



浙江大学计算机学院  
数字媒体与网络技术

# Digital Asset Management

## 数字媒体资源管理

# 5. Streaming multimedia



任课老师：张宏鑫  
2020-11-09

# Keys of Streaming Media

- Algorithms (\*\*)
- Standards (\*\*\*\*\*)
- Complete End-to-End systems (\*\*\*)
- Research Frontiers(\*)

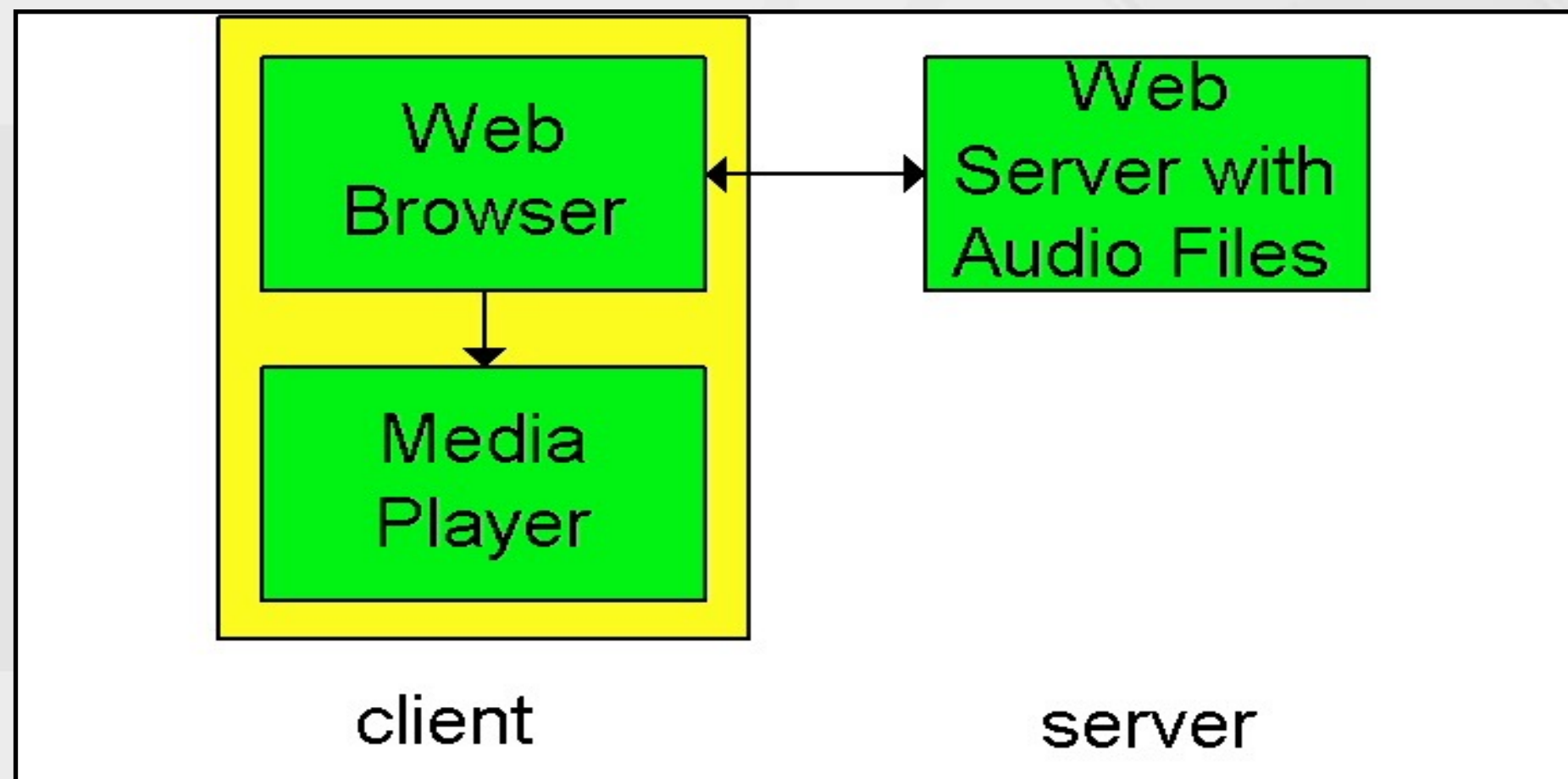


# Streaming ...

- Progressive streaming transport (PST)
  - use HTTP
  - no jump
- Real-time streaming transport
  - Real server (Real-time streaming protocol, RTSP)
  - Windows Media server (M\$ media server, MMS)
  - Quicktime server



# Streaming from Web server (1)

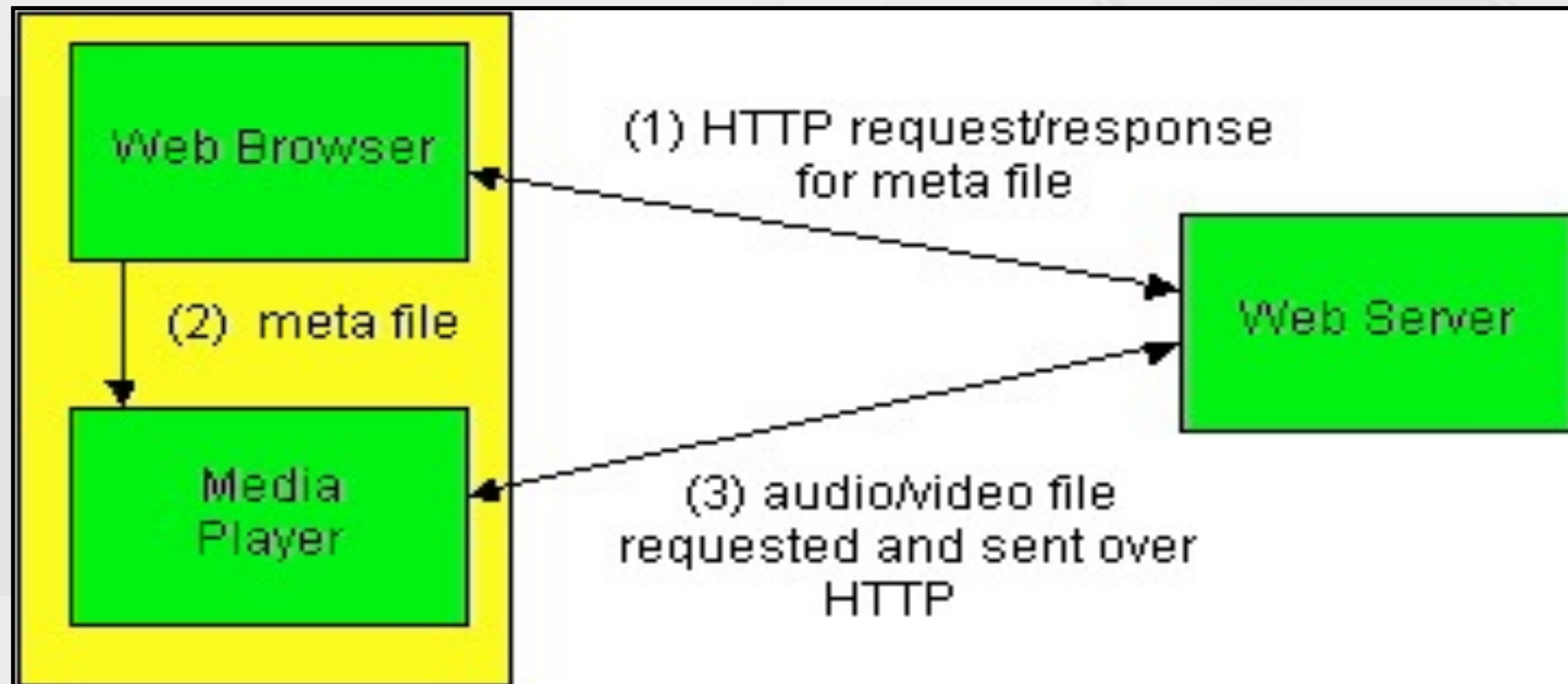


Major drawback:

