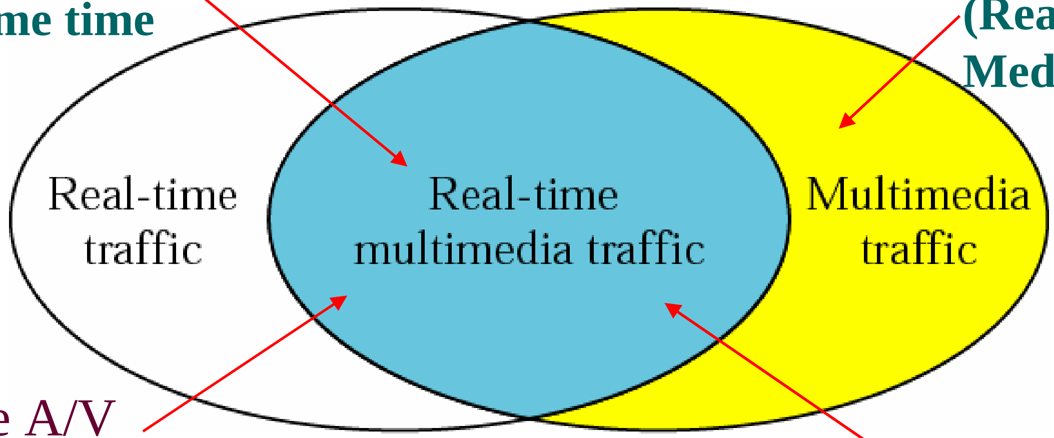
Multimedia over IP

(RTP, RTCP, SIP, RSTP)

Multimedia Traffic

The production, transmission, and use of data take place at the same time (NetMeeting).

Production, transmission and use of data take place at different times (RealPlayer, Windows Media Player)



<u>Streaming Live A/V</u>

(Like broadcast TV/radio but via Internet, Can't pause, rewind. The time between request and display 1 to 10 seconds. Continuous playout)

<u>Streaming Stored A/V</u>

(Like DVD but via Internet Prerecorded multimedia content, user may pause, rewind, forward,…The time between request and display 1 to 10 seconds After display start the playout must be continuous)

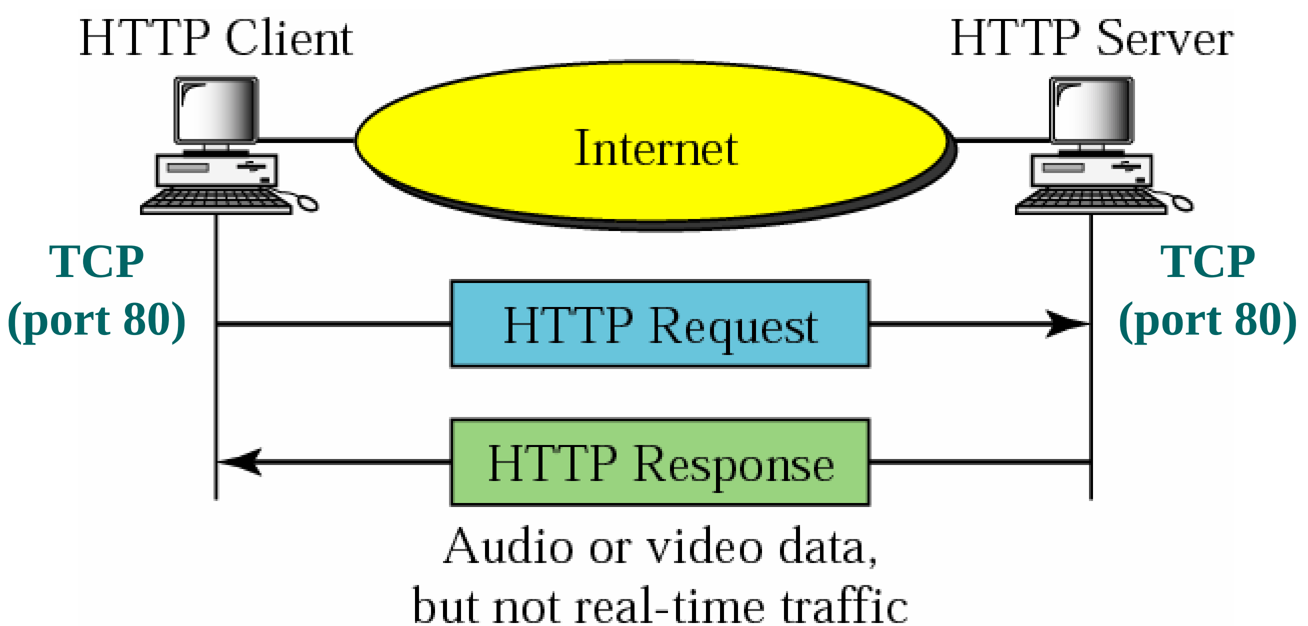
<u>Real-Time Interactive A/V</u>

(IP phone, video conferencing. Can't pause, rewind. Delay between initial request and display small: video <150ms, audio <400 acceptable. Continuous playout)

Multimedia Traffic (cont.)

Non-Real Time Multimedia Traffic

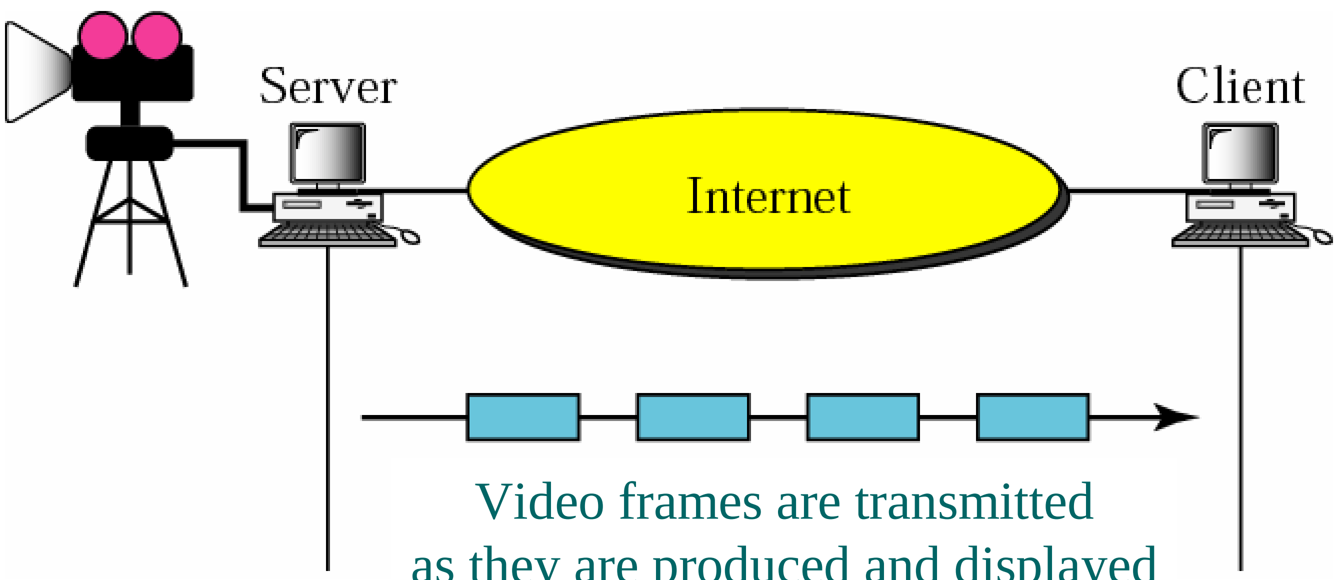
Example: downloading of a video from the Internet. The video has already been made; it's a finished product. A client HTTP is used to download the video from an HTTP server and the user views the video at a later time. The production, transmission, and use all happen at different times.



Multimedia Traffic (cont.)

Real-Time Multimedia Traffic

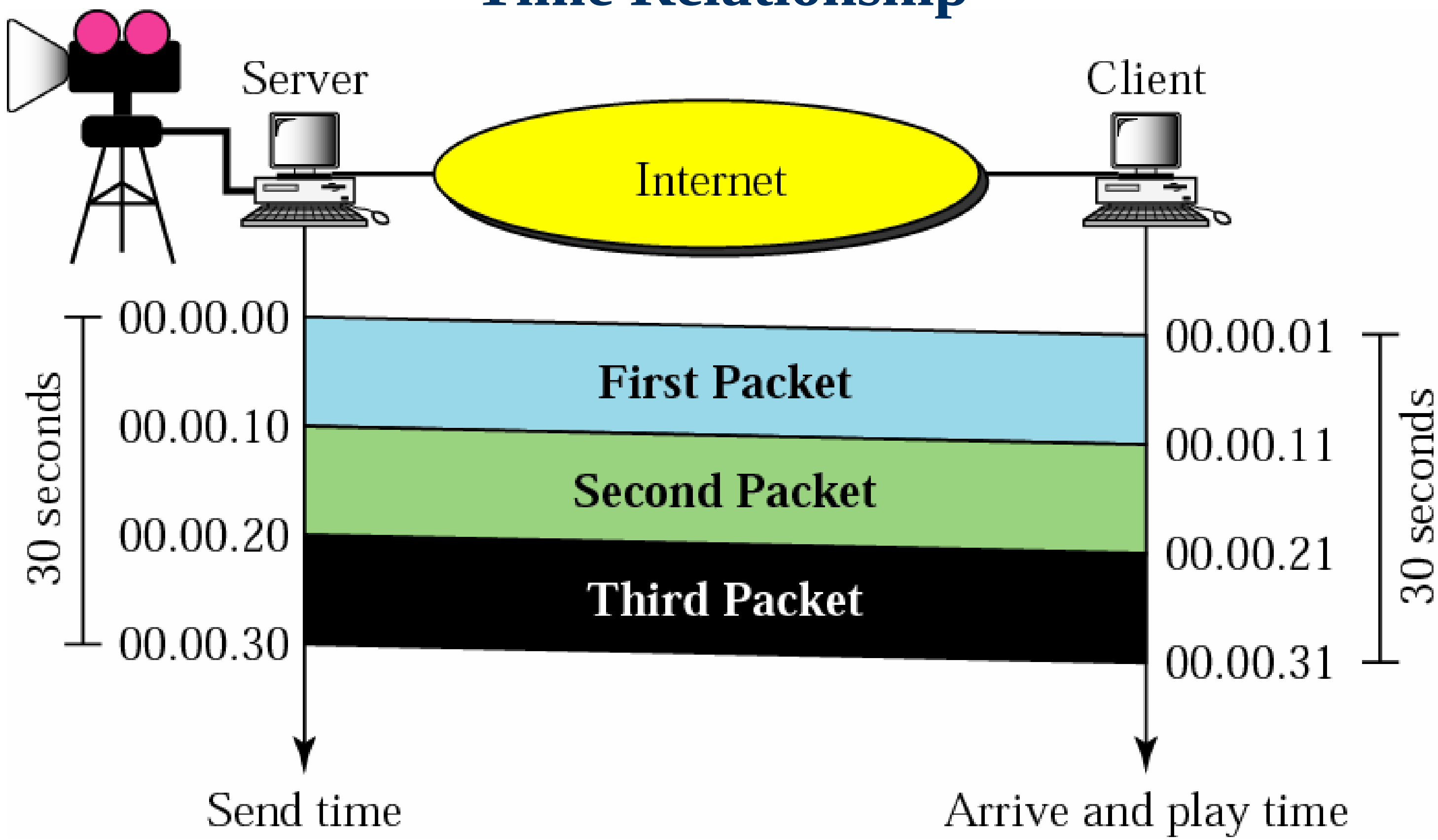
Example: a video conference in which a camera is connected to a server that transmits video information as it is produced. Everything that happens at the server site can be displayed on the computer at the client site. This is both multimedia (video) and real-time traffic (production and use at the same time).



Video frames are transmitted
as they are produced and displayed

Real-Time Multimedia Traffic

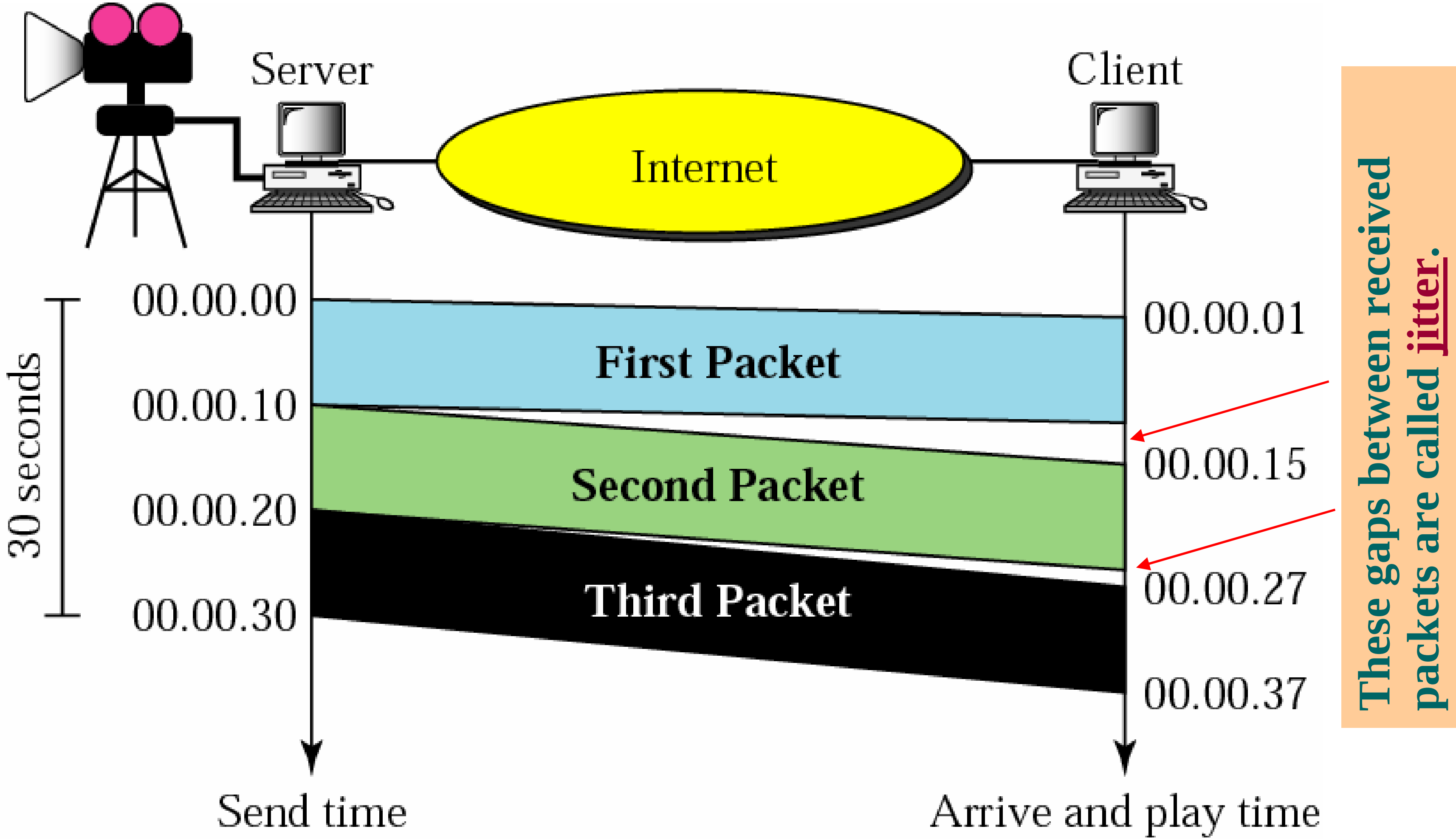
Time Relationship



Each packet holds 10 seconds of video information
Assumption: transfer delay is constant and equal 1 second (exaggerated)
The receiver sees the video with the same speed as it is created, the constant transfer delay is immaterial.

Real-Time Multimedia Traffic (cont.)

