Interprocess Communication

From **Coulouris, Dollimore, Kindberg and Blair**

Distributed Systems:

Concepts and Design

Edition 5, © Addison-Wesley 2012

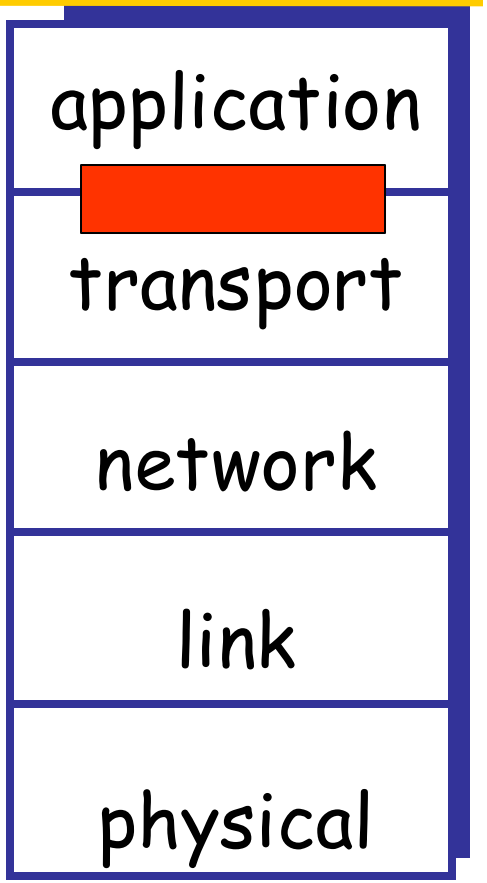
The characteristics of interprocess communication

- Messages are sent to (Internet address, local port) pairs.
- A port has exactly one receiver but can have many senders
- Processes may use multiple ports to receive messages

  - Validity: A point-to-point message service is reliable if messages are guaranteed to be delivered despite packet drop
  - Integrity: Messages must arrive uncorrupted and without duplication
  - Ordering: messages be delivered in sender order

Application and middleware layers use the services provided by the network and transport layers through socket API.

Sockets



From Computer Networking by Kurose and Ross.

Goal: learn how to build client/server application that communicate using sockets

Socket API

- introduced in BSD4.1 UNIX, 1981
- explicitly created, used, released by apps
- client/server paradigm
- two types of transport service via socket API:
- unreliable datagram
- reliable, byte stream-oriented

socket

a *host-local*, *application-created*, *OS-controlled* interface (a "door") into which application process can both send and receive messages to/from another application process

Processes-to-process communication

Process: program running within a host.

within same host, two processes communicate using inter-process communication (shared memory defined by OS).

processes in different hosts communicate by exchanging messages using transport layer

Client process: process that initiates communication

Server process: process that waits to be contacted

- Note: applications with P2P architectures have client processes & server processes

Addressing processes

to receive messages, process must have *identifier*

host device has unique 32-bit IP address

Q: does IP address of host on which process runs suffice for identifying the process?

A: No, *many* processes can be running on the same host

identifier includes both IP address and port number associated with the process

What is a port number?

16 bits integer used by transport layer to identify end points (processes) on a host

well-known ports: 1 – 1023 Telnet 23; FTP 21; HTTP 80

registered ports: 1024 – 49151

dynamic or private ports: 49152 - 65535

To communicate, client must know the server's IP address, and port number.
How will the server know the client's IP address and port number?

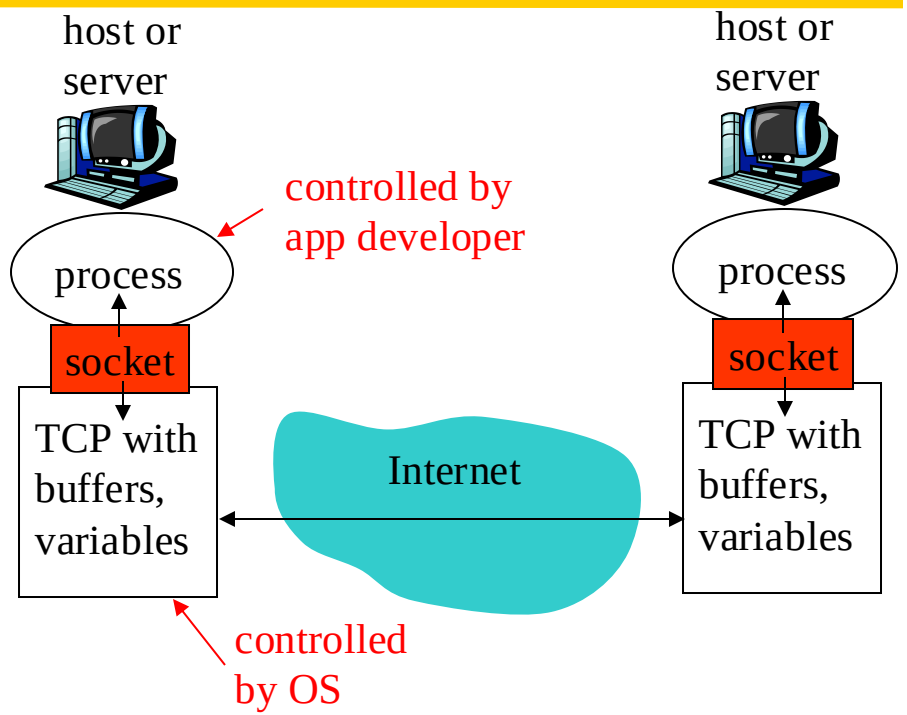
From Computer Networking by Kurose and Ross.

