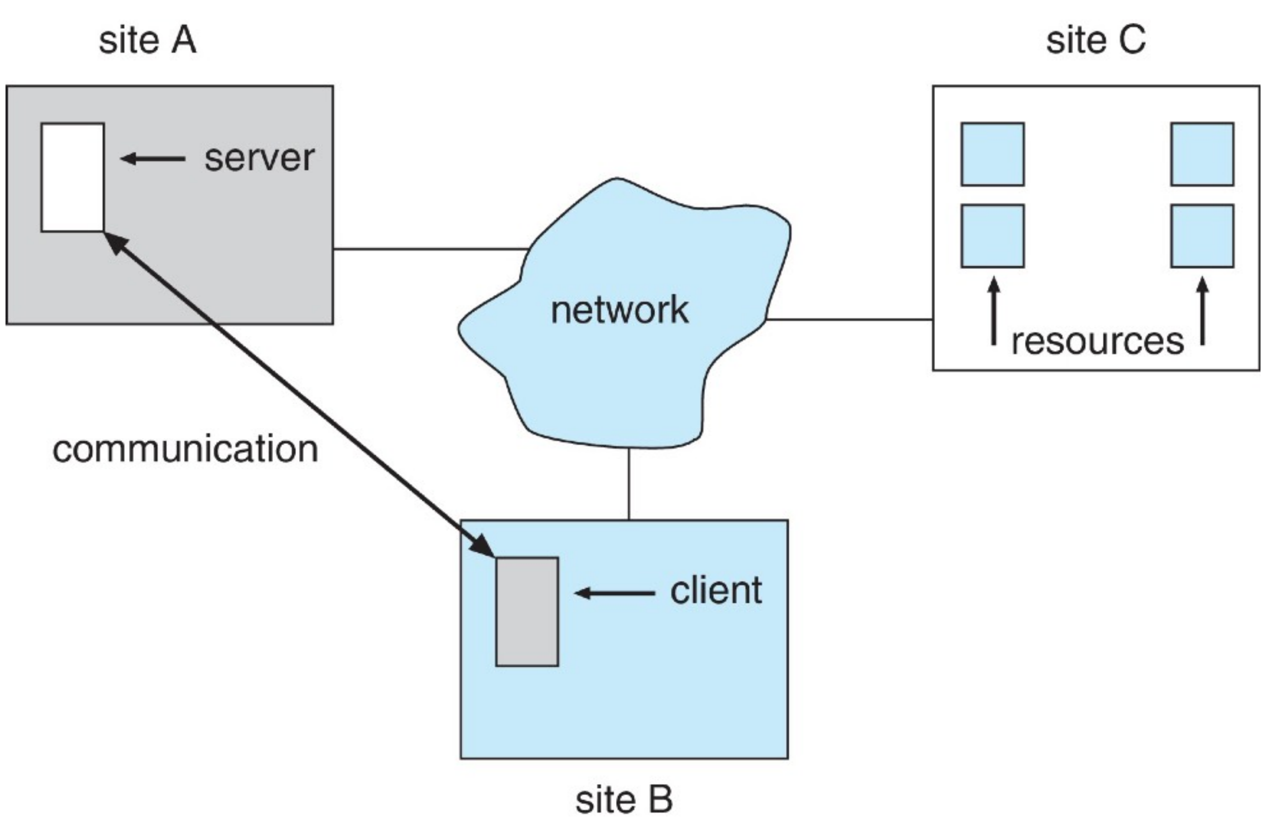
- Distributed systems
  - Motivation
  - Design issues
- Communication basics
  - TCP/IP and UDP/IP networking

Distributed systems

- A set of physically separate, loosely coupled nodes connected by a communication network (e.g. high-speed bus or the Internet)
  - each node has an independent OS along with its own resources
  - nodes are variously called processors, machines, computers, and hosts
- Today, almost all systems are distributed in some way
  - complex services are built from a large collection of machines that cooperate to provide these services; networks hook them together
- Communication over a network occurs through **message passing**
  - communication is inherently **unreliable**, i.e. messages sometimes do not reach their destination correctly due to packet loss (dropped by routers) or corruption, and node or link failure

Configurations

- Nodes may exist in a client-server, peer-to-peer, or hybrid configuration
  - in the client-server model, the server has some resource that the client wants to use
  - in the peer-to-peer model, each node may act as both client and server



Why are distributed systems so popular?

