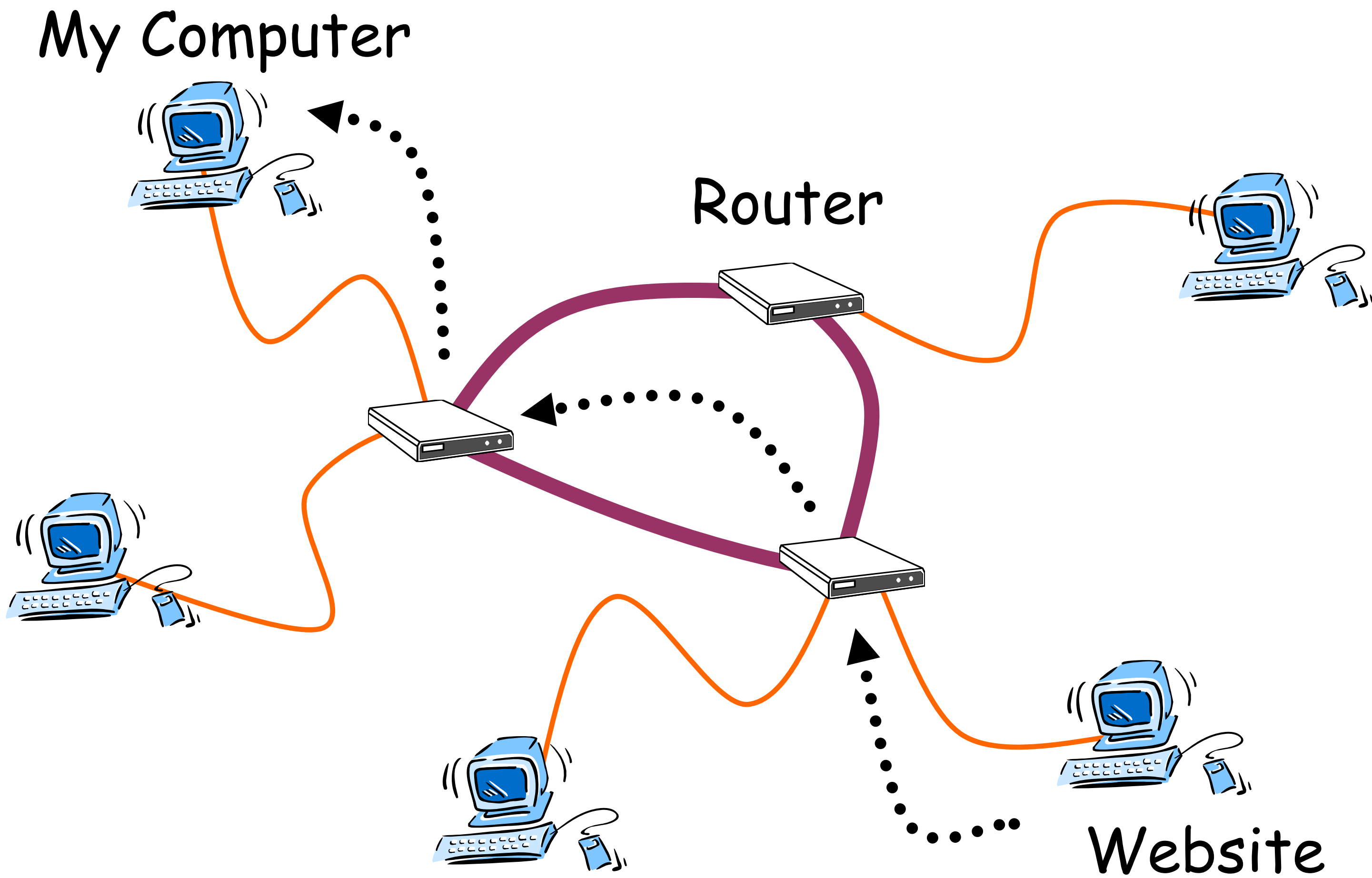
CS 40, Lecture 2:
TCP, IP, and the alphabet soup

Ramesh Johari

Outline

- IP, the Internet Protocol
- The end-to-end argument
- TCP, the Transmission Control Protocol
- UDP, the User Datagram Protocol
- Looking ahead

A simplified Internet

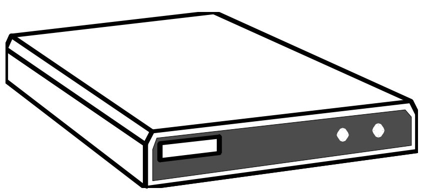


Routing and IP

How does a router know where to send packets next?