Jitter

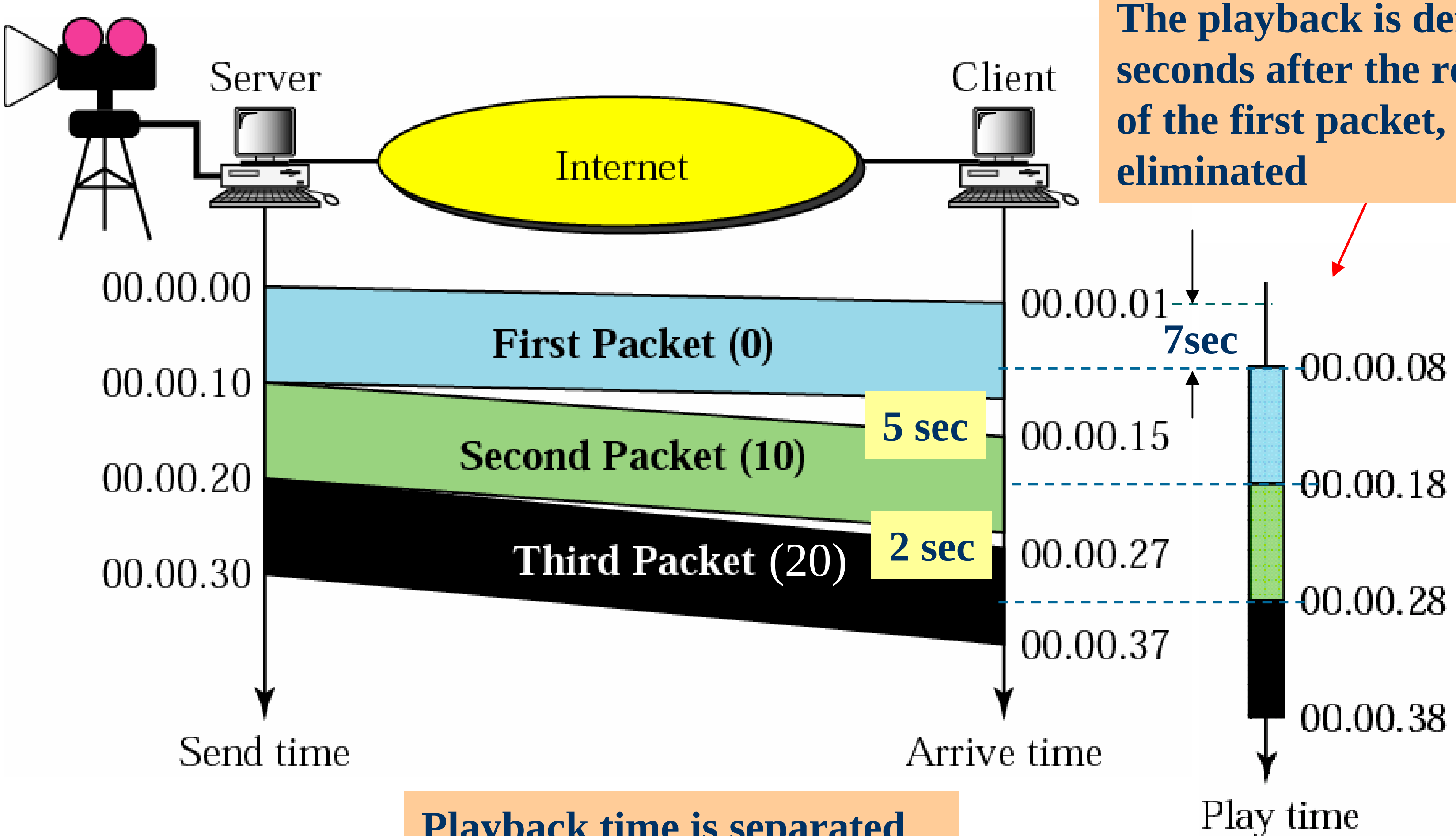


What happens if the transfer delay is not constant?

Real-Time Multimedia Traffic (cont.)

Timestamp

A solution to the jitter problem is use of timestamps - each packet has a time of the packet with respect to the first packet (time is measured at the sender's site.)



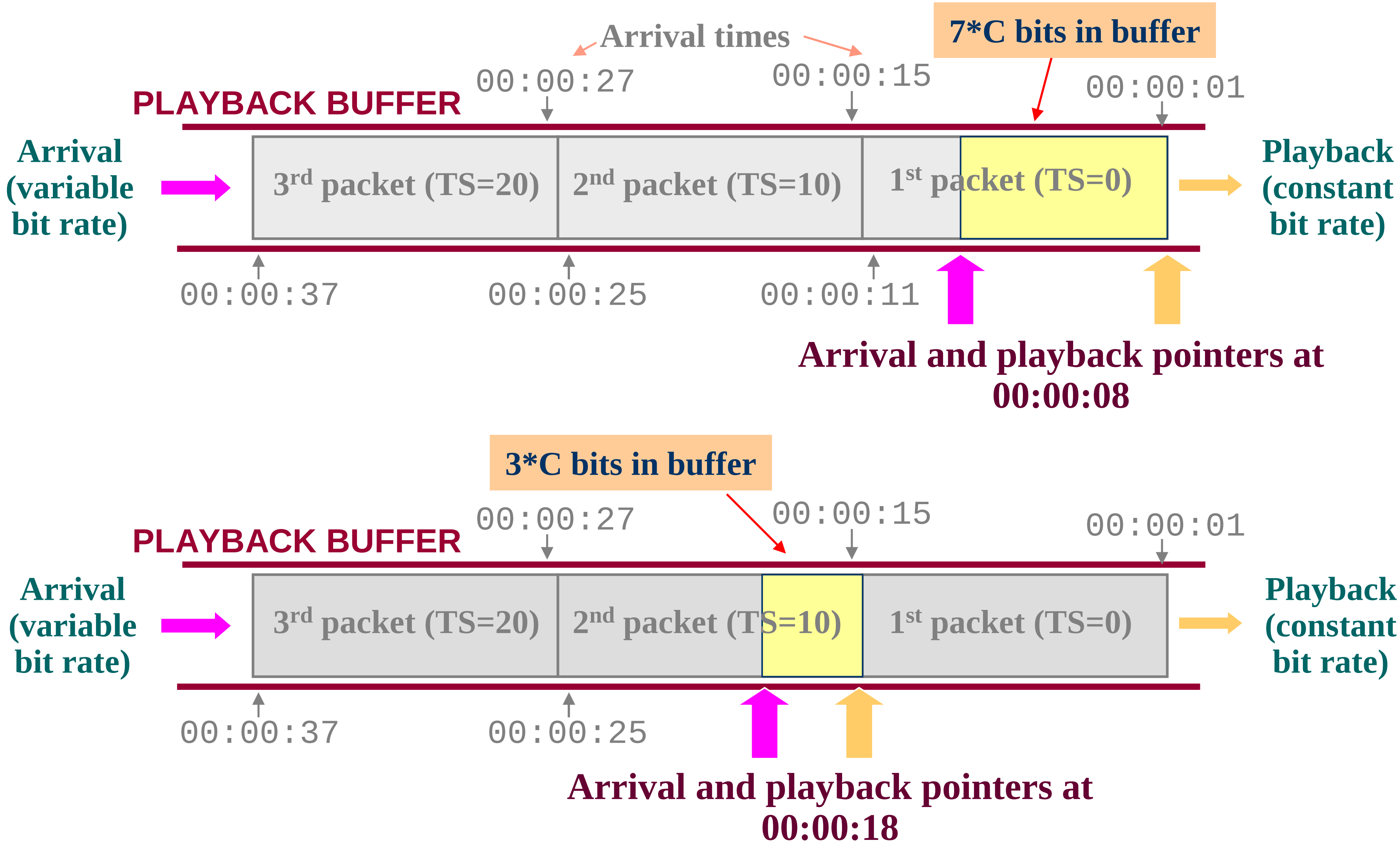
The playback is deferred 7 seconds after the reception of the first packet, all gaps eliminated

Playback time is separated from the arrival time

Real-Time Multimedia Traffic (cont.)

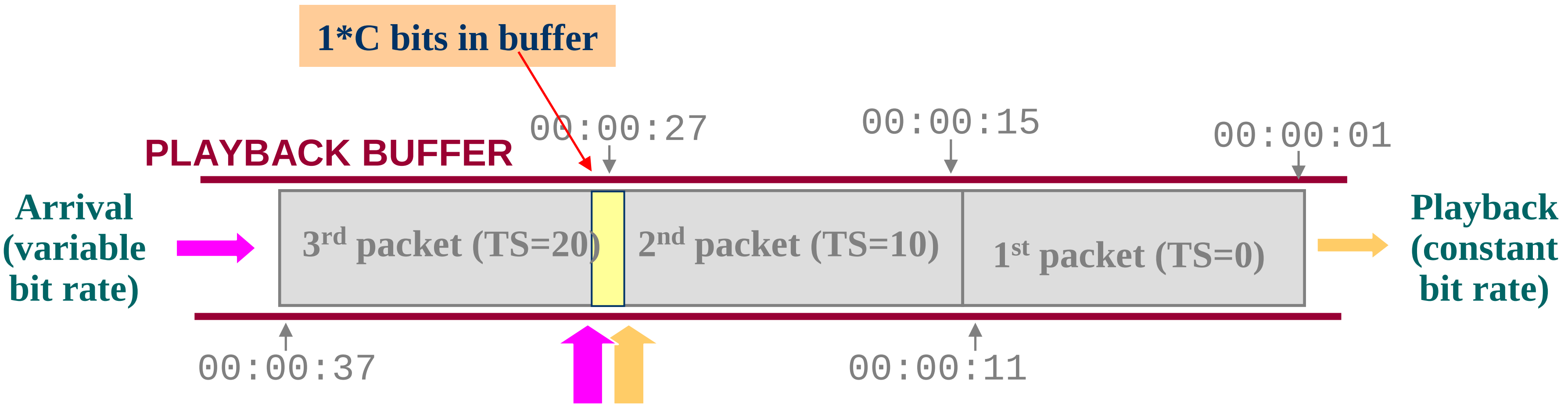
Playback Buffer

In order to separate playback time from the arrival time a playback buffer is needed.

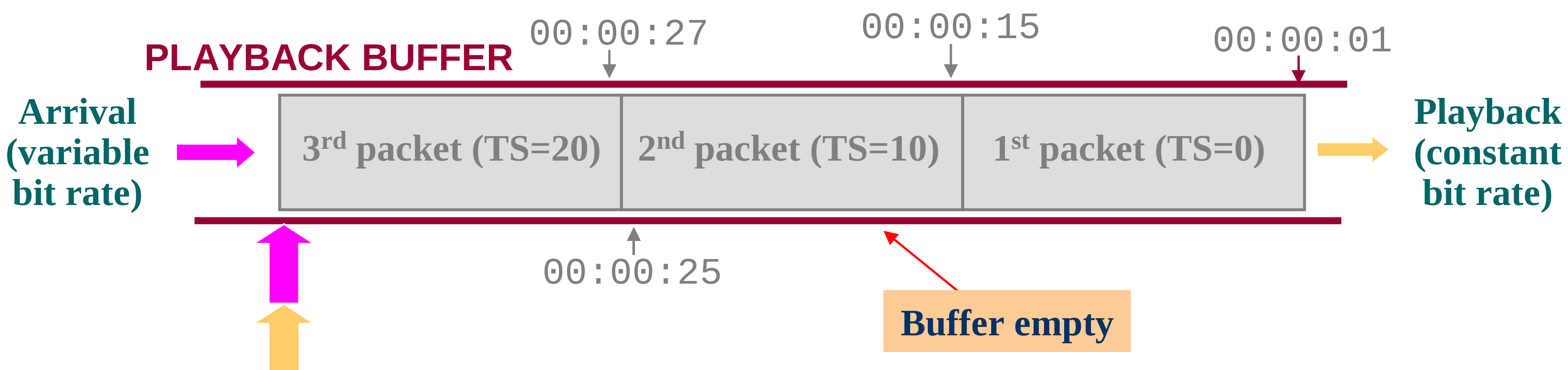


Real-Time Multimedia Traffic (cont.)

Playback Buffer (cont.)



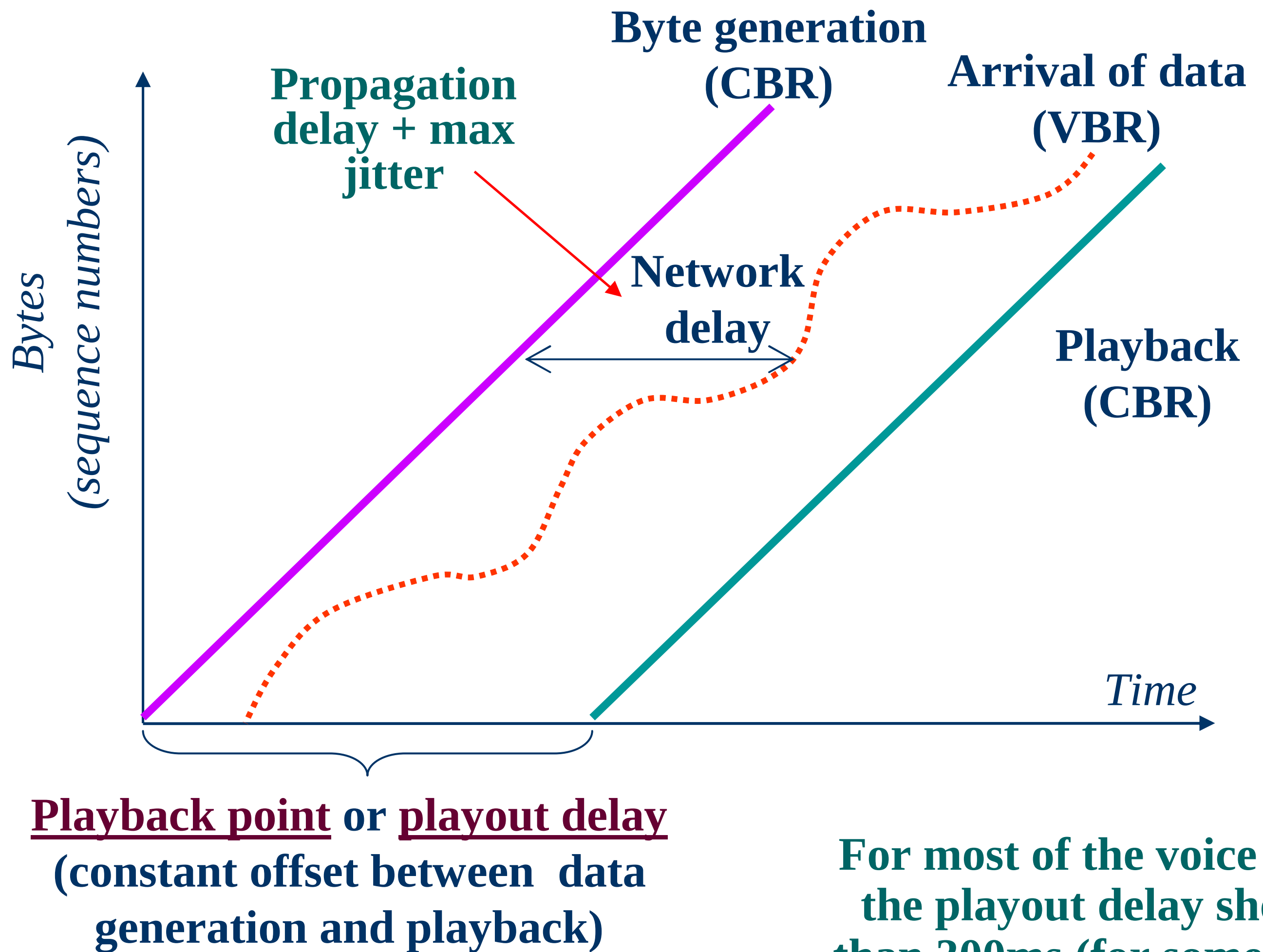
Arrival and playback pointers at 00:00:28



Arrival and playback pointers at 00:00:38

Real-Time Multimedia Traffic (cont.)