- Resource sharing
  - resources need not be replicated at each processor, e.g., shared files
  - expensive and/or scarce resources can be shared, e.g., GPUs
  - user can use specialized and licensed software on another machine
- Computational speedup
  - distribute tasks among various sites to run **concurrently**
    - they do not compete with each other for a single CPU core
  - load balancing would help
- Reliability and availability
  - resource replication results in fault tolerance (no SPOF)
    - machine failure does not imply system failure, usually performance degrades but system remains operational
  - detecting and recovering from site failure would be necessary

Design goals in distributed systems

- Robustness: the system should withstand failures
  - including failure of a link, failure of a site, and loss of messages
  - a fault-tolerant system can tolerate a certain level of failure; the degree of fault tolerance depends on the system design and the specific fault
  - detecting hardware failure is difficult (can use a heartbeat protocol)
- Transparency: the system should appear as a conventional, centralized system to users
  - user interface should not distinguish between local and remote resources
  - user mobility allows users to log into any machine and still see their environment

Design goals in distributed systems

- Scalability: the system should react gracefully to increased load (by accepting new resources)
  - data **compression** and **deduplication** can cut down on storage and network resources used
- Consistency: the cached copy of data must be consistent with the master copy
  - consistency checks must be performed periodically
  - nodes must keep track of the cached data to detect potential inconsistencies

Distributed OS

- Users are not aware of multiplicity of machines
  - accessing remote resources is similar to accessing local resources
  - data migration and process migration are handled by the distributed OS
    - data translation may be required (when they have different character-code representations, byte ordering, etc.)
- Data migration
  - transfer data by transferring the entire file or only those portions of the file that are necessary for completing the immediate task
- Computation migration
  - transfer computation rather than data across the system, for example via remote procedure calls (RPCs)
- Process migration
  - execute the entire process (computation and data), or parts of it, at different sites

Communication Basics

Definitions

- Network: one or more communication links allowing two computers to communicate
  - each computer has a network address
- Network interface: computer's interface to the network
  - each network interface card (NIC) has a unique hardware address
- Packet: network's basic transmission unit; a sequence of bits
- Protocol: a set of rules for communication that are agreed to by all parties

Network structure

- Local Area Network (LAN) covers a small geographical area (e.g., a building)
  - consists of multiple computers, peripherals (printers, storage arrays), and routers providing access to other networks
  - must be fast and reliable
- Ethernet and Wireless (WiFi) are the most common ways to build a LAN
  - Ethernet defined by IEEE 802.3 standard with speeds typically varying from 10Mbps to over 10Gbps
    - everyone taps into a single wire
    - everyone gets packets and discards them if it is not the target
  - WiFi defined by IEEE 802.11 standard with speeds typically varying from 11Mbps to over 400Mbps