media player interacts with server through intermediary of a Web browser



# Streaming from Web server (2)

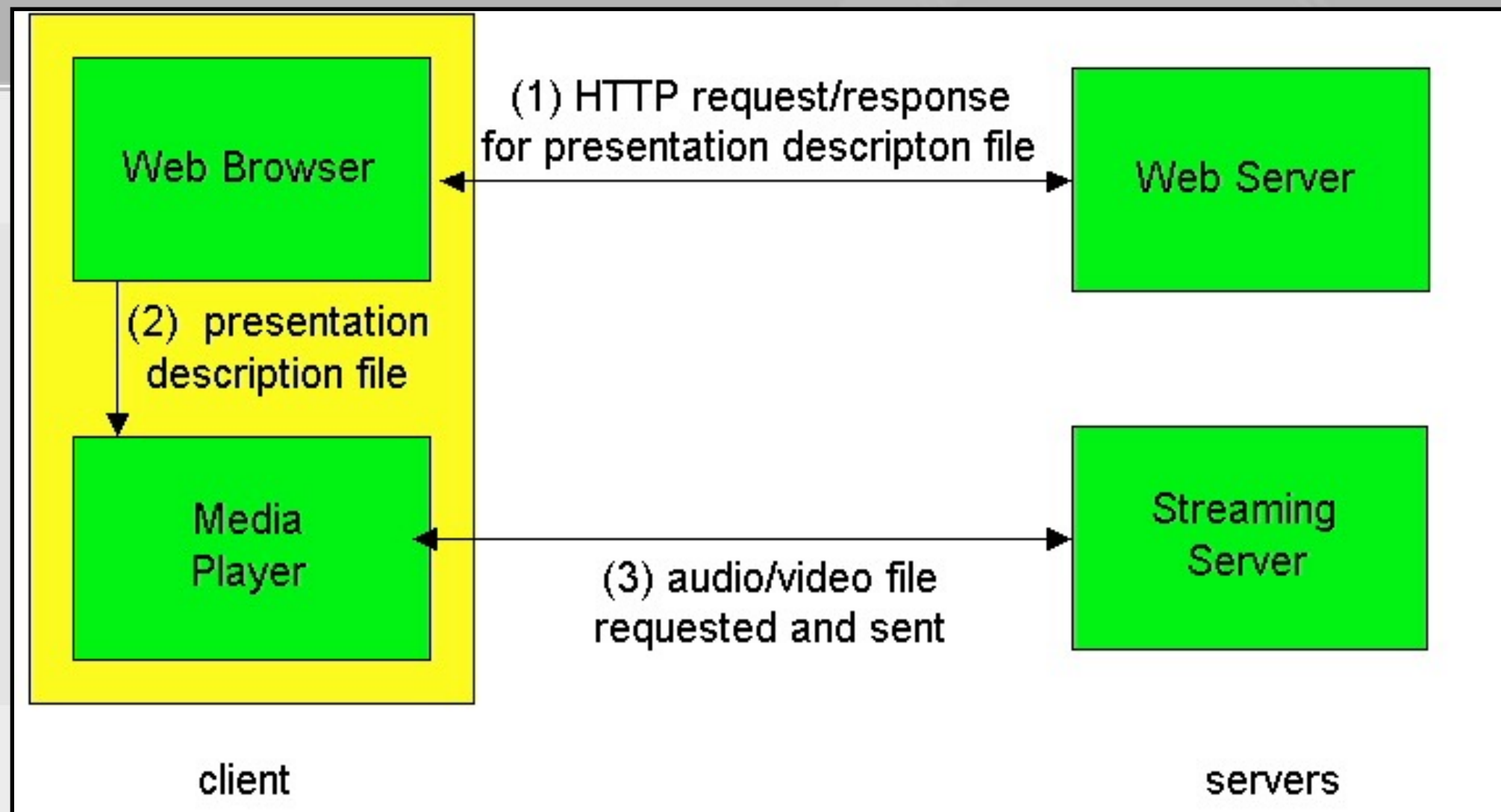


Some concerns:

- Media player communicates over HTTP, which is not designed with pause, ff, rnd commands
- May want to stream over UDP



# Streaming from a streaming server



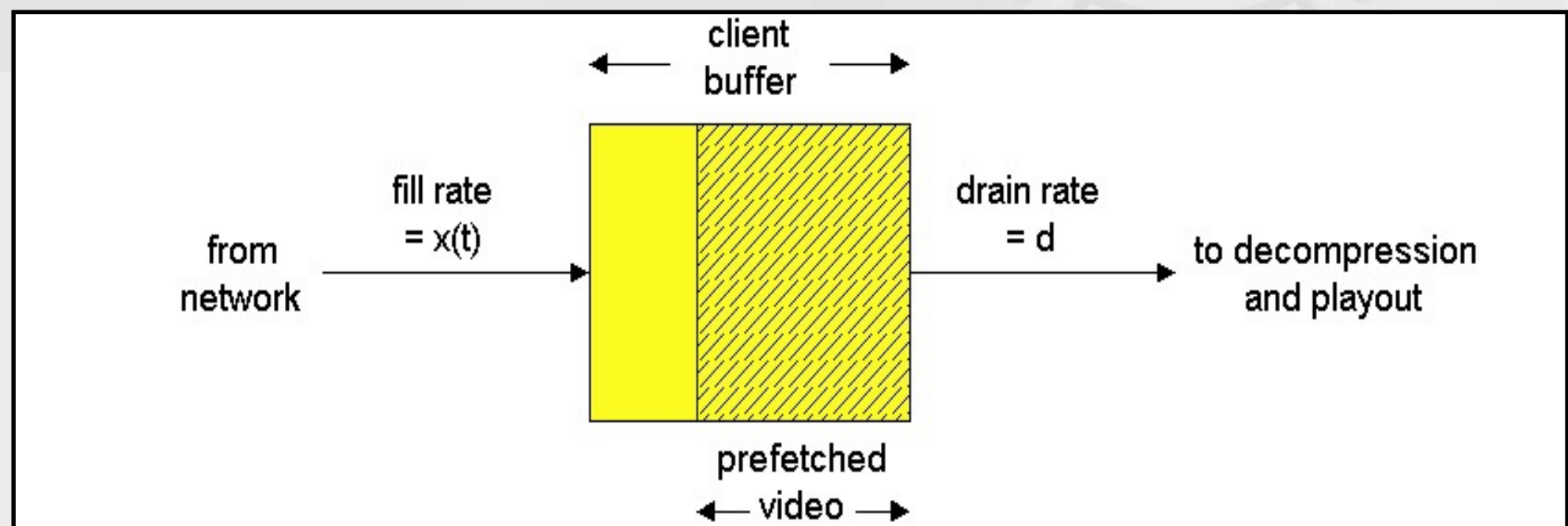
This architecture allows for non-HTTP protocol between server and media player

Can also use UDP instead of TCP.



# Options when using a streaming server

- Send at constant rate over **UDP**. To mitigate the effects of jitter, buffer and delay playback for 1-10 s. Transmit rate =  $d$ , the encoded rate. Fill rate  $x(t)$  equals  $d$  except when there is loss.
- Use TCP, and send at maximum possible rate under TCP; TCP retransmits when error is encountered;  $x(t)$  now fluctuates, and can become much larger than  $d$ . Player can use a much large buffer to smooth delivery rate of TCP.





# Streaming Protocols

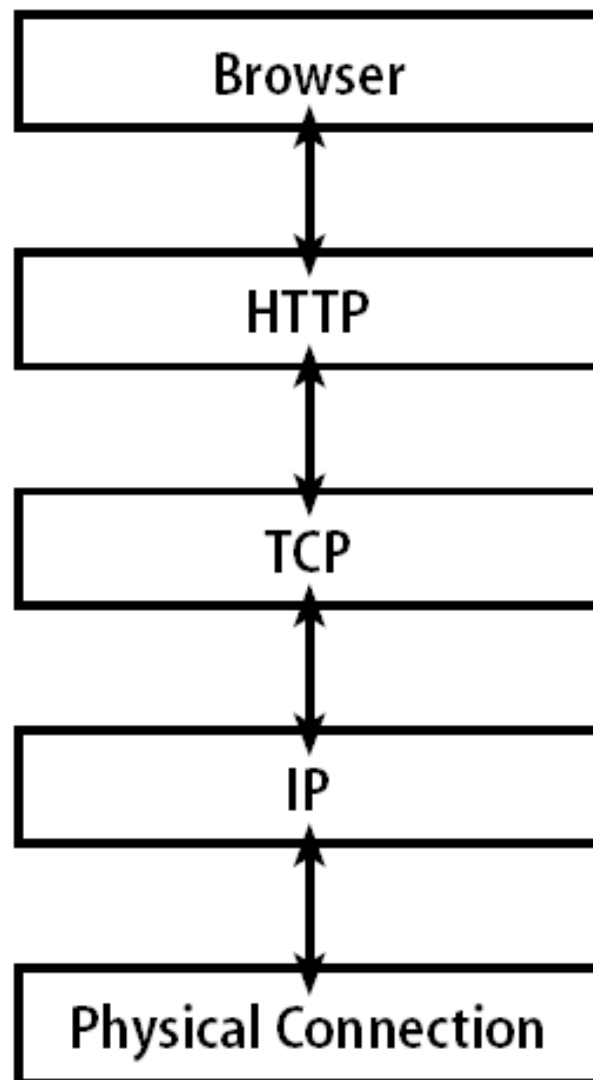
- **RTP (Real-time Transport Protocol)**
  - layered on top of **UDP**
  - RTP enables a one-way stream, transmitting media from the server to the client
- **RTSP (Real-time Streaming Protocol)**
  - similar to **HTTP** and to **FTP**—protocols used by Web servers
  - However, **HTTP** and **FTP** cannot be used for true streaming
    - both layered on top of **TCP**
  - a two-way protocol which uses TCP to communicate, and
  - usually layered on top of RTP
- **RTCP (Real-time Transport Control Protocol)**





# Streaming Protocols

## Viewing a Web page



# Real Time Streaming Protocol: RTSP

## HTTP

- Designers of HTTP had fixed media in mind: HTML, images, applets, etc.
- HTTP does not target stored continuous media (i.e., audio, video, SMIL presentations, etc.)

