Computer Networks

Wenzhong Li

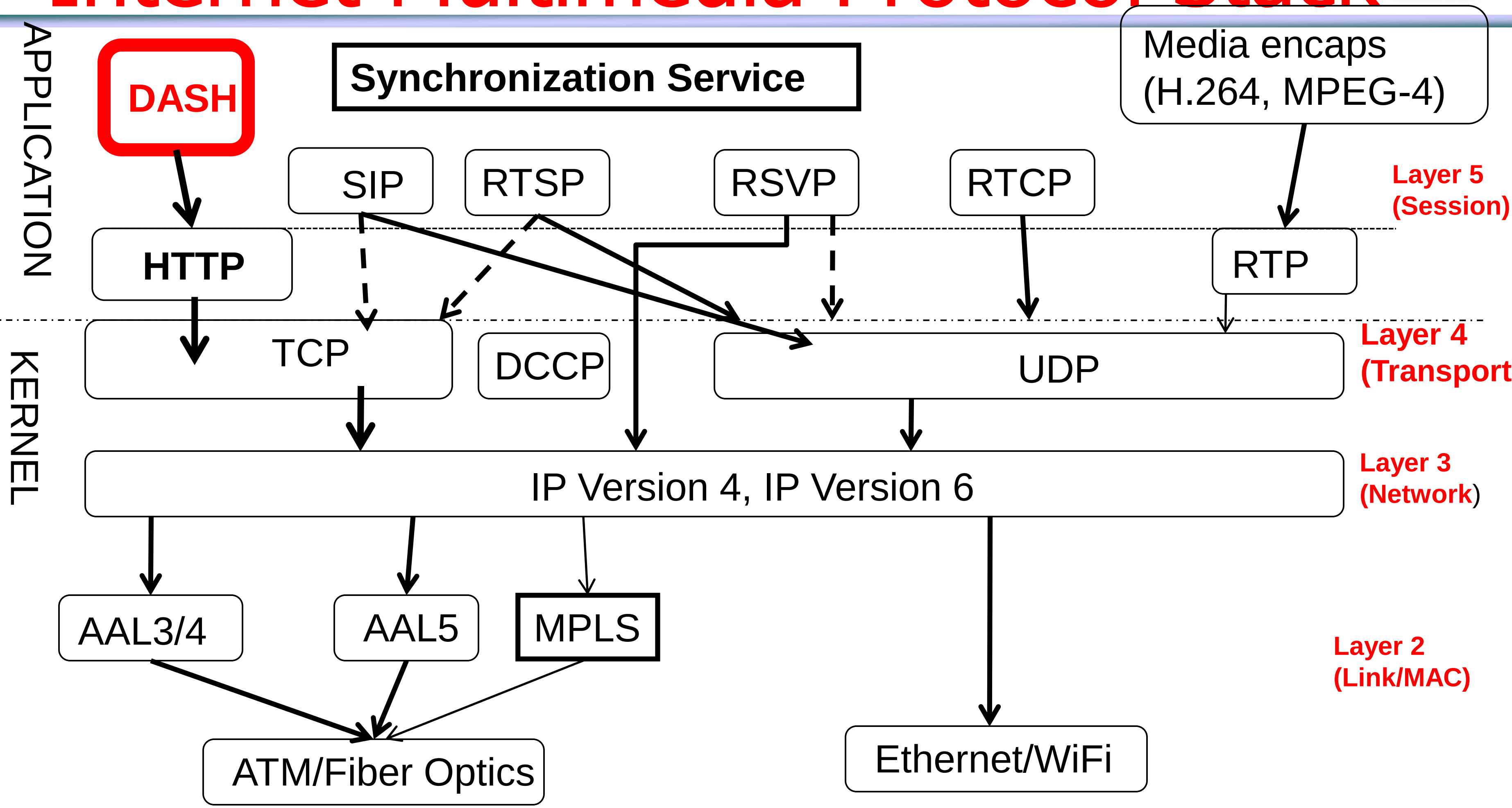
Nanjing University

Chapter 5. End-to-End Protocols

- Transport Services and Mechanisms
- User Datagram Protocol (UDP)
- Transmission Control Protocol (TCP)
- TCP Congestion Control
- Real-time Transport Protocol (RTP)
- Session Initiation Protocol (SIP)
- Real Time Streaming Protocol (RTSP)

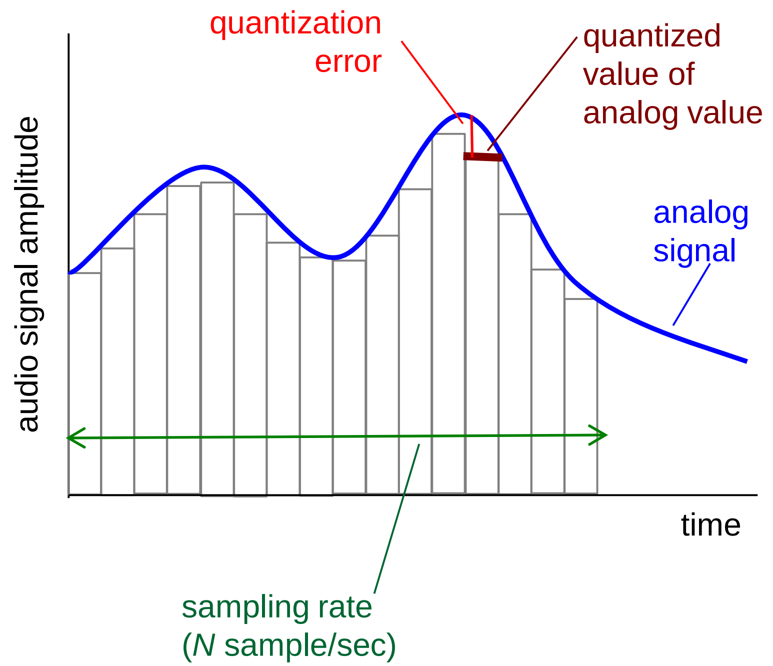
Multimedia Networking

Internet Multimedia Protocol Stack



Multimedia: audio

- Analog audio signal sampled at constant rate
  - telephone: 8,000 samples/sec
  - CD music: 44,100 samples/sec
- Each sample quantized, i.e., rounded
  - e.g., 2^8=256 possible quantized values
- Example rates
  - CD: 1.411 Mbps
  - MP3: 96, 128, 160 kbps
  - Internet telephony: 5.3 kbps and up