Playback Buffer (cont.)



For most of the voice applications the playout delay should be less than 300ms (for some applications even less than 150 ms is required)

Real-Time Multimedia Traffic (cont.)

Adaptive playout algorithm

How to determine the size of playout delay?

$$D = ED + \beta EV$$

where:

D – Playout delay (set for each talk spurt)
ED – Estimated average packet delay
EV – Estimated average packet delay variation
β – safety factor (usually $\beta = 4$)

The computation of ED and EV is done for each packet and is based on timestamps:

$$ED_i := \alpha ED_{i-1} + (1-\alpha)(r_i - t_i)$$
$$EV_i := \alpha ED_{i-1} + (1-\alpha)\,|r_i - t_i - ED_i|$$

where:

α – weighting factor (typically $\alpha = 0.998$)
r_i – time the packet i is received
t_i – the timestamp of the i-th packet

Real-Time Multimedia Traffic (cont.)

Requirements for Real-Time Multimedia Traffic

- In order to ensure playback timing and jitter removal timestamps are required.
- In order to ensure the presence and order of data a sequence number is required.
- Most of the real-time multimedia applications are video conferencing where several clients receive data. Therefore the multicast mode is preferred.
- In order to deal with congestion a mechanism for sender notification and change of encoding parameters must be provided.
- In order to display streams generated by different standards the choice of encoding must be provided.
- In order to display audio and video streams (and user data like titles) within a single A/V session mixers are required.
- In order to use high bit rate streams over a low-bandwidth network the translators are required (multimedia payload encoders and decoders)

Real-Time Multimedia Traffic (cont.)

Why real-time data can not use TCP?

- TCP forces the receiver application to wait for retransmission(s) in case of packet loss, which causes large delays.
- TCP cannot support multicast.
- TCP congestion control mechanisms decreases the congestion window when packet losses are detected ("slow start"). Audio and video, on the other hand, have "natural" rates that cannot be suddenly decreased.
- TCP headers are larger than a UDP header (40 bytes for TCP compared to 8 bytes for UDP).
- TCP doesn't contain the necessary timestamp and encoding information needed by the receiving application.
- TCP doesn't allow packet loss. In A/V however loss of 1-20% is tolerable. (This kind of loss can be compensated by FEC, see later)

Real-Time Multimedia Traffic (cont.)

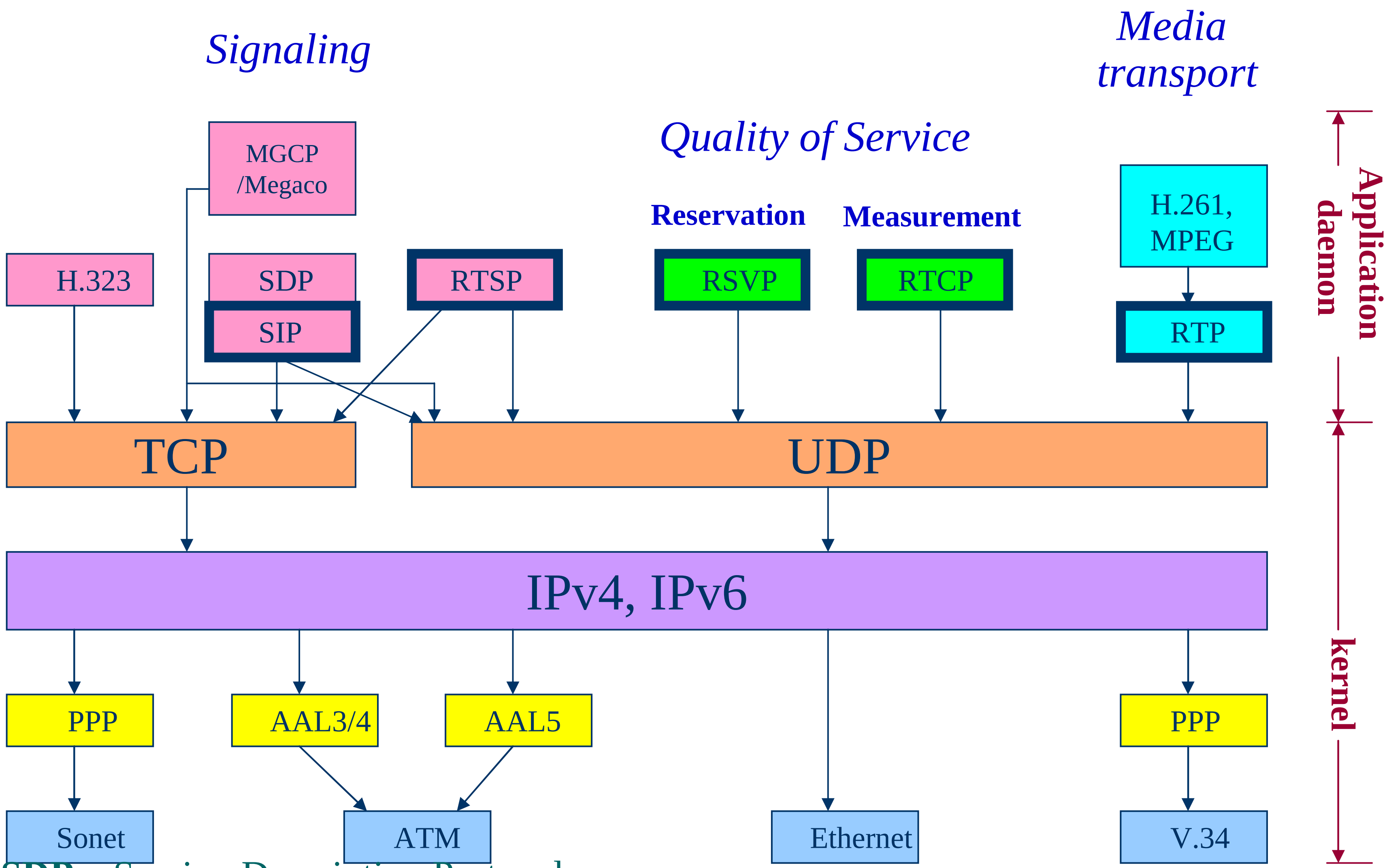
Protocols

There are several related protocols which support the real-time traffic over Internet (here are listed the most important ones which will be discussed in the following slides):

- **RTP** (Real-Time Protocol)
  -- Used for real-time data transport (extends UDP, sits between application and UDP)
- **RTCP** (Real-time Control Protocol)
  -- Used to exchange control information between sender and receiver, works in conjunction with RTP
- **SIP** (Session Initiation Protocol)
  -- Provides mechanisms for establishing calls over IP
- **RTSP** (Real-Time Streaming Protocol)
  -- Allows user to control display (rewind, FF, pause, resume,…)
- **RSVP** (Resource Reservation Protocol, discussed in next chapter)
  -- Intended to add determinism to connectionless information, provides QoS to some extent.

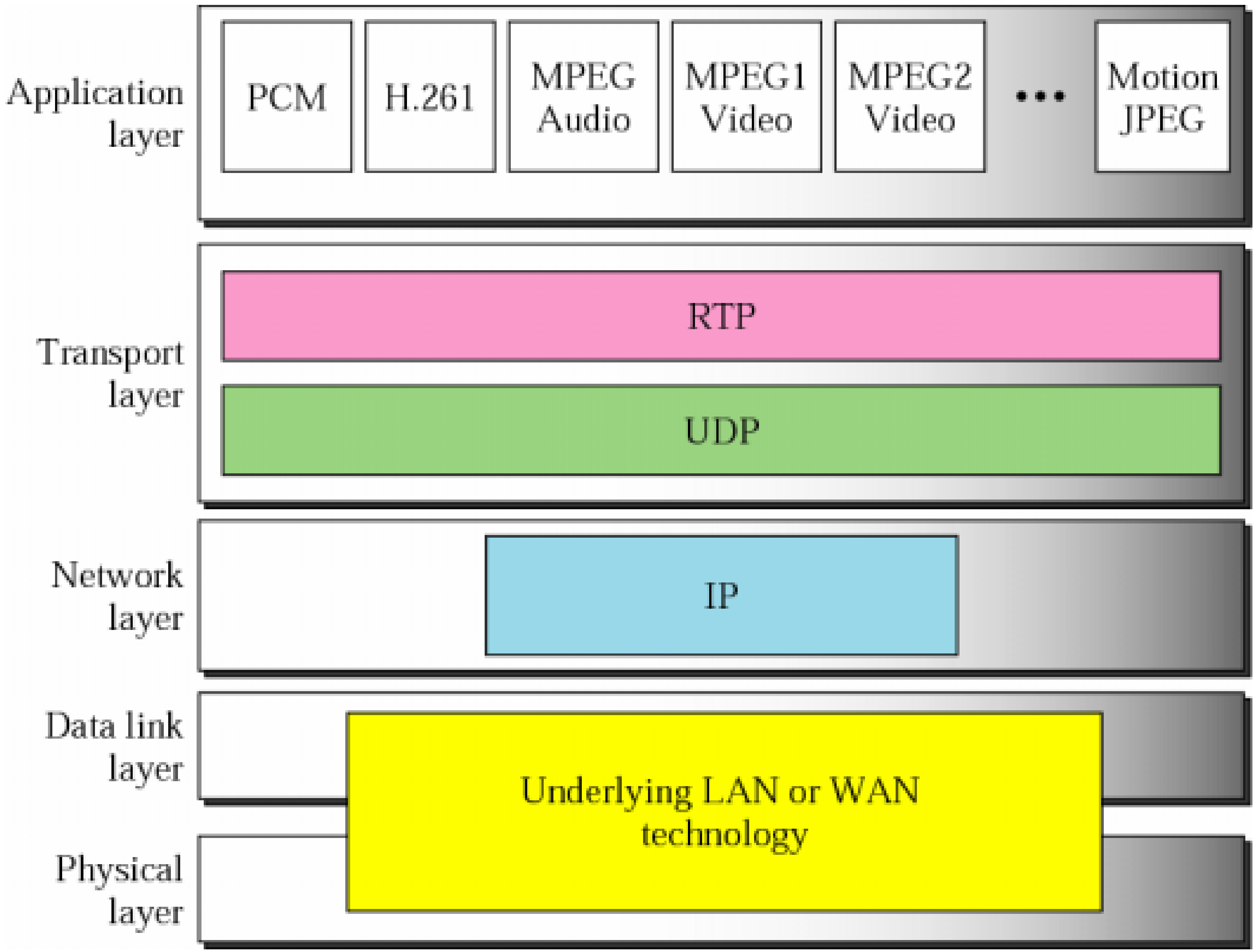
Real-Time Multimedia Traffic (cont.)

Protocols (cont.)



SDP – Session Description Protocol

Real Time Protocol (RTP)



RTP (cont.)

RFC 3550 July 2003 (originally RFC 1889, January 1996)

There was a dilemma whether to implement RTP as a sublayer of transport layer or a s a part of application layer. At this point it is common that RTP is implemented as an application library, which executes in user space rather than in kernel space like all other protocol layers below RTP. (Example: JMF -Java Media Framework).

RTP doesn't ensure real-time delivery itself, but it provides means for:

- Jitter elimination/reduction (with application's playout buffers)
- Synchronization of several audio and/or video streams that belong to the same multimedia session.
- Multiplexing of audio/video streams that can belong to different ?????
- Translation of video a/v streams from one encoding type to another

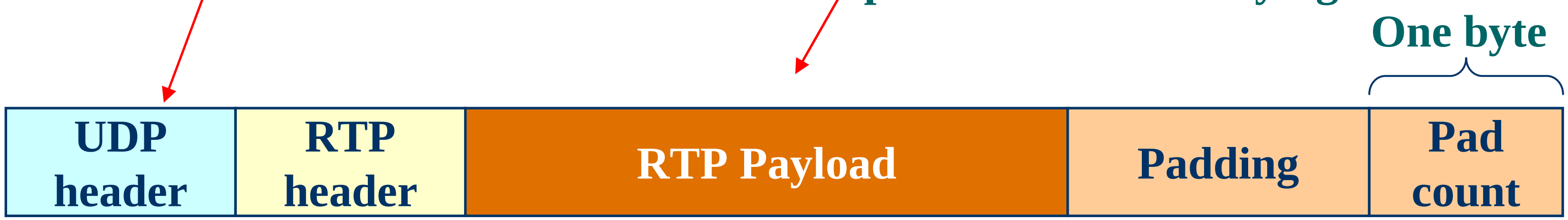
With the help of RTCP, RTP also provides "hooks" for adding reliability and flow/congestion control which is implemented within the multimedia application (this property is sometimes called: application-level framing)

RTP (cont.)

RTP Packets

The size of the payload is a compromise between overhead and packetization delay. In addition, it is desirable that the multimedia packets are as small as possible so that the packet loss doesn't decrease the quality of reception (for example 20 ms of uncompressed voice has 160 samples (bytes). Loss of 20 ms of voice is tolerable. Loss of larger packets can be annoying.

RTP packets are encapsulated in UDP datagrams (TCP can not be used due to overhead and to the fact that most of real time traffic is multicast)



Variable size, depending on the application (i.e. when needed). Designed to be small to decrease the overhead.

Some encoding algorithms require fixed block sizes, therefore a padding is needed

Port number is an even number randomly assigned at the session initiation time

RTP (cont.)

RTP Header

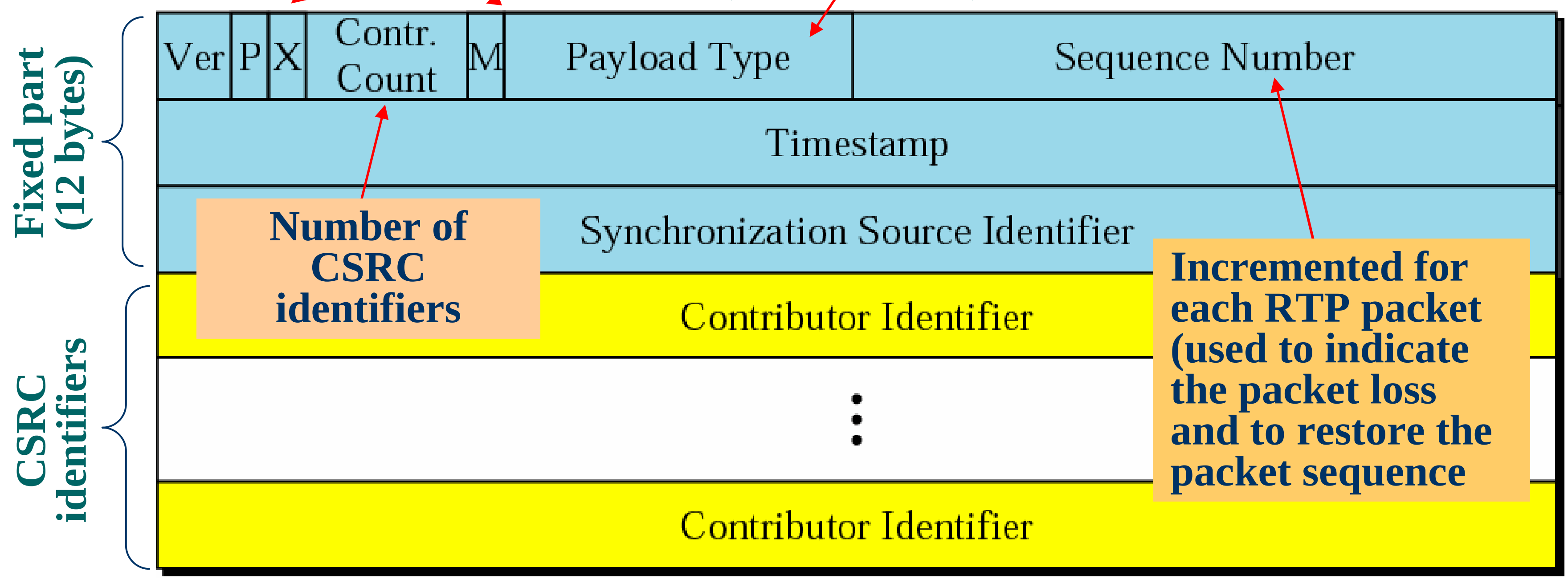
Version number (current version = 2)

Padding bit (1 – packet contains padding)

Extension bit (1 – fixed header is followed by an extension header which is required by some applications)

Marker bit 1 – marks the frame boundary (e.g. beginning of a talkspurt in audio, the end of frame in video)

(see next slide)



SSRC – Synchronization source (source that is generating the RTP packets for this session)
CSRC – Contributing source (used by a mixer to identify the contributing sources)

RTP (cont.)

