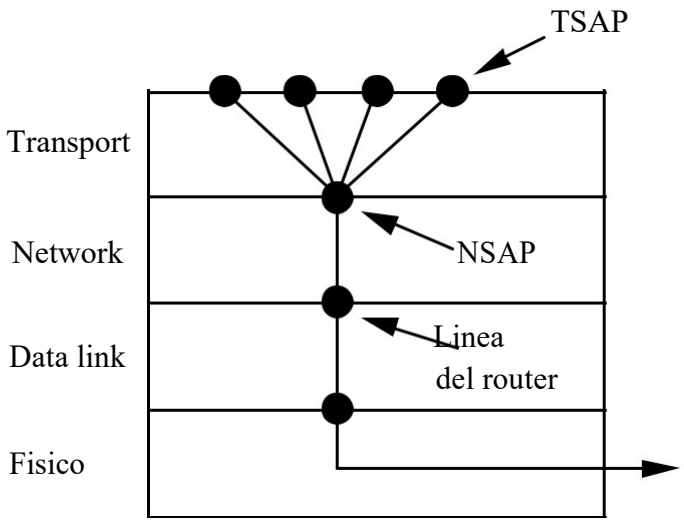
Multiplexing

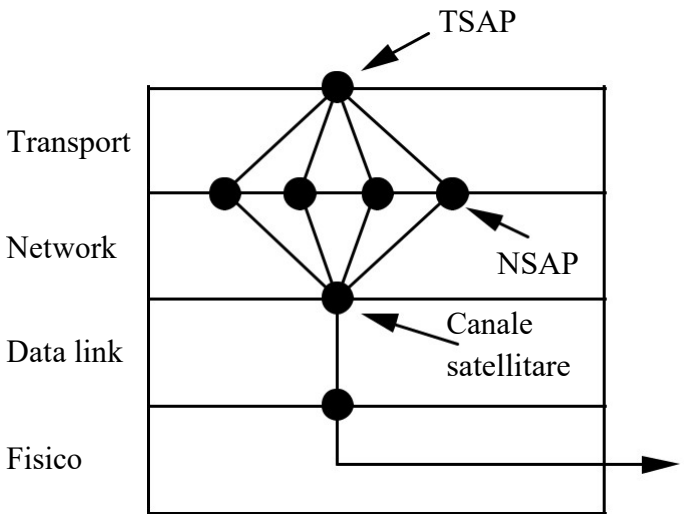
Una importante opportunità è data dal multiplexing delle conversazioni di livello transport su quelle di livello network.

Se le connessioni di livello network (in una rete che le offre) sono costose da istituire, allora è conveniente convogliare molte conversazioni transport su un'unica connessione network (***upward multiplexing***).



Upward multiplexing

Viceversa, se si vuole ottenere una banda superiore a quella consentita a una singola connessione network, allora si può guadagnare banda ripartendo la conversazione transport su più connessioni network (***downward multiplexing***).



Downward multiplexing

Il livello transport in Internet

Il livello transport di Internet è basato su due protocolli:
TCP (*Transmission Control Protocol*);
UDP (*User Data Protocol*).

Il secondo, UDP, è di fatto IP con l'aggiunta di un breve header, e fornisce un servizio di trasporto datagram (quindi non affidabile).

Il protocollo TCP è stato progettato per fornire un flusso di byte affidabile, da sorgente a destinazione, su una rete non affidabile.

Dunque, offre un servizio reliable e connection oriented, e si occupa di:

- **accettare dati** dal livello application;
- **spezzarli in segment**, il nome usato per i TPDU (dimensione massima 64 Kbyte, tipicamente circa 1.500 byte);
- **consegnarli al livello network**, eventualmente ritrasmettendoli;
- **ricevere segmenti dal livello network**;
- **rimetterli in ordine**, eliminando buchi e doppioni;
- **consegnare i dati, in ordine**, al livello application.

E' un servizio full-duplex con gestione di ack e controllo del flusso.

Il protocollo TCP

Le caratteristiche più importanti sono le seguenti:

- ogni byte del flusso TCP è numerato con un numero d'ordine a 32 bit, usato sia per il controllo di flusso che per la gestione degli ack;
- un segmento TCP non può superare i 65.535 byte;
- un segmento TCP è formato da:
  - uno header, a sua volta costituito da:
  - una parte fissa di 20 byte;
  - una parte opzionale;
  - i dati da trasportare;

TCP usa un meccanismo di sliding window di tipo go-back-n con timeout. Se questo scade, il segmento si ritrasmette. Si noti che le dimensioni della finestra scorrevole e i valori degli ack sono espressi in numero di byte, non in numero di segmenti.

