



Prof. F. De Rango

*Corso di*

***SISTEMI TELEMATICI***

***a.a. 2009-2010***

**Lo strato di Trasporto**

# Scenario



- Internet è composta da host connessi a reti a commutazione di pacchetto, a loro volta interconnesse tramite router
- Gli host sono connessi alla rete e, dal punto di vista della rete, sono sorgenti e destinazioni dei pacchetti
- I processi sono gli elementi attivi negli host che producono/consumano i messaggi (secondo la definizione di processo come un programma in esecuzione). Terminali, file, dispositivi I/O comunicano tra loro tramite processi
- Un processo deve distinguere tra più flussi di comunicazione con altri processi, quindi si assume che ogni processo abbia un numero di porte attraverso le quali comunicare con le porte di altri processi
- Tutta la comunicazione è di tipo inter-processo e fornisce flussi bidirezionali di dati su connessioni logiche tra le porte dei processi coinvolti

# Strato di Trasporto



- Il servizio di comunicazione fornito dal livello di trasporto può essere:
  - affidabile con garanzia di consegna dei messaggi nel corretto ordine
  - non affidabile nel quale viene implementata di fatto la sola funzionalità di indirizzamento
- Naturalmente, il servizio realmente fornito all'applicazione dipende anche dal livello rete sottostante
- Nella suite IP sono definiti due tipi di trasporto:
  - ✕ TCP (Transmission Control Protocol), orientato alla connessione e affidabile (RFC 793)
  - ✕ UDP (User Datagram Protocol), senza connessione e non affidabile (RFC 768)

# Strato di Trasporto



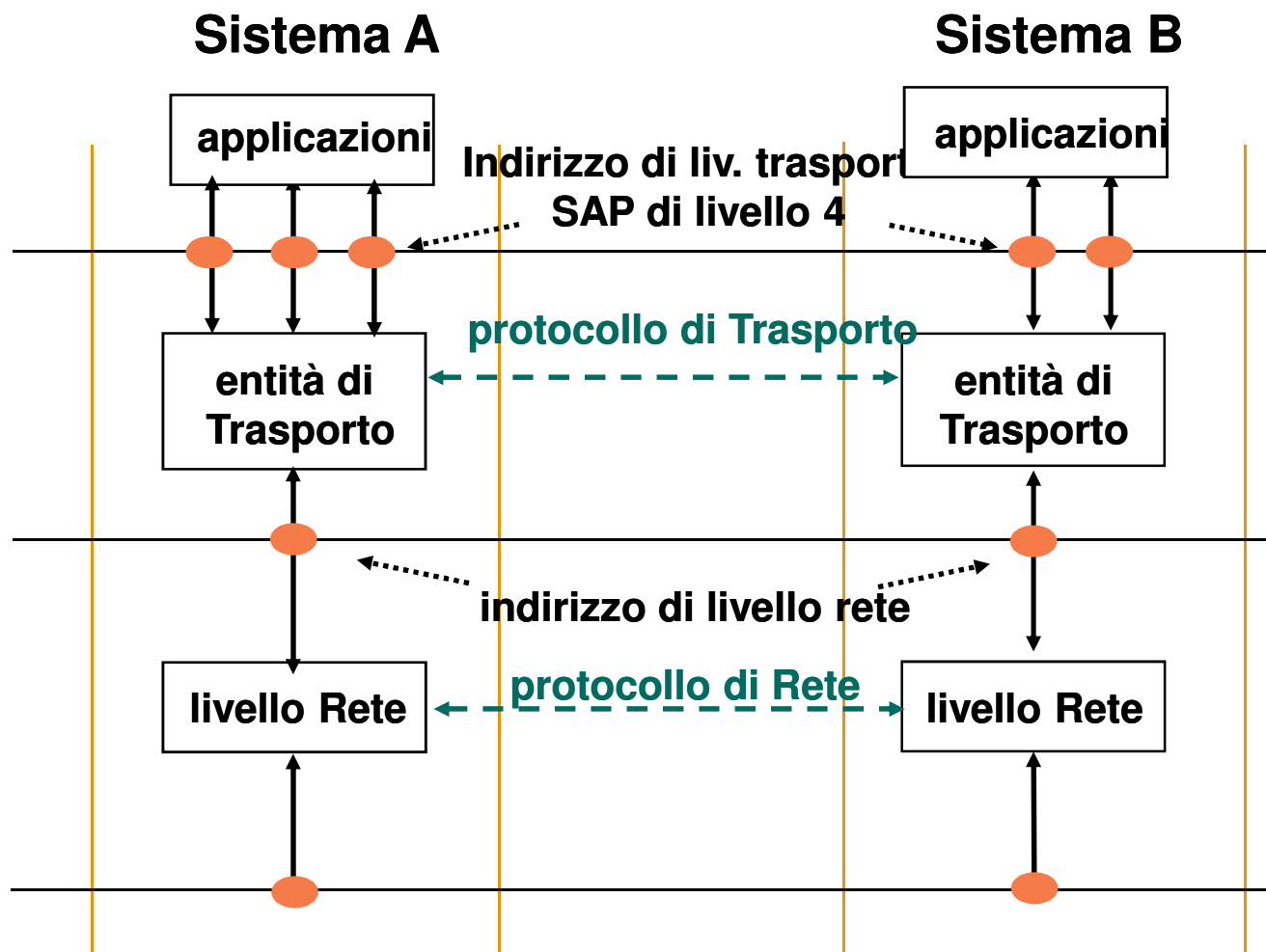
Prof. F. De Rango

- Il livello di trasporto nelle reti IP è implementato solo nei sistemi finali e serve a far colloquiare gli applicativi che sono in esecuzione nelle macchine remote
- Quindi, il primo compito che deve svolgere il livello di trasporto è quello di indirizzare i SAP su cui sono attestati i diversi processi applicativi in esecuzione su un host, ad es. HTTP, FTP, SMTP, ecc.
- Lo strato di trasporto è responsabile di distinguere, all'interno di uno stesso host, il processo applicativo destinatario (o sorgente) dei dati o i diversi utenti che fanno uso di uno stesso host

# Strato di Trasporto



Prof. F. De Rango



# Strato di Trasporto



Prof. F. De Rango

- L'indirizzo IP identifica l'host e non gli utenti o i processi attestati all'host; l'indirizzo usato dai protocolli di trasporto è il numero di porta
- Ogni elaboratore contiene un insieme di punti logici di accesso/destinazione detti “porte”; ogni porta è individuata da un intero positivo; è il sistema operativo a mettere in corrispondenza ogni porta con il relativo processo
- In terminologia OSI, una porta non è altro che un SAP dello strato 4 situato tra gli strati di trasporto e applicativo, che identifica univocamente una specifica entità di destinazione responsabile del processo di destinazione

