



# Reti di Calcolatori

**Prof. Roberto Canonico**

**Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione**

**Corso di Laurea in Ingegneria Informatica**

---

## RTP/RTCP e DASH

**I lucidi presentati al corso sono uno strumento didattico  
che NON sostituisce i testi indicati nel programma del corso**



## Nota di Copyright

Questo insieme di trasparenze è stato ideato e realizzato dai ricercatori del Gruppo di Ricerca COMICS del Dipartimento di Informatica e Sistemistica dell'Università di Napoli Federico II. Esse possono essere impiegate liberamente per fini didattici esclusivamente senza fini di lucro, a meno di un esplicito consenso scritto degli Autori. Nell'uso dovranno essere esplicitamente riportati la fonte e gli Autori. Gli Autori non sono responsabili per eventuali imprecisioni contenute in tali trasparenze né per eventuali problemi, danni o malfunzionamenti derivanti dal loro uso o applicazione.

### Autori:

Simon Pietro Romano, Antonio Pescapè, Stefano Avallone,  
Marcello Esposito, Roberto Canonico, Giorgio Ventre

# Trasferimento di informazioni multimediali su Internet



- Il trasferimento di informazioni multimediali su Internet mediante la tecnica del **file transfer** è tipicamente realizzato adottando il protocollo applicativo HTTP, il quale si appoggia sul protocollo di trasporto TCP
- Per la trasmissione in **streaming** sono adottate due tecniche:
  - mediante un protocollo ad-hoc (RTP) su UDP
  - mediante HTTP su TCP



- RTP sta per “real-time transport protocol”
- definito dal Working Group “Audio/Video Transport” dell’ IETF inizialmente in RFC 1889 (gennaio 1996)
  - <http://www.ietf.org/rfc/rfc1889.txt>
- e successivamente in RFC 3550 (luglio 2003)
  - <http://www.ietf.org/rfc/rfc3550.txt>
- RTP offre un servizio di livello trasporto specificamente progettato per i requisiti di flussi multimediali
- I pacchetti RTP sono incapsulati in datagrammi UDP
  - Un protocollo di livello trasporto su un altro di livello trasporto



- RTP è un protocollo concepito secondo il modello *Application Level Framing* proposto in
  - Clark D., and D. Tennenhouse, “Architectural Considerations for a New Generation of Protocols”, IEEE Computer Communications Review, Vol. 20(4), September 1990.
- è concepito per essere implementato direttamente nelle applicazioni, e non come uno strato aggiuntivo dello stack protocollare
- offre le funzionalità minimali richieste dalla trasmissione di flussi continui tipici delle applicazioni multimediali
- è neutrale rispetto alla codifica utilizzata (MPEG, ecc...)



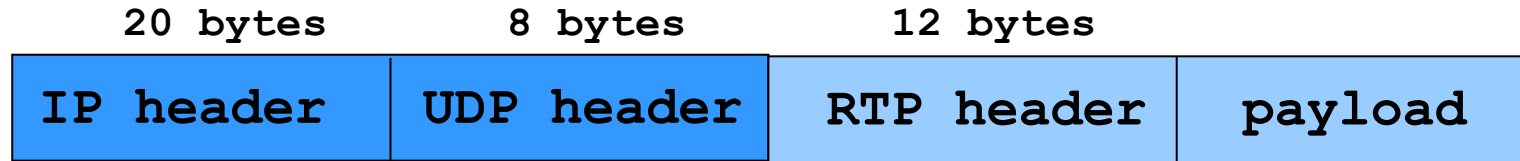
- RTP fornisce informazioni di tempificazione (timestamp) per consentire
  - **sincronizzazione intra-media**: ricostruzione della corretta tempificazione della sequenza di pacchetti ricevuti
  - **sincronizzazione inter-media**: finalizzata a mantenere “al passo” flussi multimediali trasmessi separatamente (es. audio e video: sincronizzazione “lip-sync”)