Multimedia: video

spatial coding example: instead of sending *N* values of same color (all purple), send only two values: color value (*purple*) *and number of repeated values* (N)

- Video: sequence of images displayed at constant rate
  - e.g. 24 images/sec
- Digital image: array of pixels
  - each pixel represented by bits
- Coding: use redundancy *within* and *between* images to decrease # bits used to encode image
  - spatial (within image)
  - temporal (from one image to next)
- Examples:
  - MPEG 1 (CD-ROM) 1.5 Mbps
  - MPEG2 (DVD) 3-6 Mbps
  - MPEG4 (often used in Internet, < 1 Mbps)

frame *i*

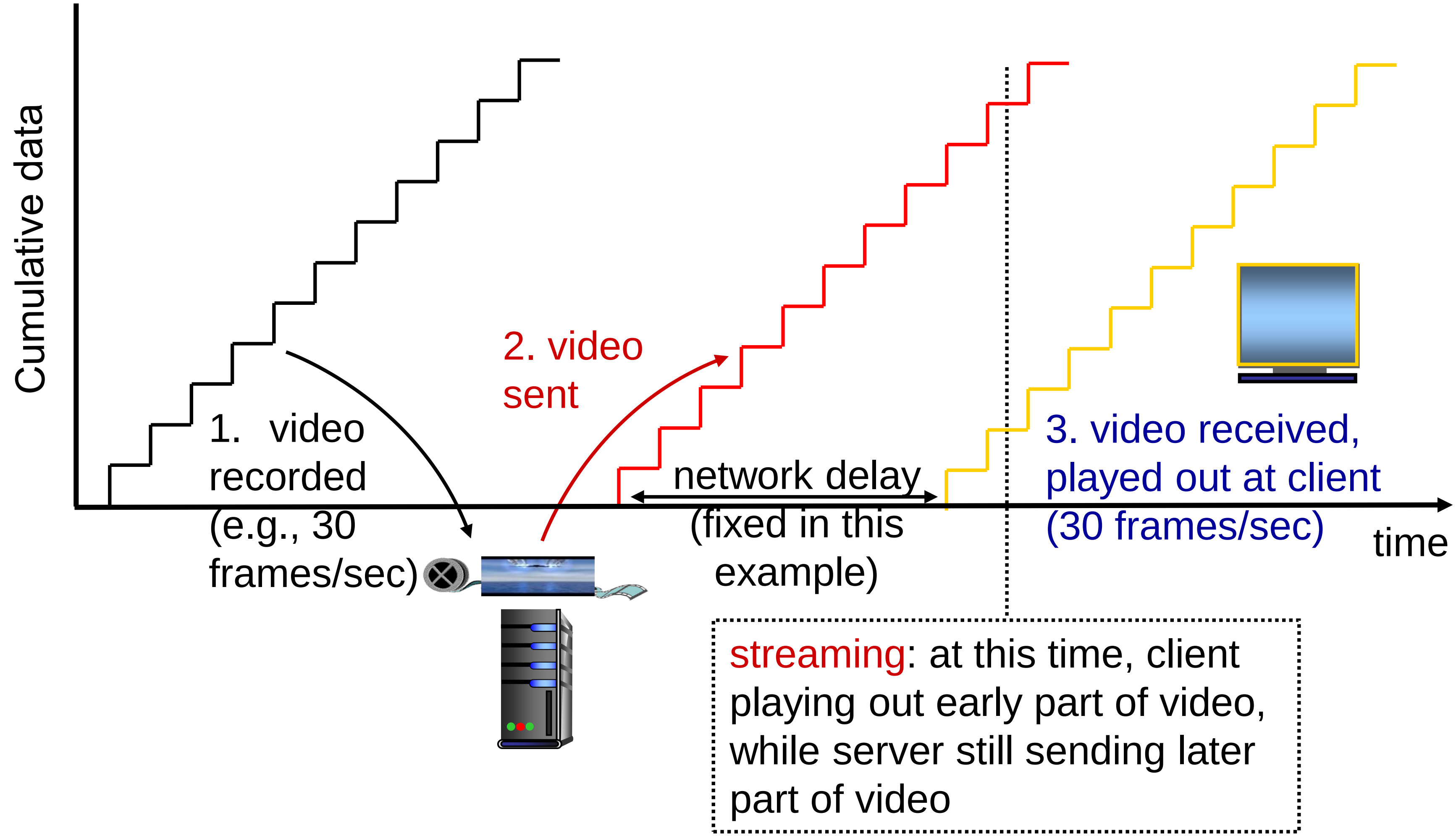
temporal coding example: instead of sending complete frame at i+1, send only differences from frame i

frame *i+1*

Three application types

- *Streaming, stored* audio, video
  - *streaming:* can begin playout before downloading entire file
  - *stored (at server):* can transmit faster than audio/video will be rendered (implies storing/buffering at client)
  - e.g., YouTube, Netflix, Hulu
- *Conversational* voice/video over IP
  - interactive nature of human-to-human conversation limits delay tolerance
  - e.g., Skype
- *Streaming live* audio, video
  - e.g., live sporting event (futbol)