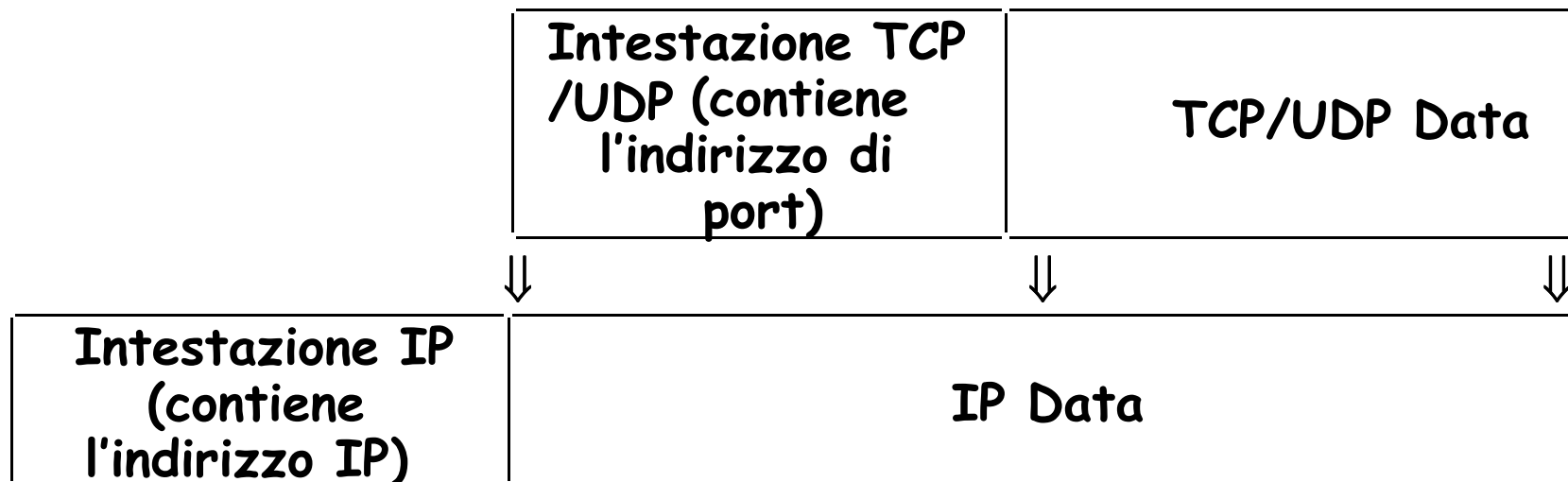
# Strato di Trasporto



- L'indirizzo completo TCP/IP è costituito dall'insieme di indirizzo IP e del numero di porta e identifica univocamente un processo in esecuzione su un host
- La divisione dei compiti fra lo strato di trasporto (UDP, TCP) e IP è la seguente:
  - ✗ lo strato IP si occupa del trasferimento dei dati fra elaboratori collegati alle reti interconnesse; quindi l'intestazione IP identifica gli host sorgente e destinazione
  - ✗ lo strato UDP (TCP) si occupa dello smistamento dei dati fra sorgenti o destinazioni multiple all'interno dello stesso host tramite il numero di porta

# Strato di Trasporto

La componente "port" è contenuta nell'intestazione del livello di trasporto, mentre la componente IP\_Address è contenuta nell'intestazione dell'unità dati di IP





# Porte TCP e UDP



Il numero di porta è il mezzo con cui un programma client indirizza un programma server; dunque, per richiedere un certo servizio, fornito da un processo (server) residente su un host remoto, un applicativo client deve aprire una connessione con la macchina di destinazione sulla porta server che individua quel particolare servizio

- ✗ ad es. un client FTP, per connettersi ad un FTP server, deve conoscere e indicare l'indirizzo IP dell'elaboratore remoto e il numero della porta associata al servizio FTP

# Porte TCP e UDP



- La porta destinazione (server port) individua il particolare servizio che l'applicativo client sta richiedendo all'applicativo server. Il valore del server port deve essere univocamente assegnato al servizio ed ha valore globale, nel senso che deve essere conosciuto da tutti i client che a tale servizio vogliono accedere
- La porta sorgente (client port) permette allo stesso client di attivare diverse connessioni o sessioni dello stesso servizio, (è quello che accade ad esempio su un PC quando si aprono diverse finestre del browser verso lo stesso sito Web). Il port sorgente viene scelto dall'applicativo client ed ha un valore solo locale.

# Porte TCP e UDP



**I numeri di porta possono essere assegnati in due modi:**

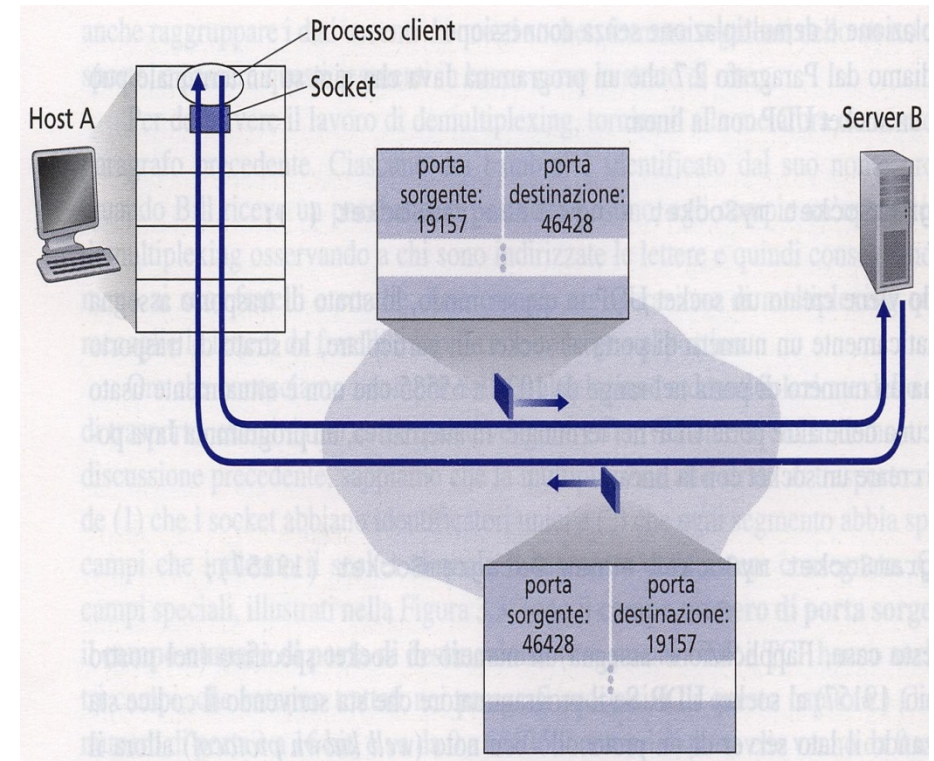
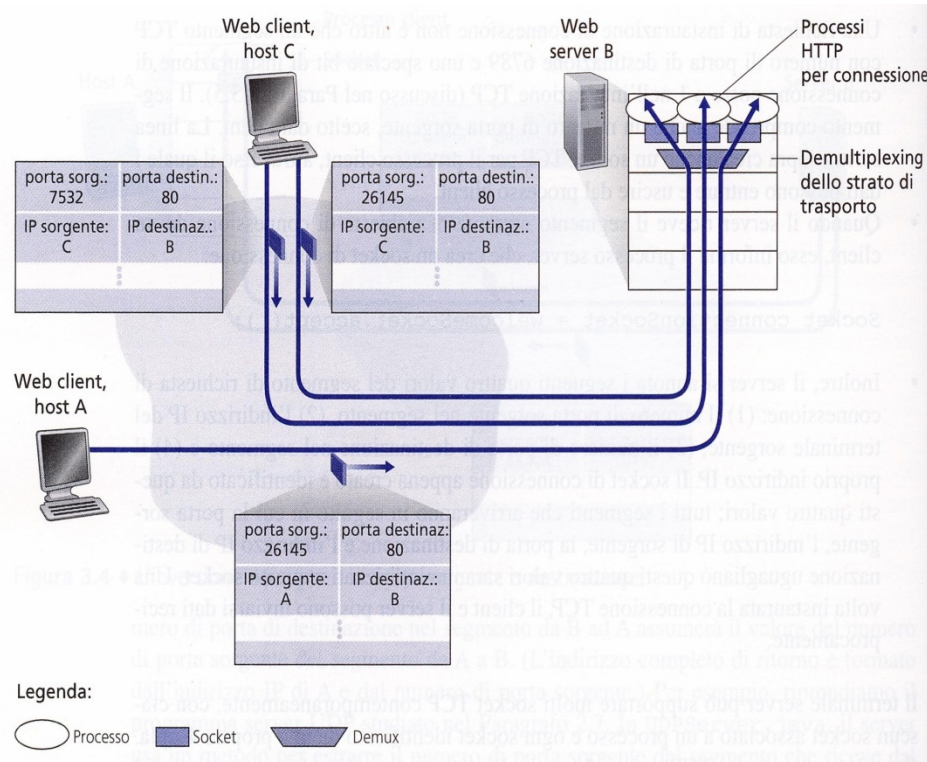
- ✗ **assegnazione universale:** assegnati dall'IANA in maniera globale e univoca a determinati processi applicativi (well-known ports)
  - ✗ **assegnazione dinamica:** assegnati dal processo applicativo sorgente dopo una fase di negoziazione col processo applicativo destinazione e validi per una determinata sessione
- **La porta è individuata da un numero naturale di 16 bit; lo spazio di numerazione è diviso in due gruppi:**
    - ✗ **da 0 a 1023 (?256)** è lo spazio riservato per le porte privilegiate o **well known port** per indirizzare un certo servizio
    - ✗ **da 1024 a 65535** è lasciato libero per le **porte utenti**, cioè quelle scelte dall'applicativo client come porta sorgente