## RTSP: RFC 2326

- Client-server application layer protocol.
- For user to control display: rewind, fast forward, pause, resume, repositioning, etc...

## What it doesn't do:

- does not define how audio/video is encapsulated for streaming over network
- does not restrict how streamed media is transported; it can be transported over UDP or TCP
- does not specify how the media player buffers audio/video

## RealNetworks

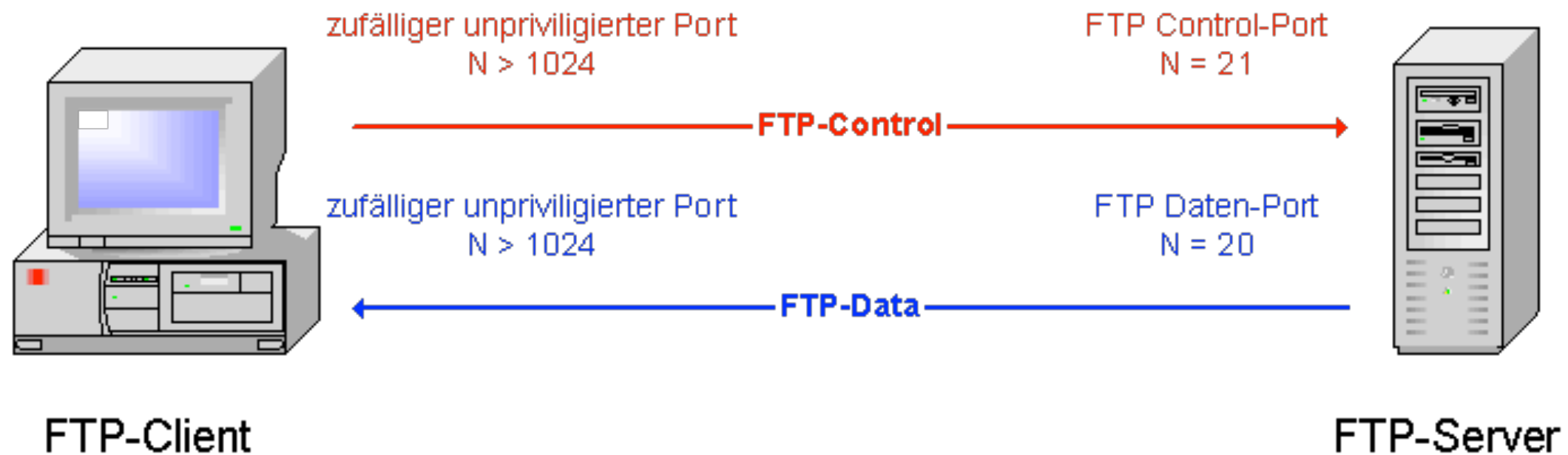
- Server and player use RTSP to send control info to each other



# FTP?

## Active FTP Beispiel

Pfeile Kennzeichnen die Richtung der Verbindung



# RTSP: out of band control

## FTP uses an “out-of-band” control channel:

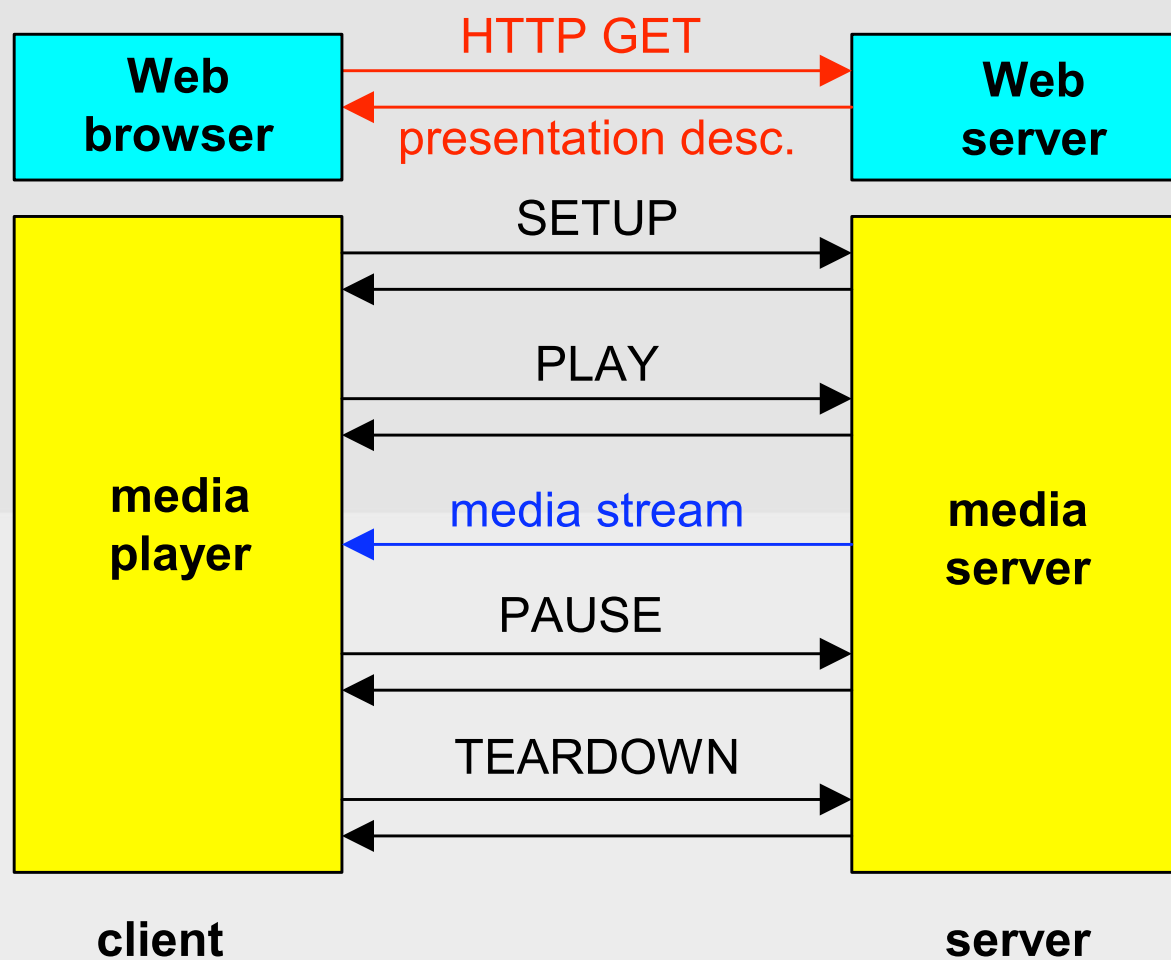
- A file is transferred over one channel.
- Control information (directory changes, file deletion, file renaming, etc.) is sent over a separate TCP connection.
- The “out-of-band” and “in-band” channels use different port numbers.

## RTSP messages are also sent out-of-band:

- The RTSP control messages use different port numbers than the media stream, and are therefore sent out-of-band.
- The media stream, whose packet structure is not defined by RTSP, is considered “in-band”.
- If the RTSP messages were to use the same port numbers as the media stream, then RTSP messages would be said to be “interleaved” with the media stream.



# RTSP initiates and controls delivery



- Client obtains a description of the multimedia presentation, which can consist of several media streams.
- The browser invokes media player (helper application) based on the content type of the presentation description.
- Presentation description includes references to media streams, using the URL method **rtsp://**
- Player sends **RTSP SETUP** request; server sends RTSP SETUP response.
- Player sends **RTSP PLAY** request; server sends RTSP PLAY response.
- Media server pumps media stream.
- Player sends **RTSP PAUSE** request; server sends RTSP PAUSE response.
- Player sends **RTSP TEARDOWN** request; server sends **RTSP TEARDOWN** response.



# RTSP session

- Each RTSP has a session identifier, which is chosen by the server.
- The client initiates the session with the SETUP request, and the server responds to the request with an identifier.
- The client repeats the session identifier for each request, until the client closes the session with the TEARDOWN request.
- RTSP port number is 554.
- RTSP can be sent over UDP or TCP. Each RTSP message can be sent over a separate TCP connection.