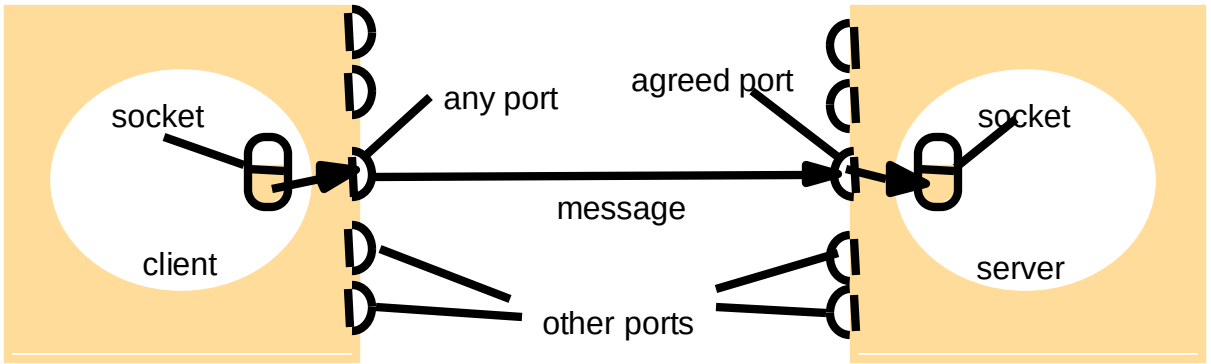
Sockets

API, an interface, gate, door between a process and transport layer

A socket must be bound to a local port

Is (IP addr, port) enough to identify a socket?



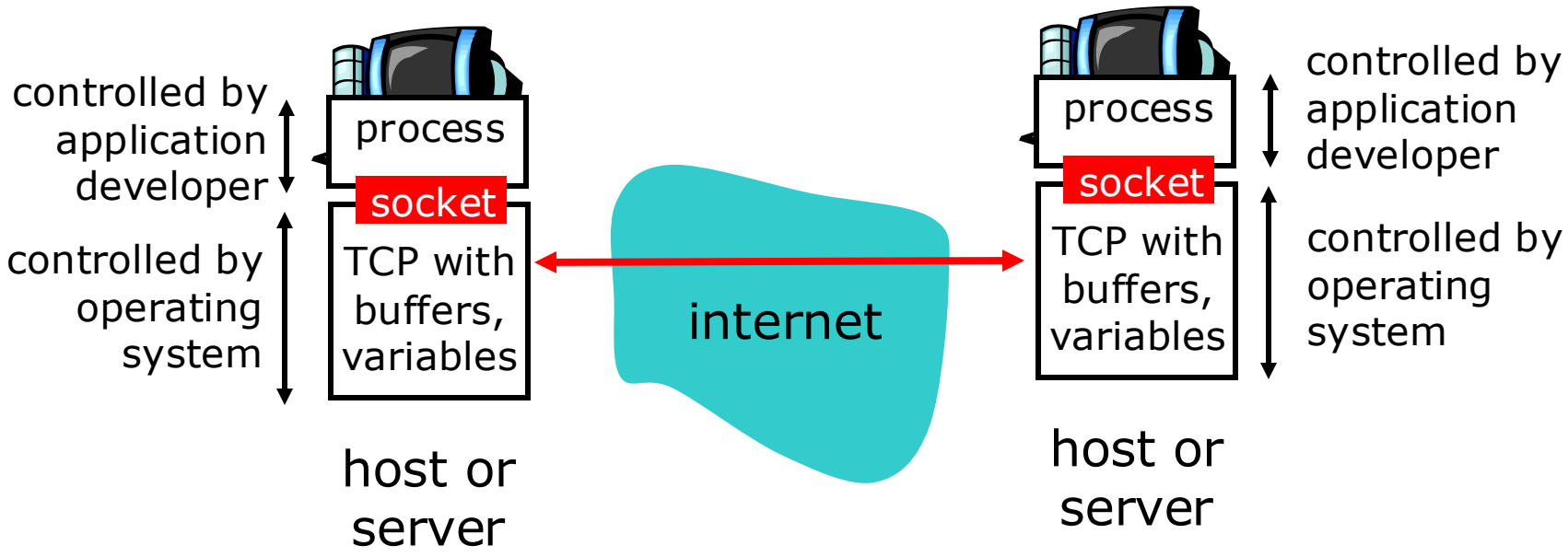


From Computer Networking by Kurose and Ross.

Socket-programming using TCP

Socket: a door between application process and end-end-transport protocol (UCP or TCP)

TCP service: reliable transfer of **bytes** from one process to another



From Computer Networking by Kurose and Ross.