# Università della Calabria D.E.I.S.

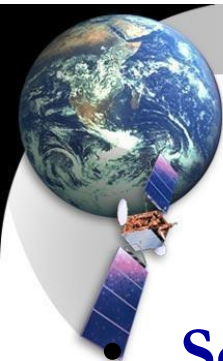
## Porte TCP e UDP



Prof. F. De Rango



## Esempi di utilizzo delle porte



# Well Known Port

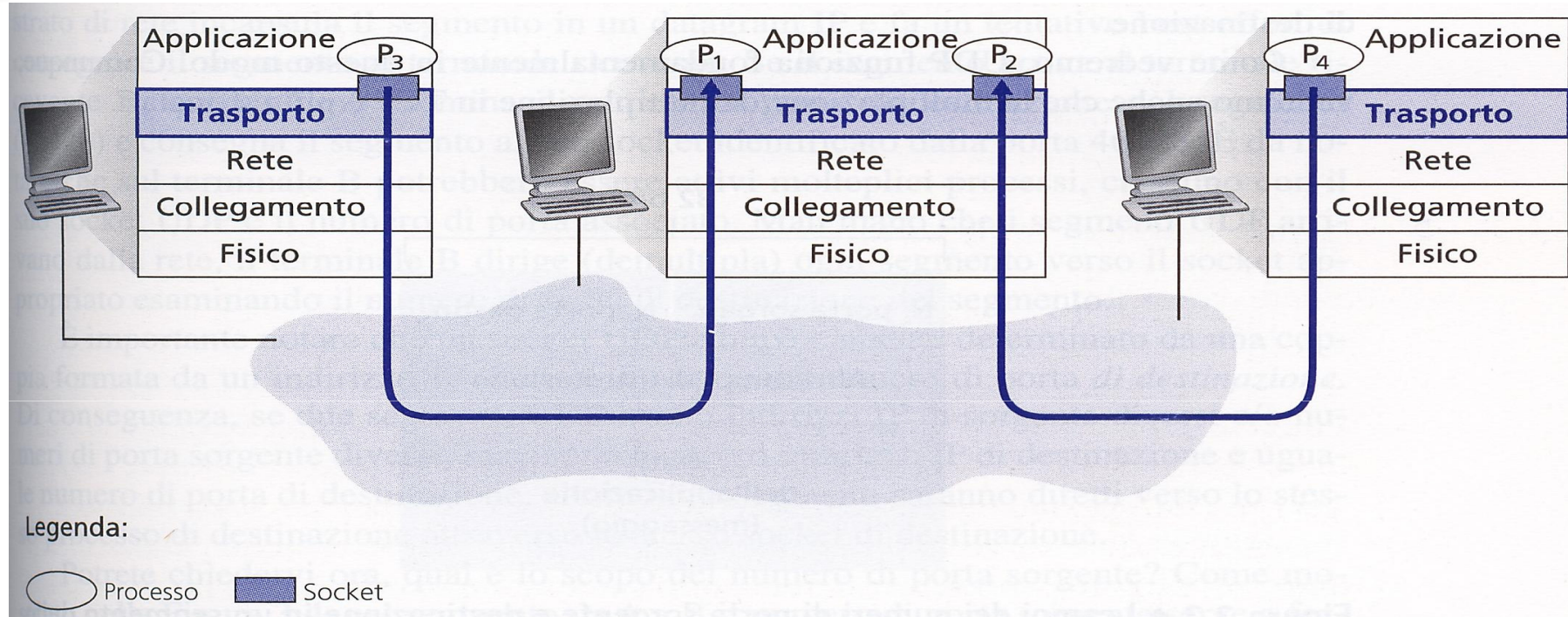
Sono associate agli applicativi principali

Prof. F. De Rango

Servizio	Porta	TCP	UDP
FTP	21		
Telnet	23		
SMTP	25		
TFTP	69		
DNS	53		
HTTP	80		
SNMP	161		



# Strato di Trasporto



Multiplexing e demultiplexing a livello trasporto



# **Il protocollo UDP**

## **(User Datagram Protocol)**

### **RFC 768**

# Protocollo UDP: funzioni



- E' il protocollo di trasporto più semplice in grado di usare il servizio di comunicazione e le funzionalità di IP facendo colloquiare processi remoti
- UDP aggiunge due funzionalità a quelle di IP:
  - ✗ l'indirizzamento delle applicazioni, cioè il de/multiplexing delle informazioni tra le varie applicazioni tramite il concetto di porta
  - ✗ un blando controllo d'errore sull'header dei messaggi, cioè una checksum (opzionale) per verificare l'integrità dei dati



# Protocollo UDP: servizio



- UDP è un protocollo che fornisce un servizio di tipo datagram, che non garantisce la consegna e che non esercita nessun controllo sul flusso e riordinamento delle unità informative emesse dall'applicazione:
  - ✗ connectionless (pacchetti fuori sequenza)
  - ✗ non affidabile (pacchetti persi)
  - ✗ senza controllo di flusso (saturazione del ricevitore)
- Non prevede meccanismi di recupero da errore, es. ritrasmissioni in caso di errori/perdite
  - ✗ eventuali meccanismi di ritrasmissione (se necessari) vengono gestiti direttamente dall'applicazione