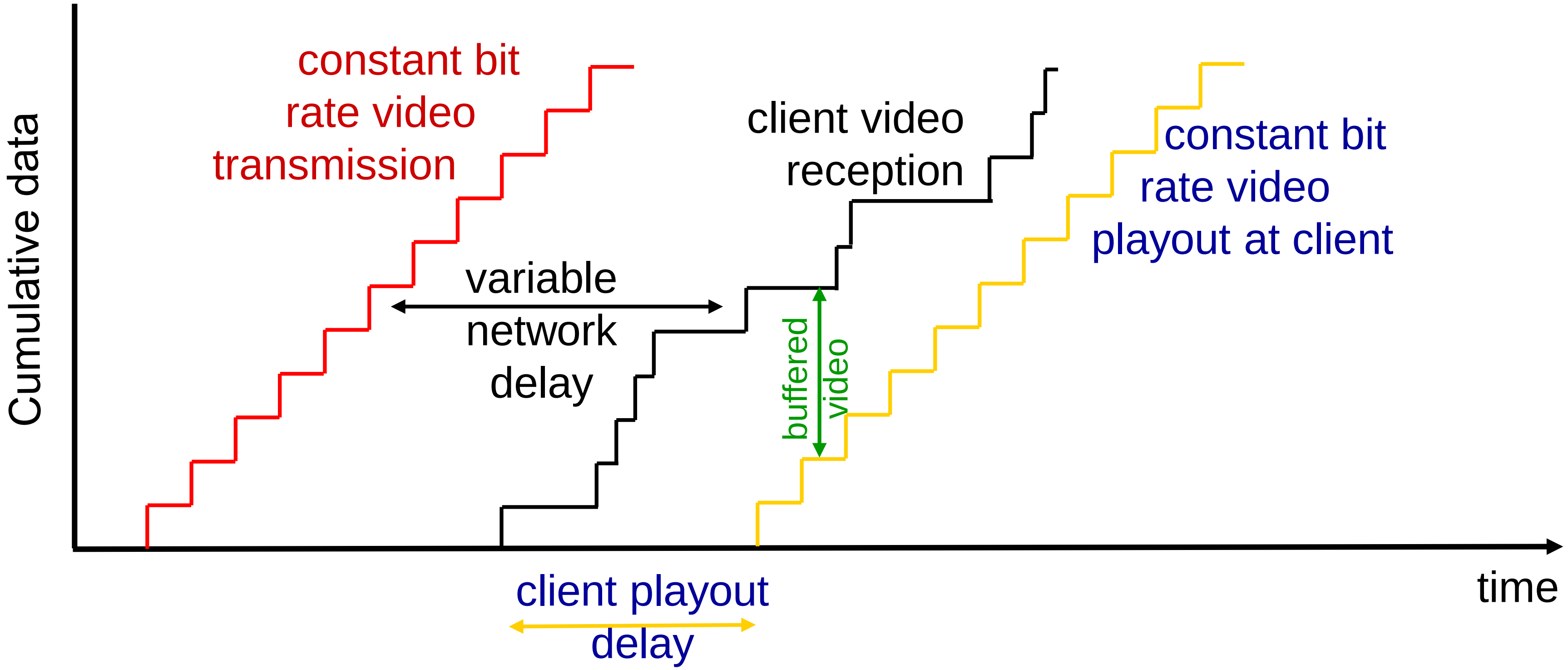
Streaming stored video:



Streaming stored video: challenges

- *Continuous playout constraint*: once client playout begins, playback must match original timing
  - … but *network delays are variable* (jitter), so will need *client-side buffer* to match playout requirements
- Other challenges:
  - client interactivity: pause, fast-forward, rewind, jump through video
  - video packets may be lost, retransmitted

Streaming stored video: revisted



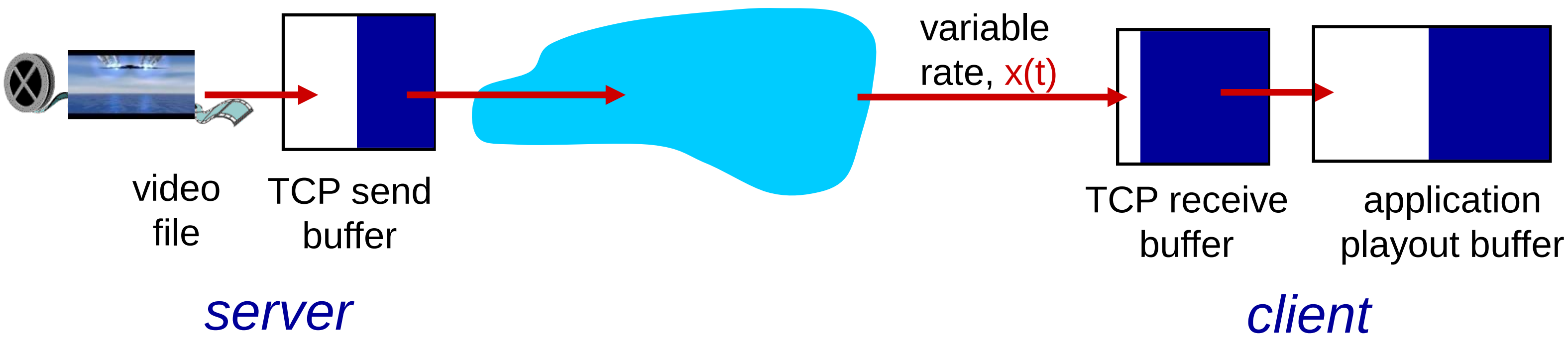
- *client-side buffering and playout delay:* compensate for network-added delay, delay jitter

Streaming multimedia: UDP

- Server sends at rate appropriate for client
  - often: send rate = encoding rate = constant rate
  - transmission rate can be oblivious to congestion levels
- short playout delay (2-5 seconds) to remove network jitter
- error recovery: application-level, timeipermitting
- RTP [RFC 2326]: multimedia payload types
- UDP may *not* go through firewalls

Streaming multimedia: 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 multimedia: DASH

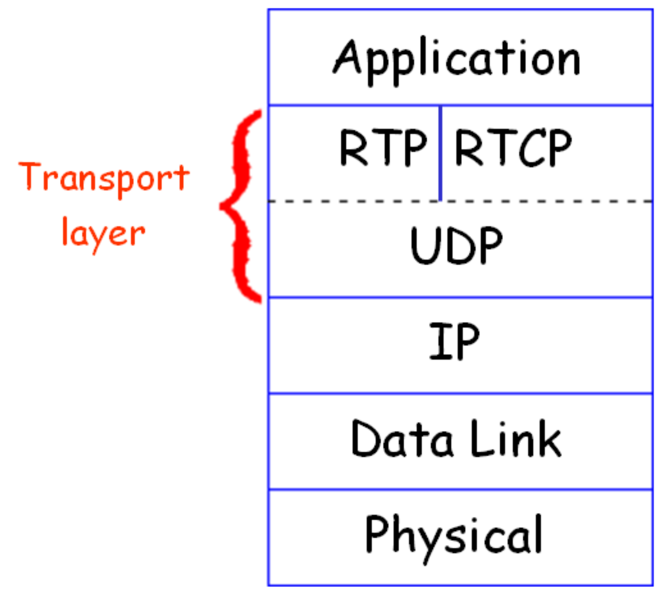
- *DASH: D*ynamic, *A*daptive *S*treaming over *H*TTP
- *server:*
  - divides video file into multiple chunks
  - each chunk stored, encoded at different rates
  - *manifest file:* provides URLs for different chunks
- *client:*
  - periodically measures server-to-client bandwidth
  - consulting manifest, 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 multimedia: DASH