Socket programming *with TCP*

Client must contact server

- server process must first be running
- server must have created socket (door) that welcomes client's contact

Client contacts server by:

- creating client-local TCP socket
- specifying IP address, port number of server process
- When client creates socket: client TCP establishes connection to server TCP

When contacted by client, server TCP creates new socket for server process to communicate with client

- allows server to talk with multiple clients
- source port numbers used to distinguish clients

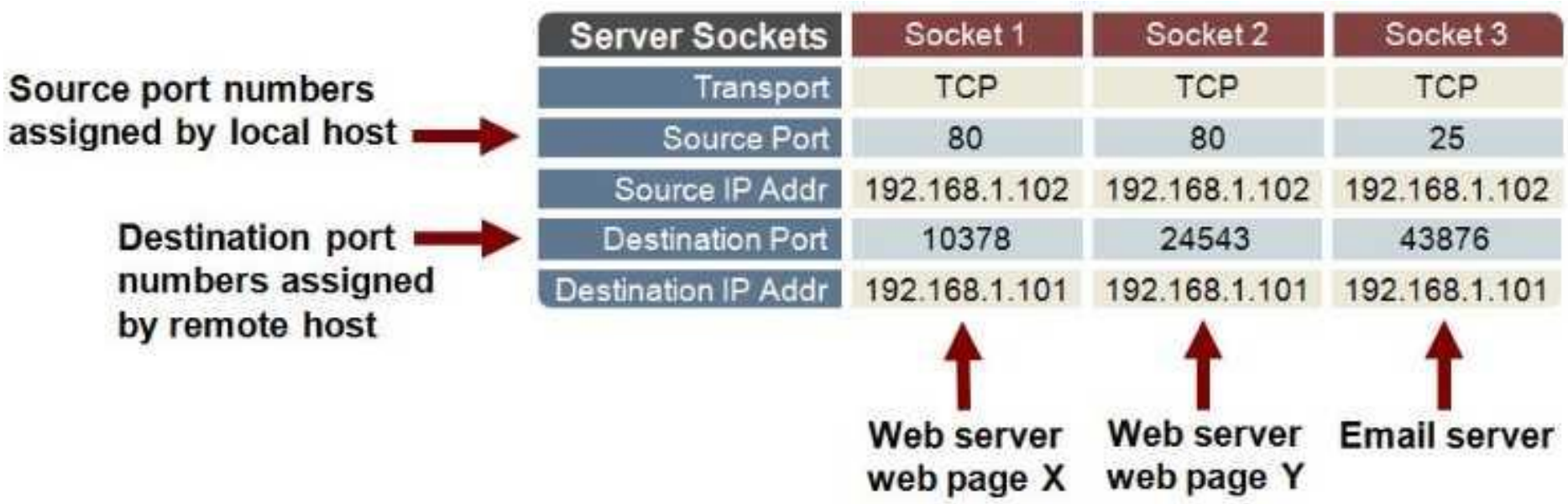
application viewpoint

TCP provides reliable, in-order transfer of bytes ("pipe") between client and server

From Computer Networking by Kurose and Ross.

Socket *with TCP*

Web server has two sockets opened: one for each web page it is serving. These sockets are differentiated by the destination port numbers.



Client/server socket interaction: TCP

Server (stand-by, waiting for requests)

1 Create socket
(Claim resources/ available phone)

2 Bind port
(Claim ID on this machine/
get a phone extension No.)

3 Listen/ accept
(Wait for connections/ Wait for phone call)

6 Send/ receive
(Communication/ Chat on phone)

7/8 Close socket
(End communication/ Hang up the phone)

Client (initiate the request)

4 Create socket
(Claim resources/ available phone)

-
(Don't care about public port/ phone #)

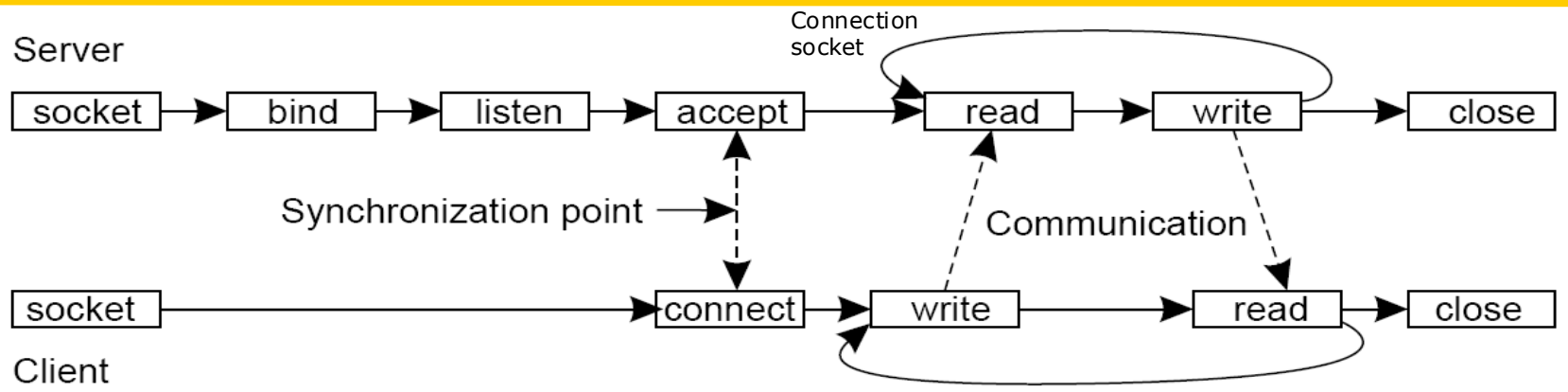
5 Connect
(connect to server/ call others)