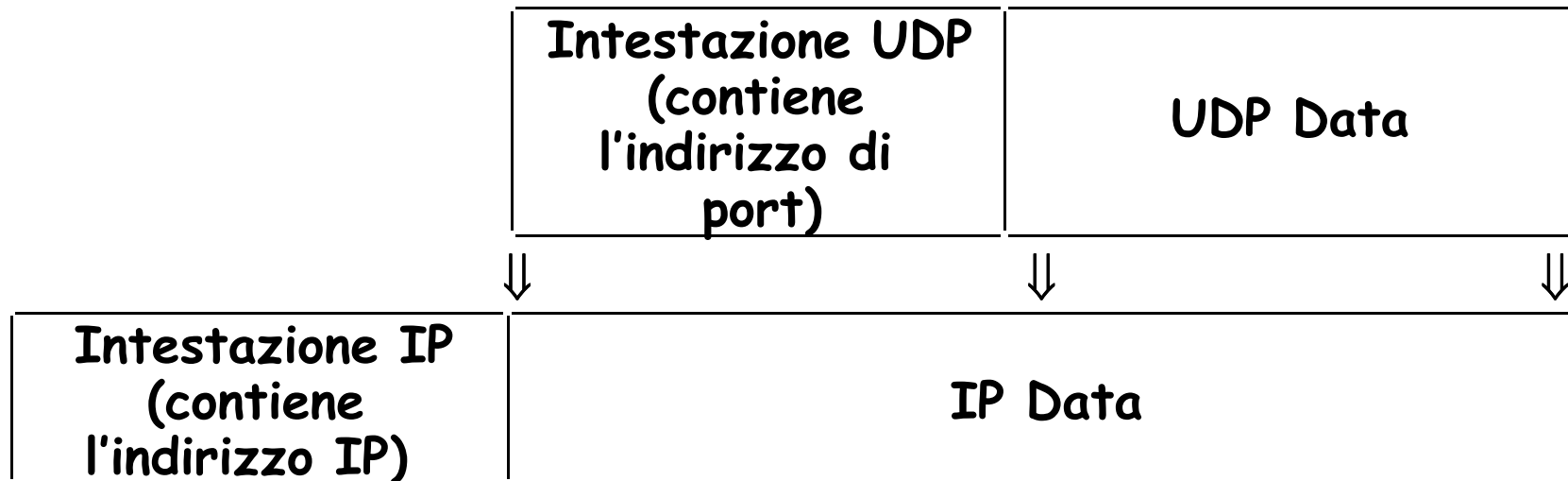
# Protocollo UDP: applicazioni

- UDP è usato da quegli applicativi che non necessitano di un trasferimento affidabile e per i quali l'overhead dovuto alla fase di apertura di un servizio di trasporto orientato alla connessione non sarebbe giustificato. Tra questi: DNS, NFS, SNMP, RIP, ecc.
- Inoltre, UDP è usato dai servizi che non possono tollerare il controllo di flusso del TCP. Tra questi, i servizi di trasporto di flussi *stream* come voce o video. In questo caso, di solito, alle funzionalità di UDP vengono aggiunte quelle del protocollo RTP (Real Time Protocol), che ha come compito principale quello di aggiungere all'header UDP le funzionalità di numerazione dei pacchetti e di time-stamp



# Pacchetto UDP

UDP accetta dal livello superiore dati senza vincoli sulla loro lunghezza, eventualmente li frammenta e li invia in datagrammi IP distinti

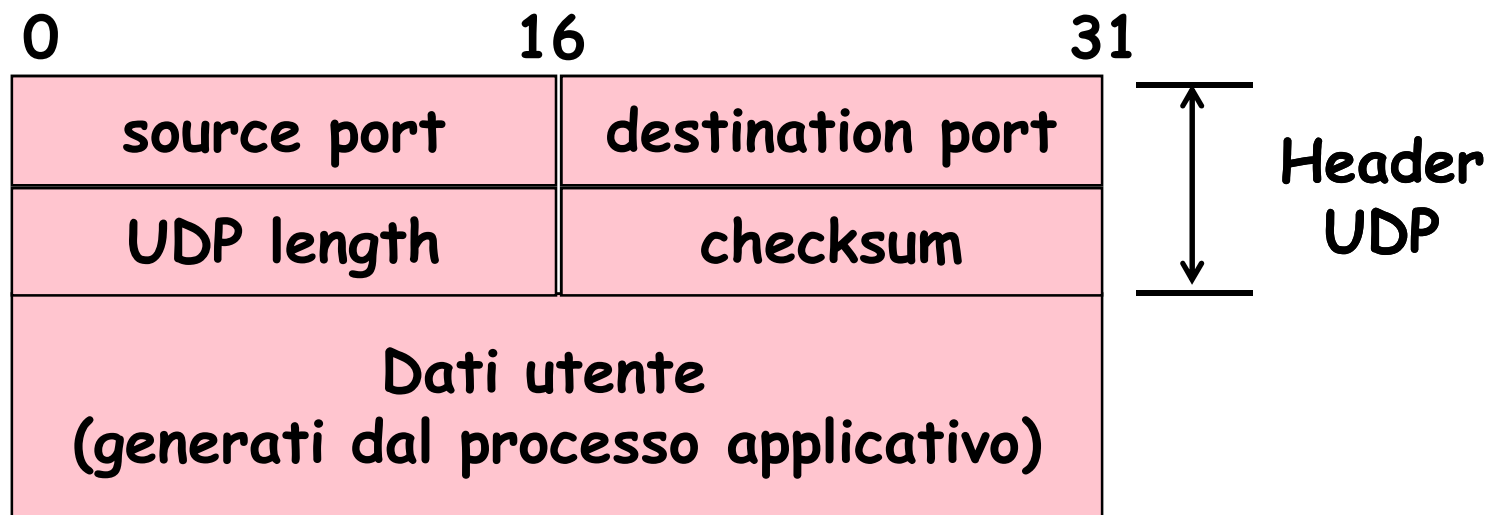




# Pacchetto UDP

L'unità dati UDP (datagramma utente) ha lunghezza variabile, viene imbustata in IP ed indirizzata con il campo Protocol pari a 17

- L'intestazione di UDP è lunga 8 byte, contro i 20 byte dell'intestazione TCP



# Pacchetto UDP



- Port number (16 bit) indirizzi delle porte sorgente e destinazione
- Length (16 bit) è la lunghezza in byte del datagramma UDP (header + dati); il minimo valore è di 8 byte, quando la parte dati è vuota
  - ✗ l'informazione è ridondante, visto che l'header UDP ha lunghezza fissa di 8 byte, la lunghezza della parte dati potrebbe essere ricavata sottraendo 8 byte al contenuto del campo length dell'header IP
- Checksum (16 bit), campo opzionale per il controllo di errore; quando non si usa (in reti altamente affidabili) si riduce il carico di processamento di un datagramma; però dato che IP non fa controllo di errore, la checksum è l'unico strumento per verificare che i dati siano giunti a destinazione correttamente

# UDP: controllo di errore



- La checksum copre tutto il datagramma utente UDP (header e dati) e anche il cosiddetto "pseudo-header", che è considerato solo ai fini del calcolo della checksum ma non viene trasmesso a destinazione
- Lo pseudo-header è costituito da alcuni campi dell'header IP: indirizzi IP sorgente e destinazione (32 bit); campo Protocol (8 bit) che identifica il protocollo UDP; più la lunghezza del datagramma utente UDP (16 bit) e 8 bit di Padding per avere una lunghezza totale che è un multiplo di 16 bit