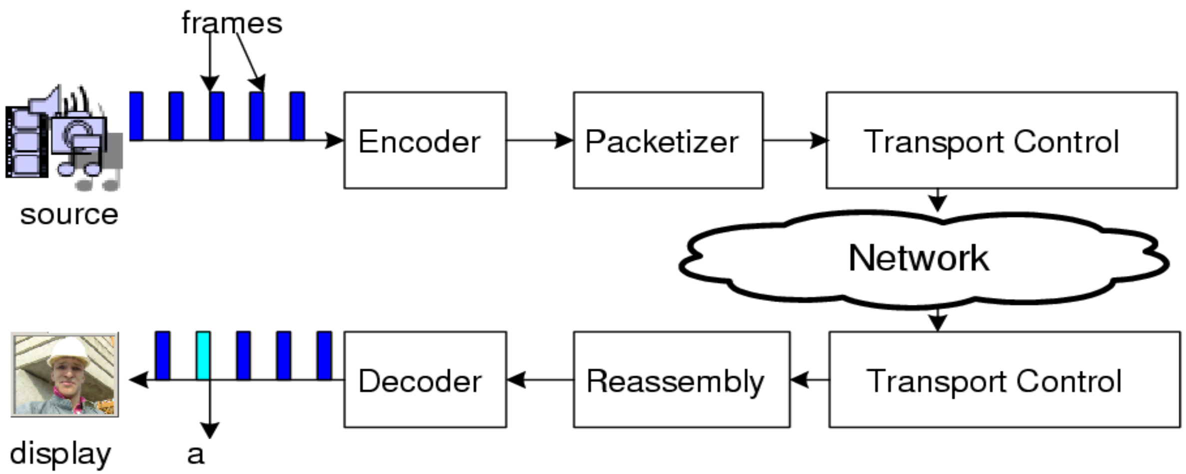
- *DASH: Dynamic, Adaptive Streaming over HTTP*
- *"intelligence"* at client: client determines
  - *when* to request chunk (so that buffer starvation, or overflow does not occur)
  - *what encoding rate* to request (higher quality when more bandwidth available)
  - *where* to request chunk (can request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

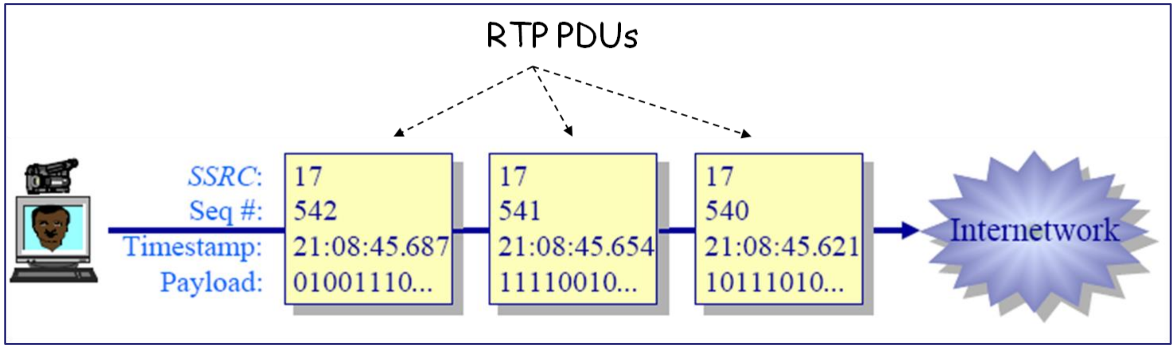
Real-time Transport Protocol (RTP)

- RFC 3550
  - Built upon UDP, i.e. RTP packets encapsulated in UDP segments
  - Specifies packet structure for packets carrying audio, video data
- RTP packet provides
  - Payload type identification
  - Packet sequence numbering
  - Time stamping

