

Programmazione e Architetture (Modulo B)

Lezione 19

TCP, UDP e uno sguardo al livello applicazione

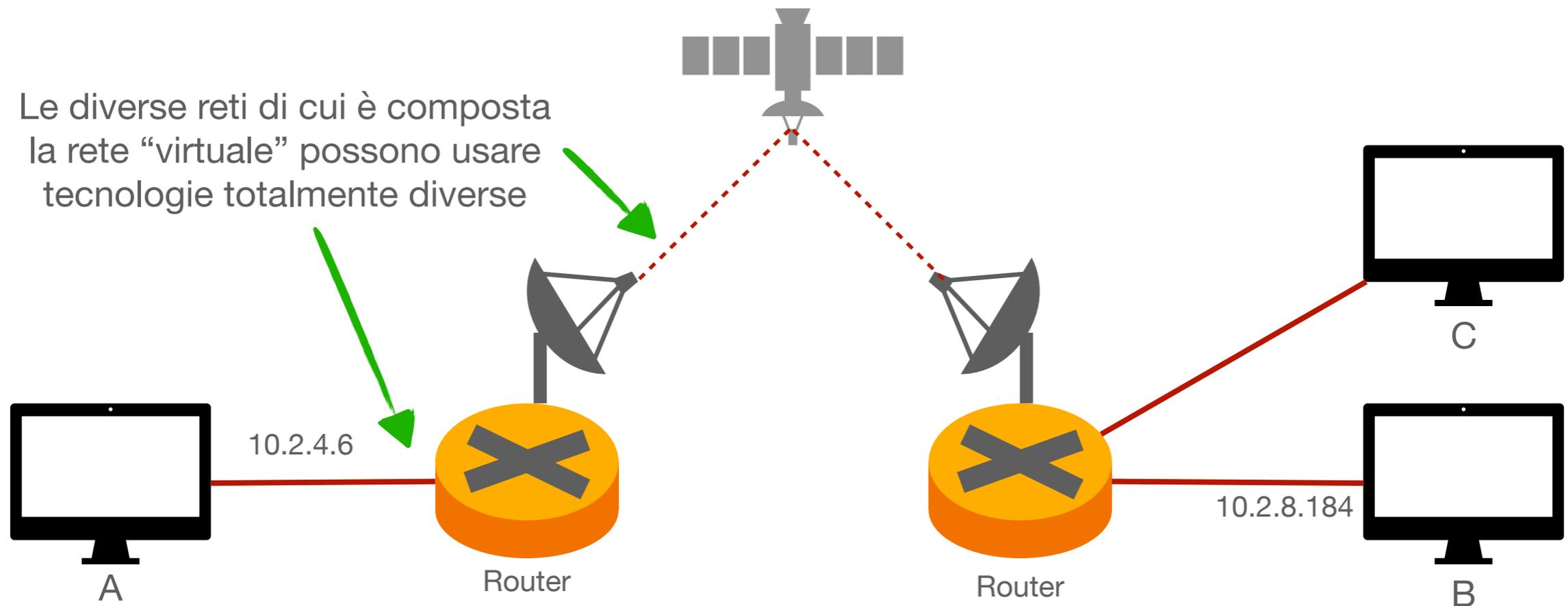
Breve riepilogo

IP e le “reti di reti”

- A ogni nodo viene assegnato un indirizzo IP, che scriveremo nella forma $a . b . c . d$ con $a, b, c, d \in \{0, 1, \dots, 255\}$
- Possiamo inviare pacchetti IP tra due macchine se sappiamo i loro indirizzi IP...
- ...anche se le macchine non sono direttamente connesse tra di loro
- Questo perché con IP abbiamo una rete “virtuale” composta da più reti interconnesse tra di loro

