Now, it's your turn

...to design a Internet protocol

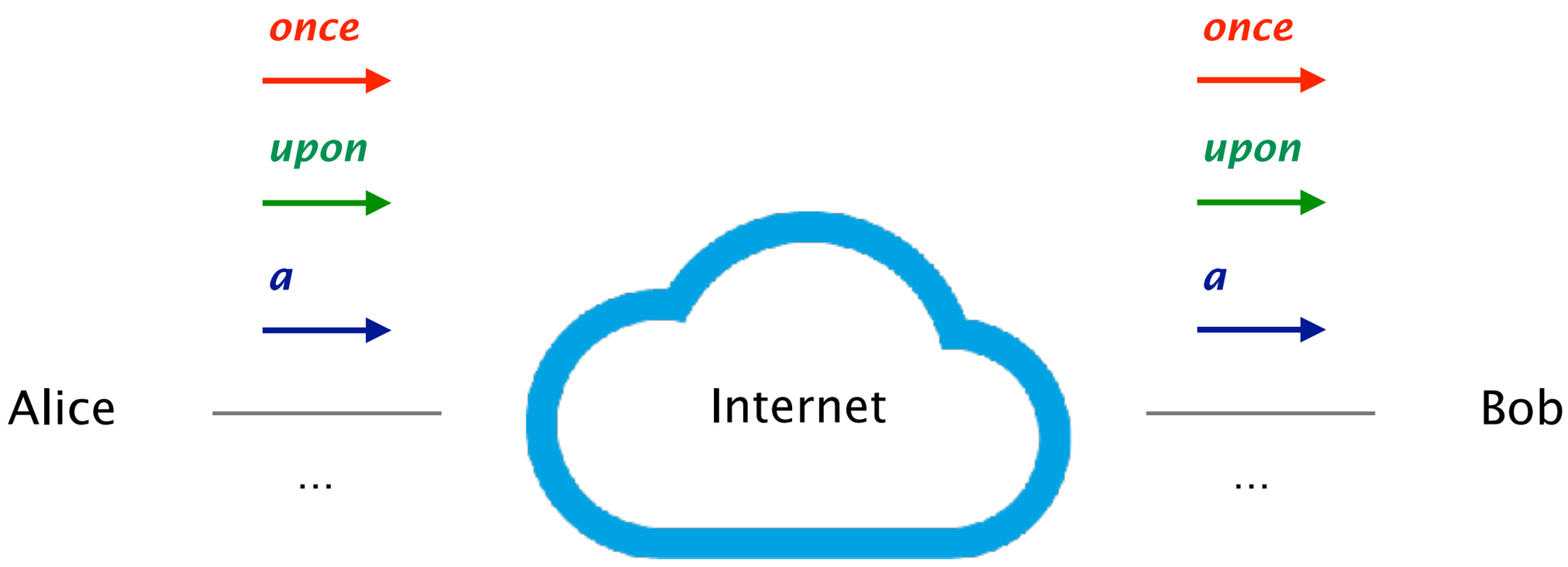
instructions given in class

I'm asking you to develop a

reliable transport protocol (sitting at L4)