This is the function of the *Internet Protocol*.

Routing tables

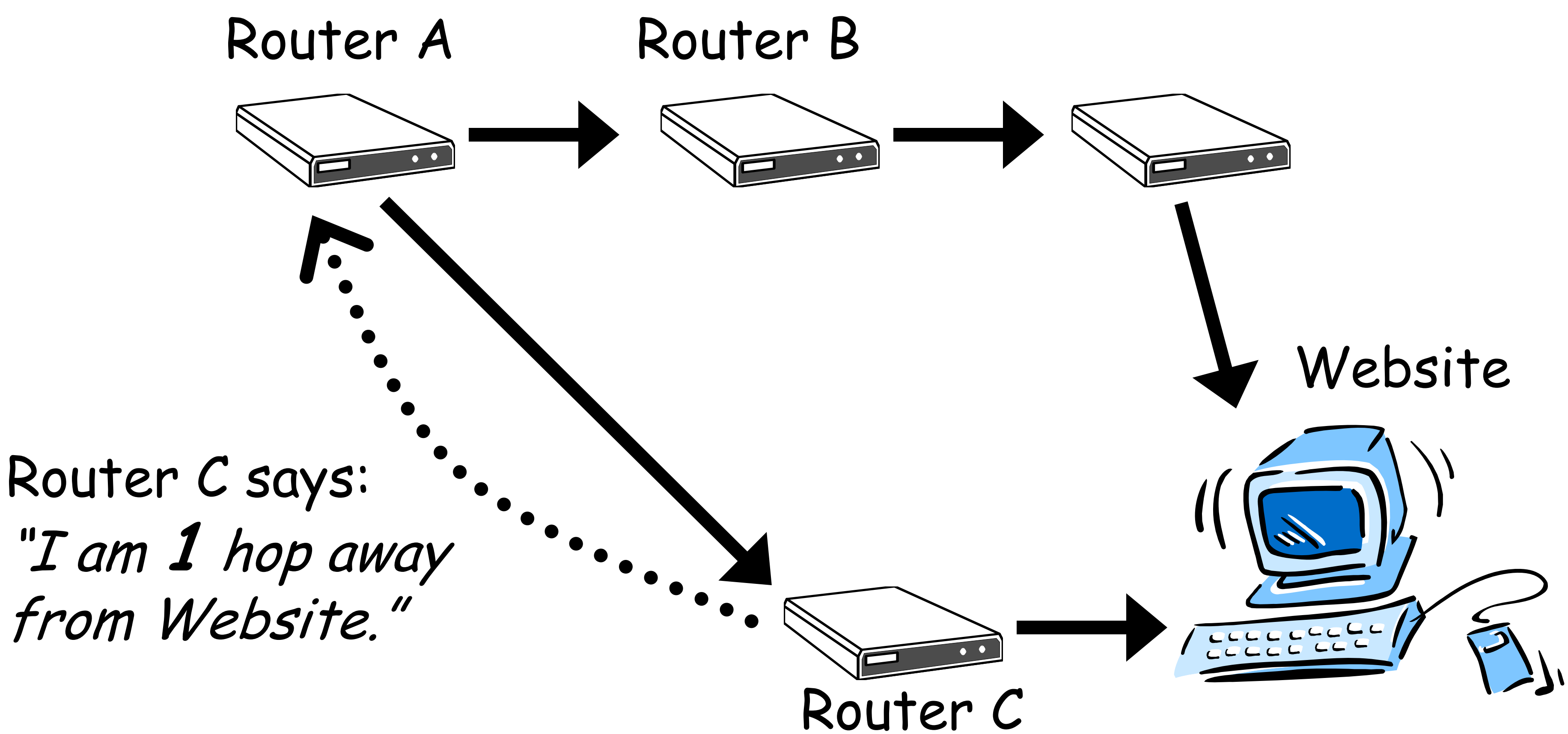


Router A