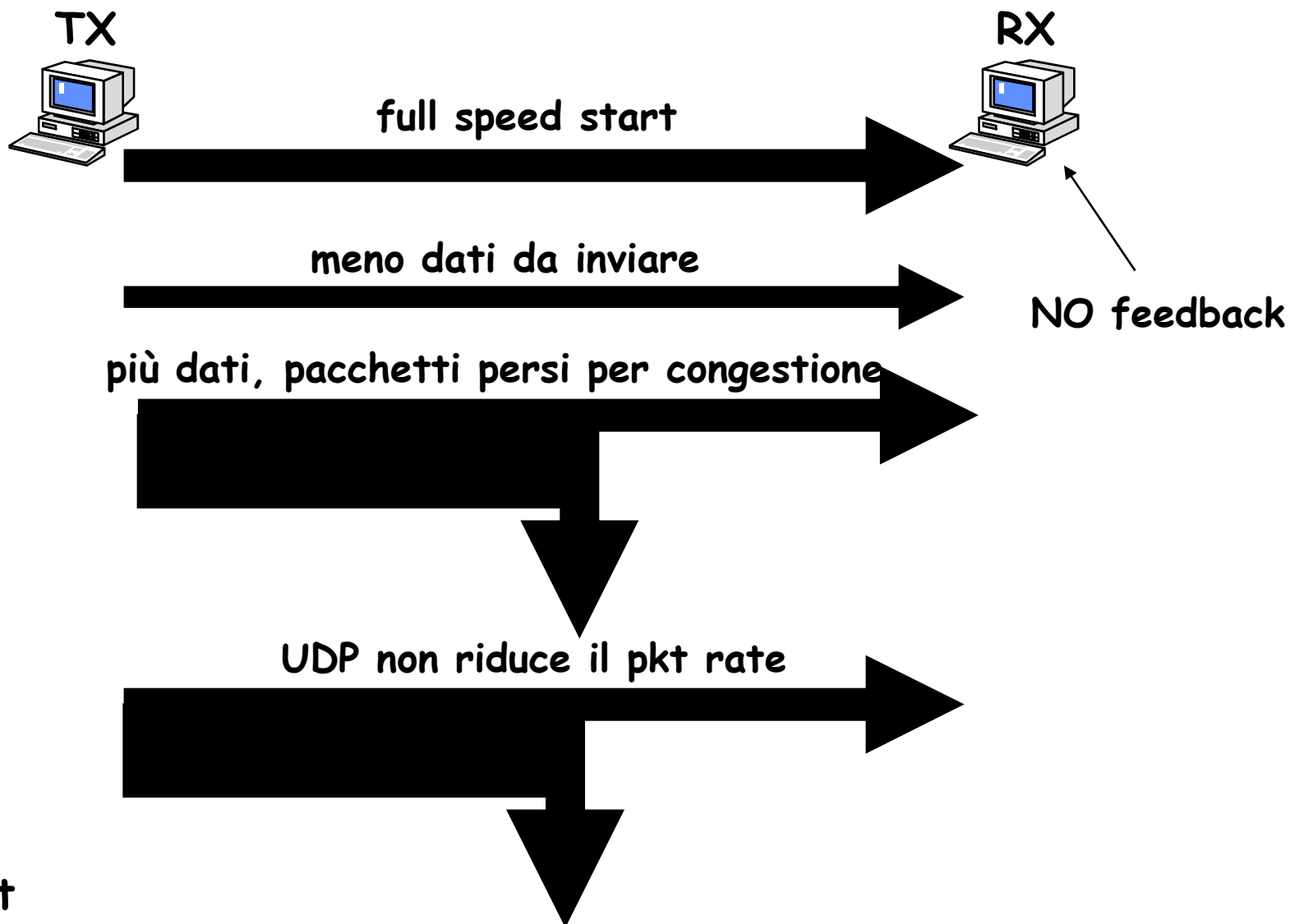
# UDP: controllo di errore



0	8	16	24	31
Source IP address				
Destination IP address				
Padding	Protocol	UDP length		

- La checksum applicata allo pseudo-header permette di verificare che il datagramma sia giunto alla destinazione corretta; il solo controllo sull'header UDP non fornirebbe questa garanzia perché non contiene l'informazione sul sistema di destinazione

# UDP: congestione







# **Il protocollo TCP (Transmission Control Protocol)**

**Definizione: RFC 793**

**Identificazione di bug: RFC 1122**

**Estensioni : RFC 1323**



# Transmission Control Protocol (TCP)

- TCP è un protocollo “con connessione” che fornisce un servizio affidabile “end-to-end” tra coppie di processi su host remoti
- TCP effettua funzioni di:
  - × indirizzamento di uno specifico utente all'interno di un host (multiplazione e de-multiplazione delle informazioni)
  - × trasferimento di un flusso informativo continuo e bi-direzionale (byte stream), ma non strutturato, di dati tra host remoti (funzione di segmentation & reassembly)
  - × gestione delle connessioni
  - × controllo e recupero di errore
  - × controllo di flusso
  - × controllo di congestione
  - × riordinamento delle unità informative

# Indirizzamento TCP



## • Multiplexing:

per permettere ai processi in un Host di usare contemporaneamente le facilità di comunicazione di TCP, il TCP fornisce un set di indirizzi o porte in ogni host

- L'indirizzo completo TCP/IP è costituito dall'insieme di indirizzo IP e numero di porta e identifica univocamente un processo in esecuzione su un host
- Tale indirizzo viene spesso indicato con il nome di “socket” ed è costituito da:

`port@IP_Address = port@Host_Id.Net_Id`

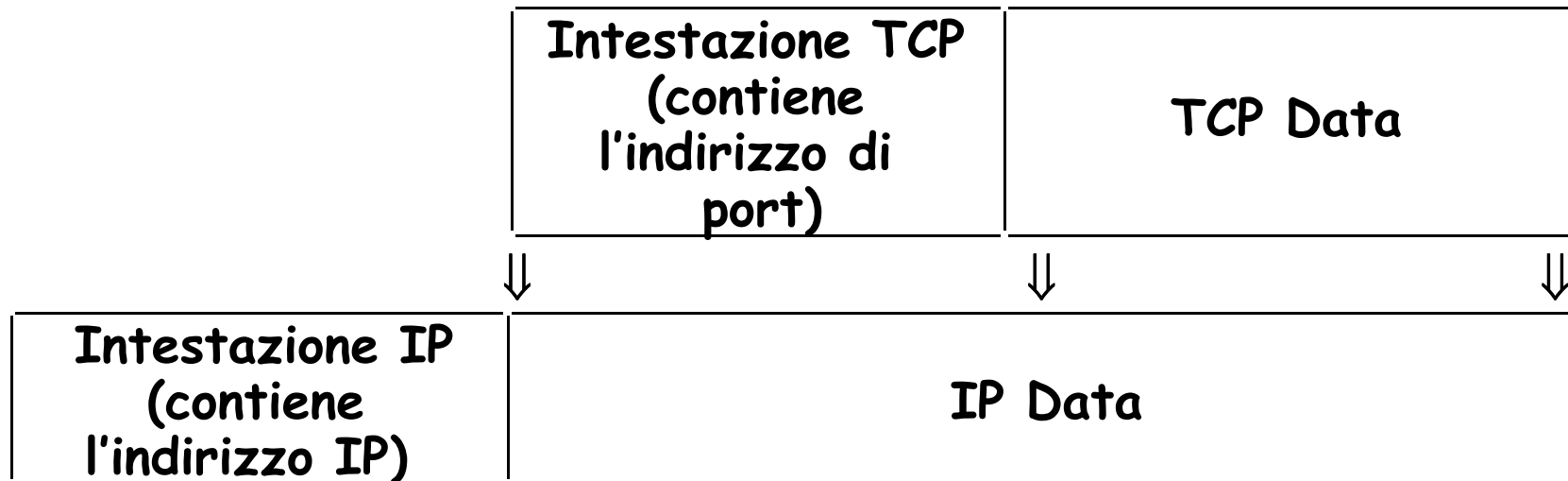
Es.: `derango@telecom.deis.unical.it`

`derango@160.97.25.95`

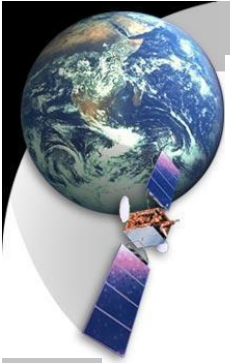
# Indirizzamento TCP



La componente "port" è contenuta nell'intestazione dell'unità dati di TCP, mentre la componente IP\_Address è contenuta nell'intestazione dell'unità dati di IP