# RTP (4)



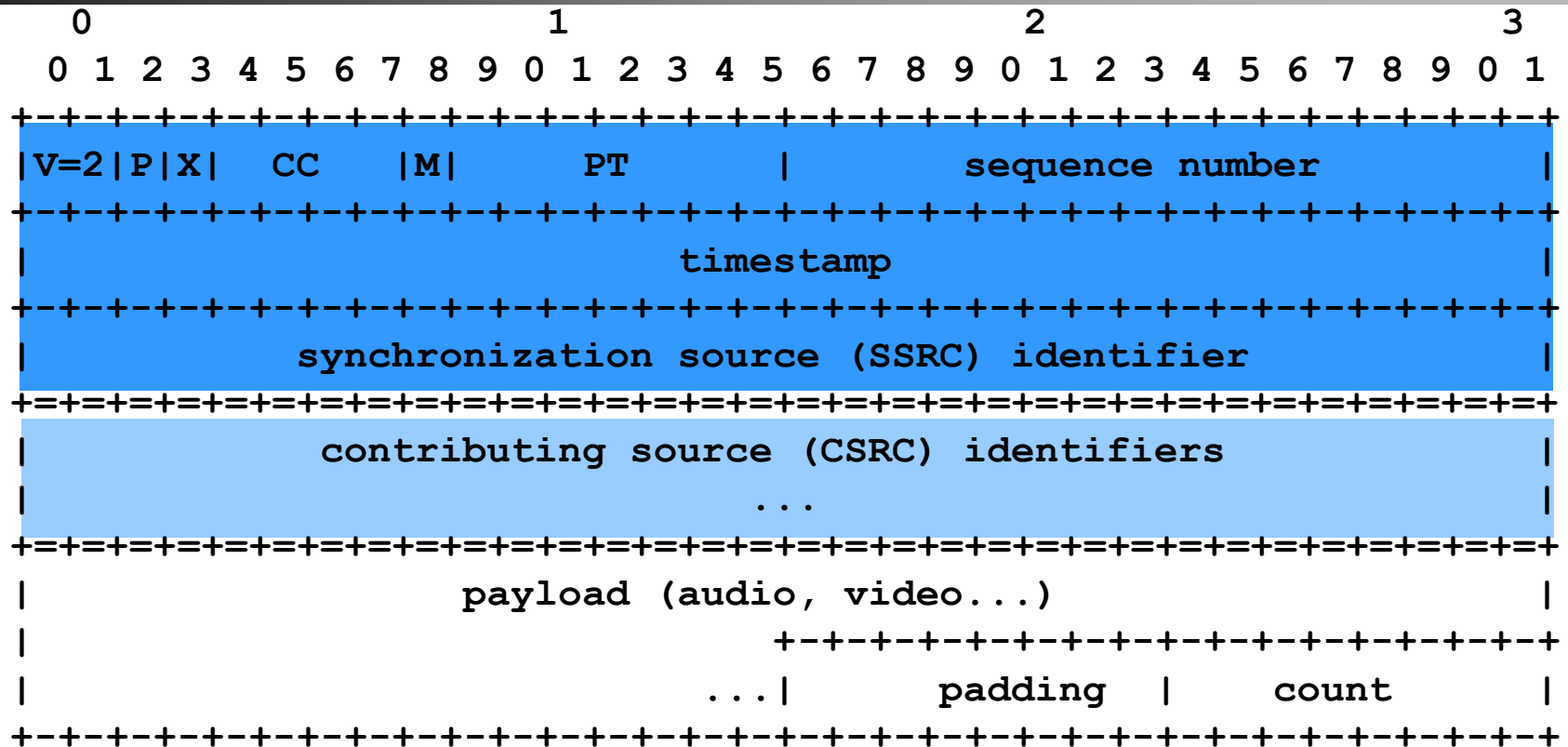
- supporta sia la trasmissione unicast che la trasmissione multicast
- i suoi meccanismi sono scalabili rispetto al numero di appartenenti al gruppo multicast
- separa la trasmissione dei dati dalla trasmissione delle informazioni di controllo
- RTP è definito congiuntamente ad un protocollo di controllo (RTCP) utilizzato per scambiare informazioni di servizio e di controllo sulla qualità della trasmissione
- fornisce informazioni necessarie a combinare flussi di informazioni differenti mediante appositi mixer software

# Incapsulamento di pacchetti RTP



- Un pacchetto RTP è trasmesso in un datagramma UDP
- L'header UDP contiene i numeri di porto sorgente e destinazione
- RTP usa numeri di porto destinazione pari per la trasmissione dei flussi dati
- Se  $2n$  è il numero di porto destinazione usato per uno flusso, il numero successivo  $2n+1$  è usato da RTCP per trasmettere le informazioni di controllo relative a quel flusso