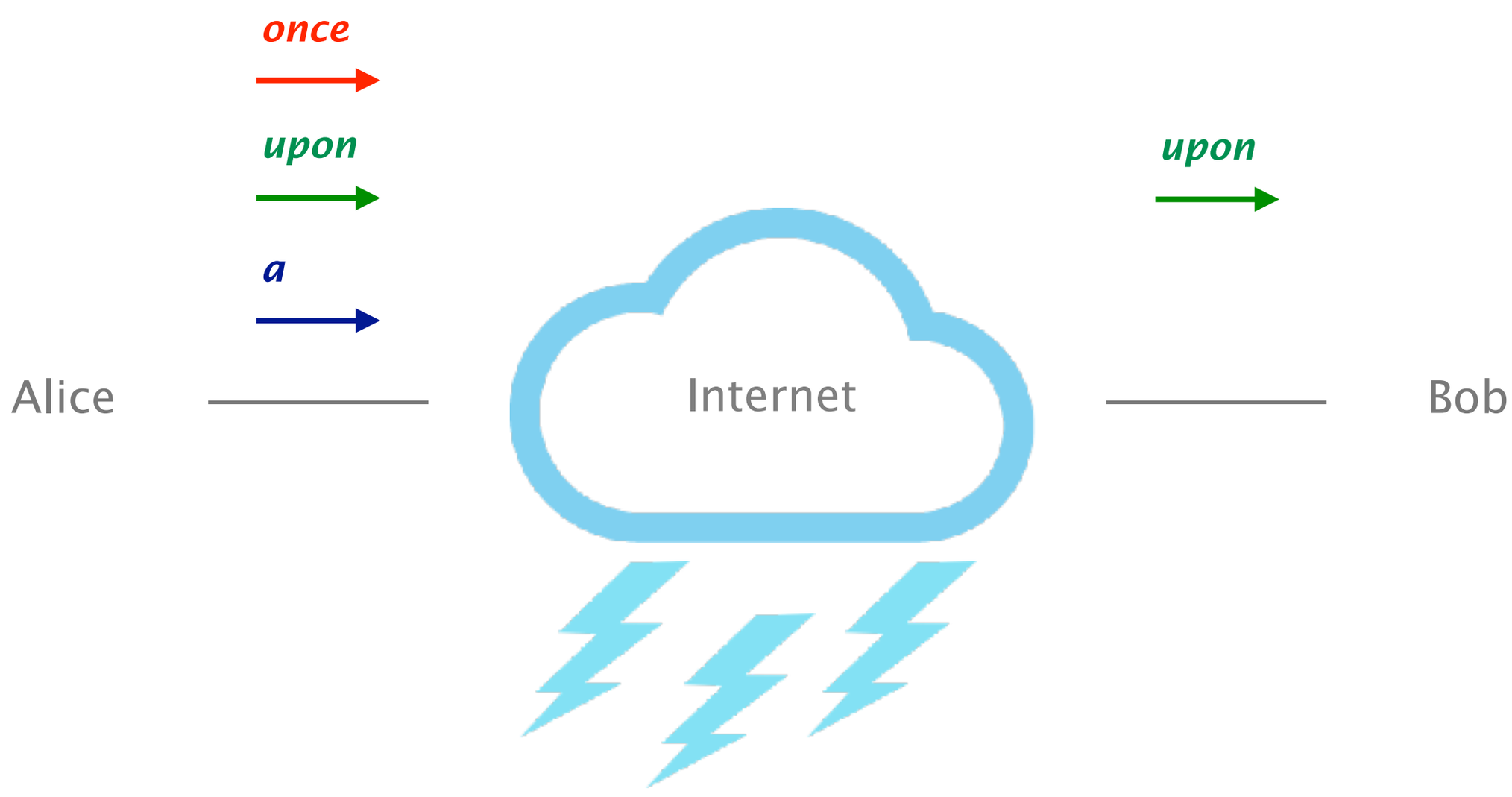
	Application	
L4	Transport	reliable end-to-end delivery
	Network	
	Link	
	Physical	

Let’s consider that Alice wants to transmit a text to Bob, word-by-word, via the Internet



The Internet (aka the Network layer) only provides a best-effort global packet delivery service