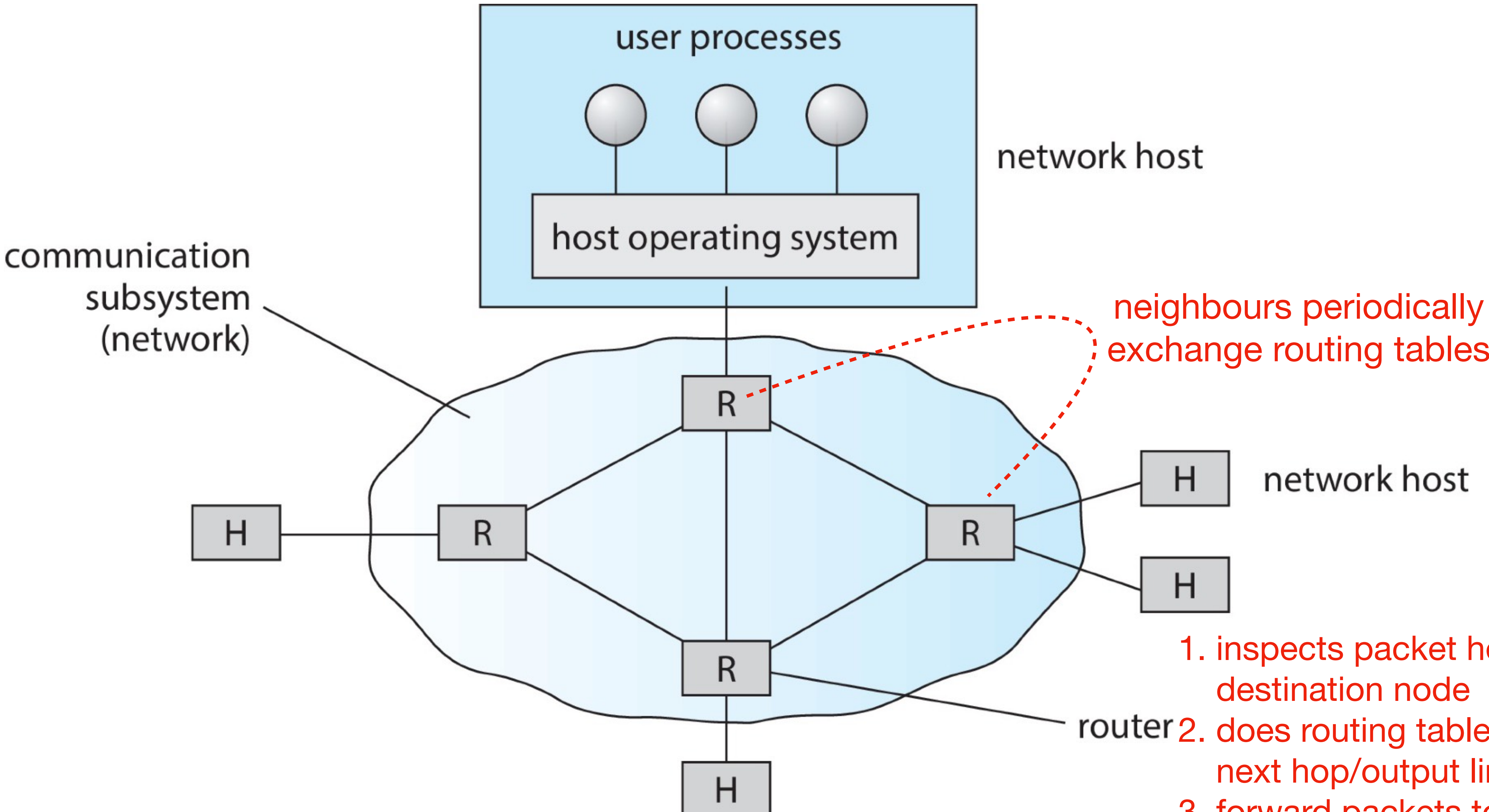
Network structure

- Wide Area Network links geographically separated sites (across the country or world)
  - WAN is slower and less reliable than LAN
  - Internet WAN enables hosts around the world to communicate
- Point-to-point connections via links
  - telephone lines, leased (dedicated data) lines, optical cables, microwave links, radio waves, and satellite channels
  - speed varies
    - many backbone providers have speeds at 40-100Gbps
    - local Internet Service Providers (ISPs) may be slower
- WANs and LANs interconnect

Principles of network communication

- Data sent through the network is divided into **packets**
- Computers at switching points control the packet flow
  - analogy: cars/roads/police -> packets/links/computers
- Resource sharing causes contention
  - just like traffic jams
- The destination computer is interrupted when a packet arrives

How is a packet delivered to destination if hosts are located on separate networks?



1. inspects packet header to identify destination node
2. does routing table lookup to find next hop/output link
3. forward packets to output link