# Header RTP



- Version (V)
- Padding (P)
- eXtension (X)
- Sequence Number
- CSRC Count (CC)
- Marker (M)
- Payload Type (PT)
- Sync. SouRCe (SSRC)



# Campi dell'header RTP

- **Payload Type:** 7 bit, specifica la codifica utilizzata per i dati (PCM, MPEG2 video, ecc.)
- **Sequence Number:** 16 bit, serve ad identificare perdite di pacchetti
- **Timestamp:** 32 bit, specifica il tempo di campionamento del primo byte del payload; serve a rimuovere il jitter introdotto dalla rete mediante bufferizzazione
- **Synchronization Source identifier (SSRC):** 32 bit, identifica la sorgente del flusso, ed è scelto casualmente dalla sorgente stessa; è introdotto per non dover fare affidamento sull'indirizzo IP per identificare la sorgente;
  - problema: sono possibili conflitti ...

# Campi dell'header RTP (2)



- **Contributing Source identifier list (CSRC):**  
sequenza di  $n$  campi da 32 bit ( $0 \leq n \leq 15$ ),  
ciascuno dei quali identifica la sorgente  
originaria in un flusso prodotto dalla “fusione”  
di flussi diversi mediante un mixer software
  - esempio: audio-conferenza a più partecipanti
    - SSRC identifica il mixer
    - CSRC indica lo speaker corrente



# Sessione RTP

- Una associazione tra un gruppo di entità che comunicano mediante RTP
- Alcune applicazioni danno vita a sessioni RTP differenti per media differenti (es. audio e video), a meno che la tecnica di codifica adottata non effettui un multiplexing di flussi differenti in un singolo flusso di dati
- Sessioni RTP differenti (es. audio e video) vengono distinte da un ricevitore mediante il port number di livello trasporto (UDP)



# RTP timestamp e numero di sequenza

- Il valore di timestamp inserito in ogni pacchetto riferisce la tempificazione dei dati inseriti nel payload rispetto ad un clock specifico per il media trasportato
  - Possono essere generati pacchetti RTP consecutivi con lo stesso timestamp
  - Il numero di sequenza identifica un pacchetto rispetto agli altri principalmente per consentire di identificare pacchetti persi
  - non possono essere generati due pacchetti con lo stesso numero di sequenza
-



- Protocollo utilizzato congiuntamente ad RTP per la trasmissione di informazioni di controllo
- I pacchetti RTCP vengono inviati con una certa periodicità e trasportano informazioni di varia natura:
  - feedback sulla qualità della ricezione dei dati (perc. pacchetti persi, ...)
  - identificazione dei partecipanti ad una sessione RTP mediante un identificativo detto CNAME
- Nel caso di trasmissione RTP tra partecipanti ad un gruppo multicast, RTCP consente ad ogni partecipante di conoscere il numero di partecipanti