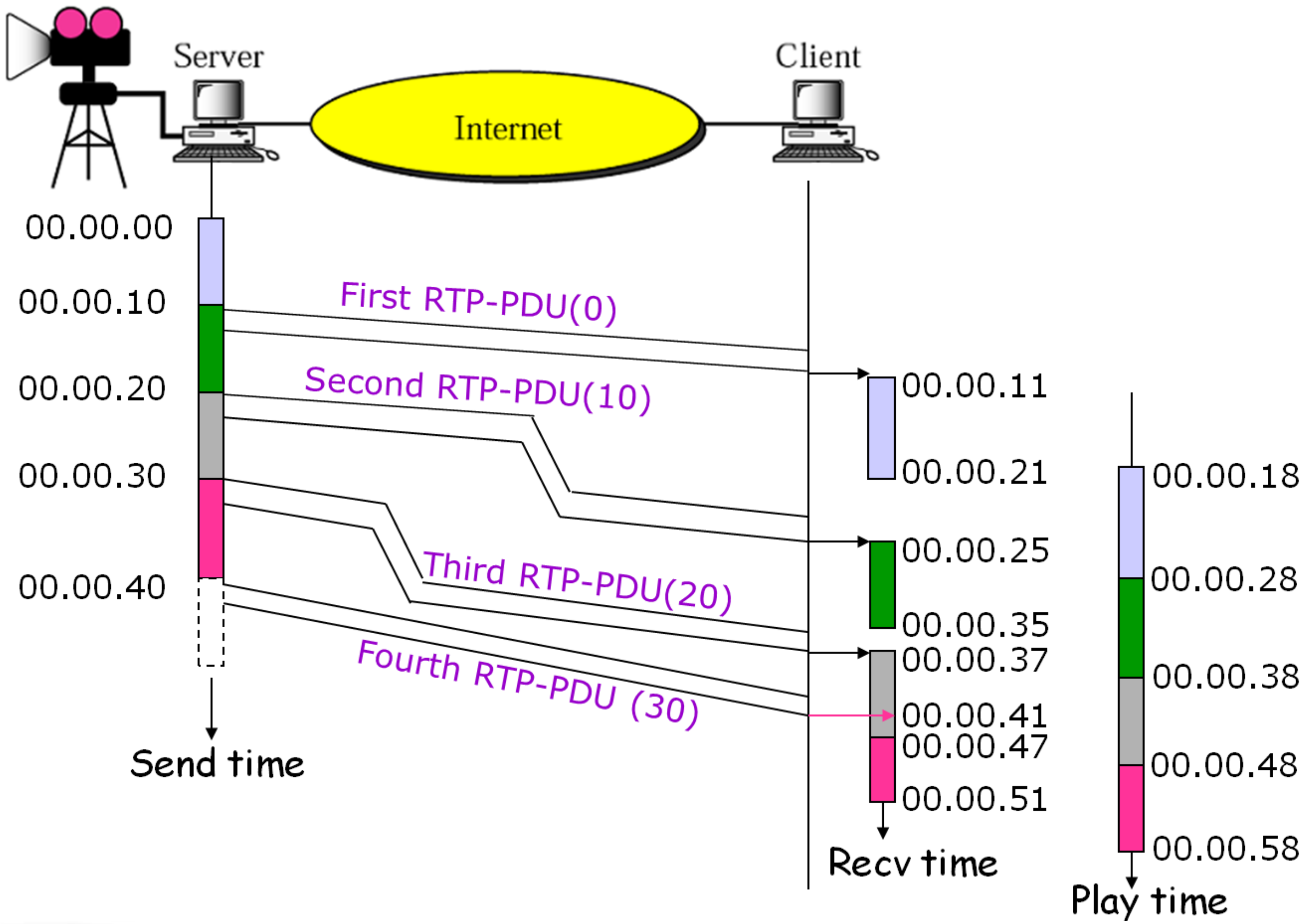


Real-time Streaming





Real-time Streaming



How does RTP works

- Timestamping - most important information for real-time applications.
  - The sender timestamp according to the instant the first octet in the packet was sampled.
  - The receiver uses timestamp to reconstruct the original timing
  - Also used for synchronize different streams; audio an video in MPEG. (Application level responsible for the actual synchronization)

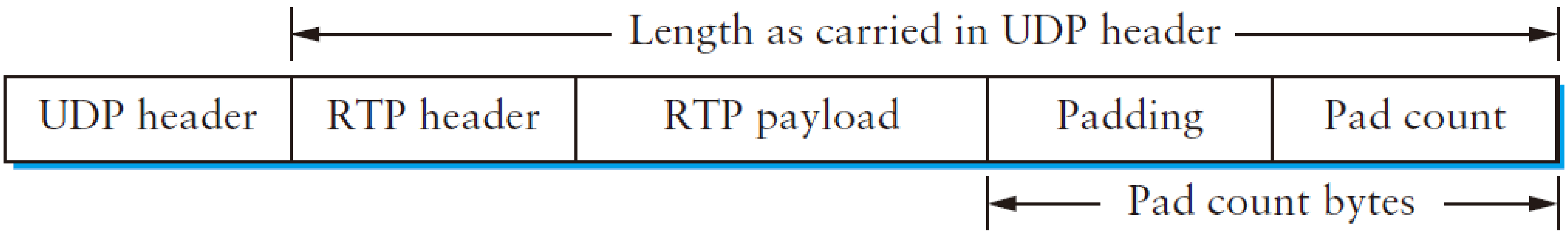
How does RTP work

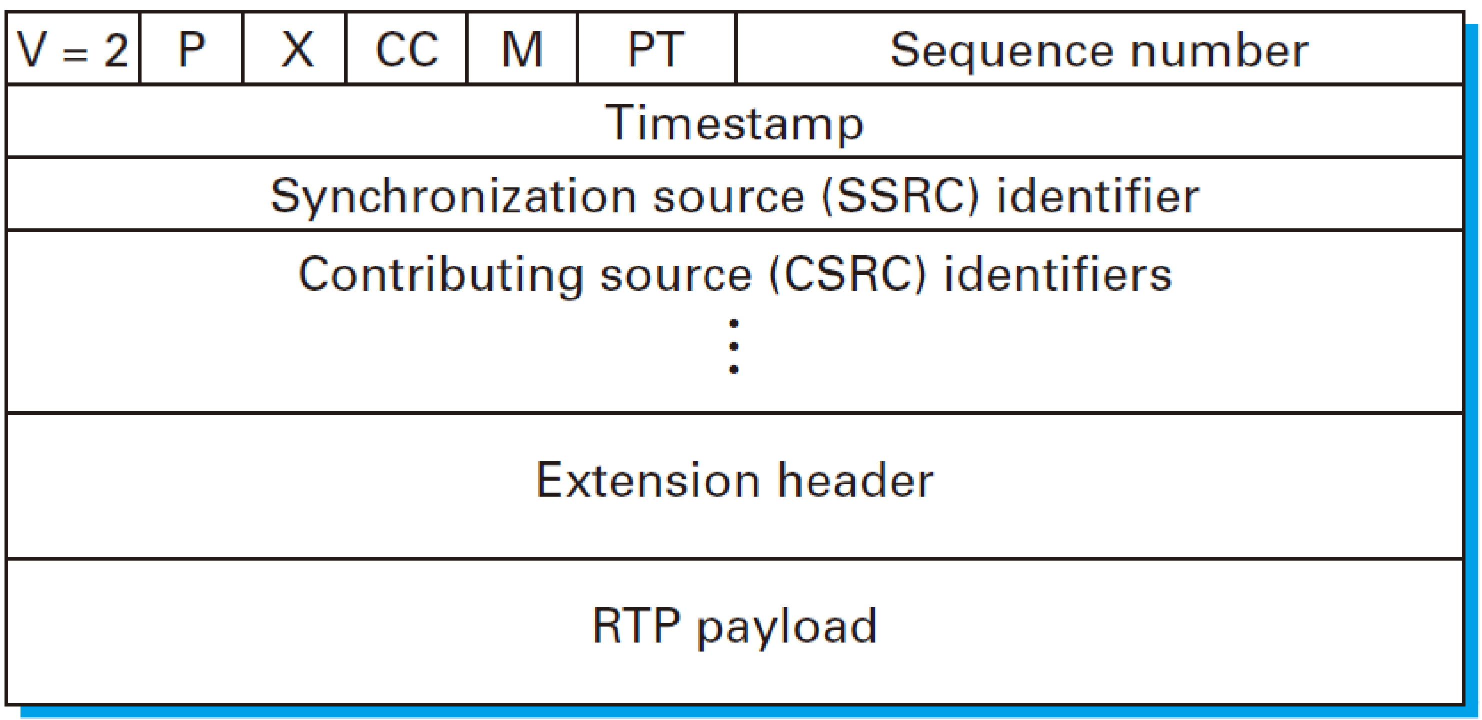
- Payload type identifier
  - specifies the payload format as well as encoding/compression schemes
  - The application then knows how to interpret the payload
- Source identification
  - Audio conference