# RTSP: **interaction example**

- C: SETUP rtsp://audio.example.com/twister/audio RTSP/1.0
- Transport: rtp/udp; compression; port=3056; mode=PLAY
- S: RTSP/1.0 200 1 OK
- Session 4231
  
- C: PLAY rtsp://audio.example.com/twister/audio.en/lofi RTSP/1.0
- Session: 4231
- Range: npt=0-
  
- C: PAUSE rtsp://audio.example.com/twister/audio.en/lofi RTSP/1.0
- Session: 4231
- Range: npt=37
  
- C: TEARDOWN rtsp://audio.example.com/twister/audio.en/lofi RTSP/1.0
- Session: 4231
- S: 200 3 OK



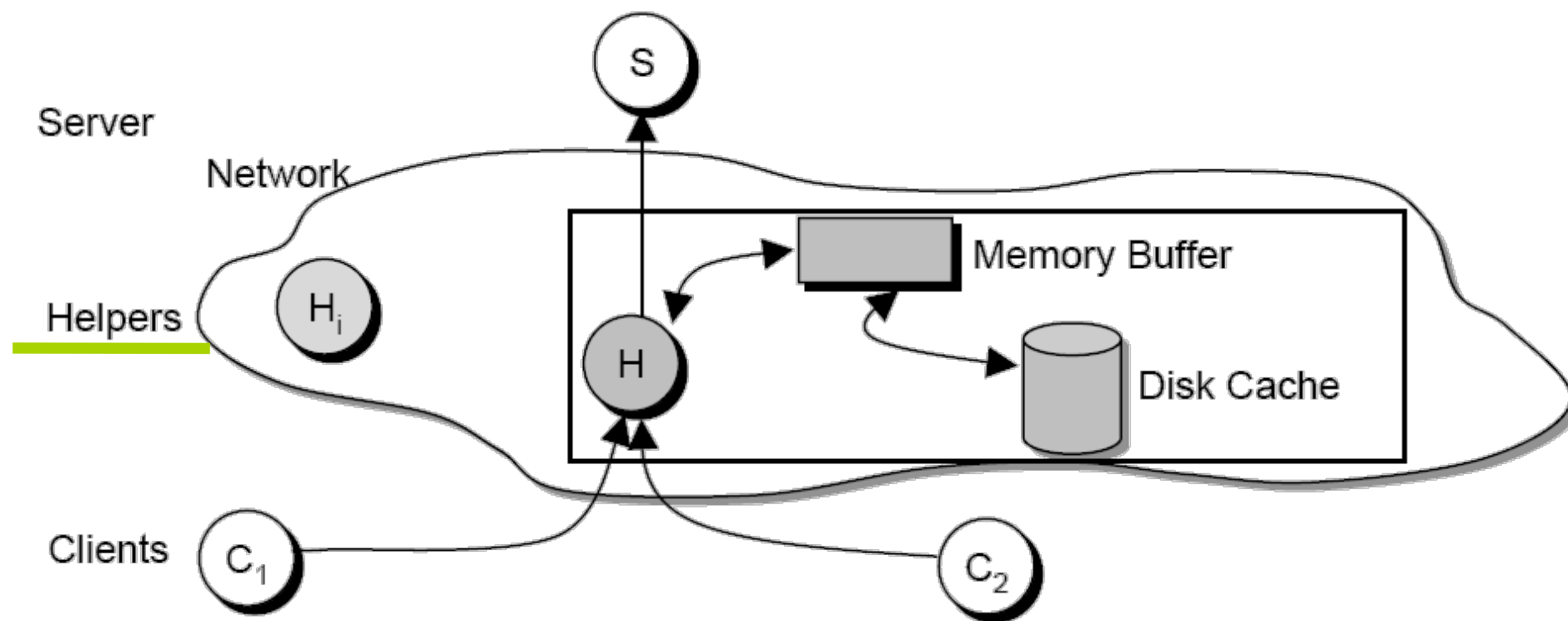


# RTSP: streaming caching

- Caching of RTSP response messages makes little sense.
- But desirable to cache media streams closer to client.
- Much of HTTP/1.1 cache control has been adopted by RTSP.
  - Cache control headers can be put in RTSP SETUP requests and responses:
    - If-modified-since: , Expires: , Via: , Cache-Control:



# Design and Implementation of a Caching System for Streaming Media



Application layer aware helper in  
the network

# RTSP: streaming caching

- **Proxy cache** may
  - hold only segments of a given media stream.
  - start serving a client from its local cache, and then have to connect to origin server and fill missing material, hopefully without introducing gaps at client.
- When origin server is sending a stream through client, and stream passes through a proxy, proxy can use TCP to obtain the stream; but proxy still sends RTSP control messages to origin server.



# Real-time interactive applications

- **PC-2-PC phone**
- PC-2-phone
  - Dialpad
  - Net2phone
- videoconference
- Webcams



# Internet phone over best-effort (1)

## Best effort

- packet delay, loss and jitter

## Internet phone example

- now examine how packet delay, loss and jitter are often handled in the context of an IP phone example.
  - Internet phone applications generate packets during talk spurts
  - bit rate is 64 kbps during talk spurt
- during talk spurt, every 20 msec app generates a chunk of 160 bytes = 8 kbytes/sec \* 20 msec
  - header is added to chunk; then chunk+header is encapsulated into a UDP packet and sent out
  - some packets can be lost and packet delay will fluctuate.
  - receiver must determine when to playback a chunk, and determine what do with missing chunk



# Internet phone (2)

## packet loss

- UDP segment is encapsulated in IP datagram
- datagram may overflow a router queue
- TCP can eliminate loss, but
  - retransmissions add delay
  - TCP congestion control limits transmission rate
- Redundant packets can help

## end-to-end delay

- accumulation of transmission, propagation, and queuing delays
- more than 400 msec of end-to-end delay seriously hinders interactivity; the smaller the better



# Internet phone (2)

## delay jitter

- consider two consecutive packets in talk spurt
- initial spacing is 20 msec, but spacing at receiver can be more or less than 20 msec

## removing jitter

- sequence numbers
- timestamps
- delaying playout



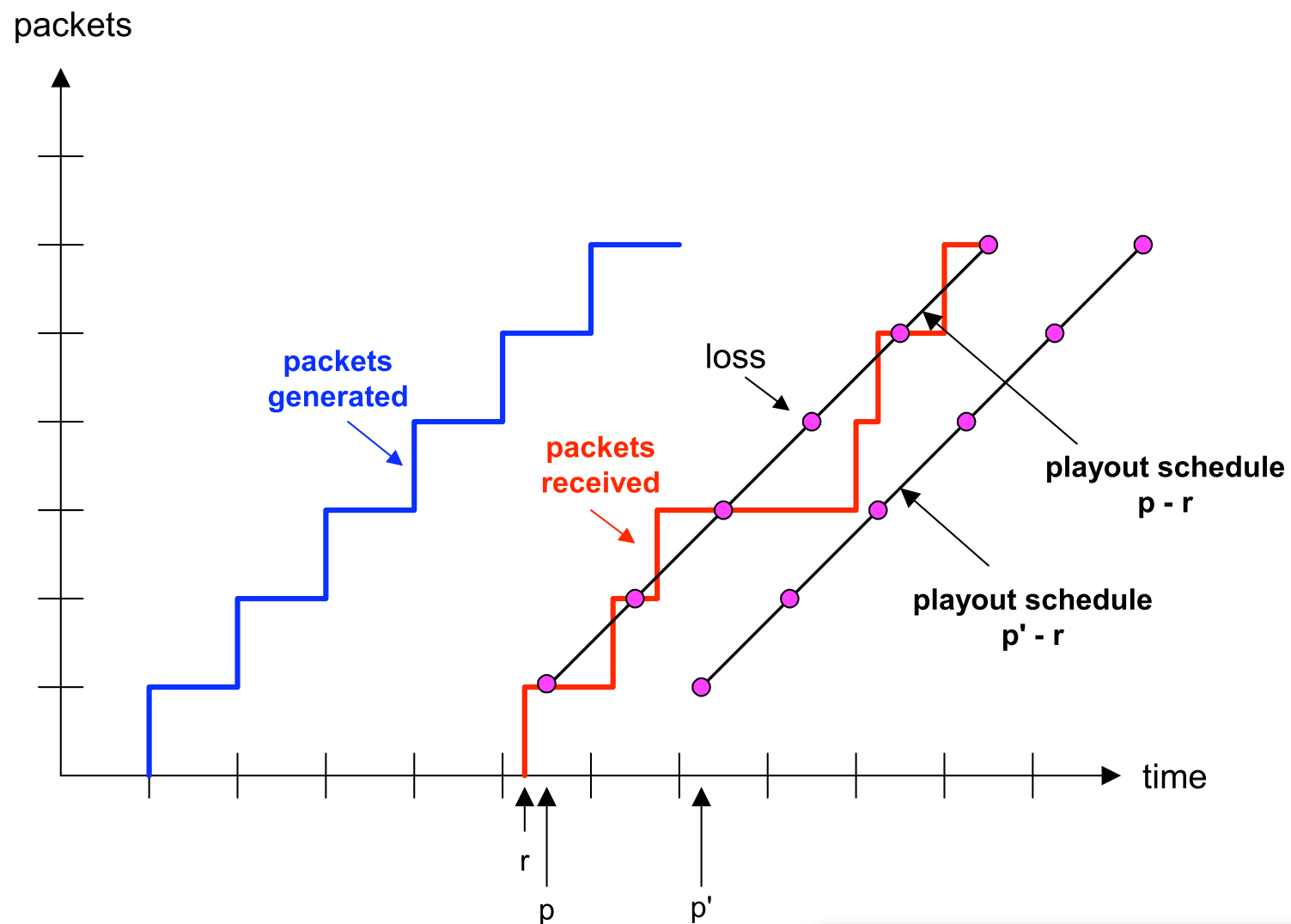


# Internet phone (3): fixed playout delay

- Receiver attempts to playout each chunk at exactly  $q$  msec after the chunk is generated.
    - If chunk is time stamped  $t$ , receiver plays out chunk at  $t+q$ .
    - If chunk arrives after time  $t+q$ , receiver discards it.
  - Sequence numbers are not necessary.
  - Strategy allows for lost packets.
- Tradeoff for  $q$ :
    - larger  $q$ : less packet loss
    - smaller  $q$ : better interactive experience



# Internet phone (4): fixed playout delay



- Sender generates packets every 20 msec during talk spurt.
- First packet received at time  $r$
- First playout schedule: begins at  $p$
- Second playout schedule: begins at  $p'$

# Adaptive playout delay (1)

- Estimate network delay and adjust playout delay at the beginning of each talk spurt.
- Silent periods are compressed and elongated.
- Chunks still played out every 20 msec during talk spurt.