La vita di un pacchetto IP

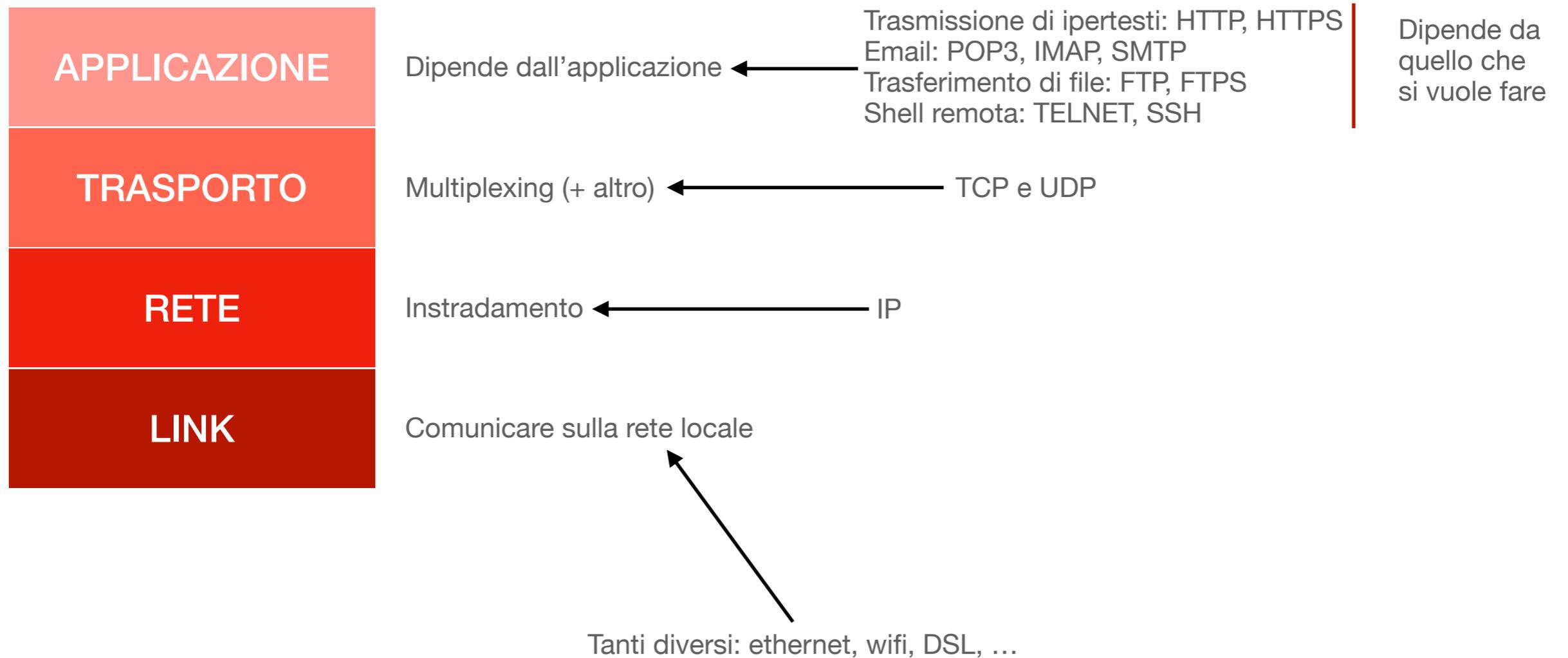
Da A a B



Ma ogni nodo è dotato di un indirizzo IP e il pacchetto può fare più salti (passare per più router) per raggiungere il nodo di destinazione