RTP and QoS

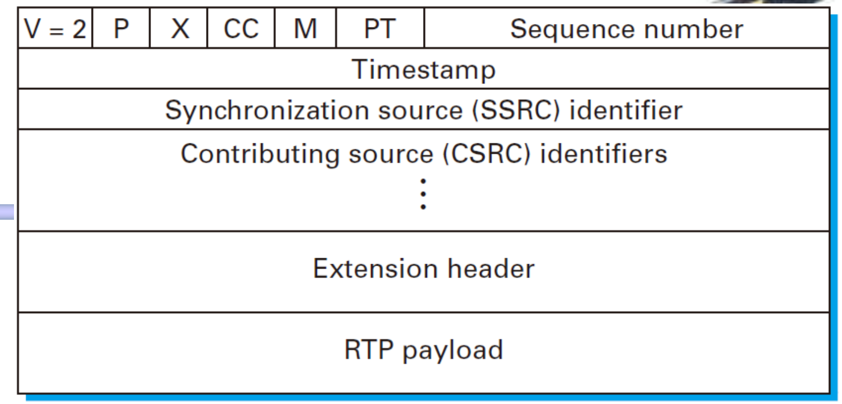
- RTP does not provide any mechanism to ensure timely data delivery
- RTP encapsulation is only seen at end systems, and unseen by intermediate routers
- Routers make no special effort for RTP packets

RTP Packets


Length as carried in UDP header
UDP header
RTP header
RTP payload
Padding
Pad count
Pad count bytes

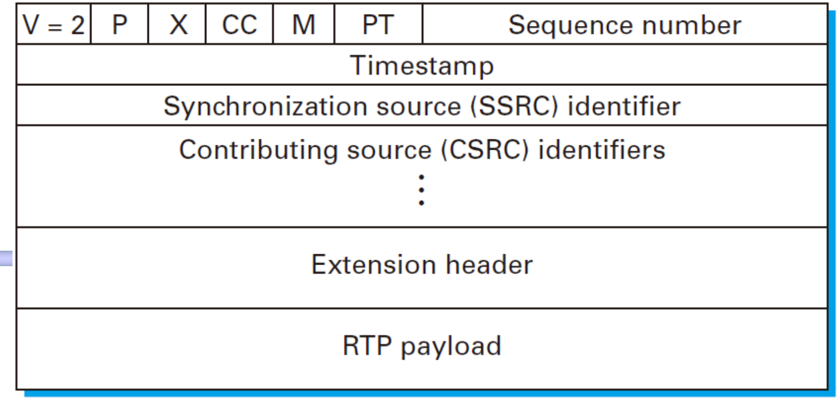

V = 2
P
X
CC
M
PT
Sequence number
Timestamp
Synchronization source (SSRC) identifier
Contributing source (CSRC) identifiers
Extension header
RTP payload

RTP Header



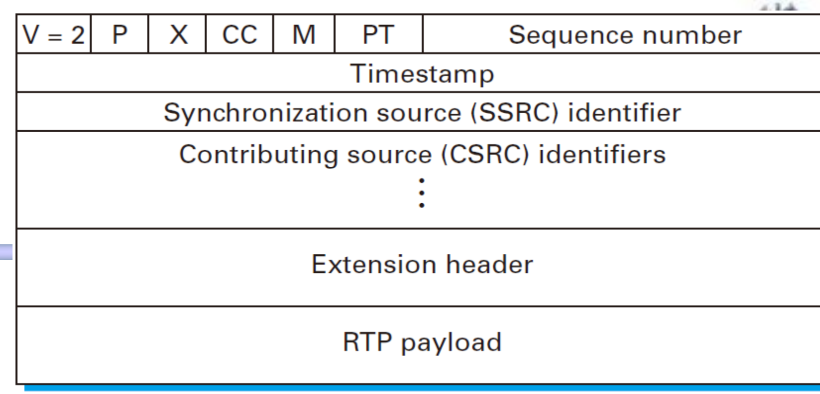
- Version (2 bits)
  - Version of the protocol, Current 2
- P (Padding) (1 bit)
  - Indicates if there are extra padding octets
- X (Extension) (1 bit)
  - Indicates presence of an extension header
- CC (CSRC Count) (4 bits)
  - Number of CSRC identifiers
- M (Marker) (1 bit)
  - Indicates special relevance for the application

RTP Header



- Payload Type (7 bits)
  - Indicates type of encoding currently being used
  - Sender can change encoding in middle of session
- Example payload:
  - 0: PCM mu-law, 64 kbps
  - 3: GSM, 13 kbps
  - 7: LPC, 2.4 kbps
  - 26: Motion JPEG
  - 31: H.261
  - 33: MPEG2 video

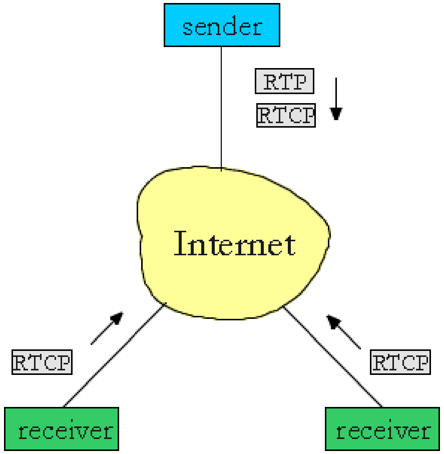
RTP Header



- Sequence Number (16 bits)
  - Increment by 1 for each RTP data packet sent
- Timestamp: (32 bits)
  - Sampling instant of first octet, used for play-back
- SSRC (Synchronization source identifier) (32 bits)
  - Uniquely identifies the source of a stream in a RTP session
- CSRC (Contributing source IDs) (32 bits)
  - Enumerate contributing sources to a stream
- Extension header
  - Specific header for certain payload type

RTP Control Protocol (RTCP)

- Specifies report PDUs exchanged between sources and destinations
  - Receiver reception report
    - feedback of data delivery
    - Packet lost, jitter, timestamps
  - Sender report
    - Intermedia synchronization, number of bytes sent
  - Source description report