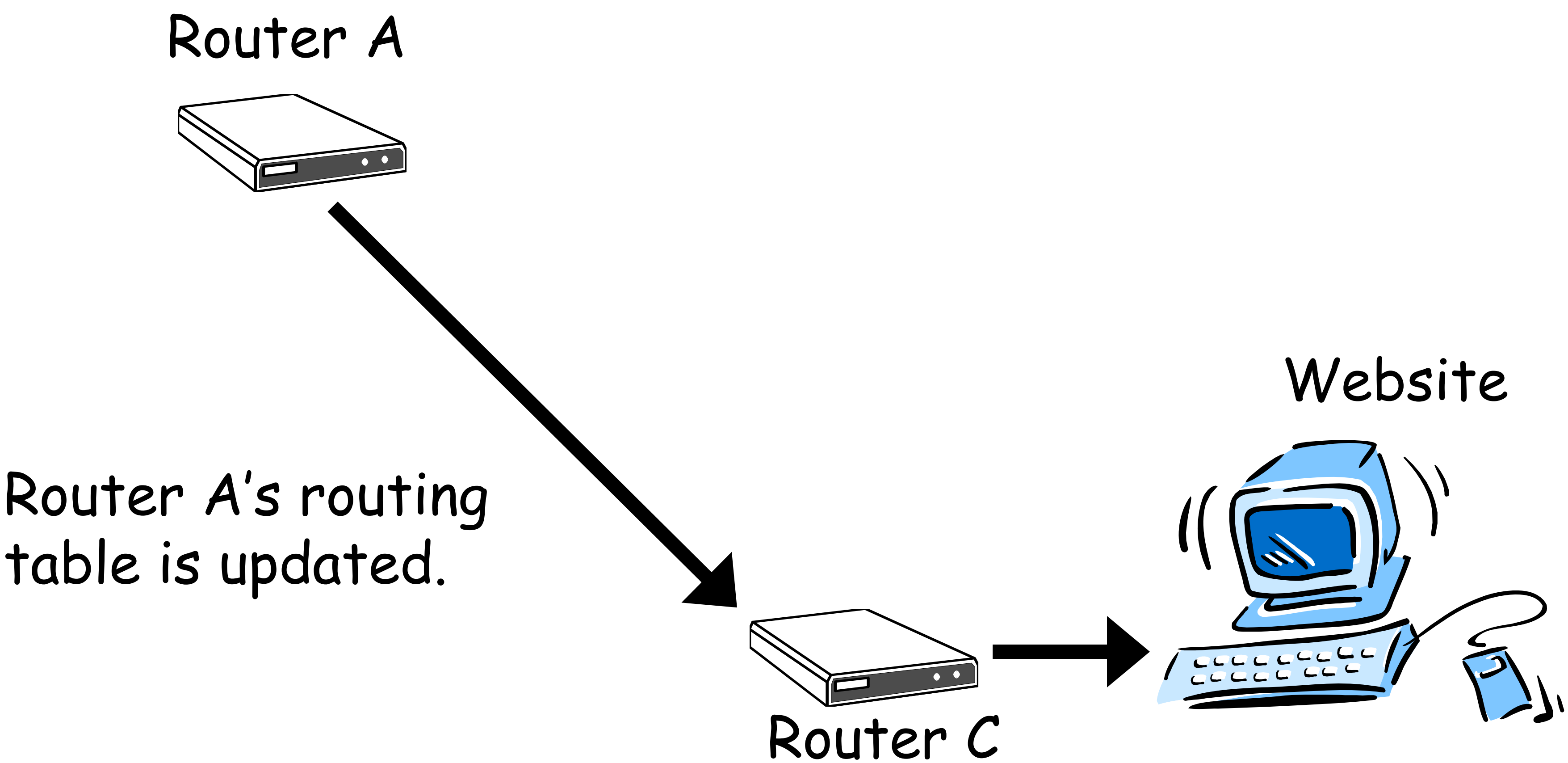
Packet's Destination	Next Router	Distance to Destination
Website	Router B	2 Hops