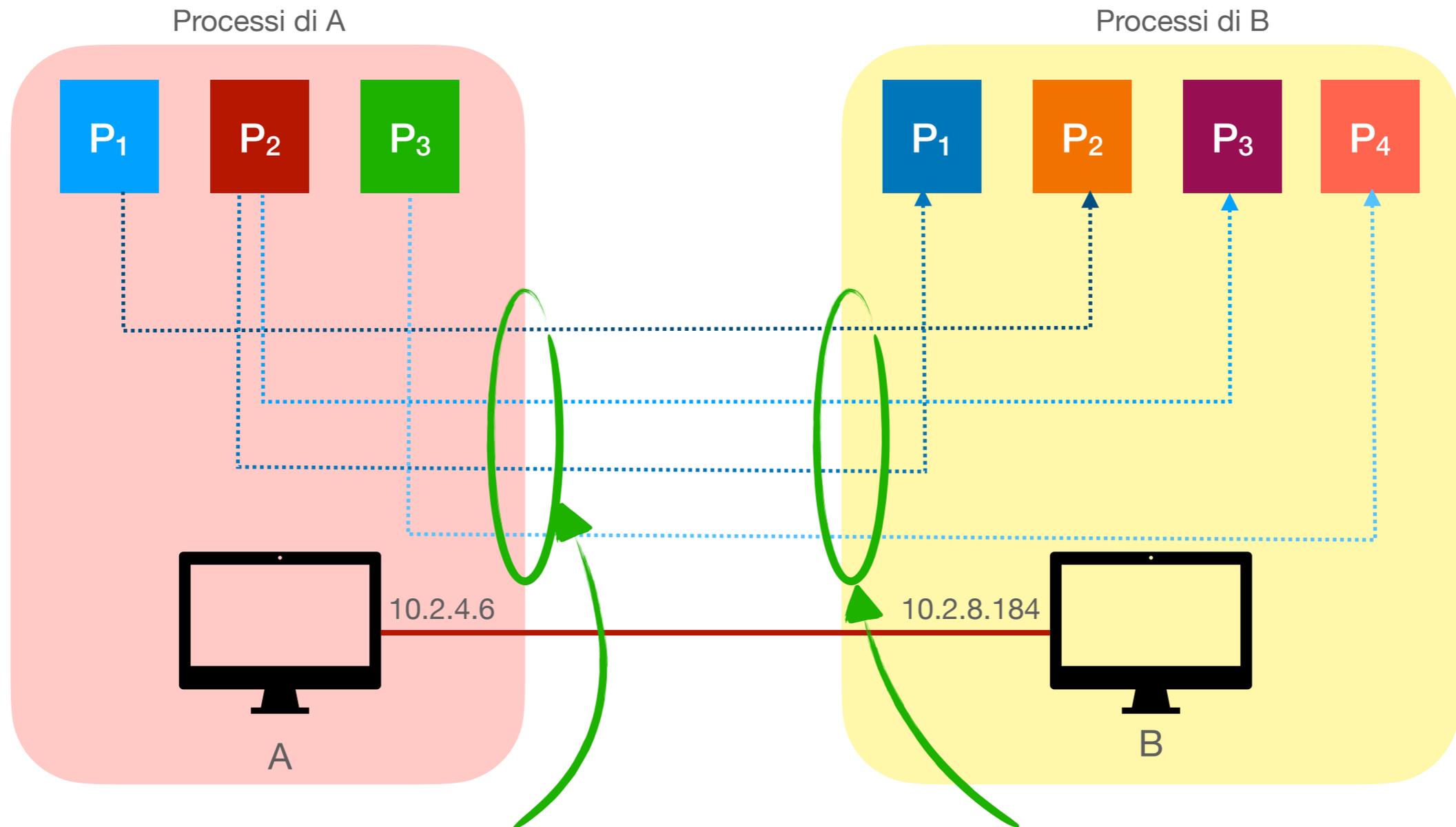
Internet Protocol Suite

Protocolli di comunicazione



Multiplexing e Demultiplexing

Comunicazione tra processi di A e processi di B



Multiplexing

Dobbiamo far passare i messaggi di processi diversi attraverso un unico canale

Demultiplexing

Dobbiamo separare i messaggi che ci arrivano tra i diversi processi

Multiplexing e Demultiplexing

Porte in UDP e TCP

- Possiamo fare in modo che ogni processo acquisisca un identificativo che, all'interno di una macchina, usiamo per decidere a che processo inviare i dati
- Questo identificativo è detto “porta” ed è un numero senza segno di 16 bit (quindi da 0 a 65535)
- Possiamo quindi dire cose come “consegnare a 10.6.7.123 sulla porta 8080” e questo ci permette di identificare il processo a cui “girare” il messaggio
- I numeri di porta da 0 a 1023 sono i **well-known port numbers** e il loro utilizzo è solitamente ristretto a ospitare certe tipologie di protocolli a livello di applicazione

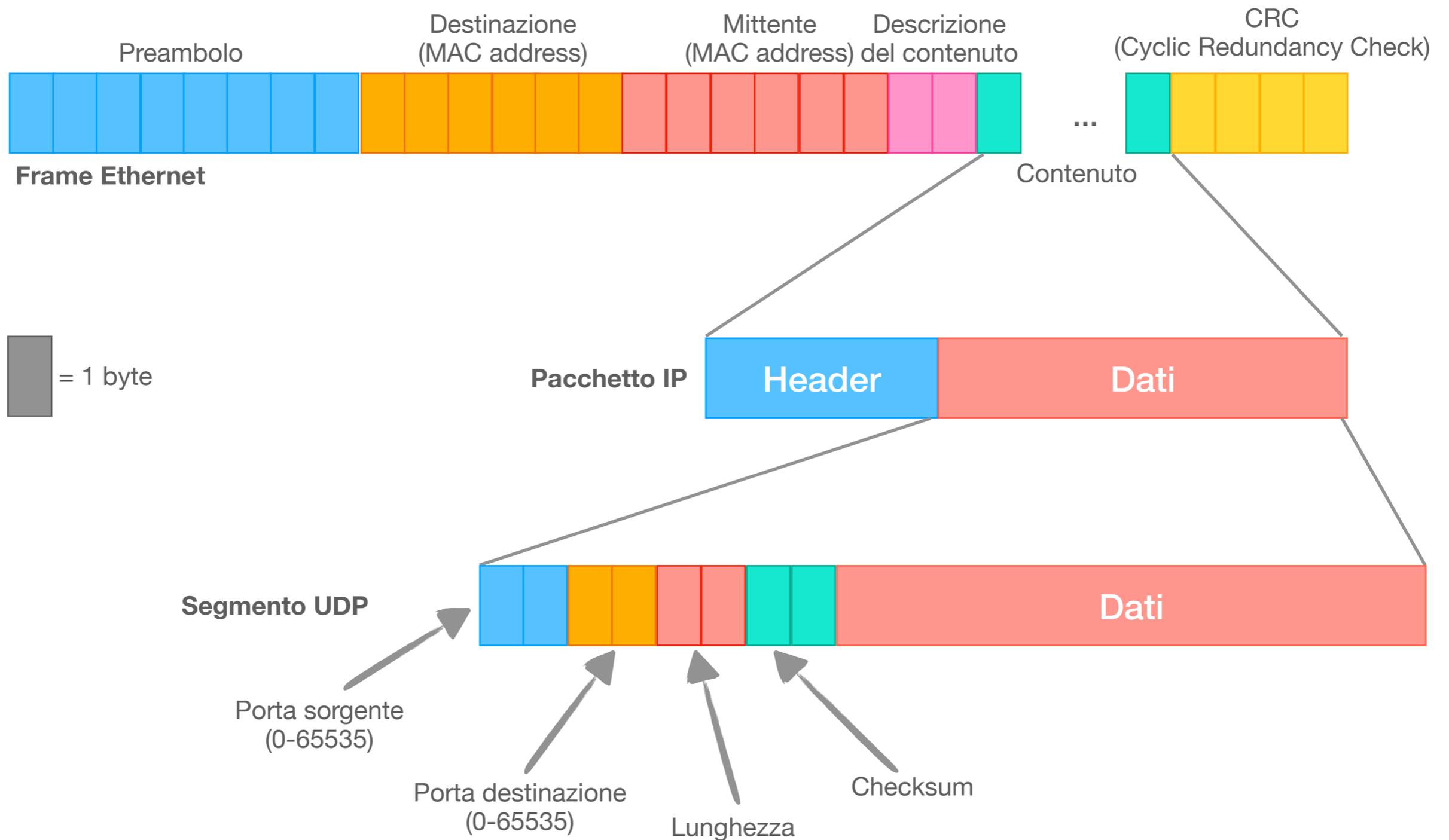
User Datagram Protocol (UDP)

Solo multiplexing/demultiplexing

- Il protocollo più semplice che ci permette di fare multiplexing e demultiplexing è UDP
- Consiste del minimo indispensabile:
 - Numeri di porta sorgente e destinazione
(serve la porta sorgente per sapere a chi rispondere)
 - Lunghezza del segmento
 - Un checksum