$t_i$  = timestamp of the  $i$ th packet

$r_i$  = the time packet  $i$  is received by receiver

$p_i$  = the time packet  $i$  is played at receiver

$r_i - t_i$  = network delay for  $i$ th packet

$d_i$  = estimate of average network delay after receiving  $i$ th packet

Dynamic estimate of average delay at receiver:

$$d_i = (1 - u)d_{i-1} + u(r_i - t_i)$$

where  $u$  is a fixed constant (e.g.,  $u = .01$ ).

# Adaptive playout delay (2)

Also useful to estimate the average deviation of the delay,  $v_i$ :

$$v_i = (1 - u)v_{i-1} + u |r_i - t_i - d_i|$$

The estimates  $d_i$  and  $v_i$  are calculated for every received packet, although they are only used at the beginning of a talk spurt.

For first packet in talk spurt, playout time is:

$$p_i = t_i + d_i + Kv_i$$

where  $K$  is a positive constant. For this same packet, the play out delay is:

$$q_i = p_i - t_i$$

For packet  $j$  in the same talk spurt, play packet out at

$$p_j = t_j + q_i$$

# Adaptive playout (3)

## How to determine whether a packet is the first in a talkspurt:

- If there were never loss, receiver could simply look at the successive time stamps.
  - Difference of successive stamps  $> 20$  msec, talk spurt begins.
- But because loss is possible, receiver must look at both time stamps and sequence numbers.
  - Difference of successive stamps  $> 20$  msec and sequence numbers without gaps, talk spurt begins.



# Recovery from packet loss (1)

- Loss: packet never arrives or arrives later than its scheduled playout time

## forward error correction (FEC): simple scheme

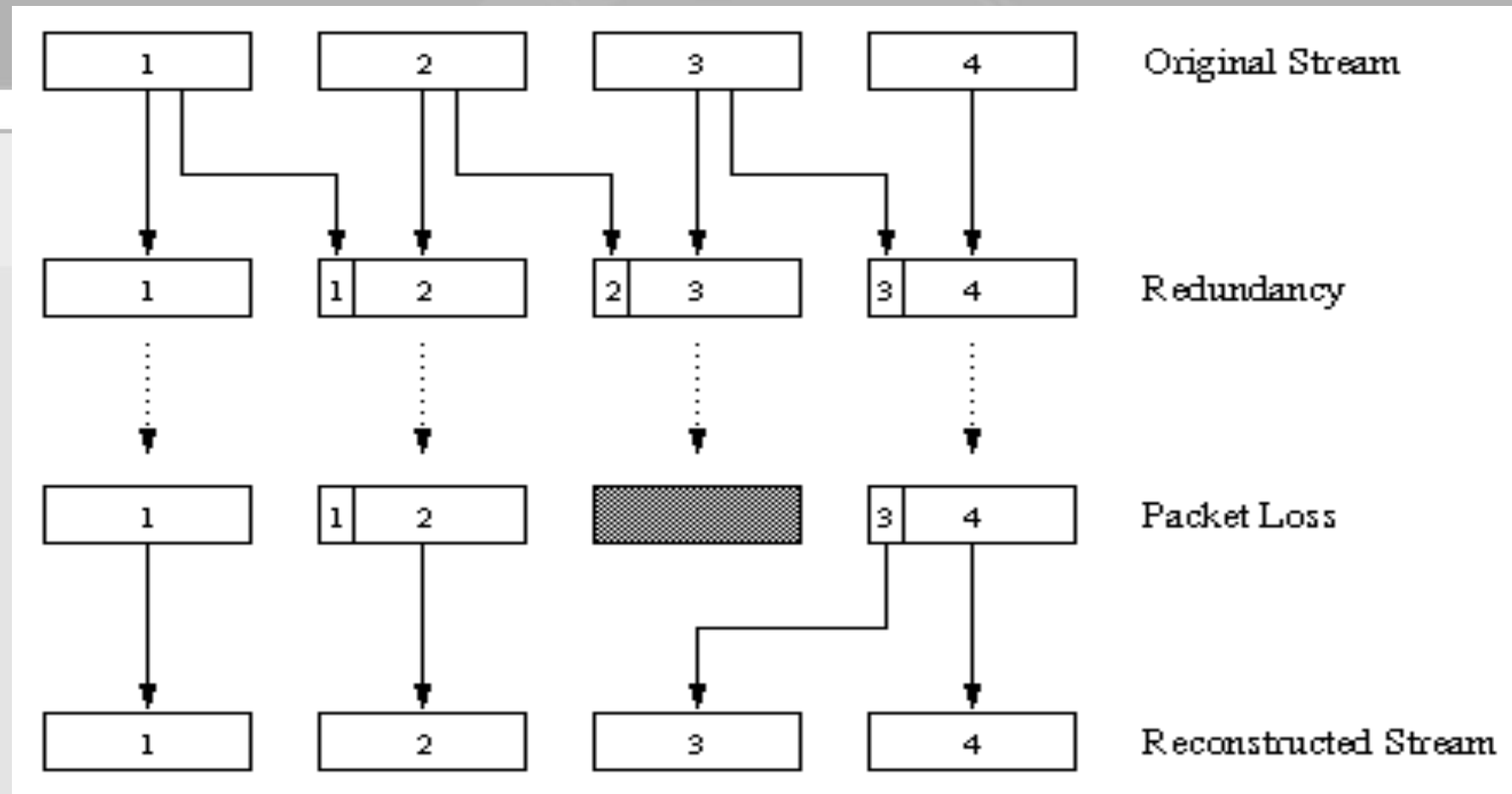
- for every group of  $n$  chunks create a redundant chunk by exclusive OR-ing the  $n$  original chunks
- send out  $n+1$  chunks, increasing the bandwidth by factor  $1/n$ .
- can reconstruct the original  $n$  chunks if there is at most one lost chunk from the  $n+1$  chunks
- Playout delay needs to be fixed to the time to receive all  $n+1$  packets
- Tradeoff:
  - increase  $n$ , less bandwidth waste
  - increase  $n$ , longer playout delay
  - increase  $n$ , higher probability that 2 or more chunks will be lost



# Recovery from packet loss (2)

## 2nd FEC scheme

- “piggyback lower quality stream”
- send lower resolution audio stream as the redundant information
- for example, nominal stream PCM at 64 kbps and redundant stream GSM at 13 kbps.
- Sender creates packet by taking the  $n$ th chunk from nominal stream and appending to it the  $(n-1)$ st chunk from redundant stream.