6 Send/ receive
(Communication/ Chat on phone)

7/8 Close socket
(End communication/ Hang up the phone)

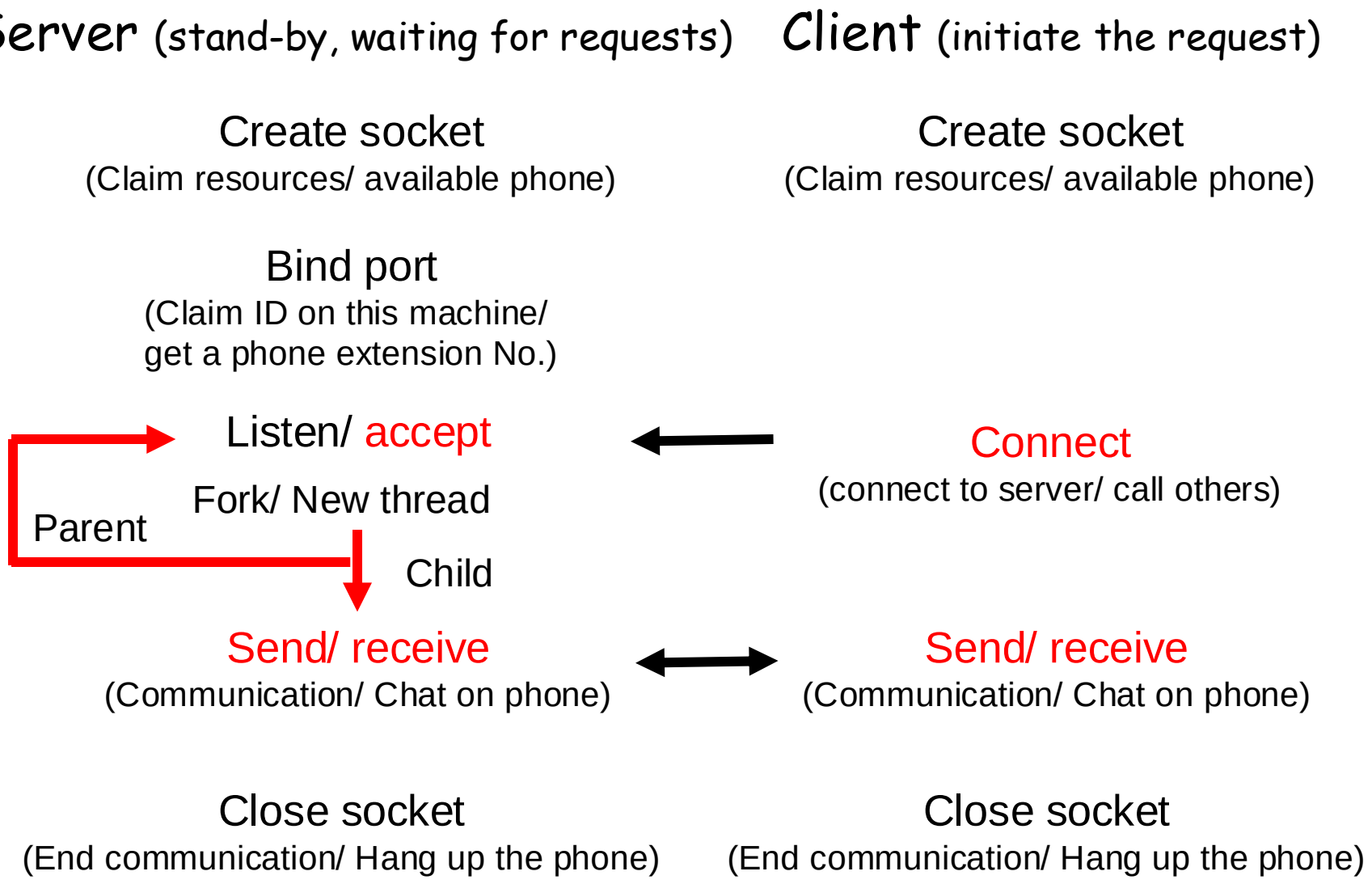
Red word: wait for the other side

TCP Socket Primitives

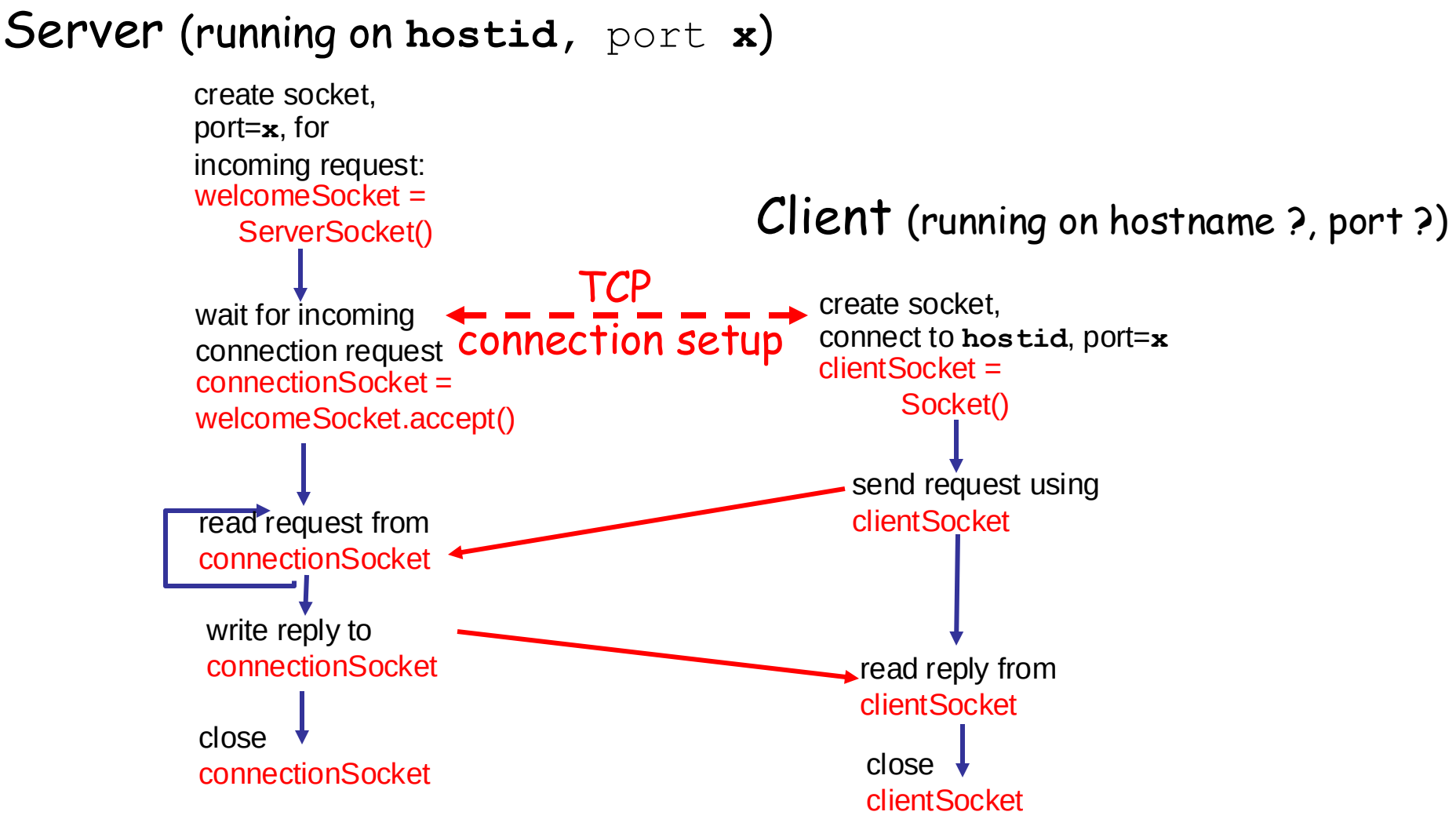


Primitive	**Function**
Socket	Create a new communication endpoint
Bind	Attach a local address to a socket
Listen	Announce willingness to accept connections
Accept	Block caller until a connection request arrives
Connect	Actively attempt to establish a connection
Send	Send some data over the connection
Recv	Receive some data over the connection
Close	Release the connection

Client/server socket interaction: TCP



Client/server socket interaction: TCP



Connection-oriented TCP

- TCP socket identified by 4-tuple:
  - source IP address
  - source port number
  - destion IP address
  - destion port number
- Receive host uses all four values to direct segment to appropriate socket