Segmento UDP

Frame che contiene un pacchetto che contiene un segmento



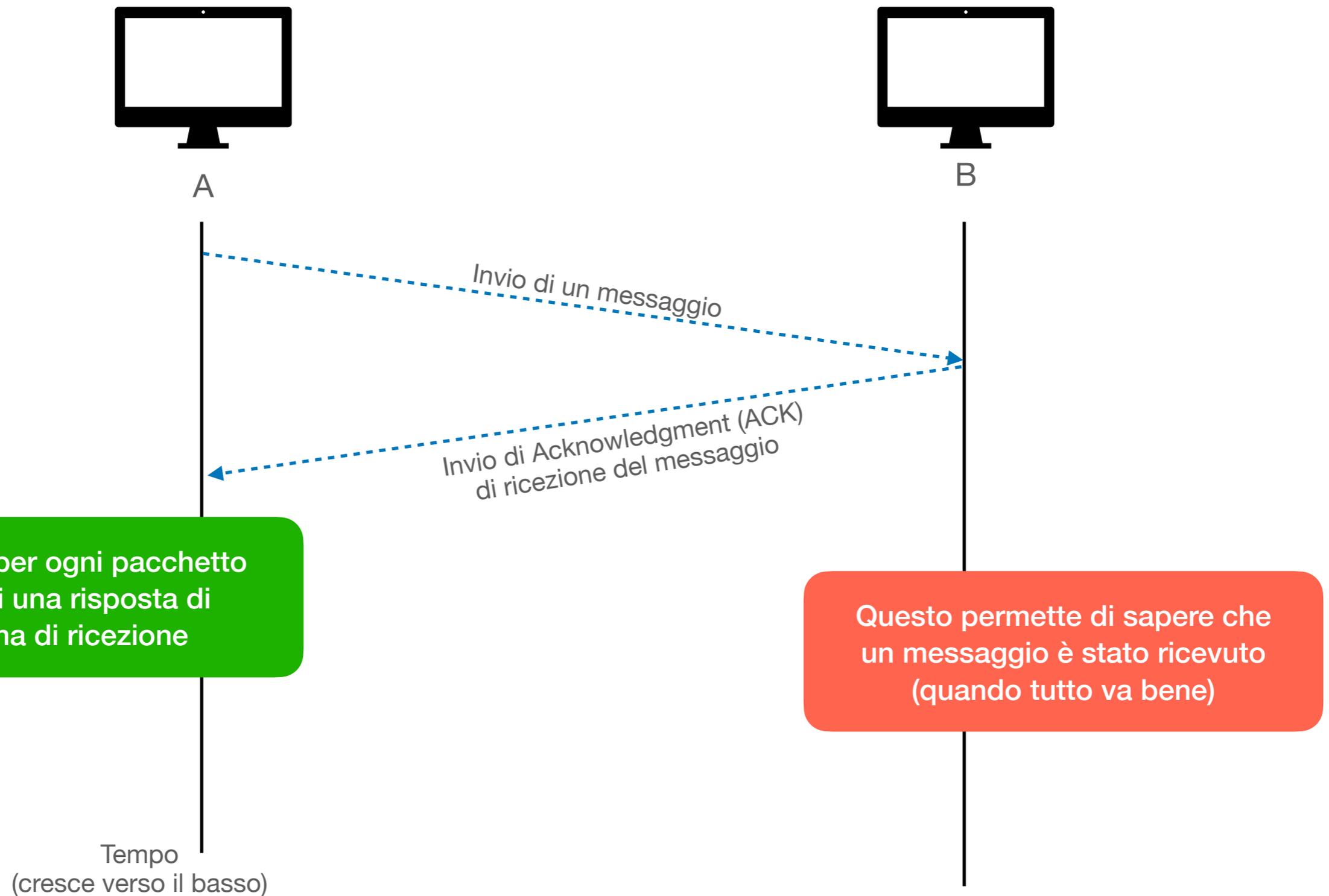
Ottenere una comunicazione affidabile

Diverse tipologie di errore

- IP non ci assicura che i messaggi arrivino a destinazione o che arrivino nell'ordine corretto
- Dobbiamo stabilire come ottenere comunque un canale affidabile
- Possiamo:
 - Avere che i messaggi vengono danneggiati
 - Perdere i messaggi