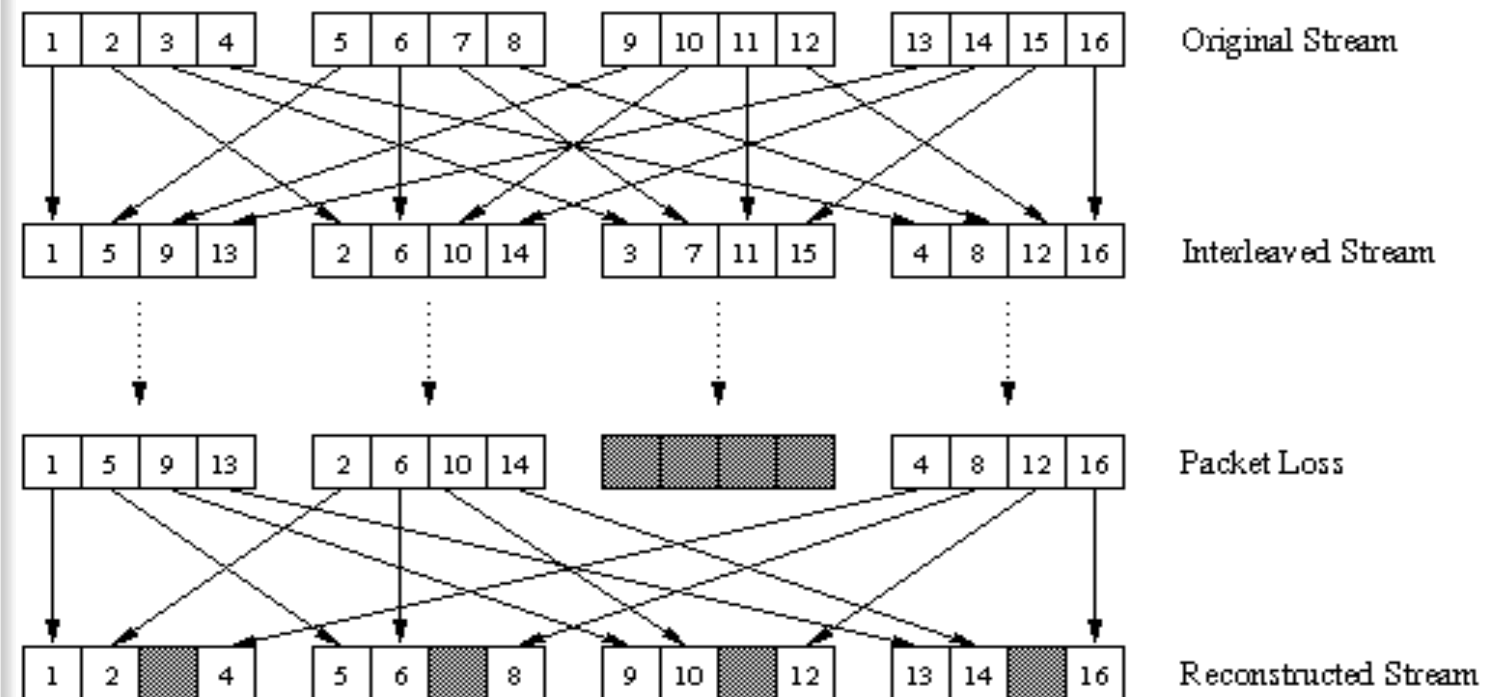
- Whenever there is non-consecutive loss, the receiver can conceal the loss.
- Only two packets need to be received before playback
- Can also append  $(n-1)$ st and  $(n-2)$ nd low-bit rate chunk



# Recovery from packet loss (3)

## Interleaving

- chunks are broken up into smaller units
- for example, 4 5 msec units per chunk
- interleave the chunks as shown in diagram
- packet now contains small units from different chunks



- Reassemble chunks at receiver
- if packet is lost, still have most of every chunk

# Recovery from packet loss (4)

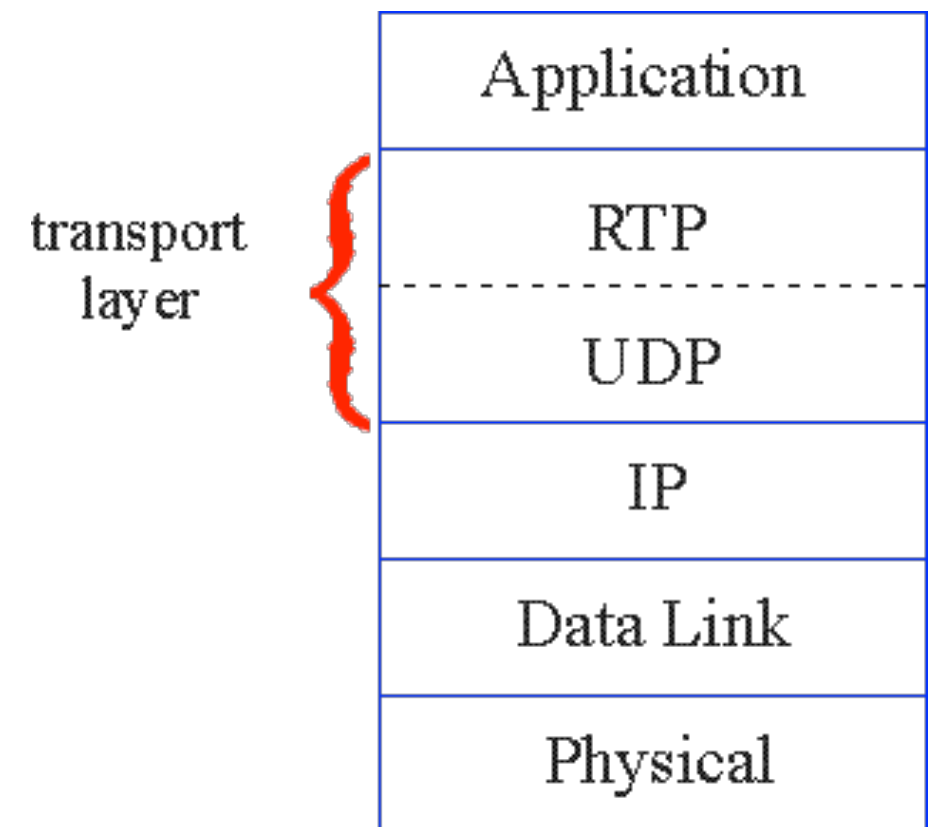
## Receiver-based repair of damaged audio streams

- produce a replacement for a lost packet that is similar to the original
- can give good performance for low loss rates and small packets (4-40 msec)
- simplest: repetition
- more complicated: interpolation



# Real-Time Protocol (RTP)

- RTP specifies a packet structure for packets carrying audio and video data: RFC 1889.
- RTP packet provides
  - payload type identification
  - packet sequence numbering
  - timestamping
- RTP runs in the end systems.
- RTP packets are encapsulated in UDP segments
- Interoperability: If two Internet phone applications run RTP, then they may be able to work together



# Real-Time Protocol (RTP)

- Provides standard packet format for real-time application
- Typically runs over UDP
- Specifies header fields below



RTP Header

- **Payload Type:** 7 bits, providing 128 possible different types of encoding; eg PCM, MPEG2 video, etc.
- **Sequence Number:** 16 bits; used to detect packet loss



# RTP Header

**Payload Type (7 bits):** Used to indicate the type of encoding that is currently being used.



**RTP Header**

If a sender changes the encoding in the middle of a conference, the sender informs the receiver through this payload type field.

- Payload type 0: PCM mu-law, 64 Kbps
- Payload type 3, GSM, 13 Kbps
- Payload type 7, LPC, 2.4 Kbps
- Payload type 26, Motion JPEG
- Payload type 31. H.261
- Payload type 33, MPEG2 video

**Sequence Number (16 bits):** The sequence number increments by one for each RTP packet sent; may be used to detect packet loss and to restore packet sequence.

# Real-Time Protocol (RTP)

- **Timestamp**: 32 bits; gives the sampling instant of the first audio/video byte in the packet; used to remove jitter introduced by the network
- **Synchronization Source identifier (SSRC)**: 32 bits; an id for the source of a stream; assigned randomly by the source



RTP Header

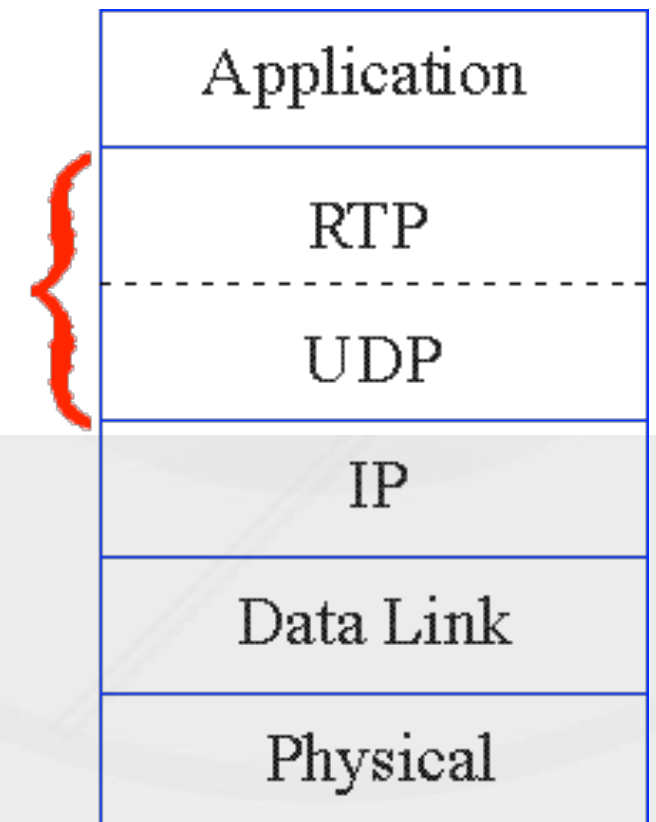


# RTP runs on top of UDP

RTP libraries provide a transport-layer interface that extend UDP:

- port numbers, IP addresses
- error checking across segment
- payload type identification
- packet sequence numbering
- time-stamping

transport  
layer





# RTP Example

- Consider sending 64 kbps PCM-encoded voice over RTP.
  - Application collects the encoded data in chunks, e.g., every 20 msec = 160 bytes in a chunk.
  - The audio chunk along with the RTP header form the RTP packet, which is encapsulated into a UDP segment.
- RTP header indicates type of audio encoding in each packet; senders can change encoding during a conference. RTP header also contains sequence numbers and timestamps.