Socket programming *with UDP*

UDP: no “connection” between client and server

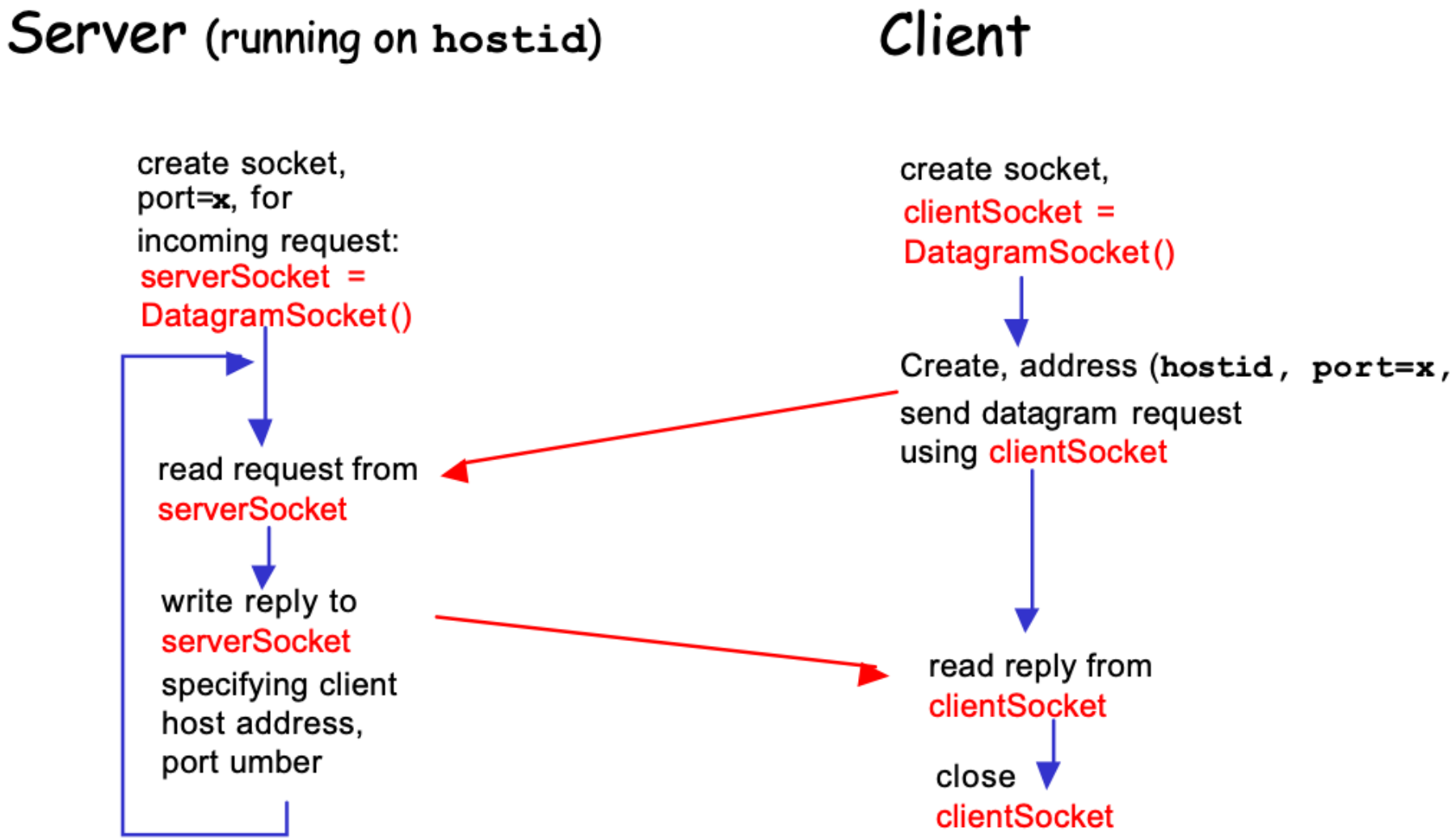
- no handshaking
- sender explicitly attaches IP address and port of destination
- server must extract IP address, port of sender from received datagram

UDP: transmitted data may be received out of order, or lost

application viewpoint

UDP provides unreliable transfer of groups of bytes (“datagrams”) between client and server

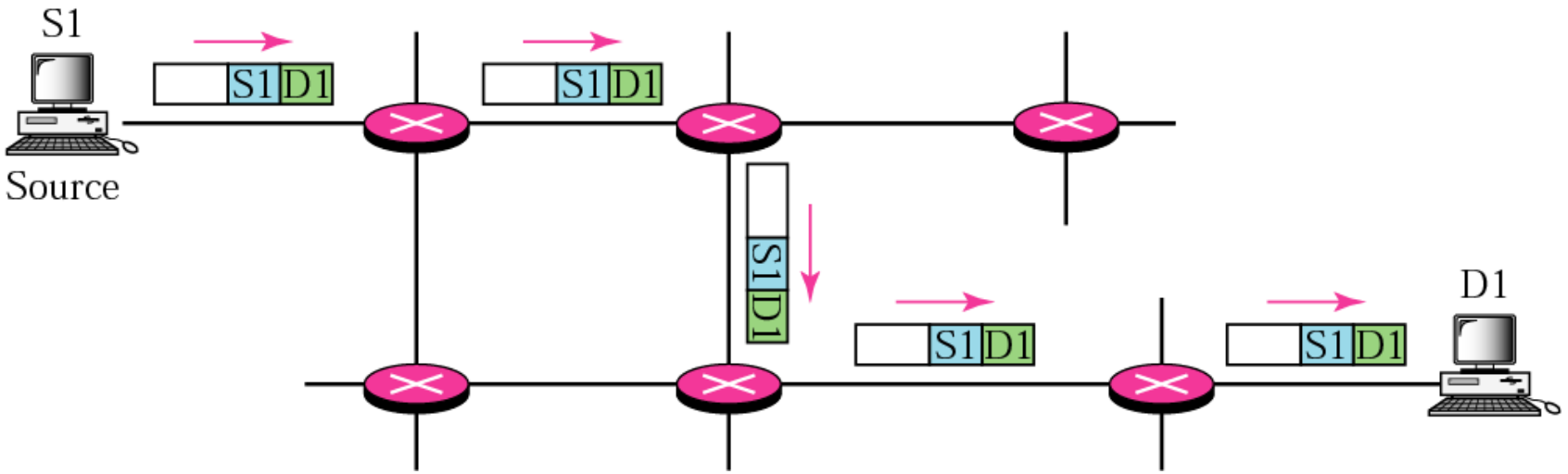
Socket programming *with UDP*



From Computer Networking by Kurose and Ross.

Multicast Communication
At Network Layer

Unicast Communication



In unicast routing, the router forwards the received packet through only one of its interfaces.