# Indirizzamento TCP



Prof. F. De Rango

- Questo significa che tutte le connessioni in atto tra due specifici host usano gli stessi indirizzi IP di sorgente e di destinazione. Saranno perciò distinte solo a livello TCP
- Ne segue che queste connessioni possono essere viste come multiplate su un unico indirizzo IP, ovvero su un unico “canale” IP di comunicazione (non su una connessione IP, dato che IP è connectionless)

# Indirizzamento TCP



Prof. F. De Rango

- Una connessione TCP è identificata dalla coppia di socket (sorgente e destinazione) associata ai due processi (end-point) che hanno stabilito la connessione
- Un numero di porta può essere usato per più connessioni, ma l'insieme dei socket di sorgente e destinazione identifica univocamente una connessione
  - ad es., un server web può avere più connessioni contemporaneamente attive sulla porta 80 e le distingue sulla base dell'indirizzo IP e del numero di porta dei client

# Indirizzamento TCP



- Un end-point può essere impegnato allo stesso tempo in più connessioni TCP

Prof. F. De Rango

(21; 151.100.37.13)

(21; 128.10.2.3)

(21; 18.26.0.36)

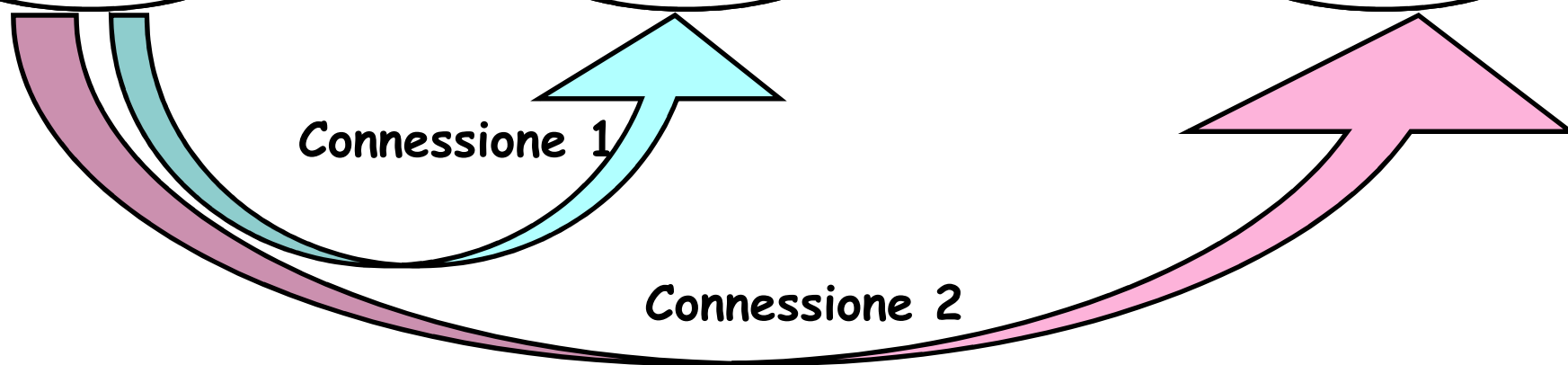
End-point  
A

End-point  
B

End-point  
C

Connessione 1

Connessione 2



# Indirizzamento TCP



- L'assegnazione del numero di porta può essere:

× statico

- l'identificativo è staticamente associato all'applicazione
- sono utilizzati identificativi inferiori a 255

Numero	Applicazione	Numero	Applicazione
7	Echo	37	Time
21	FTP	53	Domain Name Server
23	TELNET	103	X400 Mail Service
25	SMTP	119	NNTP (USENET New Transfer Prot.)



# Indirizzamento TCP



× dinamico

- l'identificativo è assegnato direttamente dal sistema operativo al momento dell'apertura della connessione

- Per rendere user-friendly l'utilizzo delle interfacce utente, alcuni processi applicativi effettuano l'indirizzamento automatico delle porte

- se si vuole attivare una sessione telnet sarà l'applicativo ad indirizzare automaticamente i dati alla porta 23