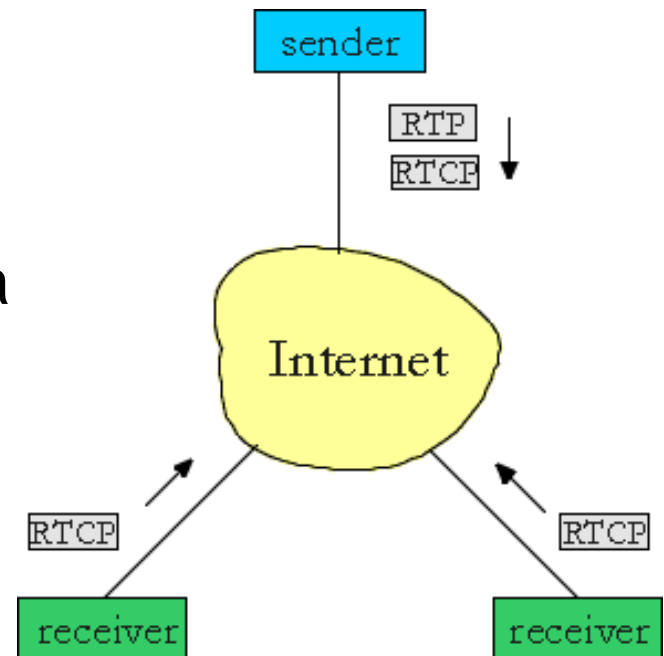
# Messaggi RTCP

- Il protocollo RTCP definisce cinque tipi diversi di messaggi:
  - **Source Report (SR)**
  - **Receiver Report (RR)**
  - **Source Description (SD)**
  - **BYE**
  - **APP**
- I messaggi di tipo *report* contengono statistiche sul numero di pacchetti inviati, numero di pacchetti ricevuti, percentuale di pacchetti persi, jitter dei tempi di interarrivo, ecc. e servono a monitorare la qualità della trasmissione
- I messaggi di tipo *description*, invece, descrivono la sorgente del flusso (contengono tra l'altro il CNAME)
- BYE serve a notificare l'uscita da una sessione
- APP è un tipo di messaggio le cui funzioni sono definibili dall'applicazione



# Banda usata da RTCP

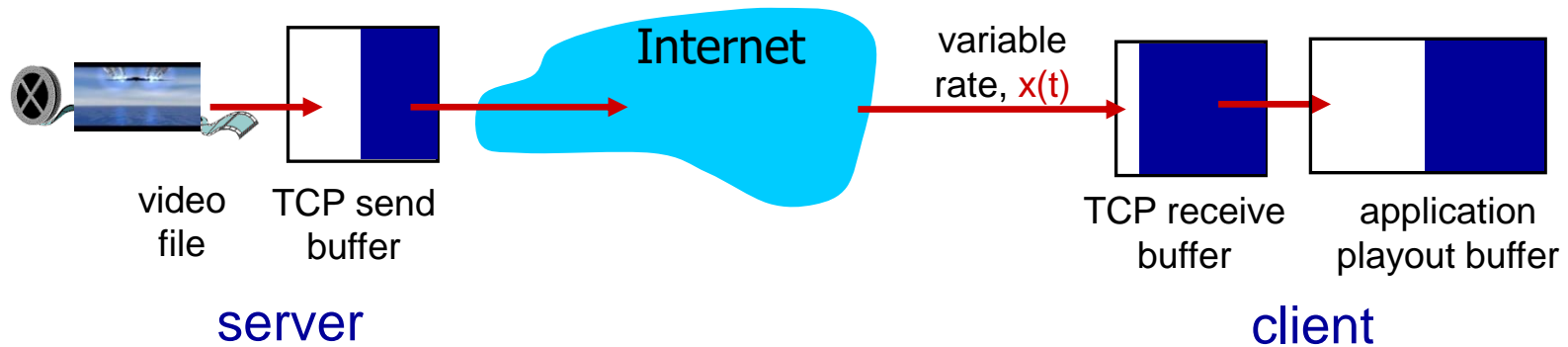
- Nel caso di trasmissione multicast, ciascun ricevitore invia periodicamente (allo stesso gruppo multicast) i report RTCP
- Cosa succede se il numero di membri del gruppo diventa molto elevato ?
- Per contenere il traffico di controllo, si inserisce una minima forma di coordinamento:
- L'intervallo temporale tra due report è proporzionale al numero di partecipanti alla sessione
- in modo che la banda consumata da RTCP non superi il 5% della banda usata dalla sessione



# Streaming di dati multimediali con HTTP



- multimedia file retrieved via HTTP GET
- send at maximum possible rate under TCP



- fill rate fluctuates due to TCP congestion control, retransmissions (in-order delivery)
- larger playout delay: smooth TCP delivery rate
- HTTP/TCP passes more easily through firewalls

# Streaming di dati multimediali con DASH



- *DASH*: *D*ynamic, *A*daptive *S*treaming over *H*TTP
- *server*:
  - divides video file into multiple chunks
  - each chunk stored, encoded at different rates
  - *manifest file*: provides URLs for different chunks
- *client*:
  - periodically measures server-to-client bandwidth
  - consulting manifest, requests one chunk at a time
    - chooses maximum coding rate sustainable given current bandwidth
    - can choose different coding rates at different points in time (depending on available bandwidth at time)



# Streaming di dati multimediali con DASH (2)

- *DASH: Dynamic, Adaptive Streaming over HTTP*
- “*intelligence*” at client: client determines
  - *when* to request chunk (so that buffer starvation, or overflow does not occur)
  - *what encoding rate* to request (higher quality when more bandwidth available)
  - *where* to request chunk (can request from URL server that is “close” to client or has high available bandwidth)