Multicast Communication

Broadcast – sends a single message from one process to **all** processes (hosts)

- Used for ARP in a LAN
- Hard and expensive in WAN

Multicast – sends a single message from one process to members of a group of processes (hosts)

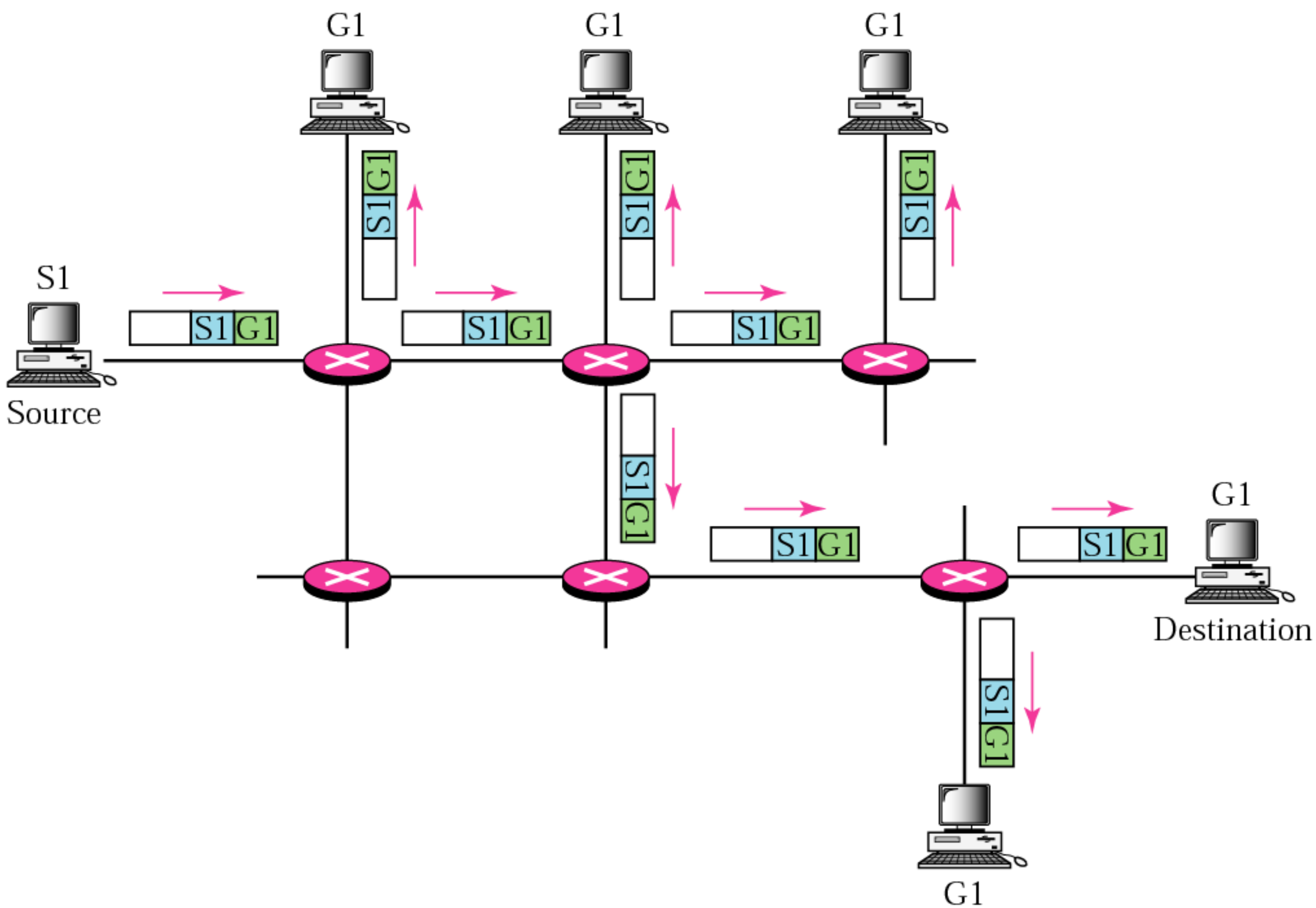
Who needs multicast?

Who should provide it?

Application, transport, network layer?

https://www.youtube.com/watch?v=C5pFaZtbISo

Multicast Communication



- In multicast routing, the router may forward the received packet through several of its interfaces