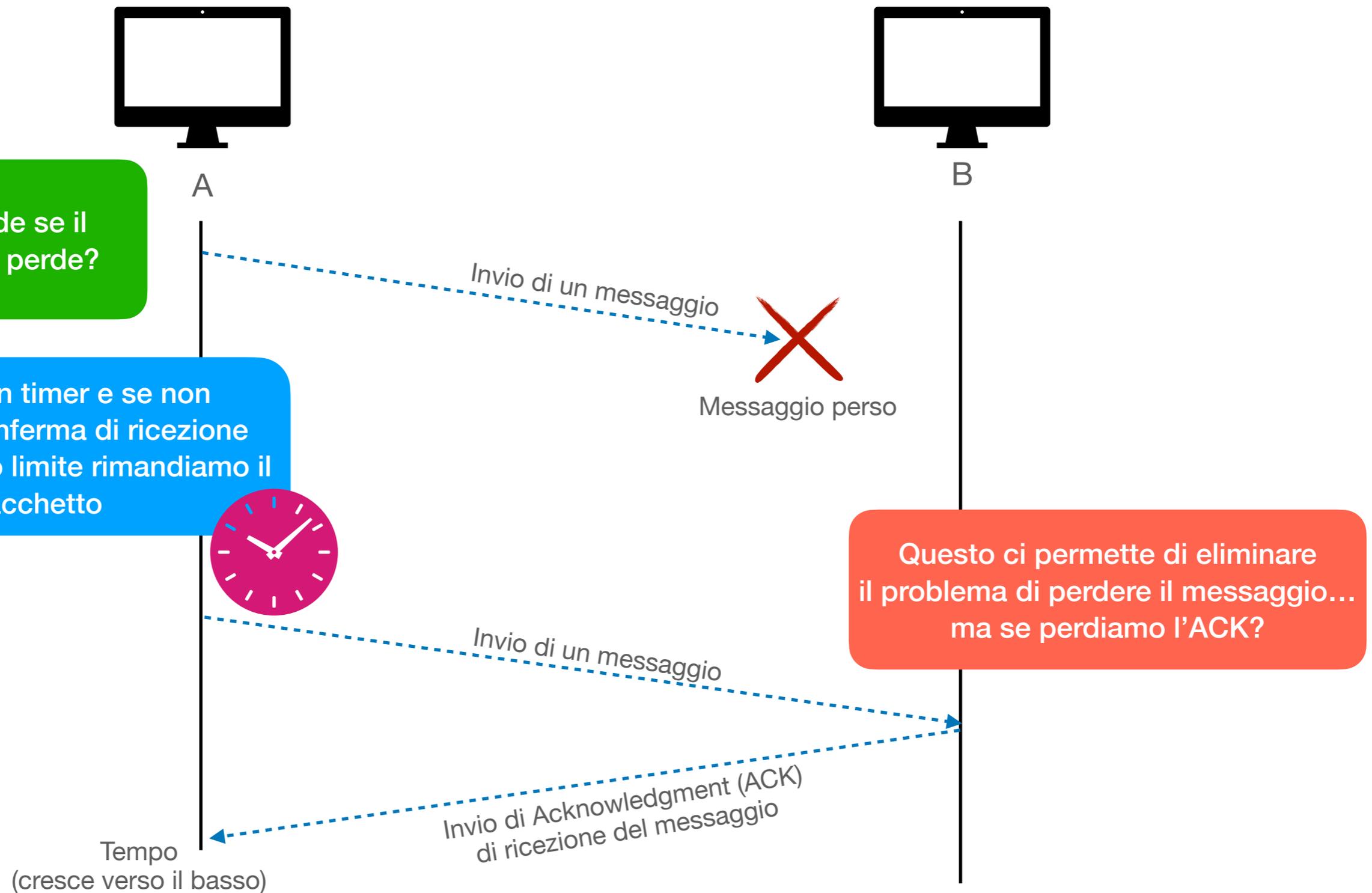
Uso degli ACK

Chiedere la conferma di ricezione



Uso degli ACK

Ritrasmettere il messaggio

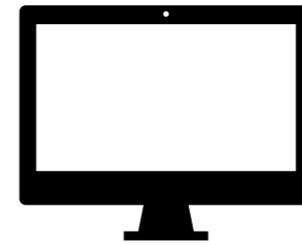


Uso degli ACK

Messaggi duplicati



A



B

Cosa succede se perdiamo l'ACK?

Rimandiamo il messaggio perché non abbiamo ricevuto l'ACK...