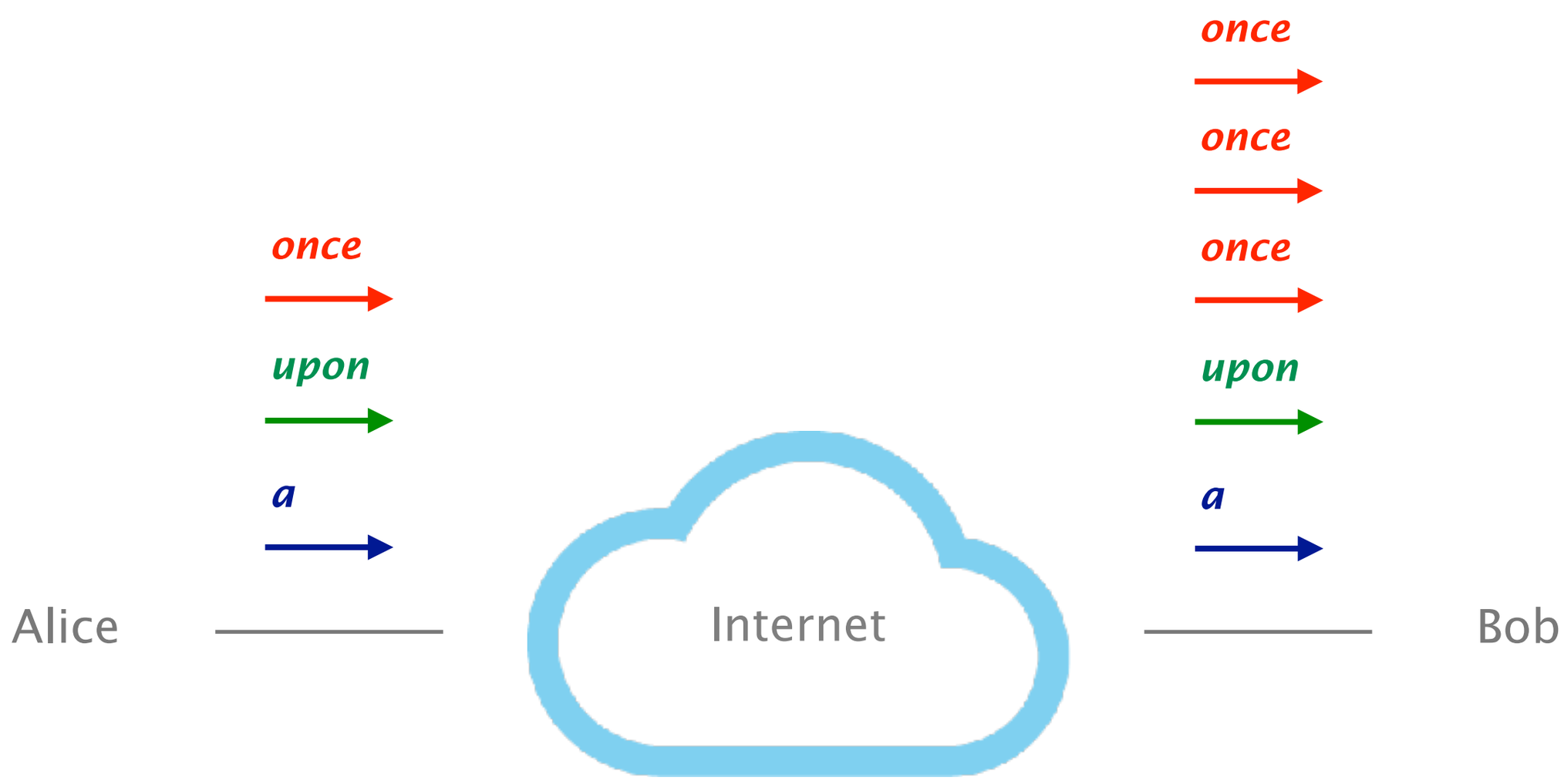
Data packets can get lost



Data packets can get corrupted

Data packets can get reordered

Data packets can get duplicated



Your job is to design a reliable transport protocol running on Alice's and Bob's computer

	property
correctness	Bob should read exactly what you've typed in the same order, without any gap
timeliness	Bob should receive the complete text as fast as possible minimize time until data is transferred
efficiency	Minimize the use of bandwidth don't send too many packets