# RTP and QoS

- RTP does **not** provide any mechanism to ensure timely delivery of data or provide other quality of service guarantees.
- RTP encapsulation is only seen at the end systems -- it is not seen by intermediate routers.
  - Routers providing the Internet's traditional best-effort service do not make any special effort to ensure that RTP packets arrive at the destination in a timely matter.
- In order to provide QoS to an application, the Internet must provide a mechanism, such as RSVP, for the application to reserve network resources.

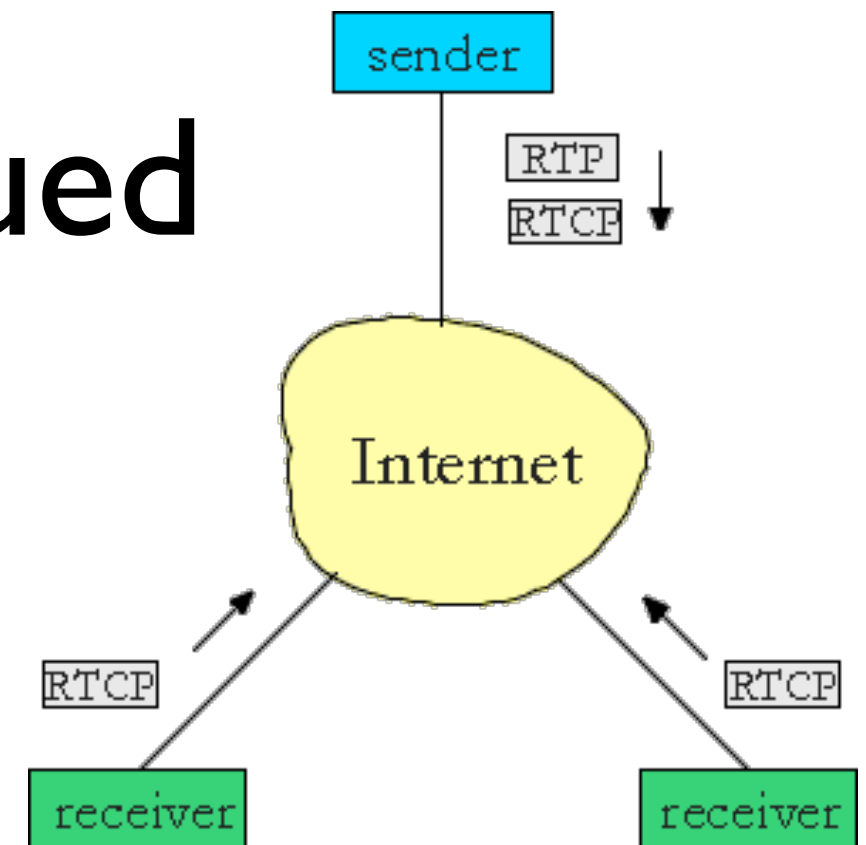
# RTP Streams

- RTP allows each source (for example, a camera or a microphone) to be assigned its own independent RTP stream of packets.
- For example, for a videoconference between two participants, four RTP streams could be opened: two streams for transmitting the audio (one in each direction) and two streams for the video (again, one in each direction).
- However, some popular encoding techniques -- including MPEG1 and MPEG2 -- bundle the audio and video into a single stream during the encoding process. When the audio and video are bundled by the encoder, then only one RTP stream is generated in each direction.
- For a many-to-many multicast session, all of the senders and sources typically send their RTP streams into the same multicast tree with the same multicast address.

# Real-Time Control Protocol (RTCP)

- Works in conjunction with RTP.
- Each participant in an RTP session periodically transmits RTCP control packets to all other participants. Each RTCP packet contains sender and/or receiver reports that report statistics useful to the application.
- Statistics include number of packets sent, number of packets lost, inter-arrival jitter, etc.
- This feedback of information to the application can be used to control performance and for diagnostic purposes.
- The sender may modify its transmissions based on the feedback.

# RTCP - Continued



- For an RTP session there is typically a single multicast address: all RTP and RTCP packets belonging to the session use the multicast address.
- RTP and RTCP packets are distinguished from each other through the use of distinct port numbers.
- To limit traffic, each participant reduces his RTCP traffic as the number of conference participants increases.

# RTCP Packets

## Receiver report packets:

- fraction of packets lost, last sequence number, average interarrival jitter.

## Sender report packets:

- SSRC of the RTP stream, the current time, the number of packets sent, and the number of bytes sent.

## Source description packets:

- e-mail address of the sender, the sender's name, the SSRC of the associated RTP stream. Packets provide a mapping between the SSRC and the user/host name.



# Synchronization of Streams

- RTCP can be used to synchronize different media streams within a RTP session.
- Consider a videoconferencing application for which each sender generates one RTP stream for video and one for audio.
- The timestamps in these RTP packets are tied to the video and audio sampling clocks, and are not tied to the wall-clock time (i.e., to real time).
- Each RTCP sender-report packet contains, for the most recently generated packet in the associated RTP stream, the timestamp of the RTP packet and the wall-clock time for when the packet was created. Thus the RTCP sender-report packets associate the sampling clock to the real-time clock.
- Receivers can use this association to synchronize the playout of audio and video.

# RTCP Bandwidth Scaling

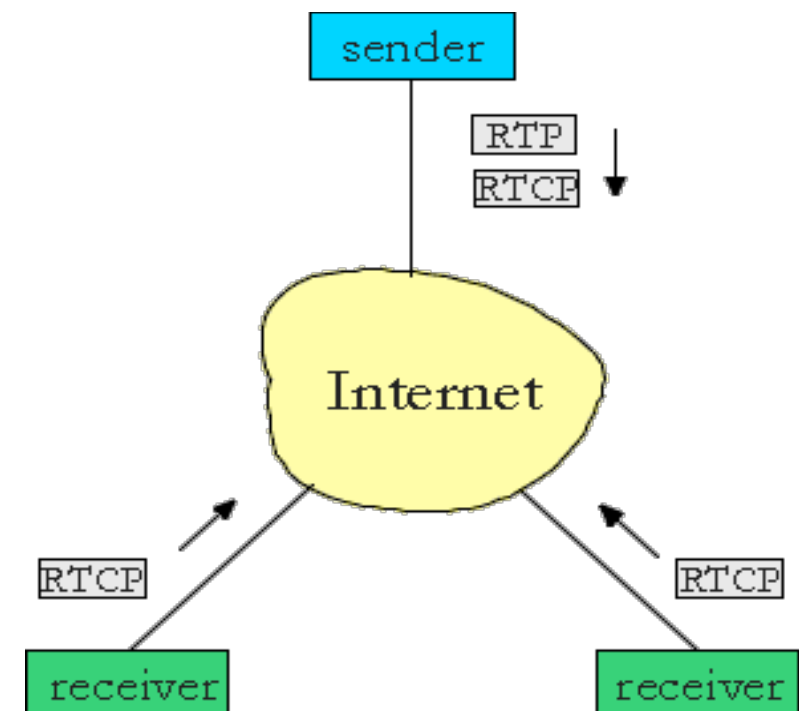
- RTCP attempts to limit its traffic to 5% of the session bandwidth.
  - For example, suppose there is one sender, sending video at a rate of 2 Mbps. Then RTCP attempts to limit its traffic to 100 Kbps.
  - The protocol gives 75% of this rate, or 75 kbps, to the receivers; it gives the remaining 25% of the rate, or 25 kbps, to the sender.
- The 75 kbps devoted to the receivers is equally shared among the receivers. Thus, if there are  $R$  receivers, then each receiver gets to send RTCP traffic at a rate of  $75/R$  kbps and the sender gets to send RTCP traffic at a rate of 25 kbps.
  - A participant (a sender or receiver) determines the RTCP packet transmission period by dynamically calculating the the average RTCP packet size (across the entire session) and dividing the average RTCP packet size by its allocated rate.





# RTP Control Protocol (RTCP)

- Protocol specifies report packets exchanged between sources and destinations of multimedia information
- Three reports are defined: Receiver reception, Sender, and Source description
- Reports contain statistics such as the number of packets sent, number of packets lost, inter-arrival jitter
- Used to modify sender transmission rates and for diagnostics purposes



# RTCP Bandwidth Scaling

- If each receiver sends RTCP packets to all other receivers, the traffic load resulting can be large
- RTCP adjusts the interval between reports based on the number of participating receivers
- Typically, limit the RTCP bandwidth to 5% of the session bandwidth, divided between the sender reports (25%) and the receivers reports (75%)



# 小结RTCP的四种功能

- 提供传送端信息传输的品质回报
- 展现层等级的识别信息，即CNAME
- 成员传送回报 => 传送的速率控制
- 提供少量session控制信息



# Streaming media servers (softwares)

- Audio
- Video
- P2P ...