Payload Type

PT	*Name*	*Type*	*RTP timestamp Clock rate (Hz)*	*References*
0	PCMU	Audio	8000	RFC 3551
1	1016	Audio	8000	RFC 3551
2	G721	Audio	8000	RFC 3551
3	GSM	Audio	8000	RFC 3551
4	G723	Audio	8000	
5	DVI4	Audio	8000	RFC 3551
6	DVI4	Audio	16000	RFC 3551
7	LPC	Audio	8000	RFC 3551
8	PCMA	Audio	8000	RFC 3551
9	G722	Audio	8000	RFC 3551
10	L16	Audio	44100	RFC 3551
11	L16	Audio	44100	RFC 3551
12	QCELP	Audio	8000	
13	CN	Audio	8000	RFC 3389

RTP (cont.)

Payload Type (cont.)

PT	*Name*	*Type*	*RTP timestamp Clock rate (Hz)*	*References*
14	MPA	Audio	90000	RFC 3551
15	G728	Audio	8000	RFC 3551
16	DVI4	Audio	11025	
17	DVI4	Audio	22050	
18	G729	Audio	8000	
25	CellB	Video	90000	RFC 2029
26	JPEG	Video	90000	RFC 2435
28	nv	Video	90000	RFC 3551
31	H261	Video	90000	RFC 2032
32	MPV	Video	90000	RFC 2250
33	MP2T	A/V	90000	RFC 2250
34	H263	Video	90000	
96-127	Dynamic Payload Types			

MPEG1 and MPEG2

For detailed payload type list see: http://www.iana.org/assignments/rtp-parameters

RTP (cont.)

Timestamp and Sequence Number

Audio:

Timestamp clock rate for audio 8000 Hz (= 125 μs)

One RTP packet caries 20 ms of audio samples, each RTP packet sent by separate UDP datagram to avoid packetization delay

Timestamp increments by 20,000 μs/125 μs = 160 (even if the packet is not sent after silence suppression)

Packet rate = 1 sec/20 ms = 50 Hz

Number of bits per RTP payload for uncompressed audio = 160*8 = 1280, for compressed audio typically 8 times less.
Sequence number increments by one for each RTP packet

RTP packet *k-1*		RTP packet *k*	
ts = x sn = y	Audio stream	ts = x+160 sn = y+1	Audio stream

RTP (cont.)

Timestamp and Sequence Number (cont.)

Video:

Timestamp clock rate for video is 90,000 Hz

One RTP packet caries one video frame (it is possible that one frame is spread between several RTP packets)

Each RTP packet sent by separate UDP datagram.
RTP packet rate = 30 Hz (sometimes 25 Hz)

Timestamp increments by (1/30)/(1/90000) = 9000/30 = 3000, or 9000/25 = 3600

Number of bits per RTP payload for uncompressed video conferencing = 352x240x12 = 1,000,000, for compressed audio typically 30 times less.
Required bit rate must cover the overhead (protocol headers and bit stream headers)

Sequence number increments by one for each RTP packet

RTP packet *k-1*		RTP packet *k*	
ts = x sn = y	Video stream	ts = x+3000 sn = y+1	Video stream

RTP (cont.)

Source Identifiers

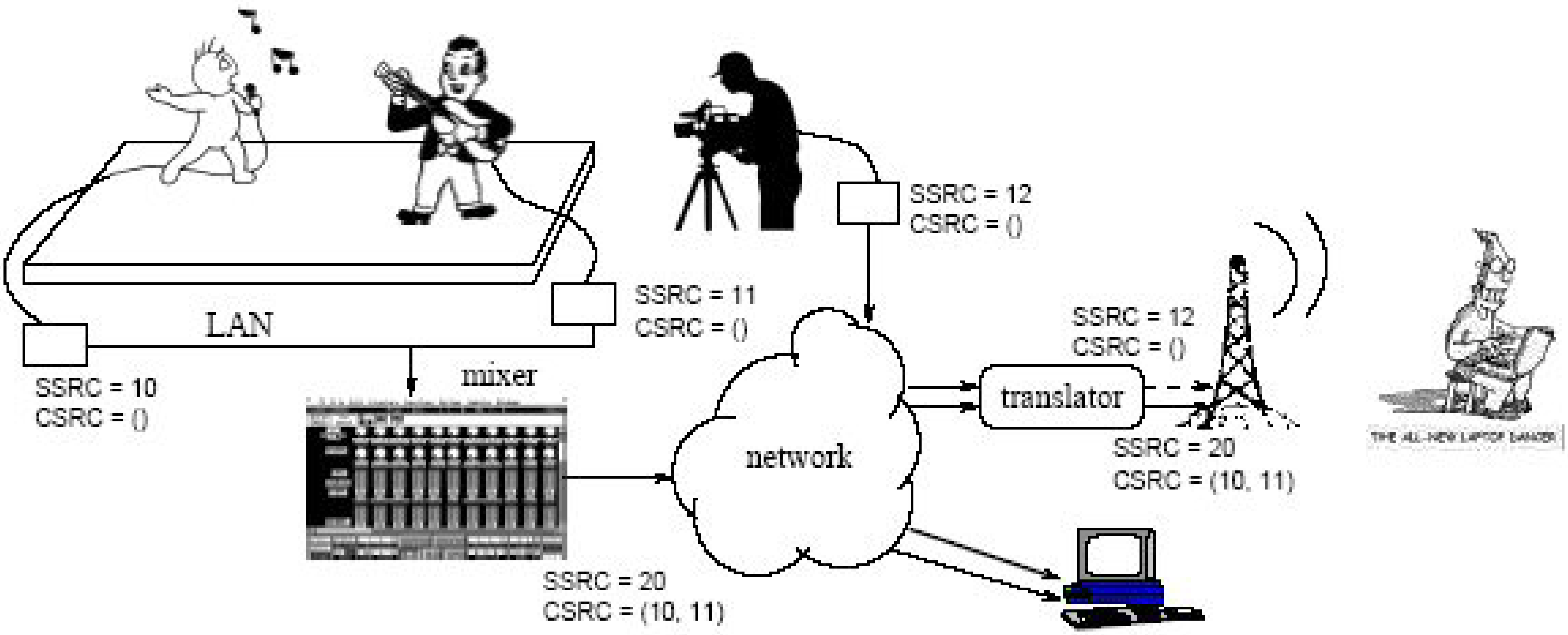
Synchronization source identifier (SSRC)

Uniquely defines the multimedia source.

The initial value is determined randomly.

Contributing source identifier (CSRC)

Used by mixers to identify sources in a mix.



Real-Time Control Protocol (RTCP)

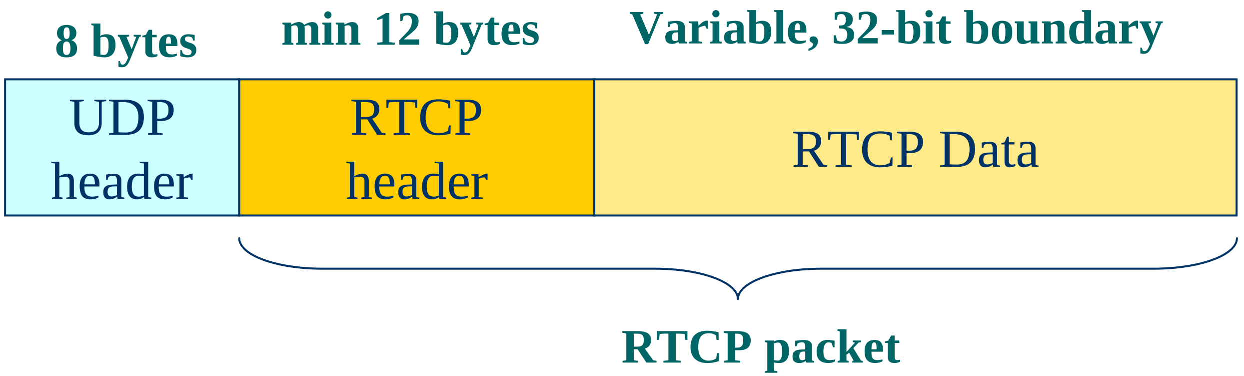
RTP is a protocol that provides basic transport layer for real time applications but does not provide any mechanism for error and flow control, congestion control, quality feedback and synchronization. For that purpose the RTCP is added as a companion to RTP to provide end-to-end monitoring and data delivery, and QoS

RTCP is responsible for three main functions:

- Feedback on performance of the application and the network
- Correlation and synchronization of different media streams generated by the same sender (e.g. combined audio and video)
- The way to convey the identity of sender for display on a user interface

RTCP (cont.)

RTCP Packets



RTCP UDP port number = RTP UDP port number + 1
(Usually one port number pair per multimedia session)

Usually several RTCP packets are combined in a bundle of several packets which are encapsulated in the same UDP datagram (to reduce the overhead due to headers).

RTCP (cont.)

Common RTCP Header

2 bits	*1 bits*	*5 bits*	*8 bits*	*16 bits*
Version	P	RR count	Packet type	Message length

Same as RTP version (v = 2)

Padding bit (same as RTP)

The number of reception report blocks contained in this packet

RTCP packet type (see next slide)

RTCP (cont.)

Packet Type

Packet Type	*Acronym*	*Description*
192	FIR	Full INTRA-frame request
193	NACK	Negative acknowledgement
200	SR	Sender report for transmission and reception statistics from active senders (periodically transmitted)
201	RR	Receiver report for reception statistics from participants that are not active senders (periodically transmitted)
202	SDES	Source description items (including CNAME – canonical name)
203	BYE	Goodbye – Indicates end of participation
204	APP	Application specific functions
207	XR	RTCP extension

References: RFC 2032, 3550, 3611

RTCP (cont.)

Sender Report

RTCP receivers provide reception quality feedback using a SR or a RR (if receiver is also a sender)

Version	P	RR count	PT = 200	Message length
SSRC of sender				

Necessary for synchronizing A/V streams

Sender information block

- NTP timestamp (the absolute wall clock time when report was sent)
- RTP timestamp (relative timestamp used in RTP packets)
- Sender’s packet count (RTP packets transmitted by this sender so far in this session)
- Sender’s octet count (same but total number of RTP payload octets)

Reception report block (for each source)

- SSRC
- Fraction lost (fraction of RTP pkts from this source lost since last rep.)
- Commutative number of lost packets
- Extended highest seq. number received (the last significant 16 bits)
- Interarrival jitter (estimate of jitter of RTP data from SSRC
- Last SR timestamp received from this source
- Delay since receiving the last SR report from this source

NTP – Network Time Protocol used to synchronize clocks of all computers in Internet. Uses UDP/IP, UTC (Universal Time Coordinated, former Greenwich Mean Time)
NTP time is independent of time zone and day-light saving time, has one millisecond accuracy.

RTCP (cont.)

Receiver Report

Reception statistics from receivers that are not senders.
Same format as SR, but without the sender's information block.

Version	P	RR count	PT = 201	Message length
SSRC of sender				

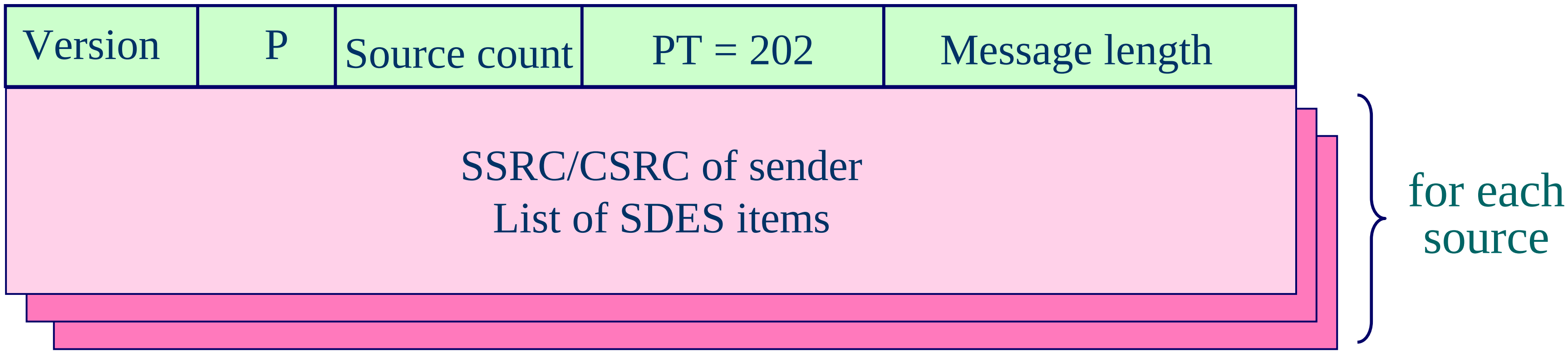
SSRC (of the sender source)
Fraction lost (fraction of RTP pkts from this source lost since last rep.)
Commutative number of lost packets
Extended highest seq. number received (the last significant 16 bits)
Interarrival jitter (estimate of jitter of RTP data from SSRC
Last SR timestamp received from this source
Delay since receiving the last SR report from this source

Report block (for each source)

RTCP (cont.)

RTCP Source Description

Used by source to provide more information about itself. Useful for user interfaces (GUI displays who is the source in a “human understandable” format)



SDES item types:

CNAME – Canonical end-point identifier unique among all participants like user@host (CNAME doesn’t change even if SSRC changes)

NAME – Real user name of source

EMAIL – E-mail address of

PHONE – Telephone number

LOC – Geographical location

TOOL – Name of application generating the stream

NOTE – Transient message describing the current stat of the source

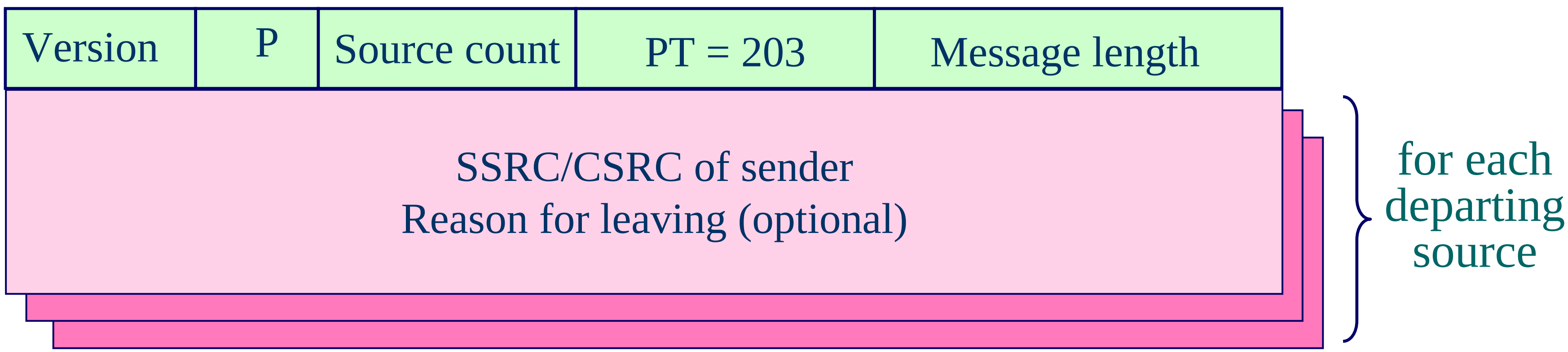
PRIV – Private experimental or application-specific extensions

END – End of SDES list

RTCP (cont.)