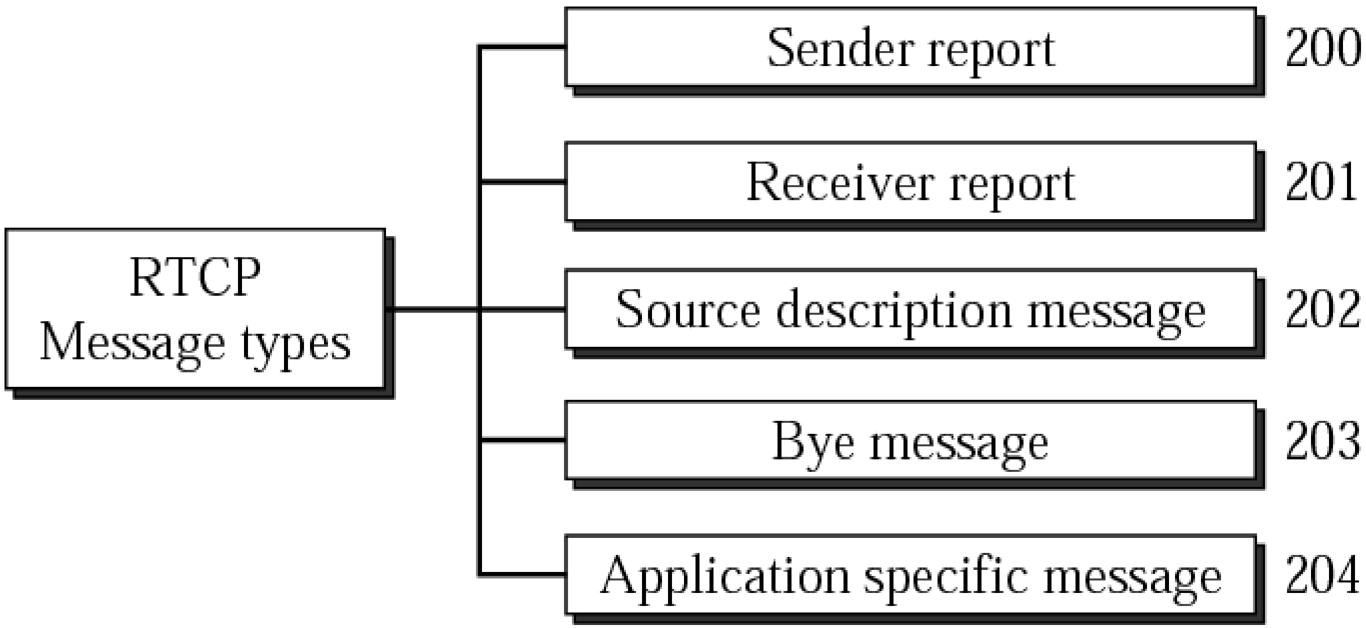


- Reports contain statistics such as the number of RTP-PDUs sent, number of RTP-PDUs lost, inter-arrival jitter

- Used by application to modify sender transmission rates and for diagnostics purposes

RTCP Message Types

- Several RTCP PDUs of different types can be transmitted in a single UDP segment



Sender/Receiver Report PDUs

<table>
<tr><td>V</td><td>P</td><td>RC</td><td>PT=200/201 → SR/RR</td><td>Length (16 bits)</td><td rowspan="2">Header</td></tr>
<tr><td colspan="5">SSRC of Sender</td></tr>
<tr><td colspan="5">NTP Timestamp, most significant word</td><td rowspan="5">Sender Info</td></tr>
<tr><td colspan="5">NTP Timestamp, least significant word</td></tr>
<tr><td colspan="5">RTP Timestamp</td></tr>
<tr><td colspan="5">Sender's PDU Count</td></tr>
<tr><td colspan="5">Sender's Octet Count</td></tr>
<tr><td colspan="5">SSRC_1 (SSRC of the 1st Source)</td><td rowspan="6">Report Block 1</td></tr>
<tr><td colspan="3">Fraction Lost</td><td colspan="2">Cumulative Number of PDU Lost</td></tr>
<tr><td colspan="5">Extended Highest sequence Number Received</td></tr>
<tr><td colspan="5">Interarrival Jitter</td></tr>
<tr><td colspan="5">Last SR (LSR)</td></tr>
<tr><td colspan="5">Delay Since Last SR (DLSR)</td></tr>
<tr><td colspan="5">SSRC_2 (SSRC of the 2nd Source)</td><td rowspan="2">Report Block 2</td></tr>
<tr><td colspan="5">… …</td></tr>
<tr><td colspan="5">Profile-Specific Extensions</td><td></td></tr>
</table>

RTCP provides the following services

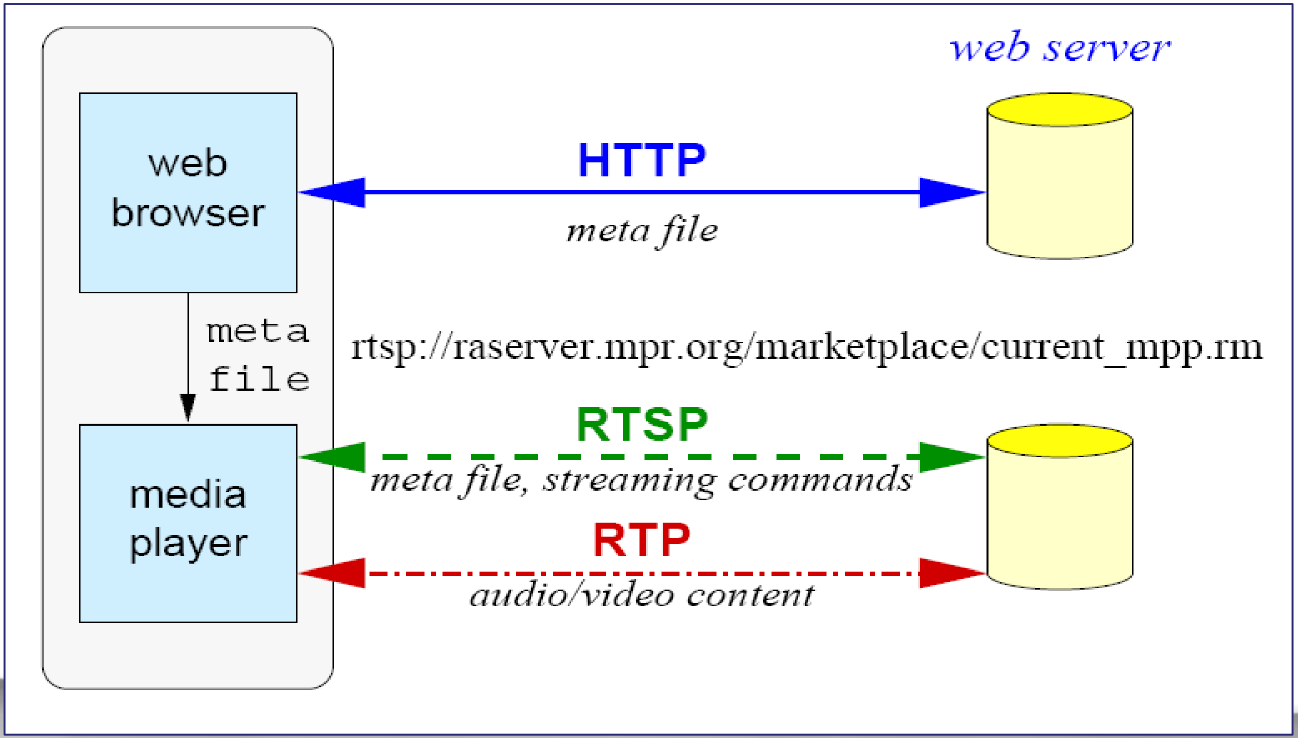
- QoS monitoring and congestion control
  - Primary function: QoS feedback to the application
  - The sender can adjust its transmission
  - The receiver can determine if the congestion is local, regional, or global
  - Network managers can evaluate the network performance for multicast distribution
- Source identification
- inter-media synchronization
- control information scaling
  - Limit control traffic (most 5 % of the overall session traffic)