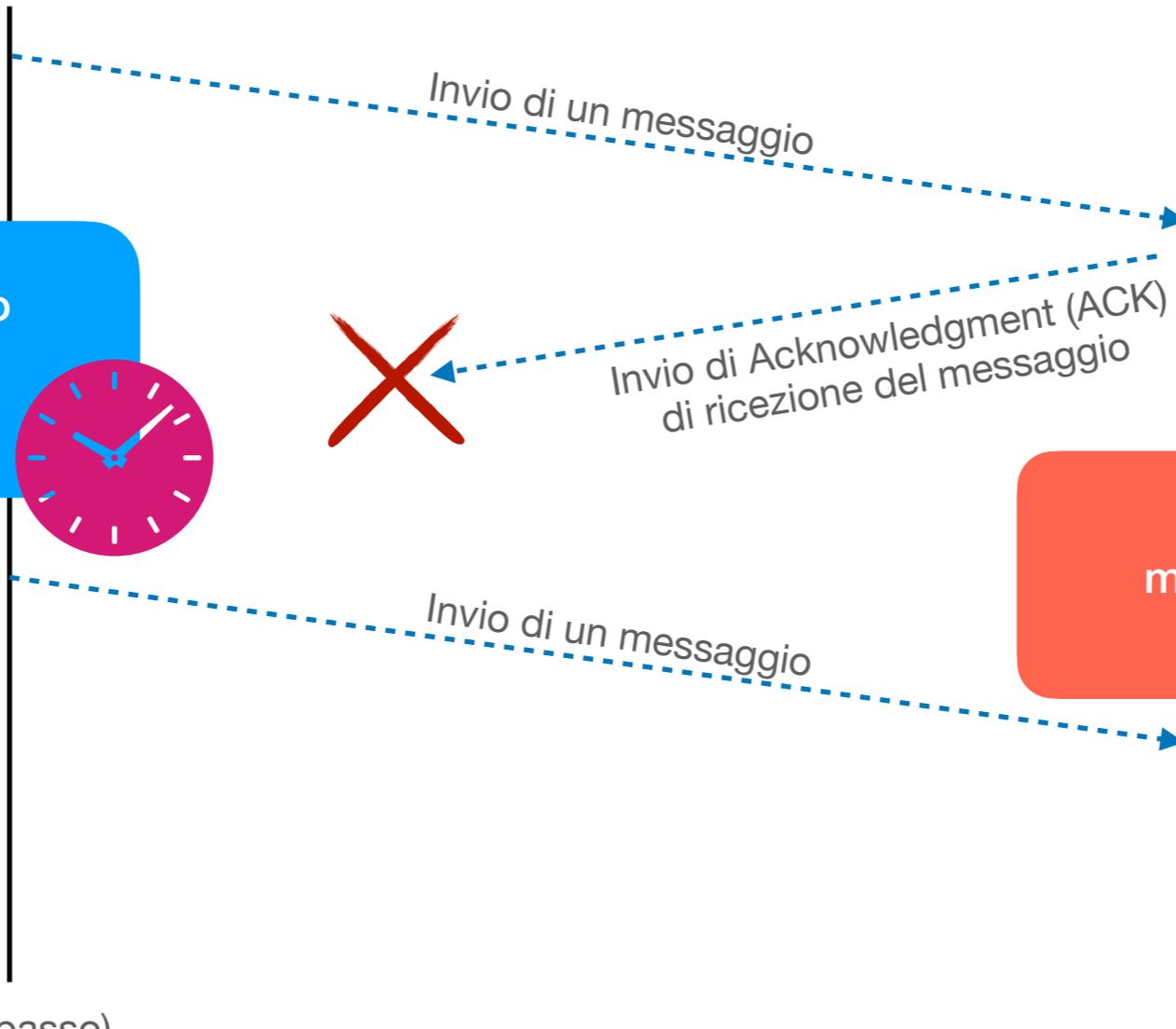
Invio di un messaggio

Invio di Acknowledgment (ACK) di ricezione del messaggio

Invio di un messaggio

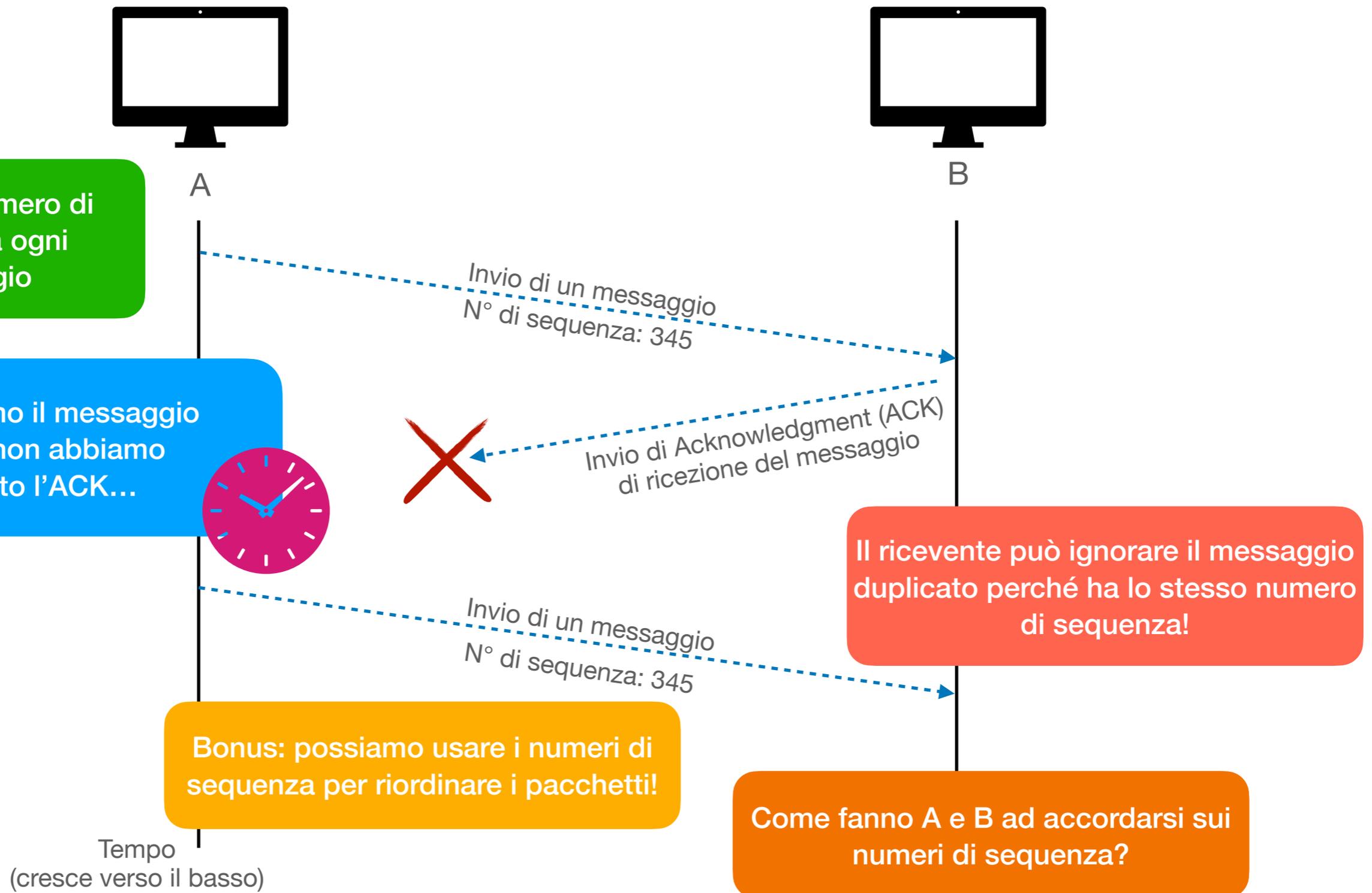
Così però abbiamo un messaggio duplicato per il ricevente!