RTCP Bye Message

Used to confirm to receivers that a prolonged silence is due to departure of one or more sources rather than network failure.



NOTE: In audio with silence suppression, no packets are sent between two talk spurts. The receiver might think that the sender has departed or the there is a network failure. The BYE message at least removes the suspicion of network failure if a source departs.

RTCP (cont.)

RTCP Application-Dependent Packet

Used to confirm to receivers that a prolonged silence is due to departure of one or more sources rather than network failure.

Version	P	Source count	PT = 204	Message length
SSRC/CSRC of sender Name (ASCII) Application-dependent data				

RTCP (cont.)

Scalability of RTCP

The volume of RTCP traffic may exceed the RTP traffic during a conference session involving larger number of participants (normally only one participant talks at the time while other participants are listening. In the meanwhile RTCP messages are sent periodically regardless if participant is talking or not.) Therefore the RTCP packet transmission rate is dynamically changed depending on the number of participants. Standard dictates that 20% of the session bandwidth is allocated to RTCP. In other words RTCP RR and SDES packets are sent for every 5 RTP packets transmitted.
In addition, 5% of the RTCP bandwidth is allocated to particular participant (CNAME).

The RTCP transmission interval of a participant is a function of the total number of participants in the RTP session that ensures required 5% of bandwidth allocation. For that purpose each participant has to continuously estimate the session size. The transmission interval is randomized to avoid synchronization effect.

RTCP messages are stackable – to amortize header overhead multiple RTCP messages can be combined and sent in a compound RTCP message.

Recovering from Packet Loss

A packet is lost if:

packet never arrived
packet arrived but corrupted (RTC test, IP and UDP checksum tests failed)
packet arrived after its scheduled playout time

RTP and RTCP do not provide error control (as TCP does via ARQ).

How can lost packets be recovered?

Three approaches:

- Forward Error Correction (FEC)
- Interleaving
- Receiver-based Repair

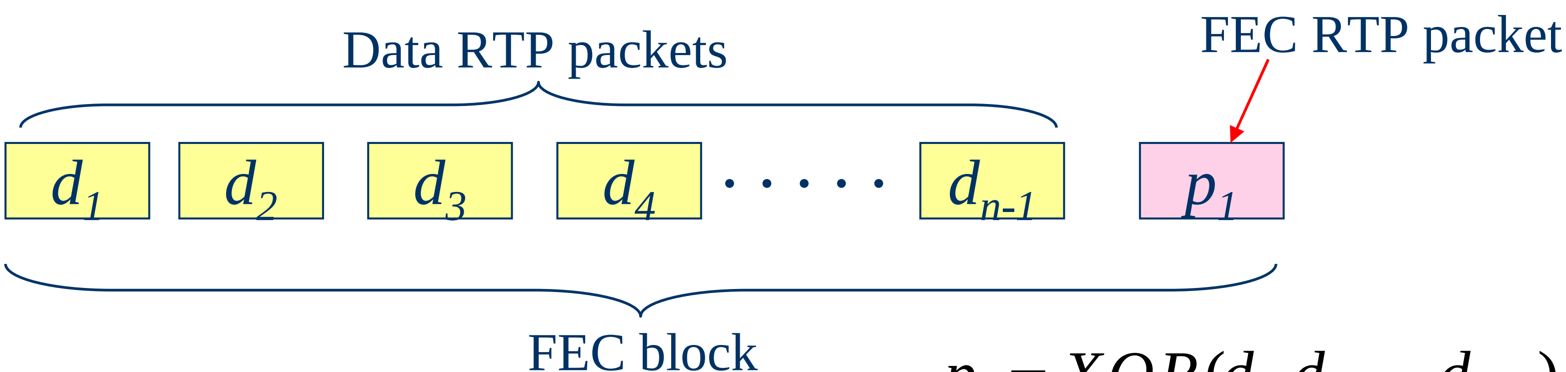
Recovering from Packet Loss (cont.)

Forward Error Correction (FEC)

Generally FEC is based on adding redundancy to transmitted data.
There are two mechanisms:

- **Using parity (FEC) packets**
- **Using redundant A/V streams**

Using parity packets



$$p_1 = XOR(d_1, d_2, \ldots, d_{n-1})$$

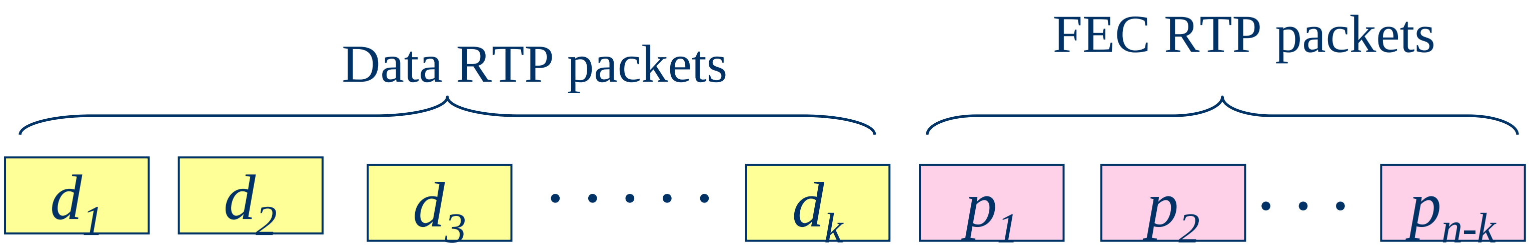
If any of k packet gets lost, it can be recovered. For example if d_3 is lost, then:

$$d_3 = XOR(d_1, d_2, d_4, \ldots, d_{n-1}, p_1)$$

Recovering from Packet Loss (cont.)

FEC (cont.)

This can be extended to several parity packets. The parity packets here can help to recover the loss of any *n-k* out of *n* packets



FEC packets are marked as <u>dynamic packet type</u> in RTP header.

A popular algorithm to generate redundant data is <u>Reed Solomon Erasure Correcting code</u> (RSE code).

In determining the FEC parameters *n* and *k*, the following must be taken into account:

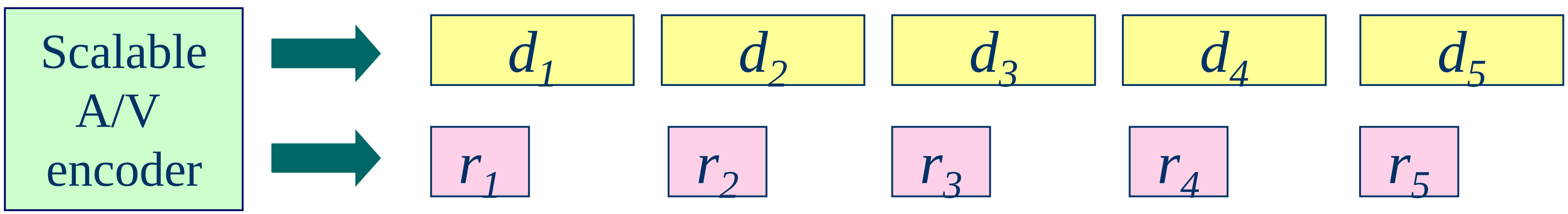
- Additional FEC packets increase the overhead, i.e. the required network bandwidth: small FEC block cause large overhead.
- Large FEC blocks cause delay in receiver (receiver needs to wait for all packets in FEC block in order to be able to do the packet recovery)

Recovering from Packet Loss (cont.)

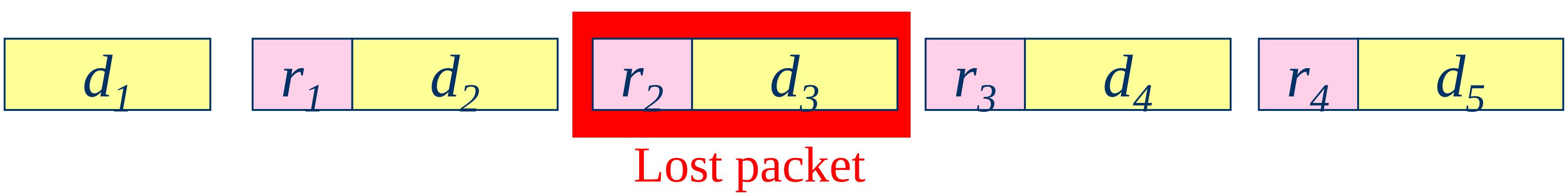
FEC (cont.)

Using Redundant A/V Streams

A scalable encoder generates two redundant streams: the enhanced-level and the base-level compressed bit streams. Enhanced-level stream has smaller compression rate, higher quality and longer streams, while base-level streams have higher compression rate, lover quality and smaller streams (example: PCM coded audio at 64 kbps and GSM encoded audio at 13 kbps).



The streams are then combined for transmission in a single bit stream.



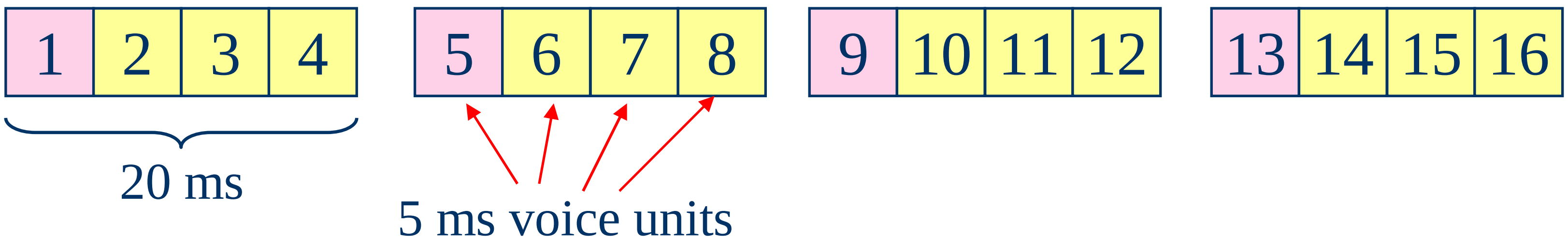
This method increases bandwidth, but the delays are smaller than in the previous case

Recovering from Packet Loss (cont.)

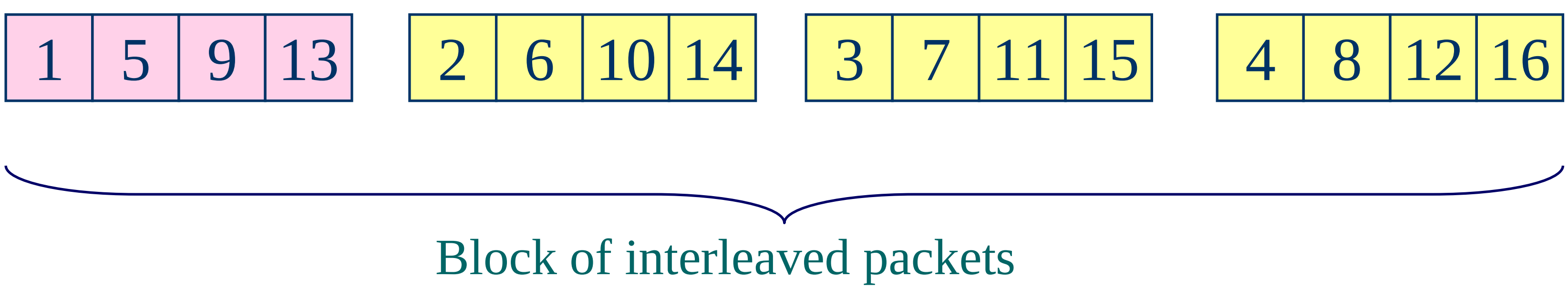
Interleaving

The FEC-based methods for data recovery increase the required bandwidth and latency. An alternative method which doesn’t increase the bandwidth is interleaving. The method is shown on Internet phone application.

Original RTP packets (each containing 20 ms of voice samples):



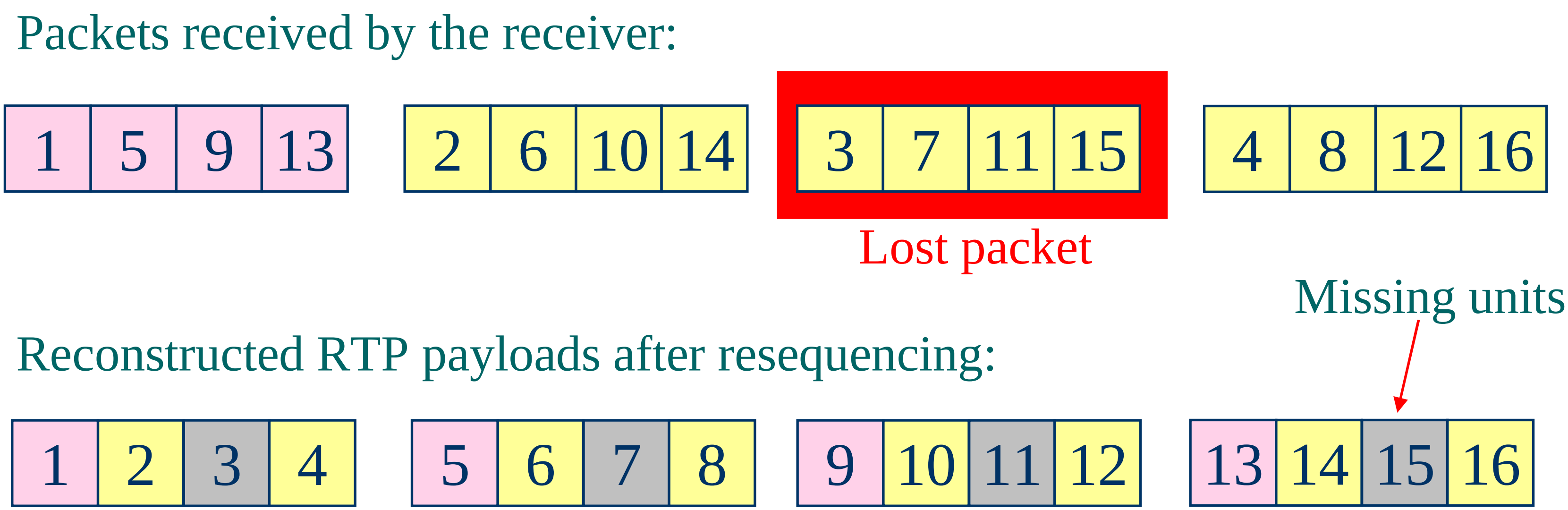
The transmission stream is generated by rearranging the voice units within RTP packets:



Recovering from Packet Loss (cont.)

Interleaving (cont.)

The FEC-based methods for data recovery increase the required bandwidth and latency. An alternative method which doesn’t increase the bandwidth is interleaving. The method is shown on Internet phone application.