WAN is implemented using routers to direct traffic from one network to another

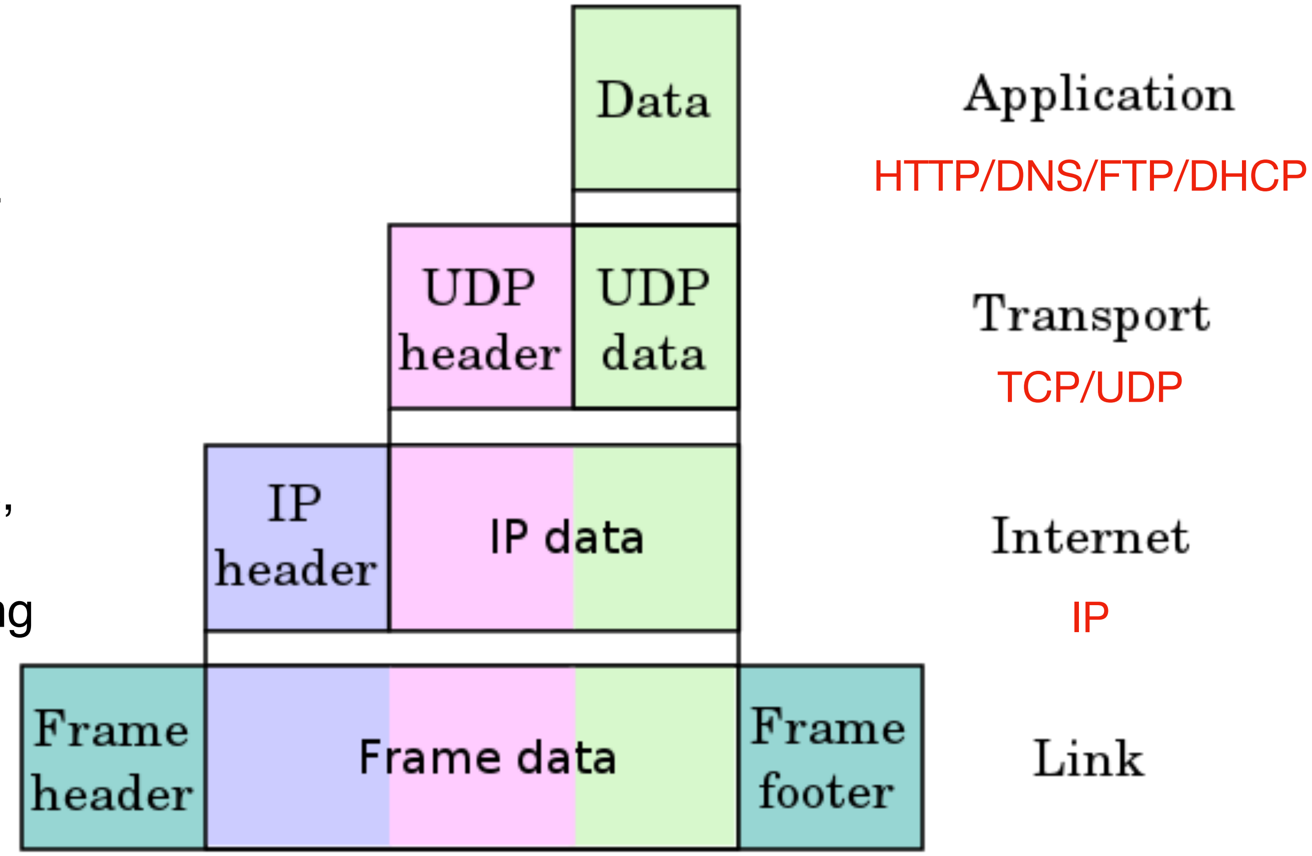
How to identify the destination system/process?

- A process on a remote system is identified by `<host-name, identifier>` pair
- Each process in a given system has a unique identifier (PID)
- Each computer system in the network has a unique name
  - Domain Name System (DNS): a global distributed database system for resolving hostname-IP address mappings
    - 32-bit (IPv4) address: e.g., 129.128.5.180
    - human readable names: e.g., gpu.srv.ualberta.ca

TCP/IP protocol stack (Internet protocol suite)

Enables end-to-end communications using functions organized into four abstraction layers

- Application layer
  - provides process-to-process data exchange for applications
- Transport layer
  - responsible for message transfer between hosts, including partitioning messages into TCP/UDP packets, maintaining packet order, and controlling flow
- Internet layer
  - responsible for routing IP packets through the network, and encoding/decoding addresses
- Link layer
  - handles the frames, or fixed-length parts of packets, including transmission over over a physical medium, and any error detection and recovery that occurred in the physical layer



routers only have the last two layers

Media Access Control (MAC) address

How does a packet move from sender (host or router) to receiver on the same LAN?