Routing Table

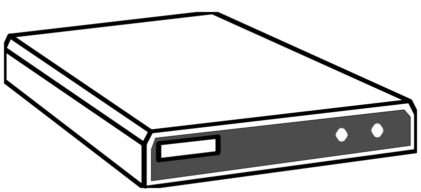
Routing table update



Routing table update



Routing tables



Router A

Packet's Destination	Next Router	Distance to Destination
Website	Router C	1 Hop

Routing Table

IP routing

Generally, IP routing is:

- *Shortest path routing:*
  A packet follows the "shortest" path available to the destination.
- *"Next hop" routing:*
  Each router only stores information about the *next hop* to the destination.

Complexities of routing

But in practice, many things are different:

- Inside their network, many providers use *traffic engineering* to shape where data travels and manage congestion
- Across providers, routes are chosen based on *business policy*, rather than shortest distance

(More on the second point next lecture.)

Packet routing

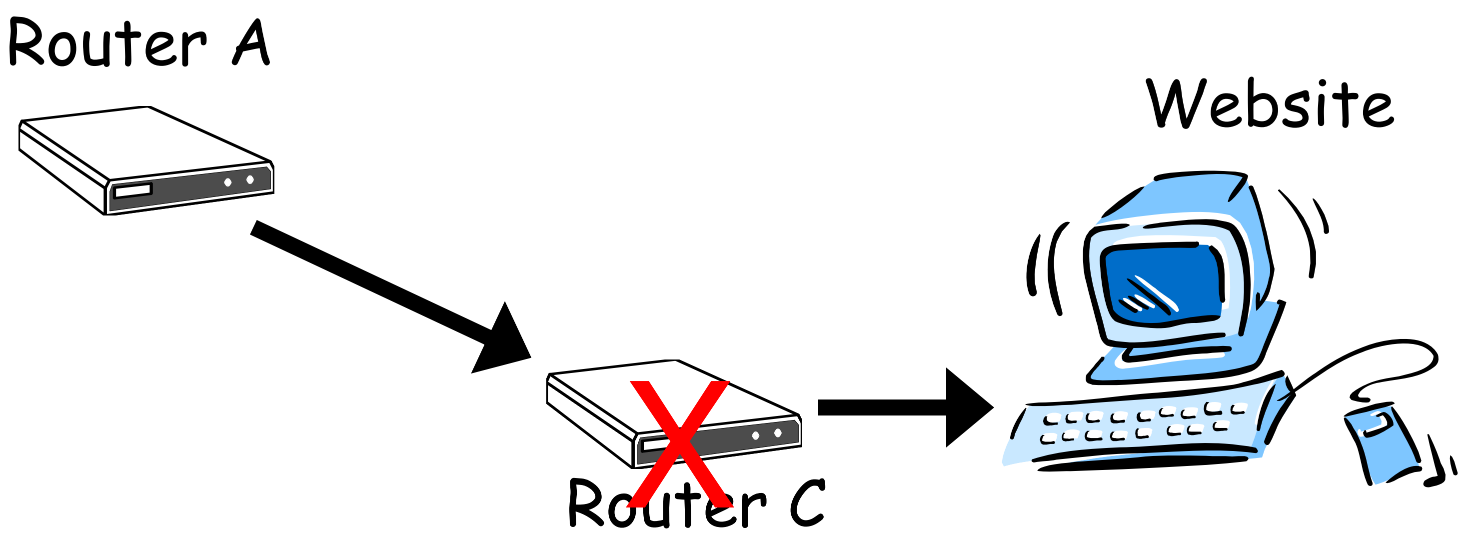
IP finds and disseminates the routes that are available.

"Packet routing":

- Suppose A wants to send a packet to B.
- A looks in its routing table, and forwards the packet to the "next hop" towards B.