32 bit

<table>
<tr><td colspan="5">Source port</td><td colspan="4">Destination port</td></tr>
<tr><td colspan="9">Sequence number</td></tr>
<tr><td colspan="9">Ack. number</td></tr>
<tr><td>TCP header len.</td><td></td><td>URG</td><td>ACK</td><td>PSH</td><td>RST</td><td>SYN</td><td>FIN</td><td>Window size</td></tr>
<tr><td colspan="6">Checksum</td><td colspan="3">Urgent pointer</td></tr>
</table>

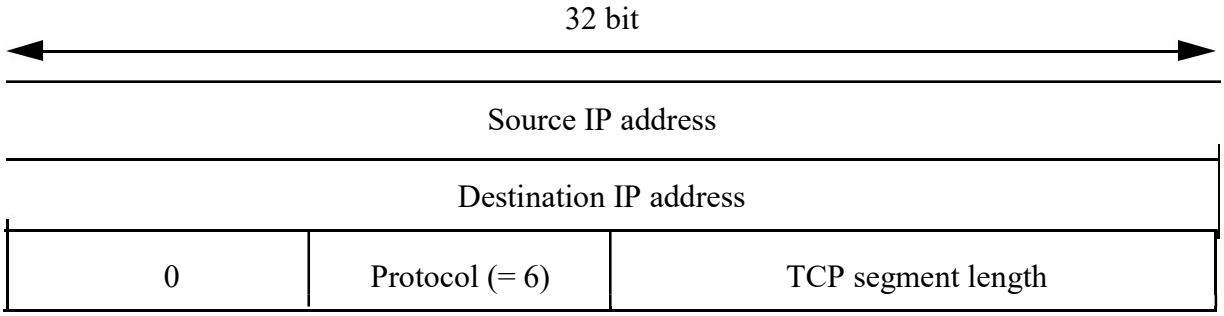
Options (zero o più parole di 32 bit)

Formato del segmento TCP

I campi dell'header hanno le seguenti funzioni:

Source port, destination port	identificano gli end point (locali ai due host) della connessione. Essi, assieme ai corrispondenti numeri IP, formano i due TSAP.
Sequence number	il numero d'ordine del primo byte contenuto nel campo dati.
Ack. number	il numero d'ordine del prossimo byte aspettato.
TCP header length	quante parole di 32 bit ci sono nell'header (necessario perché il campo options è di dimensione variabile).
URG	1 se urgent pointer è usato, 0 altrimenti.
ACK	1 se l'ack number è valido (cioè se si convoglia un ack), 0 altrimenti.
PSH	dati urgenti (***pushed data***), da consegnare senza aspettare che il buffer si riempia.
RST	richiesta di reset della connessione (ci sono problemi!).
SYN	usato nella fase di setup della connessione: • SYN=1 ACK=0 richiesta connessione; • SYN=1 ACK=1 accettata connessione.
FIN	usato per rilasciare una connessione.
Window size	il controllo di flusso è di tipo sliding window di dimensione variabile. Window size dice quanti byte possono essere spediti a partire da quello (compreso) che viene confermato con l'ack number. Un valore zero significa: fermati per un pò, riprenderai quando ti arriverà un uguale ack number con un valore di window size diverso da zero.
Checksum	simile a quello di IP; il calcolo include uno pseudoheader.
Urgent pointer	puntatore ai dati urgenti.
Options	fra le più importanti, negoziabili al setup: • dimensione massima dei segmenti da spedire; • uso di selective repeat invece che go-back-n; • uso di NAK.

Nel calcolo del checksum entra anche uno ***pseudoheader*** , in aperta violazione della gerarchia, dato che il livello TCP in questo calcolo opera su indirizzi IP.



Formato dello pseudoheader TCP

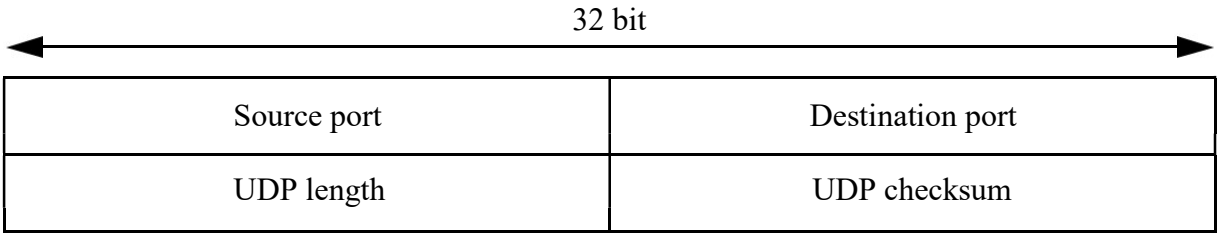
Lo pseudoheader non viene trasmesso, ma precede concettualmente l'header. I suoi campi hanno le seguenti funzioni:

Source IP address, destination IP address	indirizzi IP (a 32 bit) di sorgente e destinatario.
Protocol	il codice numerico del protocollo TCP (pari a 6).
TCP segment length	il numero di byte del segmento TCP, header compreso.

Il protocollo UDP

Il livello transport fornisce anche un protocollo non connesso e non affidabile, utile per inviare dati senza stabilire connessioni (ad esempio per applicazioni client-server).

Lo header di un segmento UDP è molto semplice:



Formato dello header UDP

La funzione di calcolo del checksum può essere disattivata, tipicamente nel caso di traffico in tempo reale (come voce e video) per il quale è in genere più importante mantenere un'elevato tasso di arrivo dei segmenti piuttosto che evitare i rari errori che possono accadere.