Who needs it?
Uses of Multicast and Its Effects

Fault tolerance based on replicated services

- Requests multicast to group of servers

Discovery in spontaneous networking

- Locate available discovery services

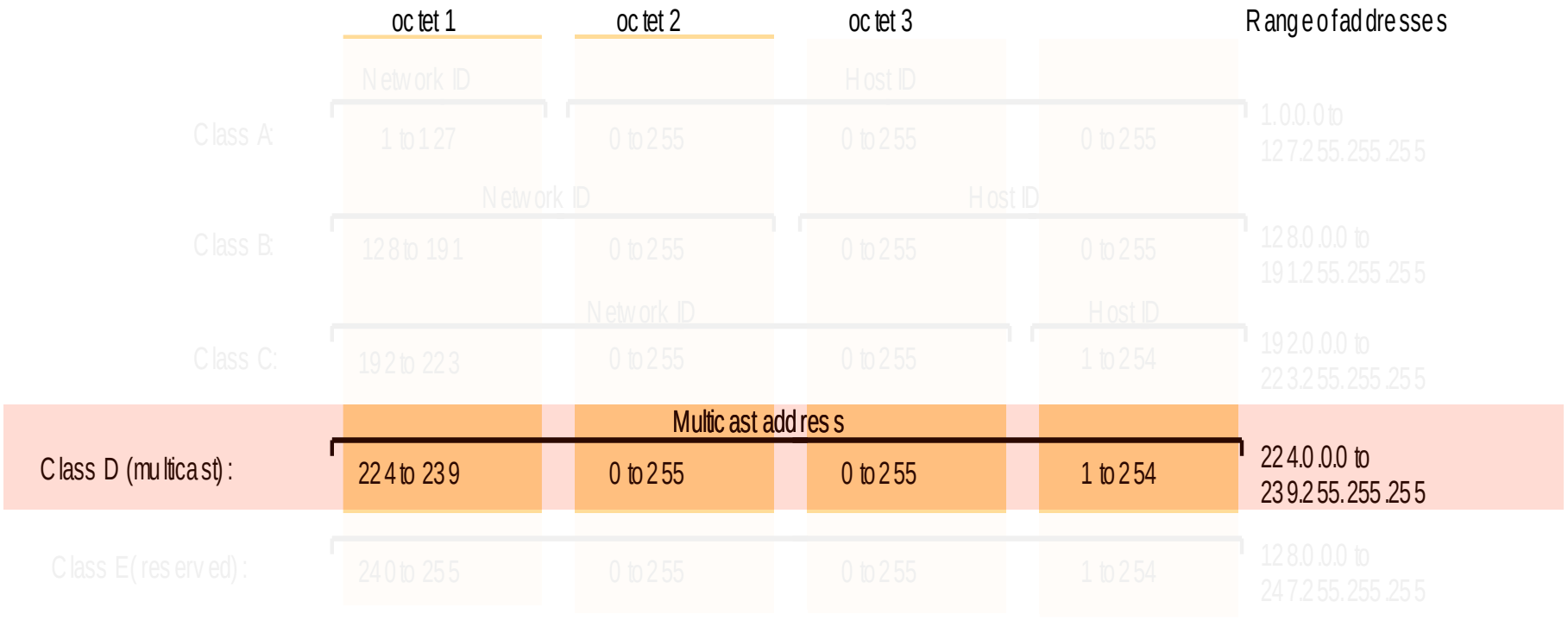
Performance from replicated data

- Multicast changes to all replicas

Propagation of event notifications in a distributed environment

- News group: news → group of interested users

Multicast IP address



224.0.0.0 to 224.0.0.255 (224.0.0.0/24) → **local** subnet multicast traffic

224.0.1.0 to 238.255.255.255 → **globally** scoped addresses

239.0.0.0 to 239.255.255.255 (239.0.0.0/8) → **administratively** scoped addresses, boundary

IP Multicast Process

Each multicast address → identify a group

Internet Group Membership Protocol (IGMP)

- Processes **register** a group with **local router** using **IGMP**

Router update its multicast routing table

Processes send message to a group

- Do not need to be a member

Router forward multicast messages

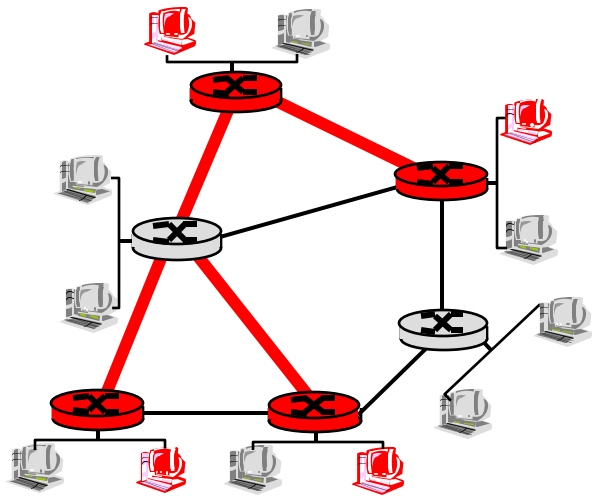
Multicast Routing Problem

Goal: find a tree (or trees) connecting routers having local multicast group members

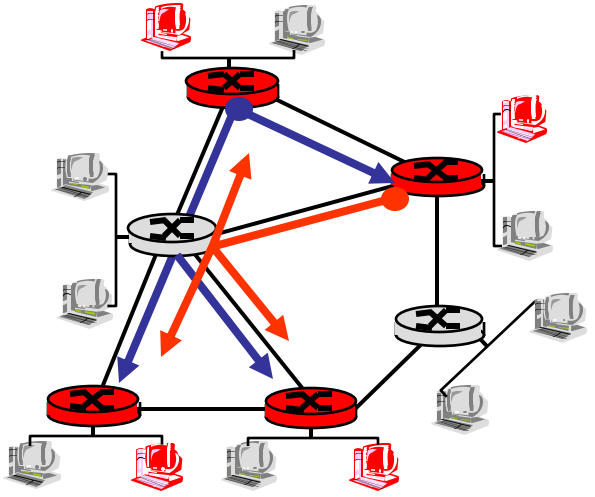
tree: not all paths between routers used

source-based: different tree from each sender to receivers

shared-tree: same tree used by all group members



Shared tree



Source-based trees

From Computer Networking by Kurose and Ross.