The number in front of you is

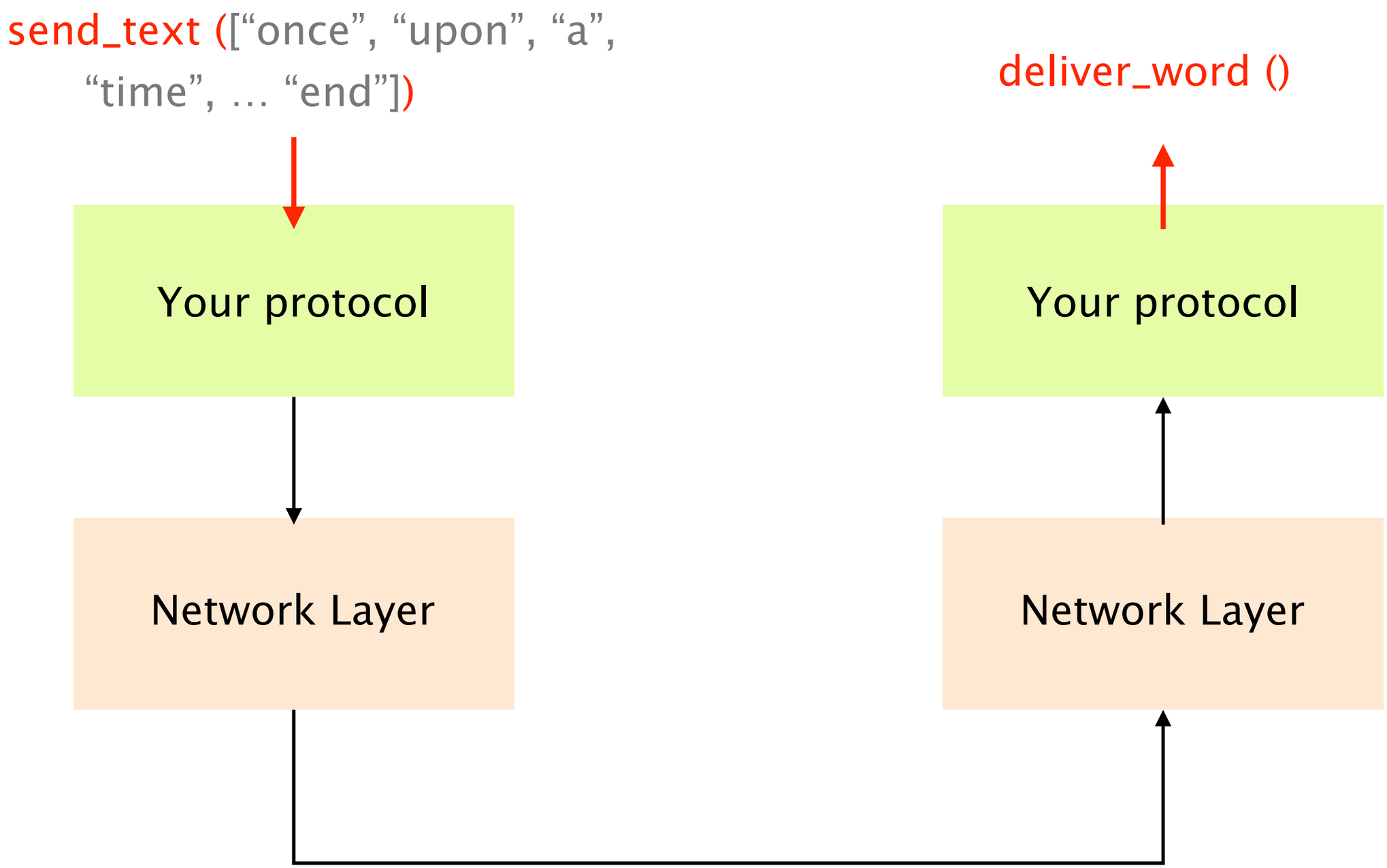
your group number

Your task

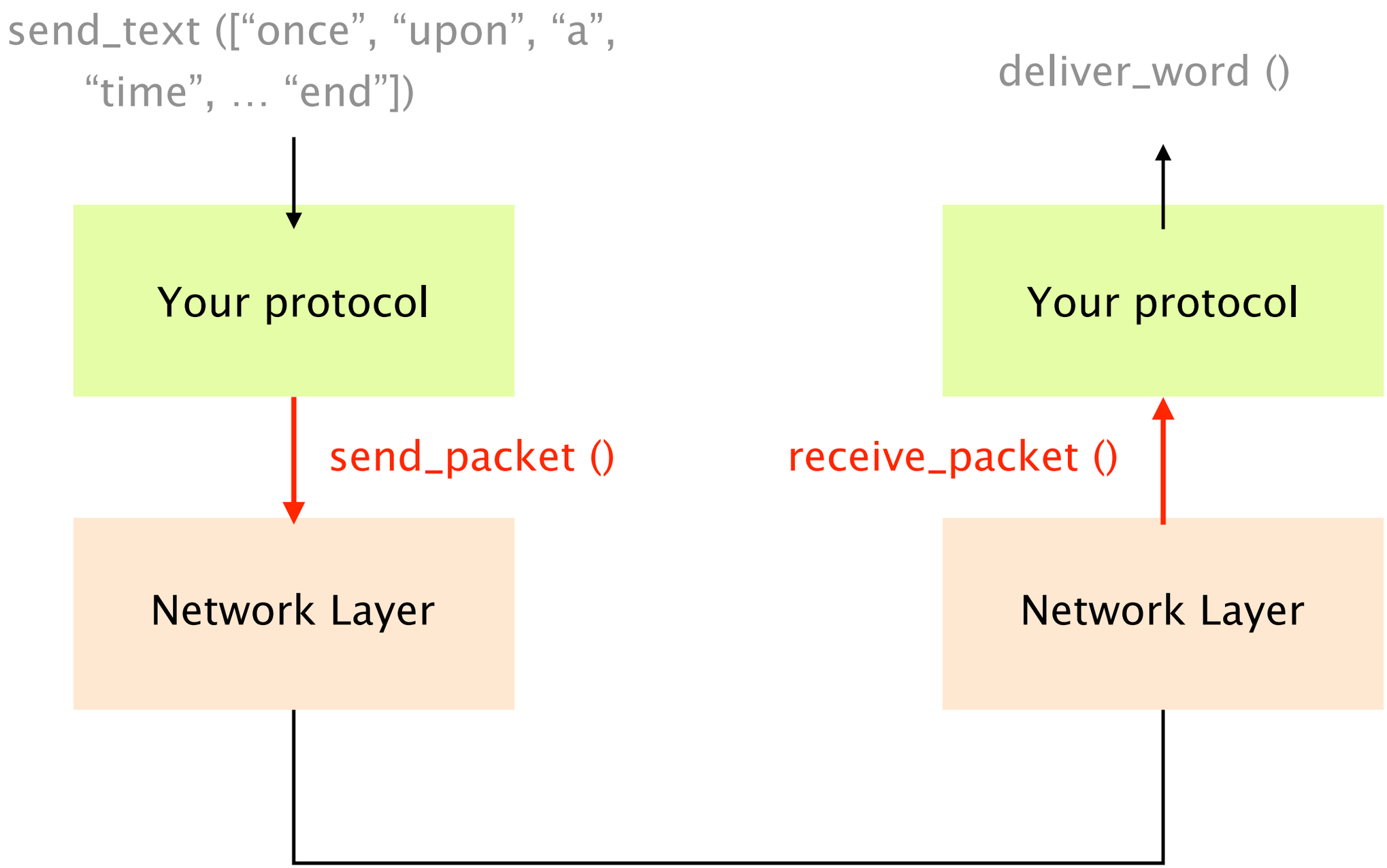
Design a protocol that can deal with packet loss, corruption, reordering and duplication

Design a protocol that can deal with packet loss, corruption, reordering and duplication