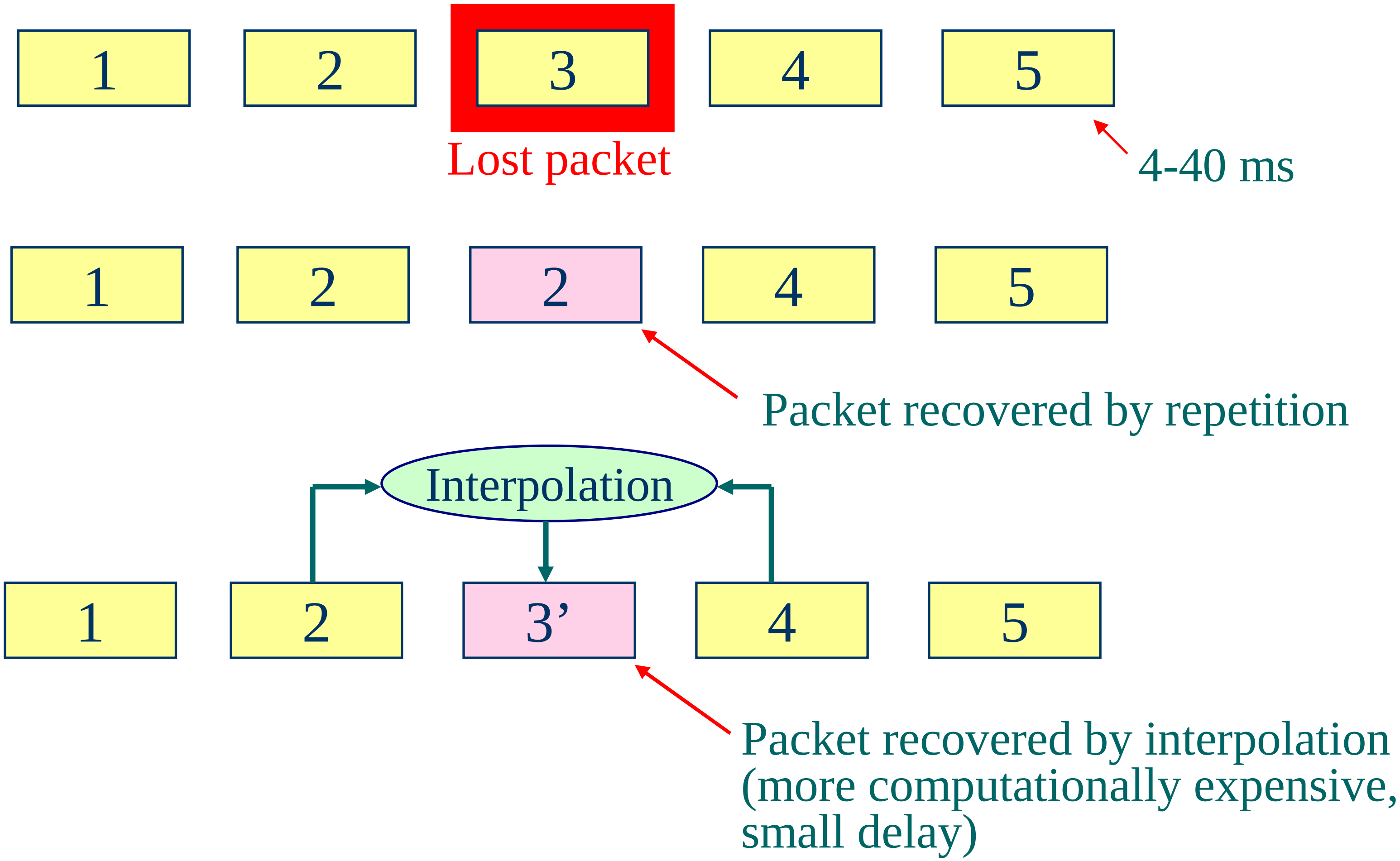
Lost packet causes small (5 ms) gaps in the restored audio stream. These gaps can’t be noticed. A loss of several packets will cause larger or more frequent gaps, which decreases the reception quality gracefully.

This method doesn’t increase bandwidth, but increases delays, which are proportional to the length of interleaving blocks. This limits the use in interactive applications like Internet phone. Good for streaming stored audio.

Recovering from Packet Loss (cont.)

Receiver-Based Repair

This method doesn't increase bandwidth requirements nor delays. Works with smaller packets. Based under the assumption that there is a small difference between two neighboring packets, which is specially true in case of voice.



Session Initiation Protocol (SIP)

RFC 3261, June 2002

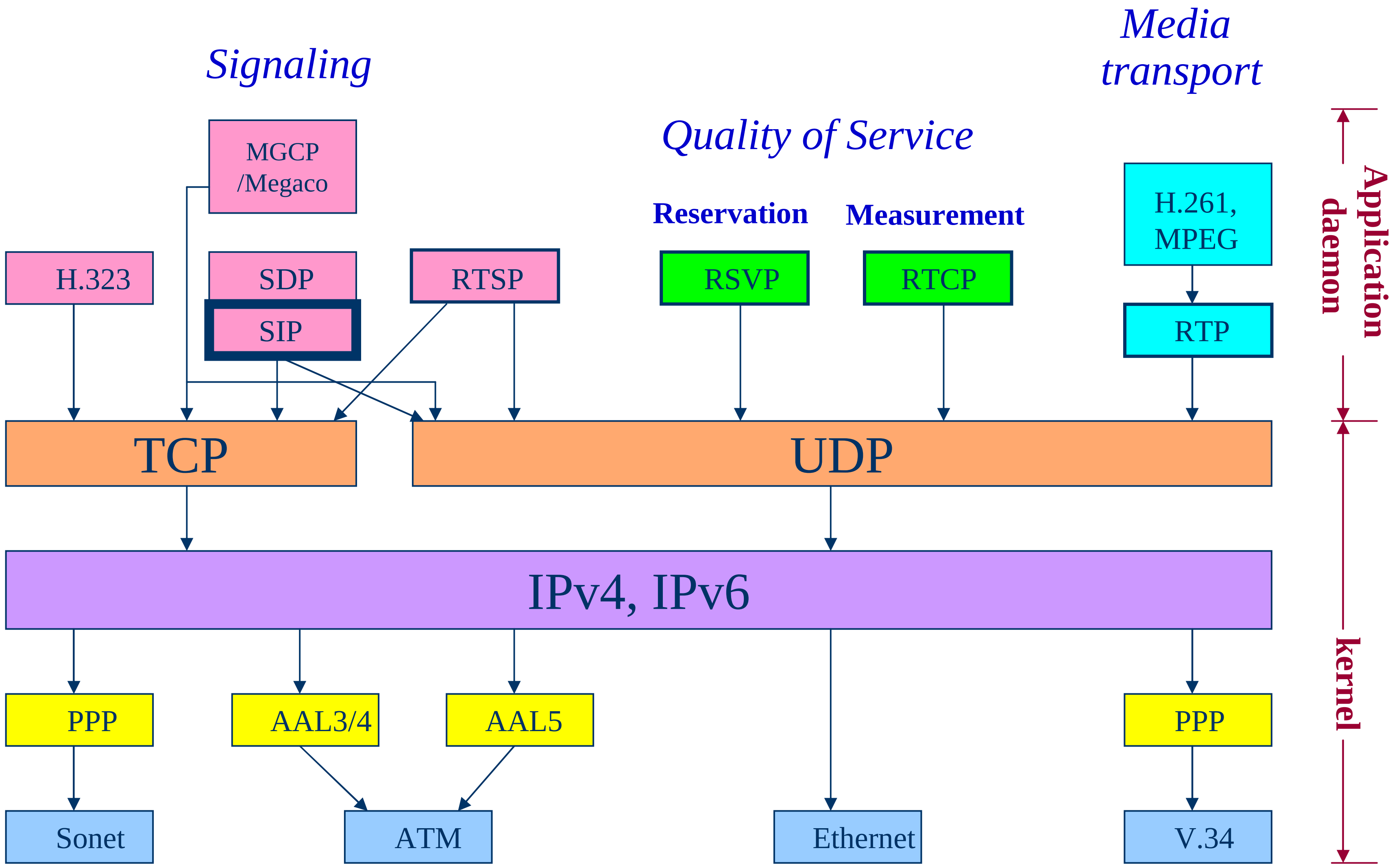
SIP is used to set up, modify and terminate an RTP-based multimedia session (like IP telephone conversation or videoconferencing).

There are several issues related to this:

- How to determine the current IP address of the callee? He/she may have several devices like Internet phone on PC (one at home, one at office), PDA, real phone/cell phone.
- The person can have dynamic IP address (DHCP), or the person is not currently available (not signed in), or doesn't want to answer at this moment.
- How can we route the call if we even do not know where the person is at the moment. How can we be notified once the person is located, or the person is available.
- Once the connection is established, how can we define the encoding type for the audio/video stream, or change the encoding type during the session,
- How can we invite new participants during the call, perform call transfer and call holding.

All these isues are addressed in SIP. An alternative protocol to SIP is **H.323**, which is more complex and more difficult to implement/configure. SIP is a simple protocol well suited for Internet.

SIP (cont.)



SDP – Session Description Protocol (used by SIP to describe the media session, not discussed here)

SIP (cont.)

Basic functions supported by SIP:

Function	*Description*
User location and registration	End points (telephones) notify SIP proxies of their location; SIP determines which end points will participate in a call.
User availability	SIP is used by end points to determine whether they will "answer" a call.
User capabilities	SIP is used by end points to negotiate media capabilities, such as agreeing on a mutually supported voice codec.
Session setup	SIP tells the end point that its phone should be "ringing;" SIP is used to agree on session attributes used by the calling and called party.
Session management	SIP is used to transfer calls, terminate calls, and change call parameters in mid-session (such as adding a 3-way conference).

SIP (cont.)

SIP is a text-based transaction protocol similar to HTTP. It uses commands (called methods) and responses. They are summarized as follows:

SIP Commands (Methods)

SIP Method	*Description*
INVITE	Invites a user to a call
ACK	Used to facilitate reliable message exchange for INVITEs
BYE	Terminates a connection between users or declines a call
CANCEL	Terminates a request, or ongoing transmission
SUBSCRIBE	Request notification of call events
NOTIFY	Event notification after an explicit/implicit subscription
REFER	Call transfer request
OPTIONS	Solicits information about a server's capabilities
REGISTER	Registers a user's current location
INFO	Used for mid-session signaling

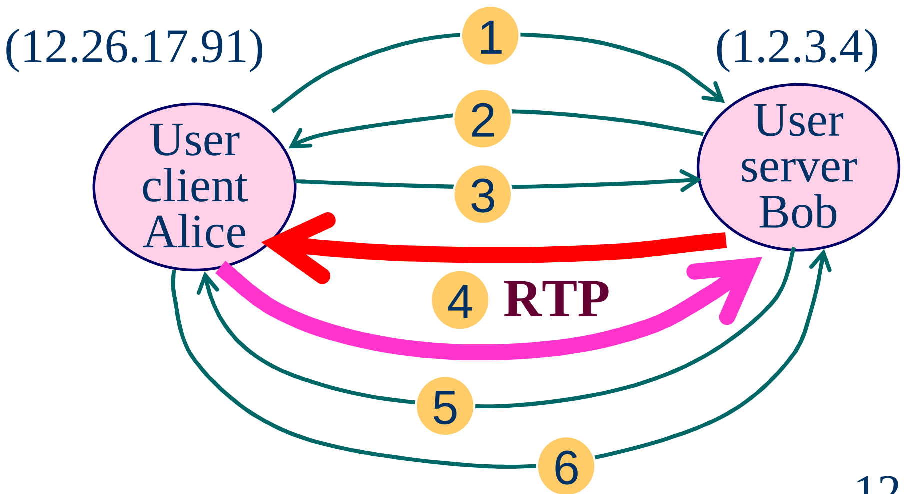
SIP (cont.)

SIP Responses

Code	*Description*
1xx	Informational (e.g. 100 - Trying, 180 - Ringing)
2xx	Successful (e.g. 200 - OK, 202 - Accepted)
3xx	Redirection (e.g. 302 - Moved temporarily)
4xx	Request Failure (e.g. 404 - Not found, 480 - Temporarily unavailable, 486 – Busy there)
5xx	Server Failure (e.g. 501 - Not Implemented)
6xx	Global Failure (e.g. 603 - Decline)

SIP (cont.)

Simple case when the calling SIP client knows the current IP address of the called client

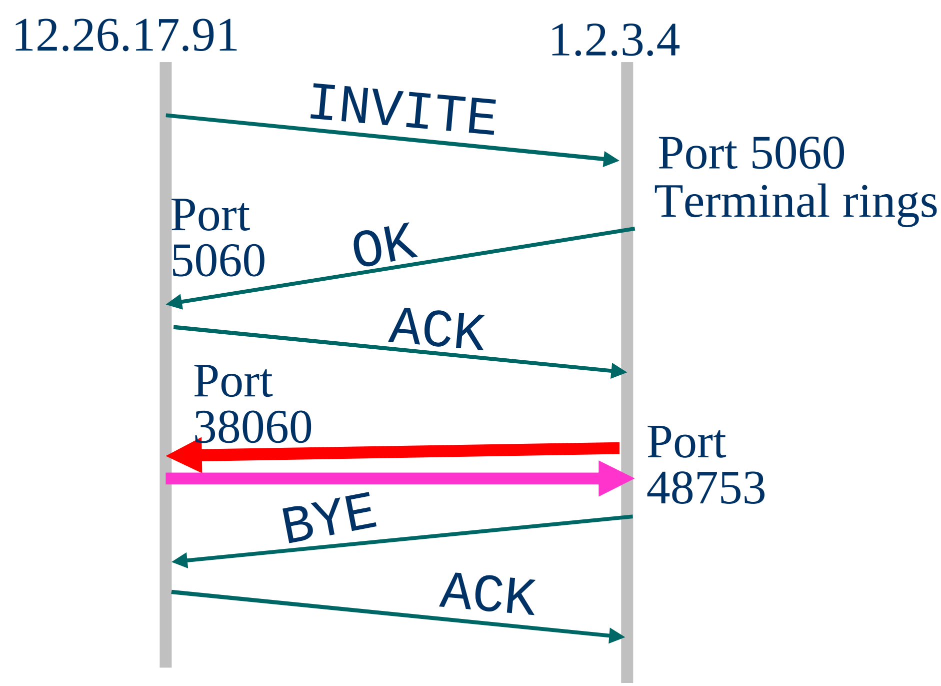


SIP response/request (over well known TCP or UDP port 5060)

Media session over RTP and UDP, port 48753 Encoding: GSM (AVP 3)

Media session over RTP and UDP, port 38060 Encoding: µ Law audio (AVP 0)

1	INVITE sip:bob@1.2.3.4 C=IN IP4 12.26.17.91 m= audio 38060 RTP/AVP 0
2	200 OK C=IN IP4 1.2.3.4 M=audio 48753 RTP/AVP 3
3	ACK
4	Media session over RTP
5	BYE
6	200 OK



SIP (cont.)

In most cases the caller doesn't know callee's IP address. Instead the caller knows the callee's e-mail address. In this case the address translation is needed.

Address translation and routing of calls are provided by additional SIP components called SIP servers.

SIP servers are logical devices (processes) that may be co-located on the same host.

In the following slides a couple of typical scenarios will be discusses, in order to capture the functionality of SIP components.

SIP (cont.)

User Agent

User agent client (end-device, calling party)

User agent server (end-device, called party)

SIP Proxy Server

An intermediary entity that acts as both a server and a client for the purpose of making requests on behalf of other clients. A proxy server primarily plays the role of routing, which means its job is to ensure that a request is sent to another entity "closer" to the targeted user. Proxies are also useful for enforcing policy (for example, making sure a user is allowed to make a call).