# Audio streaming

- 校园广播站:



- SHOUTcast + Winamp (2003)
- 豆瓣FM

# Commercial solutions

- Real Producer
- Windows Media Encoder
- QuickTime Streaming Server



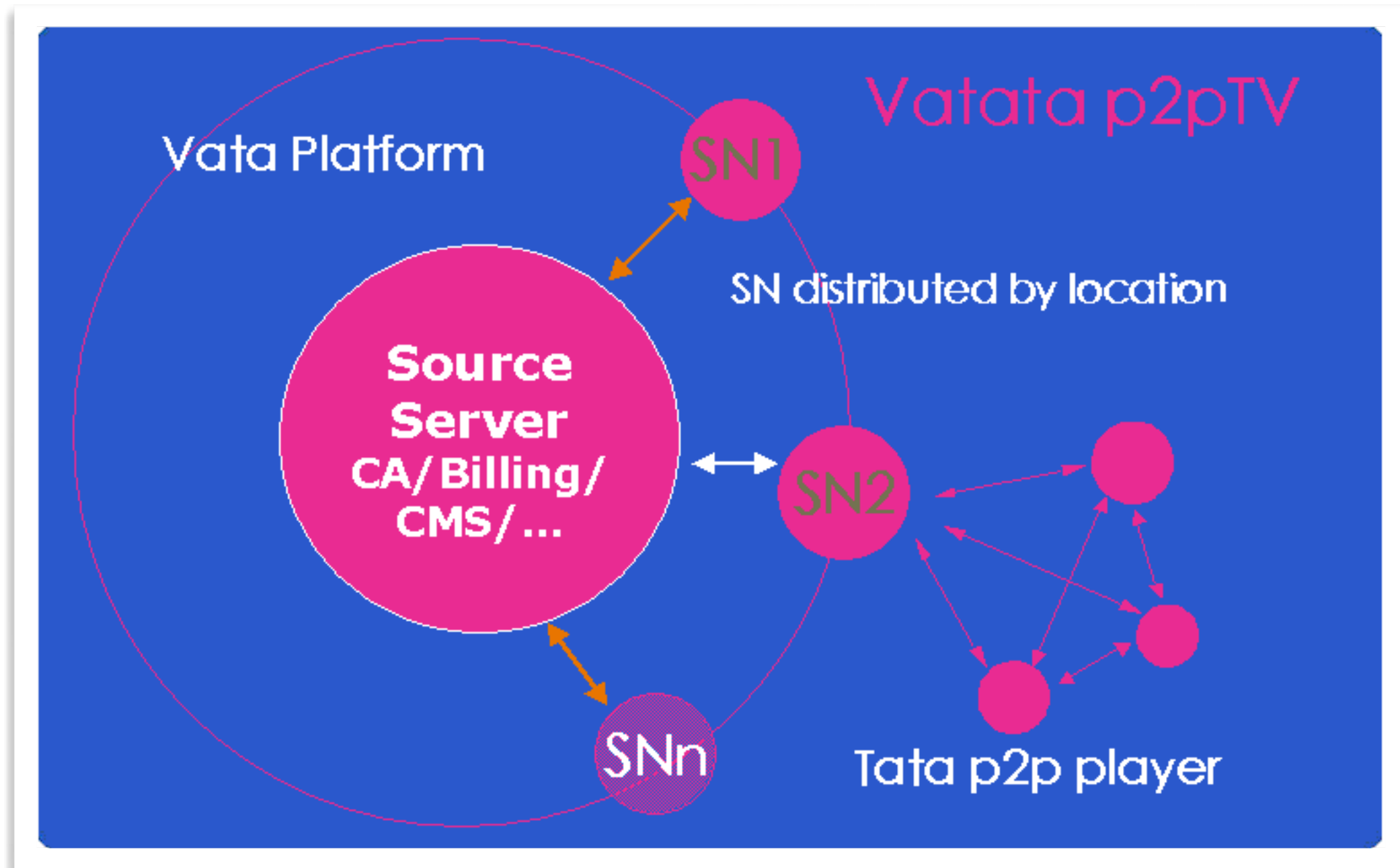
# P2P solutions

<http://clin003.com/ideas/p2p-streaming-media-server-1318/>

- CC IPTV Live Server
- PPS流媒体服务器
- MSMSream P2P流媒体
- 蓝天PPD下载点播系统
- 超限组播视像服务器
- Vatata流媒体系统目录服务器
- QVOD 流媒体点播系统 (BT, 免费试用)

**Vatata**<sup>TM</sup>

# Vatata



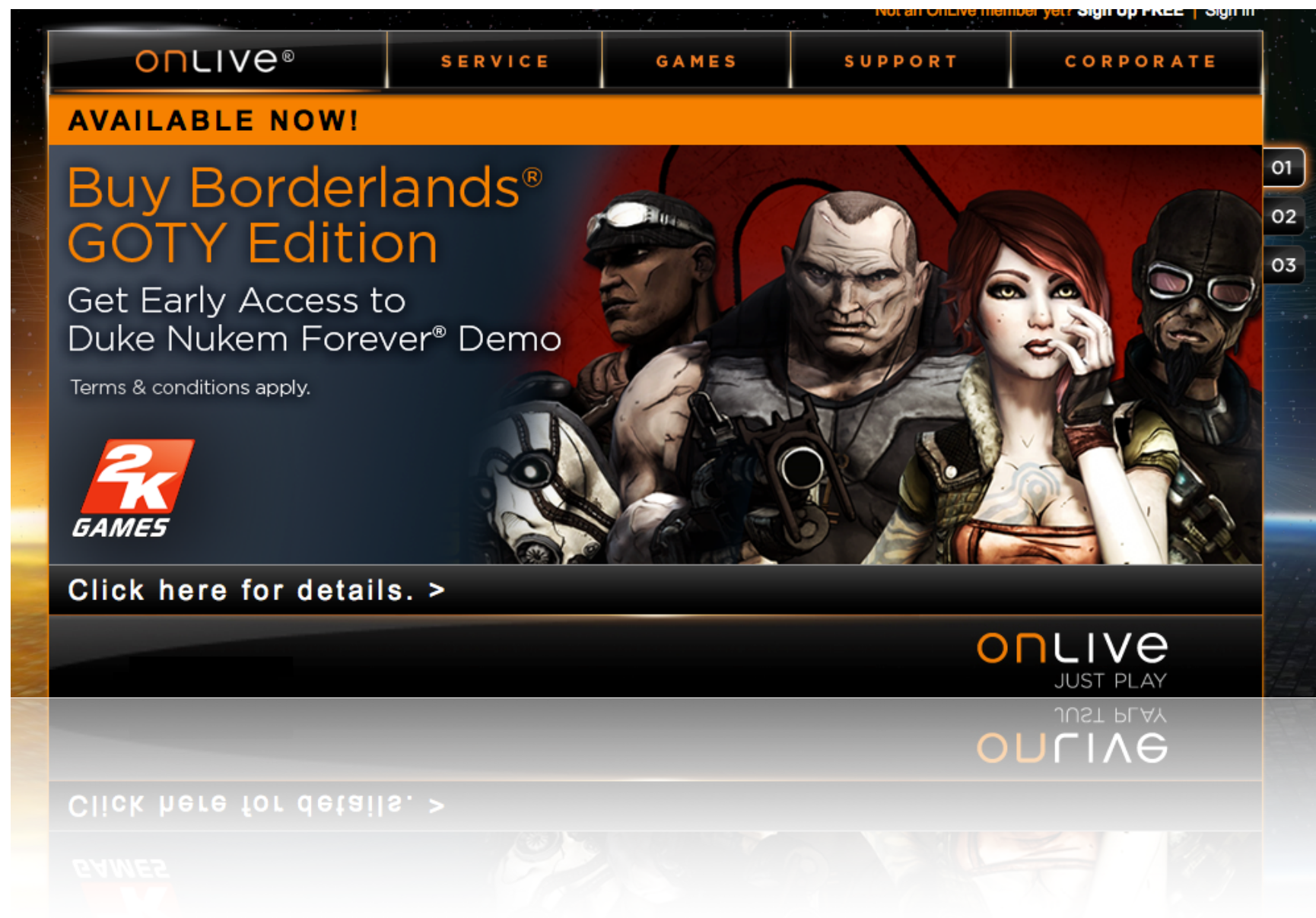


# 讨论

- 流媒体将来还会有什么样的应用？
- 流媒体在技术层面会有怎样的改变？

# Future of steaming media

- Play game through streaming!



# Streaming VR/AR media

- 3D Graphics?
- 4k/8k video?
- User interaction



<http://www.onlinereporter.com/2016/04/29/over-15-major-media-companies-are-now-creating-vr-content/>

# Reference

- Fundamentals of Multimedia, Chapter 16
  - 机械工业出版社