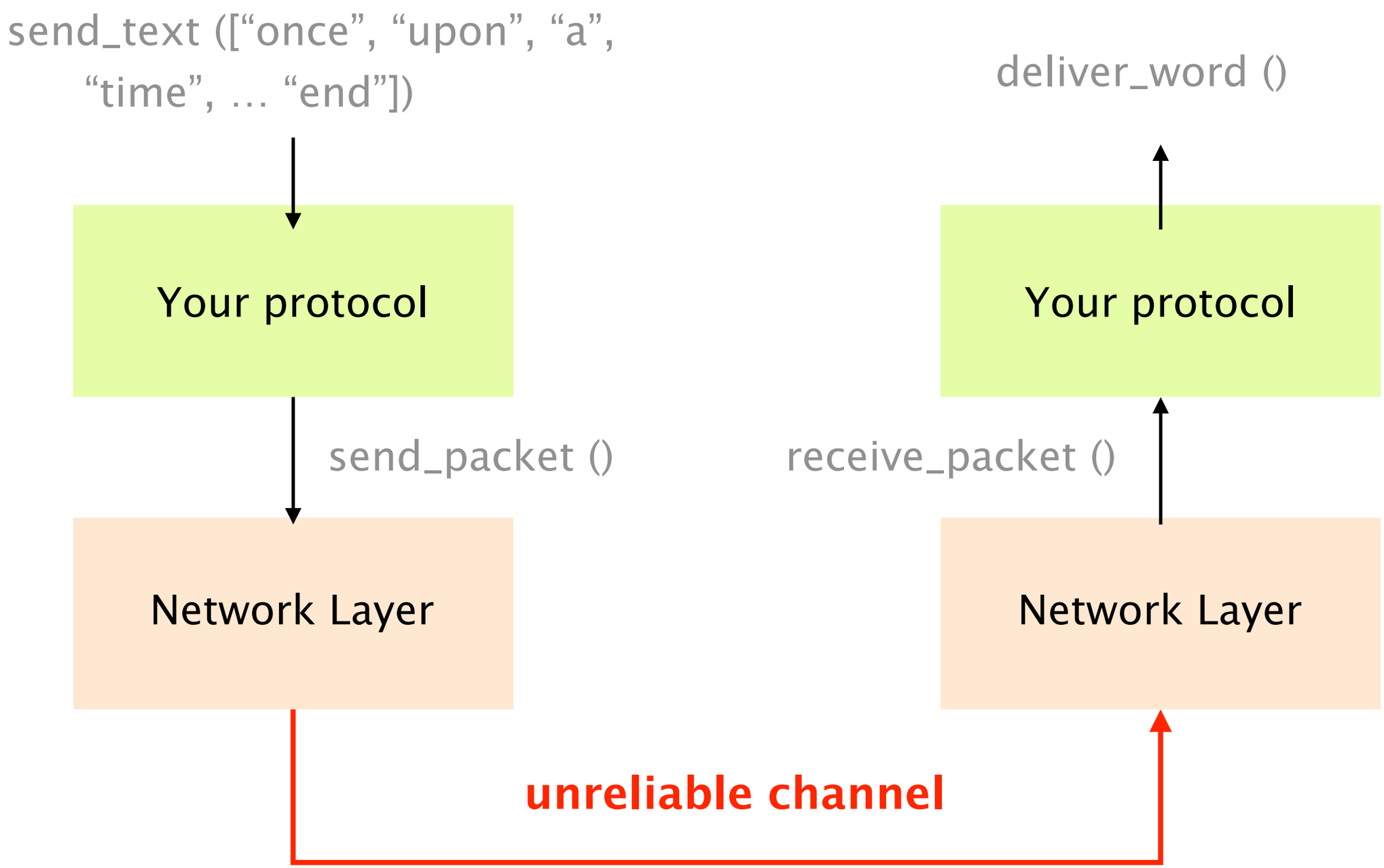
Your protocol receives a list of words on one host, and deliver them, in order, one-by-one, on another host



Your protocol uses 2 primitives of the network layer: `send_packet` and `receive_packet`



Packets can be lost, corrupted, reordered or duplicated



first Write down the pseudo-code of a protocol
that sends at most 1 word/packet at a time.
Each packet can be lost, corrupted or duplicated.

then Think about how you would extend your protocol
so that it can send *multiple* words/packets at a time.
How you deal with packet reordering?

output The procedure you run on the sender and receiver
The header(s) you need to add to the packets
An idea of how you support >1 outstanding packets

You have 15 minutes.

Any group member should be able to present its group's protocol

The basic protocols underlying the Internet are *intuitive*

The principles behind the Internet are
more about architecture than engineering

Principles

Interconnect many different networks

Ethernet, Optical Fibers, wireless, …

Scale to the entire world

both geographically and numerically

Tolerate and recover from failures

both constant and inevitable

The principles behind the Internet are

more about architecture than engineering

Architecture

what tasks get done

and *where*

Engineering

how tasks get done

The principles behind the Internet are

more about architecture than engineering

Architecture

what tasks get done

and *where*

in the network?

in the hosts?

Engineering

how tasks get done

with what technology?

Network engineering is all about optimization and balancing tradeoffs

Goals

Speed

Quality of Service

Cost

Security

Port density

…

Reliability

Solution for the single packet case

Alice

```
for word in list:
    send_packet(word);
    set_timer();

upon timer going off:
    if no ACK received:
        send_packet(word);
        reset_timer();
upon ACK:
    pass;
```

Bob

```
receive_packet(p);
if check(p.payload) == p.checksum:
    send_ack();

    if word not delivered:
        deliver_word(word);
else:
    pass;
```

The solution for the multiple packets case will be given in two lectures from now

Even in our protocol, there is a clear tradeoff between timeliness and efficiency in the selection of the timeout

```
for word in list:
    send_packet(word);
    set_timer();

upon timer going off:
    if no ACK received:
        send_packet(word);
        reset_timer();
upon ACK:
    pass
```

Too small timers will cause unnecessary retransmissions, too large timers will slow down the communication

The “right” value depends on the network conditions

Protocols have to be flexible and adapt to them