Tempo
(cresce verso il basso)



Uso degli ACK

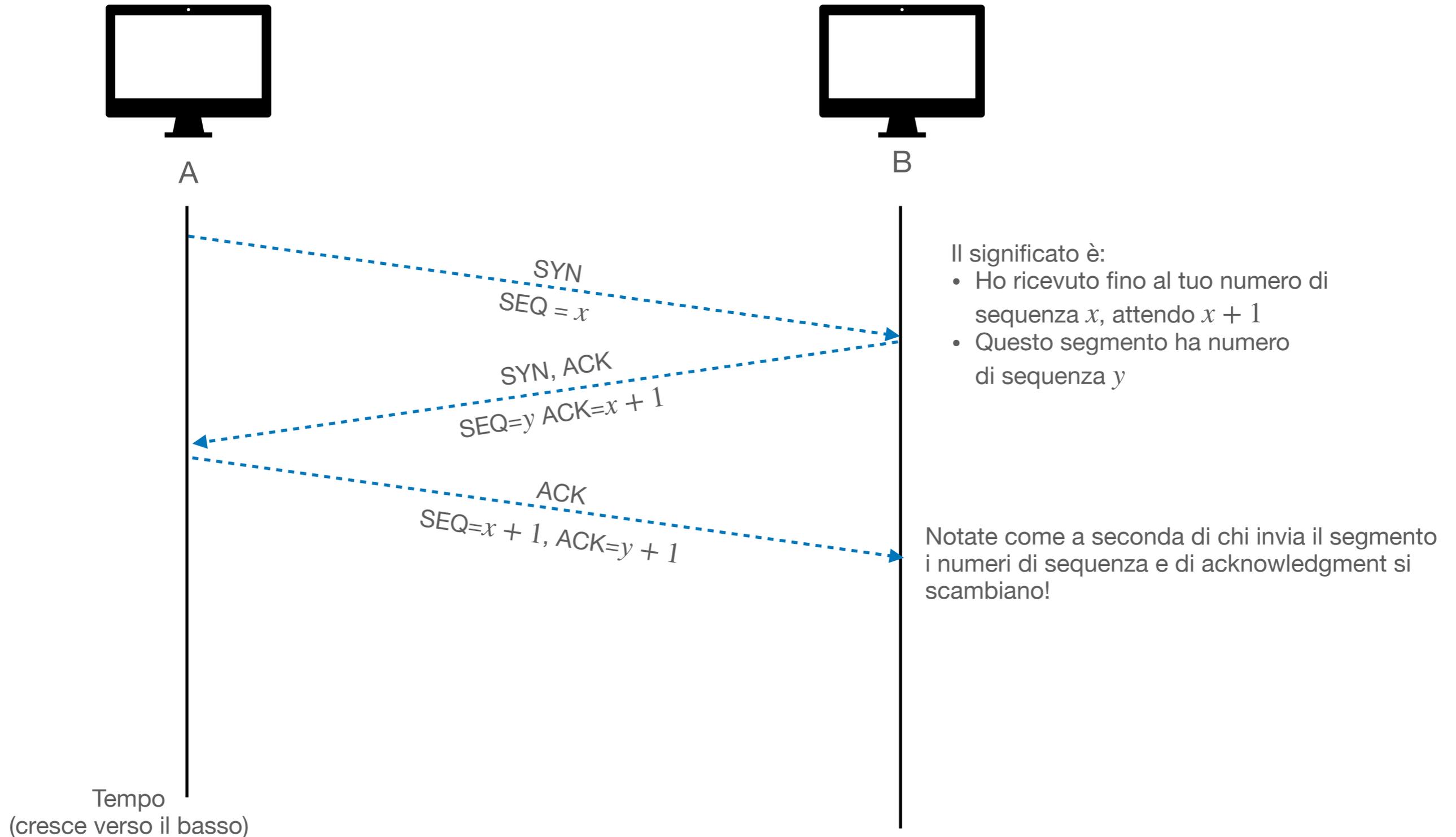
Numeri di sequenza



Stabilire connessioni

- È possibile effettuare un primo scambio di messaggio per accordarsi sui numeri di sequenza e su altri parametri
- Questo viene fatto in TCP (Transmission Control Protocol) con un **three-way handshake** (stretta di mano in tre passi)
- Per iniziare una connessione tra A e B:
 - A invia un segmento a B con un bit detto di SYN impostato a 1 e un numero di sequenza SEQ con valore x
 - B risponde con un segmento a A con i bit SYN e ACK impostati a 1, un numero di ACK pari a $x + 1$ e un numero di sequenza SEQ pari a y
 - A risponde con un segmento con il bit di ACK impostato a 1, ACK con valore $y + 1$ e SEQ $x + 1$