Reliable delivery

- Notice that IP guarantees nothing about delivery.
- In our example: what if Router C "drops" a packet?



Distributed routing

A key point:

Routing is distributed!

No “central authority” knows the entire Internet;
routing is based on local decisions.

The end-to-end argument

- Broadly, functionality that can only be provided at the ends of the network should not be put in the middle

- In the original design:
  The network was not responsible for *reliable delivery* of packets

The end-to-end argument

Why?

- Suppose the path to the destination is $A \rightarrow B \rightarrow C \rightarrow D$.
- Suppose B is responsible for making sure that packets are retransmitted from A until they are safely received.
- This is useless if C has failed, so all packets from A are eventually lost anyway!

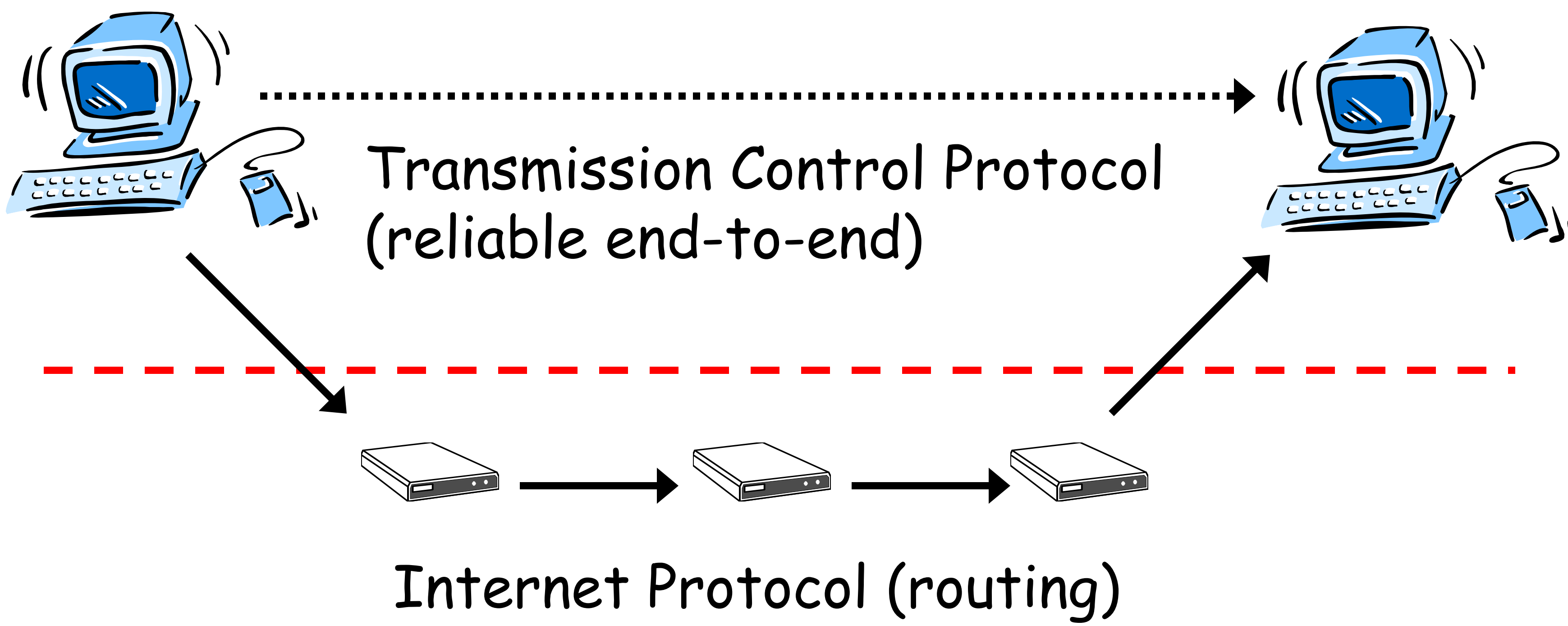
Transmission control protocol

The Internet uses TCP to provide *reliable end-to-end delivery of packets.*

Basic idea:

1. Sender sends packet.
2. Receiver sends *acknowledgment of receipt (ACK).*
3. If Sender does not get ACK, he resends packet.