SIP Registrar

Maintains user’s whereabouts in location database (location service)

Registrar is a server that accepts REGISTER requests and places the information it

receives in those requests into the location service for the domain it handles.

accepts registration requests from user

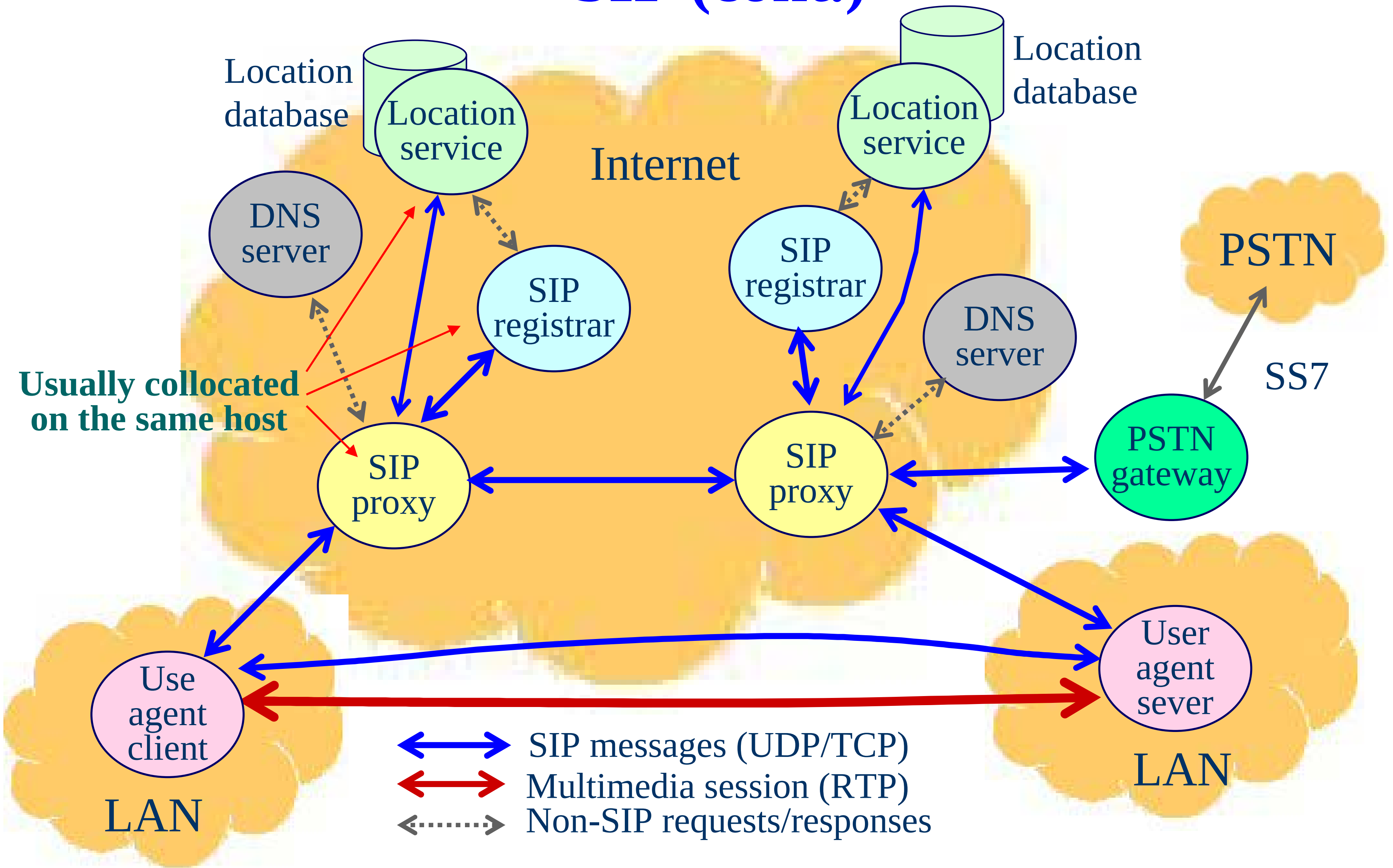
SIP Redirect Server

Redirects calls to other servers. In most cases a SIP proxy is used instead of redirect server.

Location Service

Used by SIP proxy or SIP redirect server to obtain information about callee’s possible location(s). Maintains a local database of SIP address/IP address mappings.

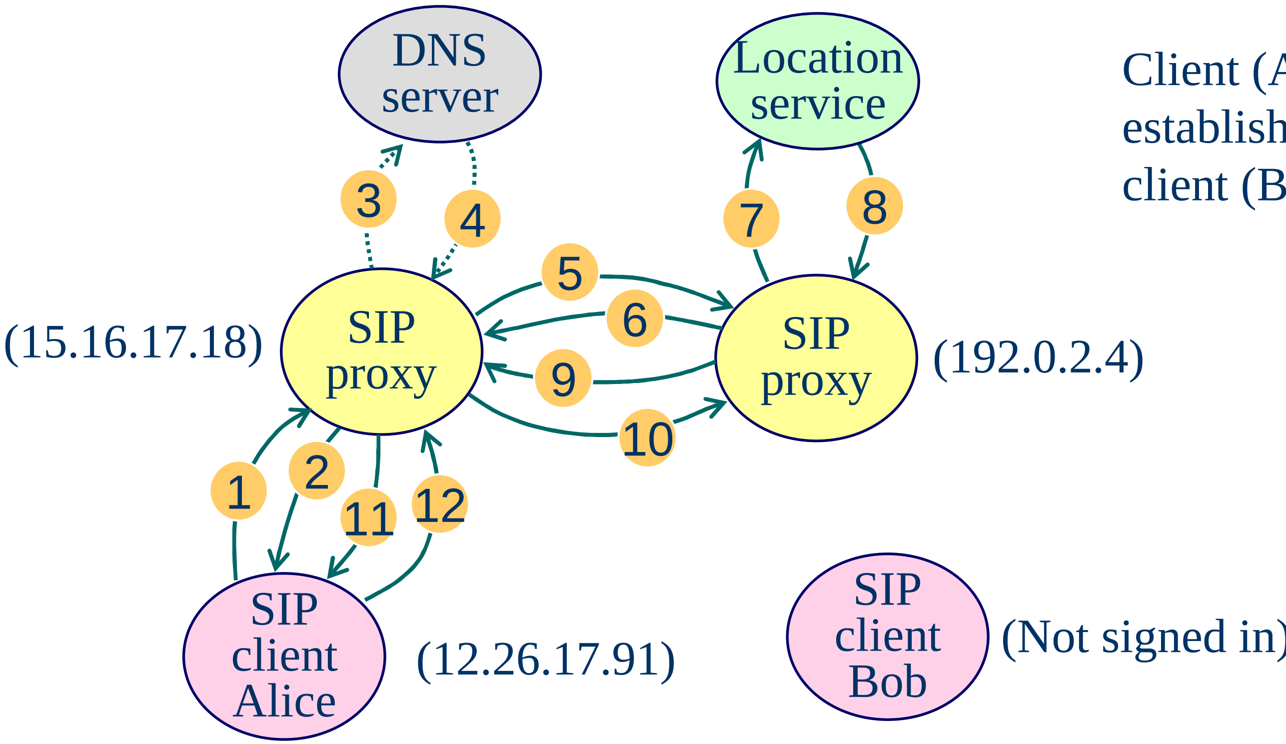
SIP (cont.)



SS7 - Signaling System 7 is an architecture for performing out-of-band signaling in support of the call-establishment, billing, routing, and information-exchange functions of the public switched telephone network (PSTN).

SIP (cont.)

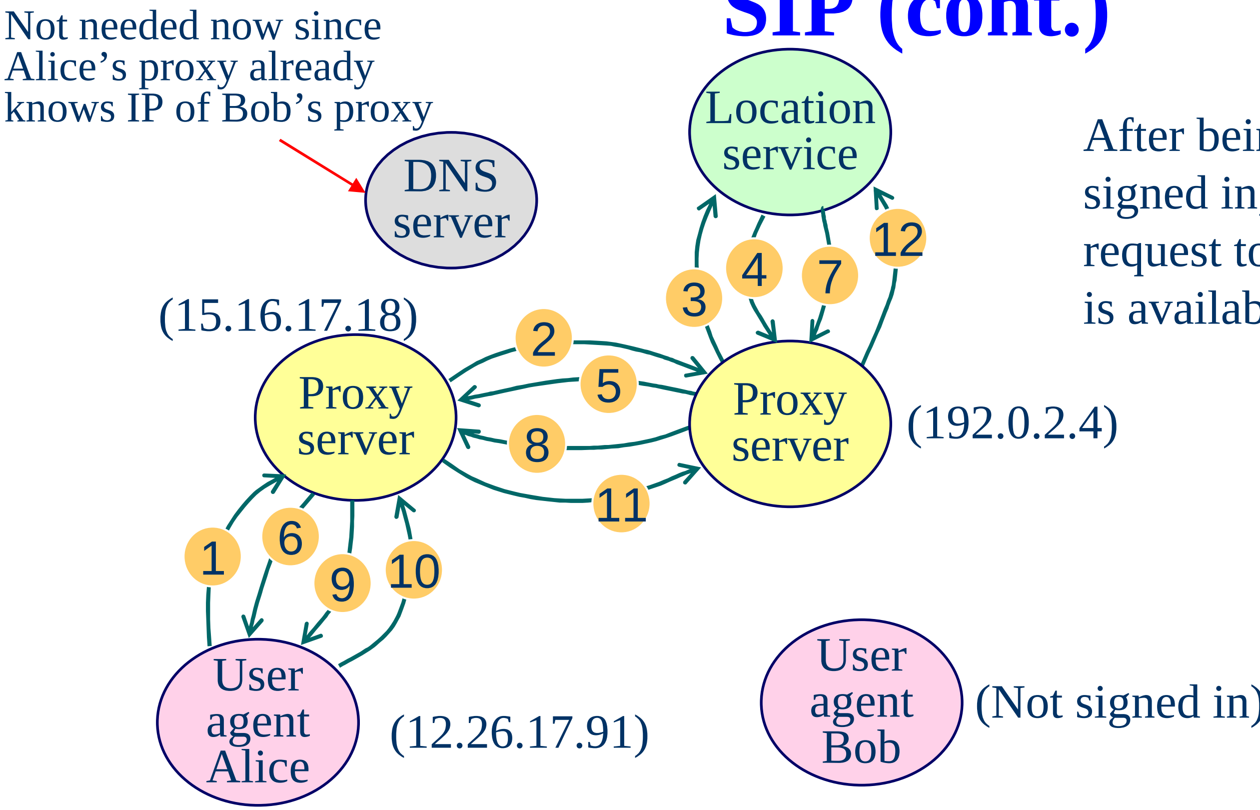
Client (Alice) is attempting to establish a session with another client (Bob) who is not signed in



1	`INVITE` `To: sip:bob@biloxi.com`
2	`100 Trying`
3	`DNS query: biloxi.com`
4	`Response: 192.0.2.4`
5	`INVITE` `To: sip:bob@biloxi.com`
6	`100 Trying`

7	`LS query: sip:bob@biloxi.com`
8	`Response: Not signed in`
9	`480 Temporarily unavailable`
10	`ACK`
11	`480 Tempoprarily unavailable`
12	`ACK`

SIP (cont.)

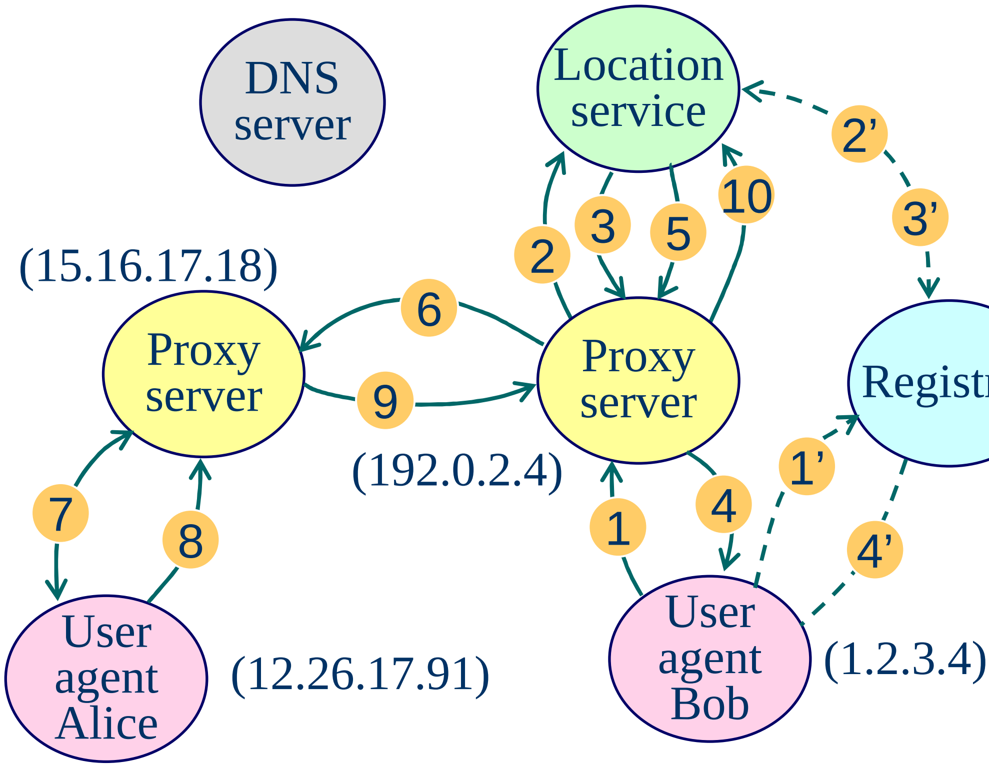


After being notified that Bob is not signed in, Alice issues a SUBSCRIBE request to be notified when Bob is available

1	SUBSCRIBE To: sip:bob@biloxi.com
2	SUBSCRIBE To: sip:bob@biloxi.com
3	SUBSCRIBE To: sip:bob@biloxi.com
4	200 OK
5	200 OK
6	200 OK

7	NOTIFY <not signed in>
8	NOTIFY <not signed in>
9	NOTIFY <not signed in>
10	200 OK
11	200 OK
12	200 OK

SIP (cont.)



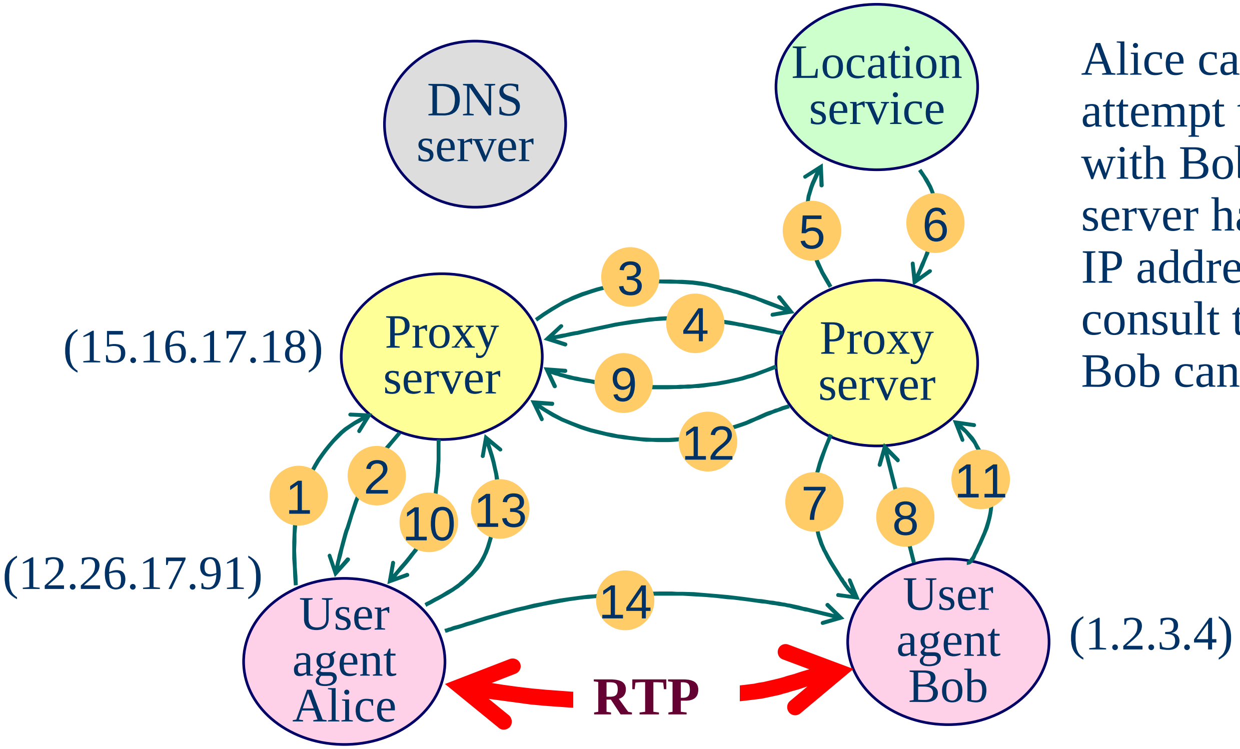
Bob eventually signed on by REGISTER request., and Alice is notified that Bob is available now.