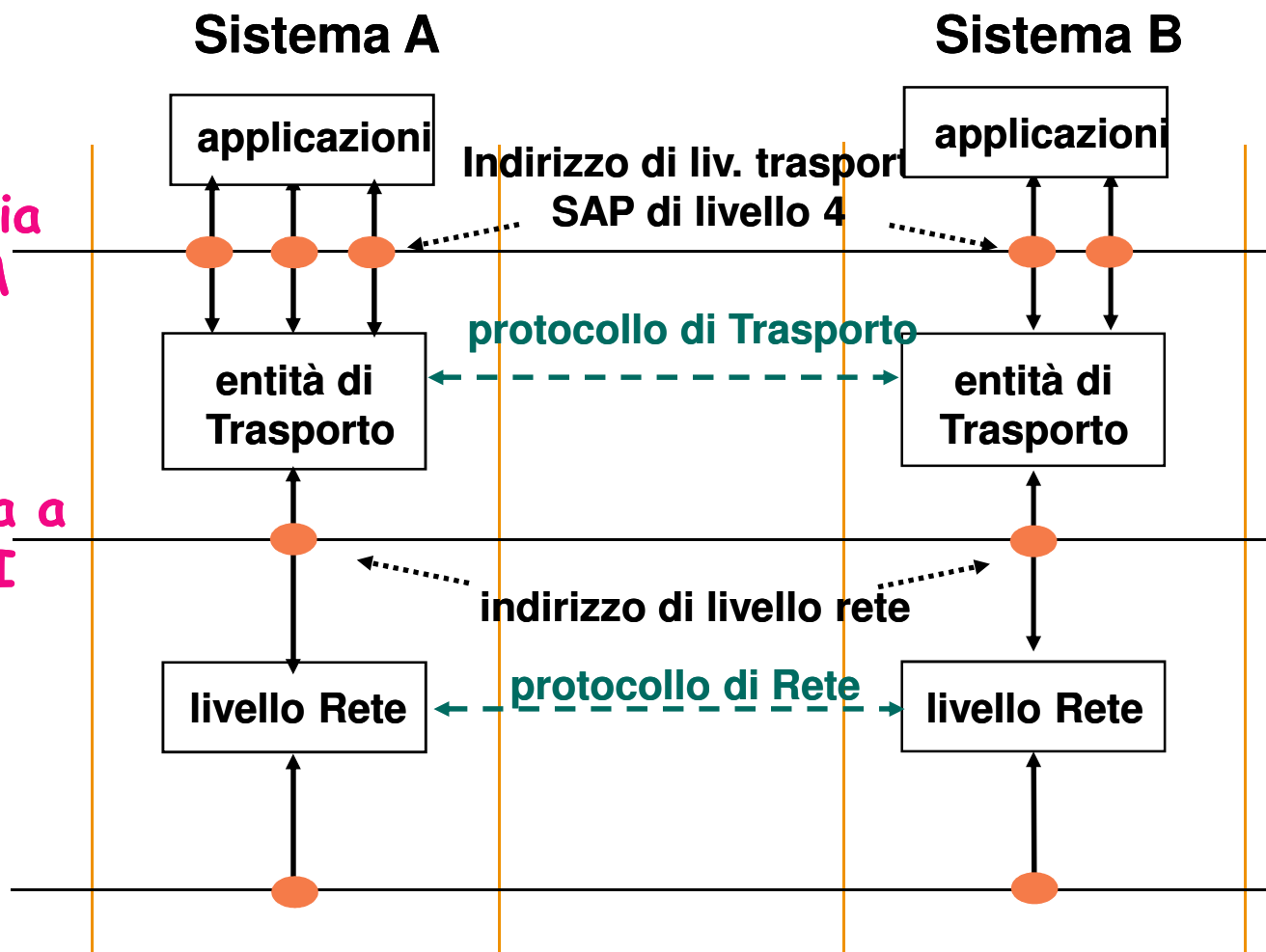
# Trasferimento dati TCP



Prof. F. De Rango

Interfaccia  
STREAM

Interfaccia a  
BLOCCHI





# Trasferimento dati TCP

- TCP accetta dal livello applicativo un flusso continuo di dati non strutturati (byte stream), li frammenta e li invia in unità dati distinte, detti segmenti
- La lunghezza massima dei segmenti viene negoziata durante la fase di apertura della connessione (MSS – Maximum Segment Size) e comunque è inferiore a 64 kbyte, dato che il segmento deve poter utilizzare il payload di un unico pacchetto IP, e dipende dall'MTU



# Trasferimento dati TCP

- Il TCP utilizza dei buffer software per immagazzinare i byte di dati finché si costruisce il segmento
- Tipicamente, è il TCP che decide quando un segmento va trasmesso, anche se l'applicazione ha dei mezzi (la funzione push) per forzare la trasmissione di segmenti prima che il TCP intervenga in modo automatico
- Anche dal lato ricevente il TCP raccoglie i byte in un buffer, che successivamente (quando il buffer si riempie o un URG flag viene ricevuto) trasmette al livello superiore sottoforma di un flusso continuo di byte; l'applicazione ricevente assorbe i byte dal buffer in base alla sua velocità di elaborazione dell'informazione



# Trasferimento dati TCP

- TCP è un protocollo orientato al byte e non al messaggio, quindi i confini tra i messaggi non devono essere mantenuti in ricezione
  - cioè, se il processo sorgente fa un'operazione di scrittura di 4 blocchi di 512 byte ciascuno su un flusso TCP, il processo ricevente può ricevere i dati in 4 blocchi di 512 byte, o in 2 blocchi di 1024 byte o in 1 blocco da 2048 byte, o in altri modi
- Ogni byte è numerato con un numero di sequenza di 32 bit



# Trasferimento dati TCP

In generale, il riempimento dei buffer dei TCP di trasmissione e ricezione determina il trasferimento dei dati, però il livello applicazione ha modo di intervenire per forzare l'invio di alcuni dati

**Sender Application -> Sender TCP (PUSH)**

l'applicazione sorgente può richiedere al TCP sorgente l'invio immediato dei dati verso il TCP destinazione, in questo caso l'applicazione usa il flag PUSH

- Es. se un utente è collegato in telnet su un terminale remoto, dopo aver digitato una linea e premuto il tasto di Carriage Return, è necessario che la linea venga immediatamente inviata all'host remoto senza bufferizzarla in attesa della linea successiva

# Dati urgenti

## Sending Application/TCP – Receiving TCP/Application (URG)

- A volte è necessario inviare dei dati urgenti (“out of band”) senza aspettare che l’entità ricevente finisca di elaborare i dati precedentemente trasmessi (molti dati possono essere in viaggio, nei router sul path e nella coda di input dell’host remoto)
  - ad esempio, in una sessione Telnet, un utente potrebbe inviare un segnale di “interrupt” (DEL o CTRL-C) che termini immediatamente l’applicazione remota
  - il segnale di interrupt deve poter essere inviato senza aspettare che l’host remoto elabori tutti i dati già inviati



# Dati urgenti

I “dati urgenti” hanno priorità su tutti gli altri dati già inviati e vengono trasmessi al processo remoto immediatamente; quando i dati urgenti sono stati elaborati, il processo remoto riprende in esame i dati “normali”

Il meccanismo usato dal TCP per i dati urgenti consiste:

- l'applicazione sorgente “setta” il bit URG del segmento per avvertire il TCP sender di smettere di accumulare i dati e inviare “immediatamente” tutto quello che ha bufferizzato verso il TCP destinazione
- appena i dati urgenti sono ricevuti dal TCP remoto, l'applicazione ricevente viene interrotta (riceve un “signal” in terminologia UNIX) in modo che smetta qualsiasi operazione e legga i dati urgenti
- la fine dei dati urgenti è marcata dall'applicazione sender che inserisce nel campo Urgent Pointer il numero di sequenza dell'ultimo byte di dati urgenti