Three-way handshake



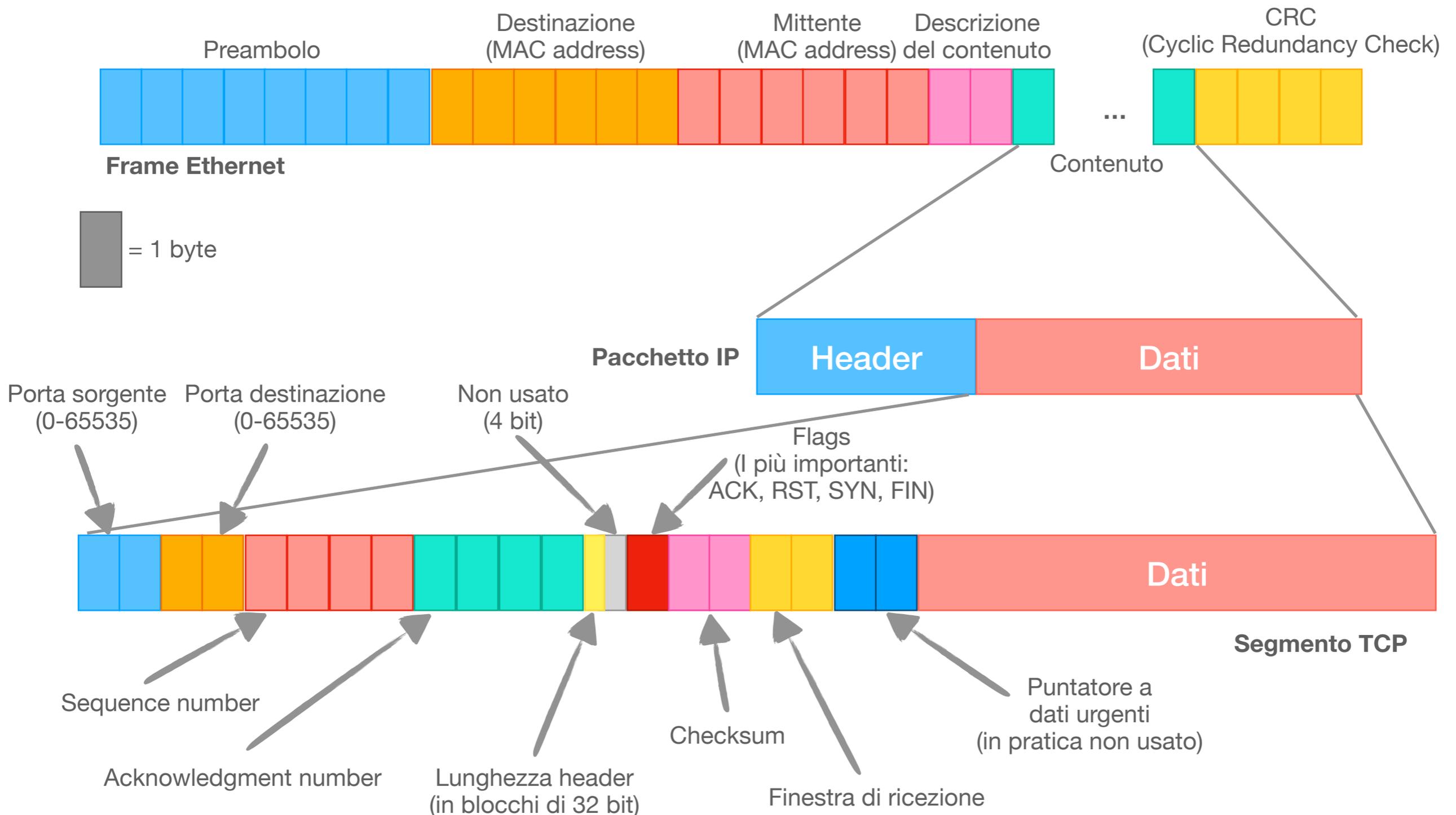
Transmission Control Protocol

TCP

- Il protocollo TCP ci fornisce tutta la struttura per avere numeri di sequenza, porte sorgenti e destinazione e stabilire connessioni
- Il protocollo TCP non è limitato all'invio di un segmento alla volta mentre è in attesa di ricevere l'ACK. Può inviare più segmenti e ricevere ACK della forma "ho ricevuto tutto fino al numero X"
- TCP fa crescere il numero di segmenti "in volo" (inviati ma di cui non è ancora ricevuto l'ACK)...
- ...a meno di non perdere dei segmenti, in quel caso il valore decresce
- Questo permette a TCP di adattarsi alla larghezza di banda disponibile

Segmento TCP

Frame che contiene un pacchetto che contiene un segmento



Protocolli a livello di applicazione

HTTP, FTP, SMTP, SSH, ...

- Le tipologie di protocolli a livello di applicazione dipendono, non sorprendentemente, dalla tipologia di applicazione
- A seconda dei requisiti di affidabilità questi possono essere costruiti “sopra” TCP o UDP
- Generalmente se si usa UDP si deve accettare che alcuni segmenti potrebbero venire persi oppure costruirsi a livello di applicazione un qualcosa di affidabile “sopra” UDP
- Spesso questi protocolli hanno delle porte riservate, così chi deve contattare per uno specifico servizio (web, invio email, etc) è già noto su quale porta contattare!

Protocolli a livello di applicazione

Alcuni protocolli

- Per il trasferimento di ipertesti (i.e., il web):
Hypertext Transfer Protocol (HTTP)
- Posta elettronica (trasmissione): Simple Mail Transfer Protocol (SMTP)
- Posta elettronica (ricezione): Internet Message Access Protocol (IMAP) o il più vecchio Post Office Protocol (POP)
- Shell remota: Secure SHell (SSH) o il non sicuro TELNET
- Associare informazioni a dei nomi di dominio (e.g., degli indirizzi IP): Domain Name System (DNS)
 - Questo è il motivo per cui potete scrivere www.units.it e non 130.186.9.36!