- every Ethernet/WiFi device has a unique medium access control (MAC) address
- if a system wants to send data to another system, it needs to perform the IP to MAC address mapping
  - using **address resolution protocol (ARP)**
  - run `arp -a` to see the content of your arp table

Internet Protocol (IP) address

- Every host has a name and an associated **IP address**
  - 32-bit (IPv4) address: approximately 4.3 billion addresses
  - 128-bit (IPv6) address: approximately 3.4×10^{38} addresses
  - a special address is reserved for local host: 127.0.0.1
- The sending system checks routing tables and locates a router to send packet to
- Each router uses the **network part** of host-id to determine where to transfer packet
- The destination system receives the packet
  - it may be a complete message, or it may need to be reassembled into a larger message spanning multiple packets

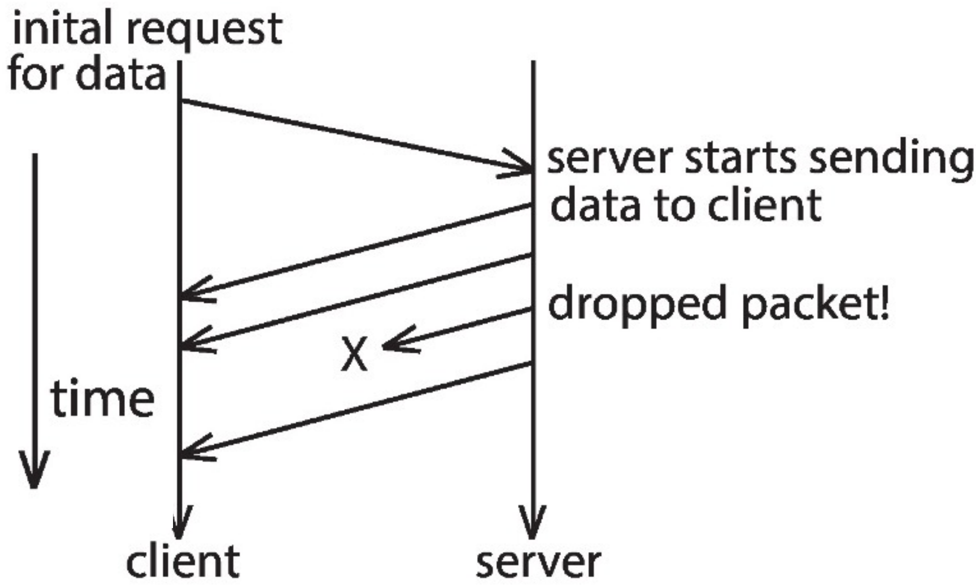
Transport layer address

- Once a host with a specific IP address receives a packet, it must somehow pass it to the correct waiting process
- Transport protocols, TCP and UDP, identify receiving and sending processes through the use of a **port** number (16 bits)
  - allows a host with a single IP address to have multiple processes sending and receiving packets
- Transport protocol can be simple or add reliability to network packet stream

User Datagram Protocol (UDP)

- UDP packets are also called **datagrams**
  - **fixed-size messages** up to some maximum size
- UDP is unreliable
  - packets may be lost or received out-of-order
  - however, packet corruption is still detected using a checksum
- UDP is connectionless
  - no connection setup at the beginning of transmission is necessary to set up state
  - also no connection tear-down at the end of transmission

Why to use unreliable communication?
many applications simply want to send data
to a destination and do not worry about packet loss



Transmission Control Protocol (TCP)

- TCP is both reliable and connection-oriented
- In addition to the port number, TCP provides abstraction to allow in-order, uninterrupted byte-stream across an unreliable network
  - whenever host sends packet, the receiver must send an **acknowledgement** packet (ACK)
  - if ACK is not received before a timer expires, the sender will **timeout** and **retransmit** the packet
    - requires keeping a copy of messages sent and not yet acknowledged
    - timeout will happen if either sender's packet or receiver's acknowledgment is dropped
    - what if the timeout is too small or too large?
  - **sequence counter** in TCP header allows the receiver to put packets in order and notice duplicate packets (`ack` was lost)
- Connections are initiated with series of control packets
  - three-way handshake (SYN, SYN+ACK, ACK)
- Connections also closed with series of control packets

TCP data transfer

- Receiver can send a cumulative ACK to acknowledge series of packets
  - server can also send multiple packets before waiting for ACKs
  - taking advantage of network throughput
- Flow of packets regulated through flow control and congestion control in TCP
  - **flow control** – prevents sender from overrunning capacity of receiver
  - **congestion control** – infers network congestion to slow down or speed up packet sending rate