TCP


Transmission Control Protocol
(reliable end-to-end)
Internet Protocol (routing)

TCP and “flow control”

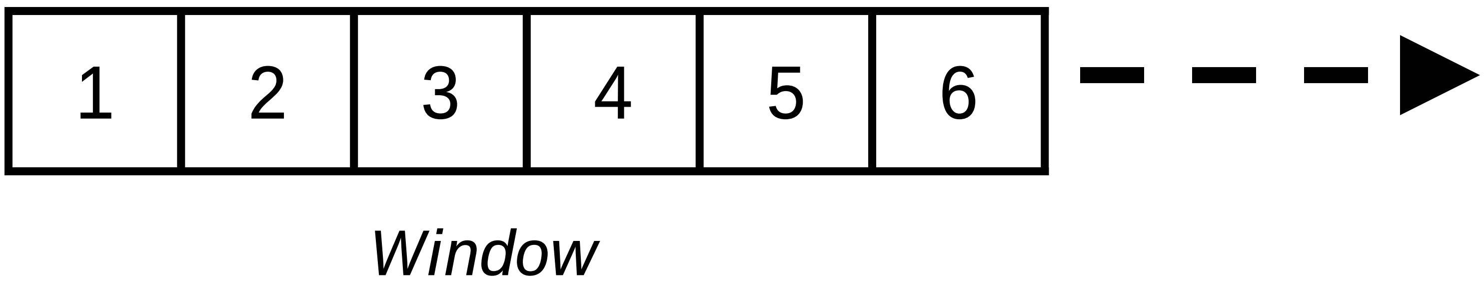
TCP also ensures that packets are not injected into the network “too fast” or “too slow.”

1	2	3	4	5	6

Window

TCP and "flow control"

First send out a "window" of packets.



TCP and “flow control”

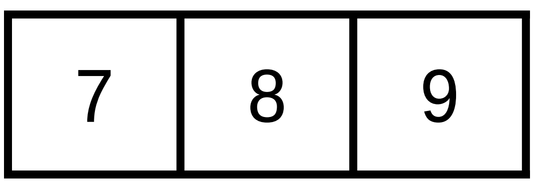
If ACKs for all are received successfully, expand window by one packet, and send again.

7	8	9	10	11	12	13

Window

TCP and “flow control”

But if even one packet is lost,
cut window in half before sending again.



Window

TCP and “flow control”

Thus the number of packets sent out after each “round trip time” (RTT) is equal to one window.

No lost packets: Window goes up 1 per RTT

Lost packet(s): Window halved per RTT

TCP and “flow control”

This mechanism serves to discover and utilize available capacity along the path to the destination.

TCP: One size fits all?

TCP is good when:

- All packets must arrive
- Delay does not matter

Example: File download.

As long as all packets arrive,
the file will be received intact.

TCP: One size fits all?

TCP is used for most common Internet tasks:

- E-mail
- Web browsing (a single page can open *many* TCP connections!)
- File downloads

TCP: One size fits all?

But what about a telephone call (e.g., VoIP, Voice over IP)?

- Even if some packets are lost, the caller can still be "heard".
- Delay is unacceptable!

UDP

UDP: User Datagram Protocol

No guarantees whatsoever;

packets are injected into the network,

and neither receipt nor sequencing are guaranteed.

UDP

But:

UDP also has the "feature" that it *never* reduces the sending rate.

This has made it a favorite protocol for "real-time" applications, such as VoIP.

TCP vs. UDP

Suppose a TCP and a UDP stream share the same link.

As the UDP stream increases its sending rate, what happens?

Why bother using TCP?

Quality of service

This is our first example of the importance of *quality-of-service*:

Some connections are *loss* sensitive.
Some connections are *delay* sensitive.

Question for discussion

You are SBC/AT&T.

You sell Internet access (DSL).

You also sell telephone service.

What happens to you when your customer buys a VoIP service from a 3rd party?