Alternative way (depends on how UA is configured)

1	REGISTER Contact: sip:bob@(1.2.3.4)
2	Update database B = bob@1.2.3.4
3	200 OK
4	200 OK
5	NOTIFY <signed in>

6	NOTIFY <signed in>
7	NOTIFY <signed in>
8	200 OK
9	200 OK
10	200 OK

SIP (cont.)



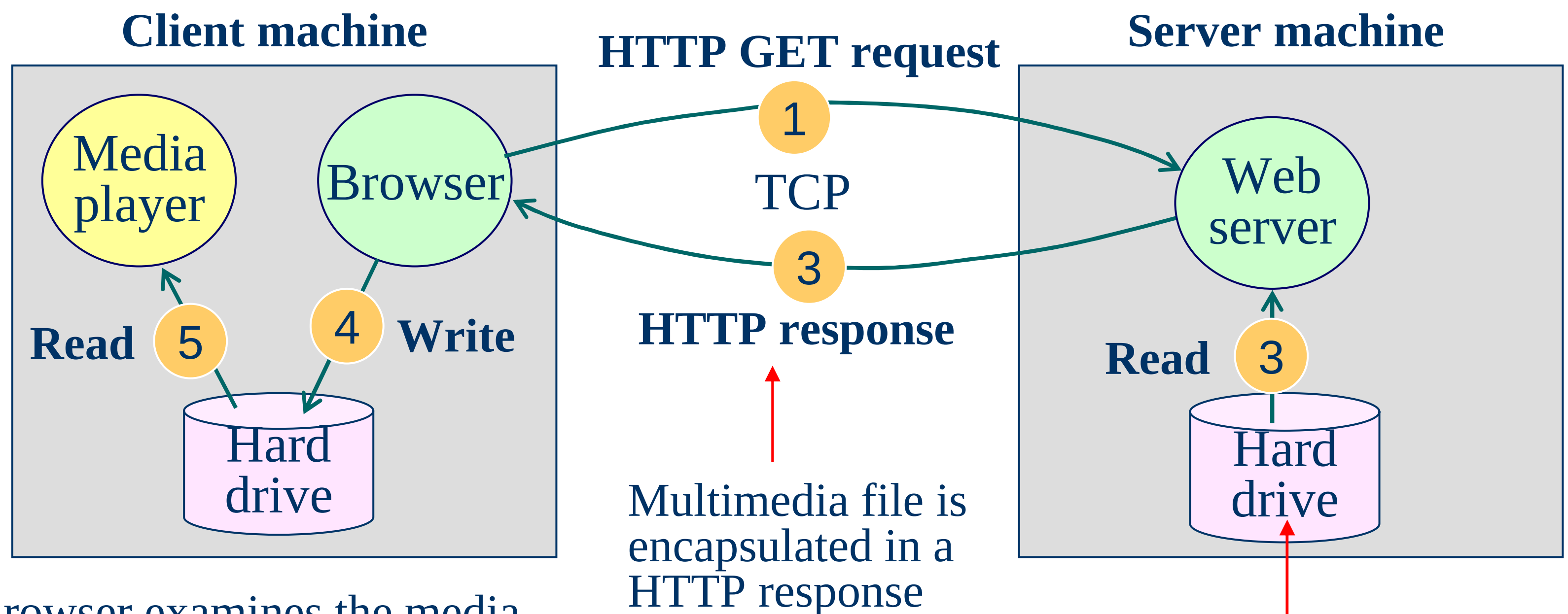
Alice can now make another attempt to establish a session with Bob. Since Alice's proxy server has already cached the Bob's IP address, there is no need to consult the DNS server. Alice and Bob can now communicate directly

1	`INVITE` `To: sip:bob@biloxi.com`
2	`100 Trying`
3	`INVITE` `To: sip:bob@biloxi.com`
4	`100 Trying`
5	`LS query: sip:bob@biloxi.com`
6	`Response: sip:bob@1.2.3.4`
7	`INVITE` `To: sip:bob@biloxi.com`

8	`180 Ringing`
9	`180 Ringing`
10	`180 Ringing`
11	`200 OK`
12	`200 OK`
13	`200 OK`
14	`ACK`

Streaming Stored Multimedia

The most straightforward (naïve) approach



Browser examines the media type (HTTP header: content type), then launches the media player. No pipelining!

Stored media file (e.g. MP3, MPEG)

Media players (helper application):

- RealPlayer (Real Networks)
- Windows Media Player (Microsoft)
- Winamp (Nullsoft)
- QuickTime (Apple)

Typical functions of the media player:

- Decoding
- Jitter removal
- Error correction
- GUI for user’s convenience

Rendering a/v

Streaming Stored Multimedia (cont.)

Problem with the straightforward approach:

Example

10 minutes of MP3 audio at typical compression rate 12 takes

$$\frac{44,000\,(s/\sec)\times 16\,(bits)\times 2\,(stereo)\times 10\,(\min)\times 60\,(\sec/\min)}{12\,(compression)\times 8\,(bits/byte)} = \frac{8,820,000\,(bytes)}{2^{20}\,(MB/byte)} \approx 8.4MB$$

To get the file from the web server over 56 kbps modem:

$$\frac{8.4\,(MB)\times 2^{20}\,(bytes/MB)\times 8\,(bits)}{56000\,(bps)} \approx \mathbf{21min}$$

Playout delay

(in fact 56 kbps can rarely be achieved, the average speed is more likely 30 kbps)

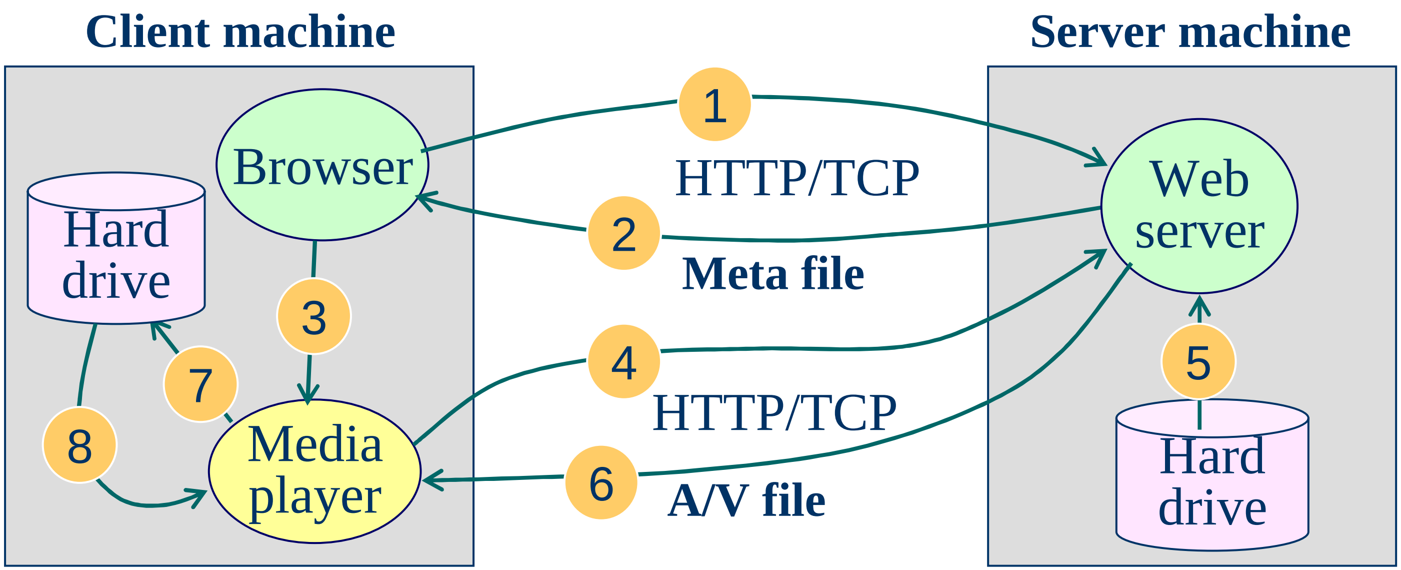
same file over 384 kbps DSL:

$$\frac{8.4\,(MB)\times 2^{20}\,(bytes/MB)\times 8\,(bits)}{384000\,(bps)} \approx \mathbf{3min}$$

How about a two hour TV movie?

Streaming Stored Multimedia (cont.)

Alternative method: direct connection between MP and web server



1 2 A meta file is transferred to browser instead of a/v file

3 Browser examines the content type in HTTP header, launches MP and passes the meta file to it

4 MP requests the a/v file directly from the web server via HTTP

5 6 7 Web server sends a/v file as an HTTP response

7 8 MP plays the a/v file.

The long playout delay is still not eliminated

Streaming Stored Multimedia (cont.)

Metafiles

A meta file is a small text file which is used to link to streaming media.

Web browsers such as Internet Explorer and Netscape Navigator were created before streaming, and consequently there was no need to incorporate ways to link to streaming media. Meta files were a work around, as they allow the players to be spawned from the web page.

Example 1: RealPlayer

Html file:

http://links.streamingwizard.com/demo/animationmodem.ram

Metafile:

rtsp://merlin.streamingwizard.com/demo/animationmodem.rm

Actual A/V file which can be rendered by RealPlayer

Streaming Stored Multimedia (cont.)

Metafiles (cont.)

Example 2: Windows Media Player

Html file:

```
http://links.streamingwizard.com/demo/businesscentre56.asx
```

Metafile:

Use Windows Media Player if available (i.e. if mms port is not closed by the firewal)

```
<asx version = "3.0">
  <entry>
    <ref href = "mms://merlin.streamingwizard.com/demo/32.wmv" />
  </entry>
  <entry>
    <ref href = "http://merlin.streamingwizard.com/demo/32.wmv" />
  </entry>
</asx>
```

Otherwise roll over to http streaming

Streaming Stored Multimedia (cont.)

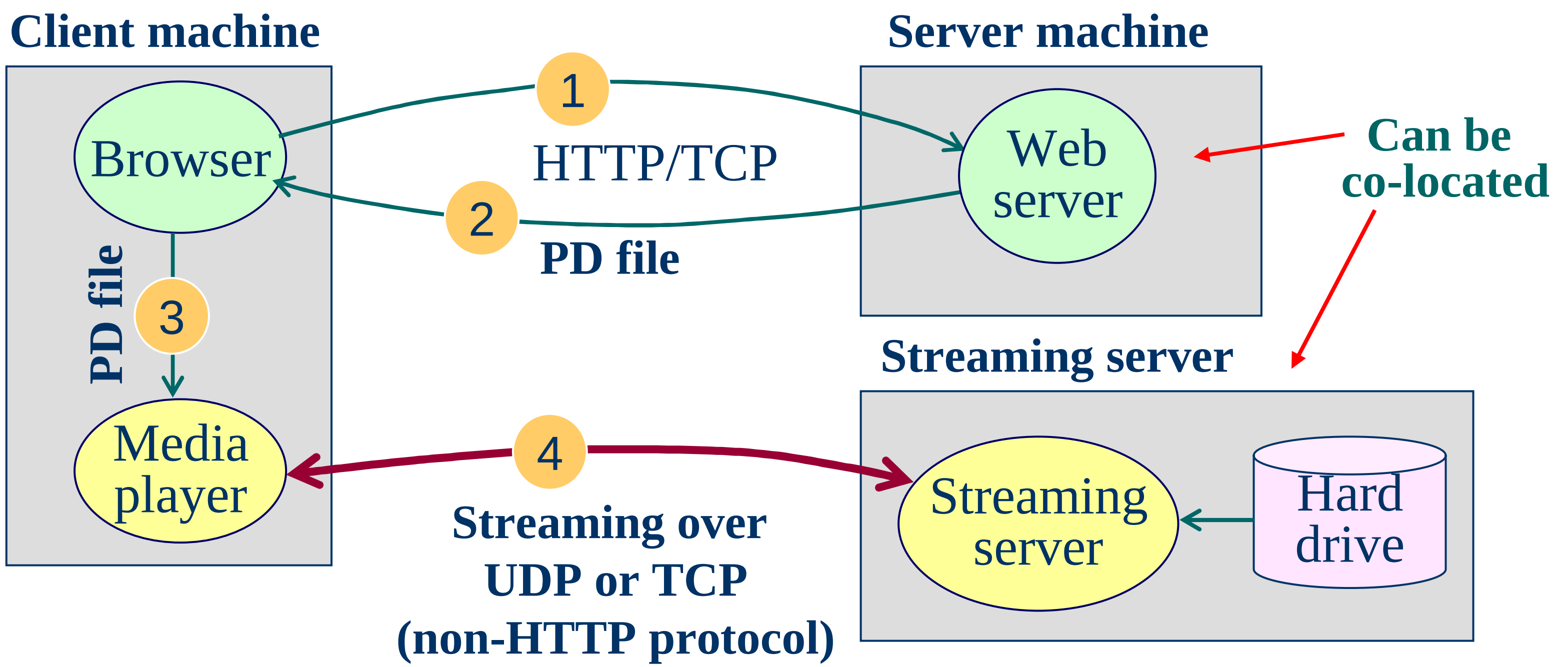
Metafiles (cont.)

Metafiles usually contain additional information relevant to the multimedia file

```
<ASX VERSION="3.0">
<title> Basic Example </title>
<copyright> Copyright 2002 </copyright>
<entry>
<author> The Author's name </author>
<title> Video clip title </title>
<ref href = "location of video file on the Internet" />
</entry>
</asx>
```

Streaming Stored Multimedia (cont.)

Using a streaming server and presentation description



1 2 A Presentation Description file is requested and transferred to the browser

3 Browser examines the content type in HTTP header, launches MP and passes the PD file to it

4 Media player requests the a/v file directly from the streaming server via HTTP

No playout delay

Streaming Stored Multimedia (cont.)

Presentation Description File

Playing multimedia files from a streaming server require more elaborate meta files, called Presentation Description files. Example:

```
<title>Twister</title>
<session>
  <group language=en lipsync>
    <switch>
      <track type=audio
         e="PCMU/8000/1"
         src = "rtsp://audio.example.com/twister/audio.en/lofi">
      <track type=audio
         e="DVI4/16000/2" pt="90 DVI4/8000/1"

        src="rtsp://audio.example.com/twister/audio.en/hifi">
    </switch>
      <track type="video/jpeg"
          src="rtsp://video.example.com/twister/video">
  </group>
</session>
```

MP can chose between two audio recordings low-fidelity and high-fidelity audio

Audio

Video

Two streams, audio and video, are played in parallel in lip-synch

A popular language for authoring of interactive audiovisual presentations is **SMIL** (Synchronized Multimedia Integration Language , pronounced "smile"). SMIL is typically used for "rich media“ (multimedia presentations which integrate streaming audio and video with images, text or any other media type).

Real Time Streaming Protocol

RFC 2326, Columbia Univ., Netscape, Real Networks, April 1998

Protocol provides "network remote control", which includes pause/resume, fast forward, rewind like VCR remote

Text-based, similar to HTTP, but not stateless (server maintains session state)
Request can be made by both, the client and the server.

RTSP messages are sent out of band over port 544 (streaming data are "in-band", different port)

Works with unicast and multicast

RTSP doesn't mandate encapsulation. Can be proprietary over UDP or TCP, or RTP (preferred)

RTSP allows media player to control the transmission of a media stream

Real Time Streaming Protocol (cont.)

RTSP Methods

Methods	*Description*
SETUP	Causes the server to allocate resources for a stream and start an RTSP session
PLAY	Starts data transmission on a stream allocated via SETUP
RECORD	Initiates recording a range of media data according to the presentation description
PAUSE	Temporarily halts a stream without freeing server resources
TEARDOWN	Frees resources associated with the stream. The RTSP session ceases to exist on the server
ANNOUNCE	Change description of media object
REDIRECT	A redirect request informs the client that it must connect to another server location
SET-PARAMETER	Set the value of a parameter for a presentation or stream specified by the URI
DESCRIBE	Get description of media object

Real Time Streaming Protocol (cont.)