# Trasferimento dati TCP

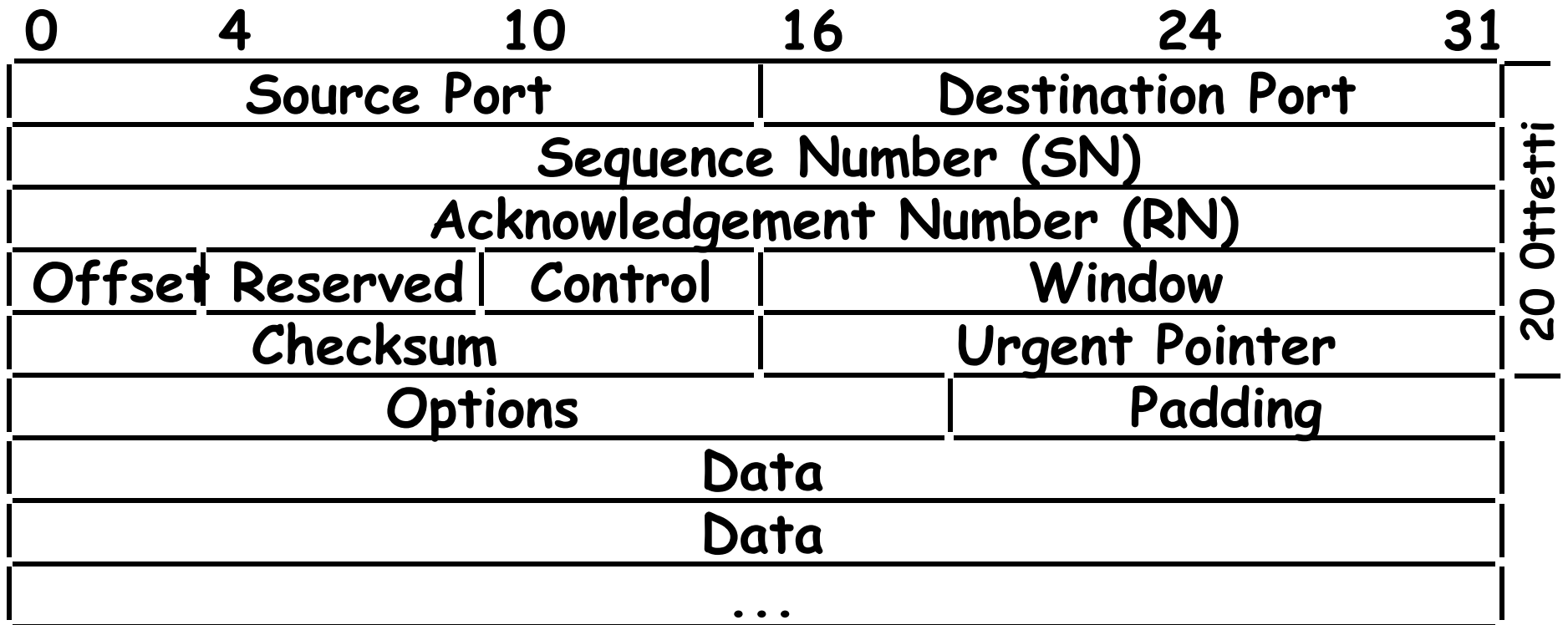
Prof. F. De Rango

- Le operazioni di segmentazione e riassetramento possono avvenire in parallelo, cioè il TCP è un protocollo full-duplex punto-punto
- Il trasferimento dati TCP avviene secondo un meccanismo di sliding window: all'invio di un segmento il sender fa partire un timer, quando il segmento arriva a destinazione il receiver invia un segmento con il riscontro del segmento ricevuto; se il timer del sender scade prima della ricezione del riscontro il sender ritrasmette il segmento
- Il TCP utilizza un unico formato di trame sia per la trasmissione di informazione d'utente che per l'informazione di servizio (apertura e chiusura della connessione, messaggi per il controllo d'errore e quello di flusso)



# Formato dell'unità dati TCP

- L'header è lungo 20 byte se il campo opzioni non viene utilizzato. Il campo dati può essere vuoto (trame di acknowledgment, connessione, ecc.).





# Formato dell'unità dati

- **Source Port (16 bit):** definisce l'indirizzo logico del processo sorgente dei dati
- **Destination Port (16 bit):** definisce l'indirizzo logico del processo destinatario dei dati
- **Sequence Number (32 bit):** numero di sequenza in trasmissione (SN); contiene il numero di sequenza del primo byte di dati contenuti nel segmento a partire dall'inizio della sessione (se  $SN=m$  ed il segmento contiene  $n$  byte il prossimo SN sarà pari a  $m+n$ )
- **Acknowledgement Number (32 bit):** numero di sequenza in ricezione (RN); nei segmenti in cui il bit ACK è 1, contiene il numero di sequenza del prossimo byte che il ricevitore si aspetta di ricevere. Il meccanismo di riscontro usato è cumulativo, così l'ack di un SN  $X$  indica che sono stati ricevuti tutti i byte fino a  $X$  escluso  $X$ . In caso di connessioni interattive bidirezionali gli ack sono inviati in piggybacking (nei segmenti di risposta contenenti dati di utente). I numeri SN e RN vengono utilizzati per il controllo d'errore e di flusso



# Formato dell'unità dati: SN e Ack

Prof. F. De Rango

I numeri di sequenza e gli ack rendono affidabile la trasmissione TCP. Ad ogni byte di dati si assegna un numero di sequenza. In ogni segmento TCP si inserisce il numero di sequenza del primo byte di dati contenuto nel segmento (SN)

- I segmenti in direzione opposta portano anche un numero di acknowledgment che è il numero di sequenza del successivo byte di dati da trasmettere atteso dal ricevitore
- Quando il TCP trasmette un segmento dati, ne conserva una copia in una coda di ritrasmissione e fa partire un timer; quando riceve l'ack per quei dati, allora cancella il segmento dalla coda. Se l'ack non è ricevuto prima della scadenza del time-out, il segmento viene ritrasmesso
- Così il TCP mantiene la corretta sequenza dei segmenti in ricezione; cioè prima di inviare una nuova sequenza di byte aspetta che la sequenza precedente venga riscontrata

# Formato dell'unità dati

- **Offset (4 bit):** contiene il numero di parole di 32 bit contenute nell'intestazione TCP (da Source port a Padding). L'intestazione TCP non supera i 64 byte ed è un multiplo di 32
- **Reserved (6 bit):** riservato per usi futuri, contiene zeri
- **Control bit (6 bit):** i bit di controllo sono 6:
  - ✗ **URG:** vale 1 quando il campo Urgent Pointer contiene un valore significativo; è usato per indicare dati urgenti che vengono trasmessi al di fuori del controllo di flusso con un meccanismo di segnalazione end-to-end tra processi remoti
  - ✗ **ACK:** vale 1 quando il campo Acknowledgement Number contiene un valore significativo
  - ✗ **PSH:** vale 1 quando l'applicazione esige che i dati forniti vengano trasmessi e consegnati all'applicazione ricevente prescindendo dal riempimento dei buffer allocati fra applicazione e TCP e viceversa (di solito il riempimento dei buffer scandisce la trasmissione e la consegna dei dati)





# Formato dell'unità dati

- **RST**: vale 1 quando un malfunzionamento impone il reset della connessione
- **SYN**: indica l'inizio della connessione; vale 1 solo nel primo segmento inviato durante la fase di sincronizzazione fra le entità TCP (3-way handshaking)
- **FIN**: indica la fine della connessione; vale 1 quando la sorgente ha esaurito i dati da trasmettere
- ✖ **Window (16 bit)**: larghezza della finestra per il controllo di flusso; il TCP ricevente riporta al TCP trasmittente il valore di una "window", che contiene il numero di byte che, a cominciare dal valore del campo Acknowledgement Number, il TCP ricevente è disposto a ricevere. Il controllo di flusso è orientato al byte
- ✖ **Checksum (16 bit)**: contiene la sequenza che permette al TCP ricevente di verificare la correttezza del segmento; protegge l'intero segmento più alcuni campi dell'header IP, cioè uno pseudo-header di 96 bit



# Formato dell'unità dati



- La checksum è il complemento a 1 del complemento a 1 della somma di tutte le word da 16 bit nell'header e nel testo; se un segmento contiene un numero dispari di byte, l'ultimo byte è riempito di zeri a destra (padding) per formare una word di 16 bit. Il pad non è trasmesso come parte del segmento
- TCP Length è la lunghezza del TCP header più la lunghezza dei dati in byte (questa quantità non viene esplicitamente trasmessa, ma è calcolata), e non tiene conto dei 12 byte dello pseudo-header

0	8	16	24	31
Source IP address				
Destination IP address				
Padding	Protocol	TCP length		



# Formato dell'unità dati

- **Urgent Pointer (16 bit):** indica i dati urgenti all'interno del segmento; contiene il numero di sequenza dell'ultimo byte di dati che devono essere consegnati con urgenza al processo ricevente. Tipicamente sono messaggi di controllo che esulano dalla comunicazione in senso stretto. A tale traffico ci si riferisce di solito con il nome di out-of-band

**Options (lunghezza variabile):** sono presenti solo raramente; le più note sono End of Option List, No-operation e Maximum Segment Size (di default è 536 byte)

- ✕ **MSS** serve per comunicare al TCP trasmittente la dimensione massima del segmento accettabile in ricezione; il campo è inviato soltanto nella richiesta iniziale di connessione (cioè, nei segmenti col bit SYN settato); se non si usa questa opzione è ammessa una qualsiasi dimensione del segmento

- **Padding (lunghezza variabile):** contiene sempre degli zeri. Serve come riempitivo per far sì che l'intestazione abbia una lunghezza multipla di 32 bit