Controllo congestione

Il protocollo TCP assume che, se gli ack non tornano in tempo, ciò sia dovuto a congestione della subnet piuttosto che a errori di trasmissione (dato che le moderne linee di trasmissione sono molto affidabili).

Dunque, TCP è preparato ad affrontare due tipi di problemi:
- scarsità di buffer a destinazione;
- congestione della subnet.

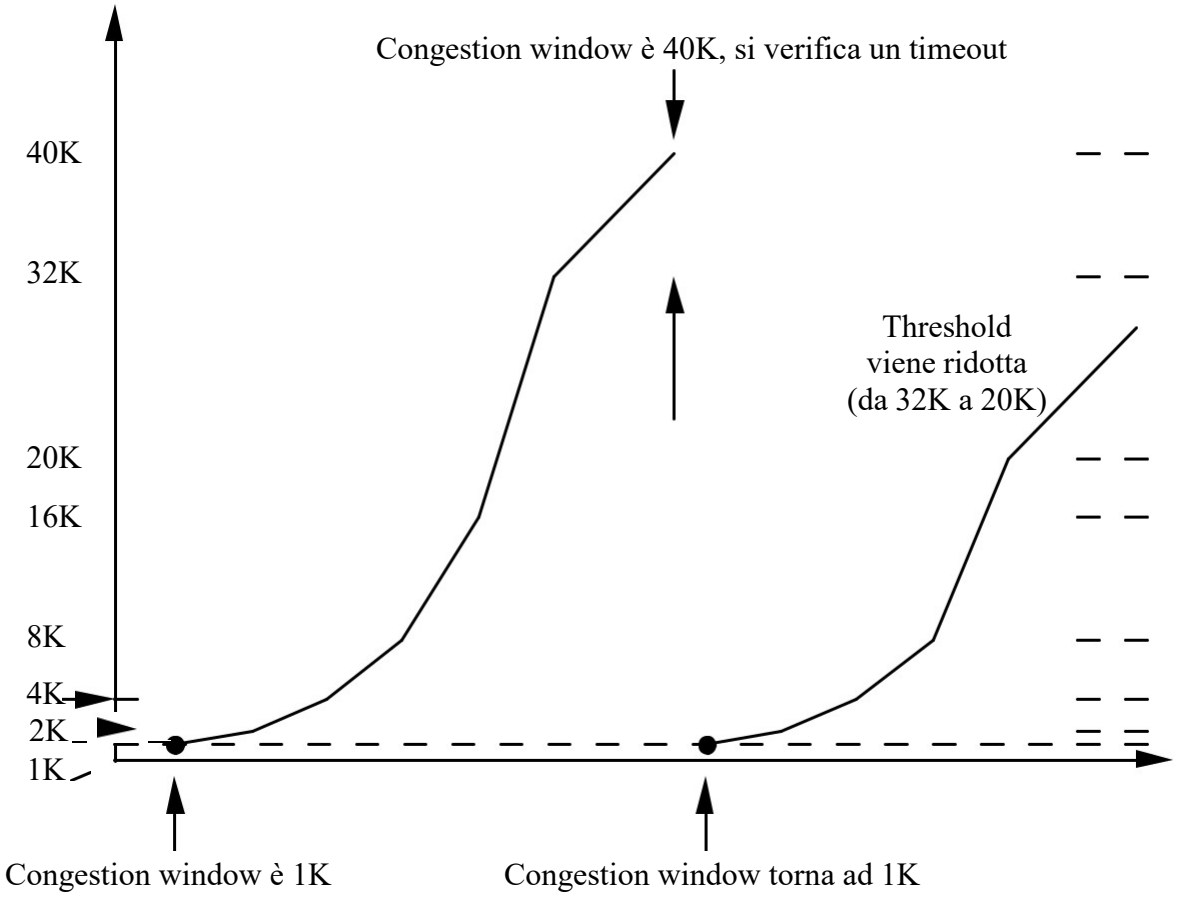
Ciascuno dei problemi viene gestito da una specifica finestra mantenuta dal mittente:
- la finestra del buffer del ricevitore (quella di cui all'esempio precedente);
- la ***congestion window***, che rappresenta quanto si può spedire senza causare congestione.

Il mittente si regola sulla più piccola delle due.
La congestion window viene gestita in questo modo:

il valore iniziale è pari alla dimensione del massimo segmento usato nella connessione;
ogni volta che un ack torna indietro in tempo la finestra si raddoppia, fino a un valore ***threshold***, inizialmente pari a 64 Kbyte, dopodiché aumenta linearmente di 1 segmento alla volta;
quando si verifica un timeout per un segmento:

il valore di threshold viene impostato alla metà della dimensione della congestion window;
la dimensione della congestion window viene impostata alla dimensione del massimo segmento usato nella connessione.

Vediamo ora un esempio, con segmenti di dimensione 1 Kbyte, threshold a 32 Kbyte e congestion window arrivata a 40 Kbyte:



Esempio di controllo della congestione TCP