Real Time Streaming Protocol (RTSP)

- RFC 2326
  - An app-level protocol that establishes and controls media sessions between end points
  - Support VCR commands: rewind, fast forward, pause, resume, repositioning, etc…
  - RTSP is an application-level protocol designed to work with RTP (and RSVP) to provide a complete streaming service over internet
  - Can built upon UDP or TCP, commands sent in ASCII text
  - Integration with web architecture, separate stream channel and control channel

RTSP Scenario

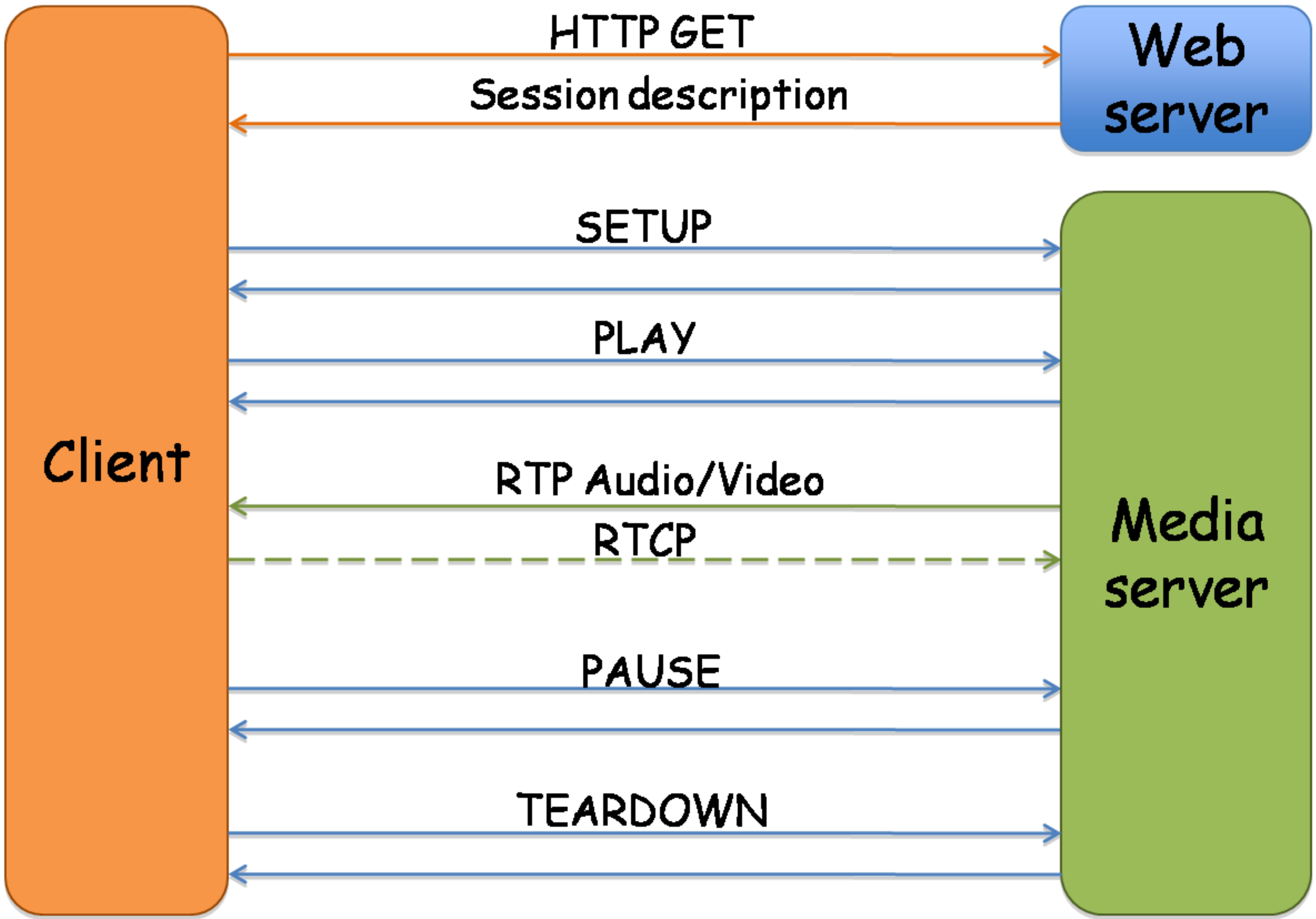
- Metafile communicated to web browser using HTTP
- Browser launches player
- Player sets up an RTSP control connection, data connection to streaming server



A Meta-File 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>
```

RTSP Operation



SETUP Example

- Specifies the transport mechanism used for streaming media
- Client can issue SETUP to change parameters for already started media

Client -> Server:

SETUP rtsp://audio.example.com/twister/audio RTSP/1.0
CSeq: 302
Transport: RTP/UDP; compression; unicast; client_port=4588-4589

Server -> Client:

RTSP/1.0 200 OK
Cseq: 302
Date: 23 Jan 1997 15:35:06 GMT
Session: 47112344
Transport: RTP/UDP; compression; unicast;
client_port=4588-4589;server_port=6256-6257

PLAY Example

- Plays from beginning to end of range specified
- Scale header can be used to change viewing rate

Client -> Server:

```
PLAY rtsp://audio.example.com/twister.en RTSP/1.0
CSeq: 833
Session: 12345678
Range: smpte=0:10:20-;time=19970123T153600Z
```

Server -> Client:

```
RTSP/1.0 200 OK
CSeq: 833
Date: 23 Jan 1997 15:35:06 GMT
Range: smpte=0:10:22-;time=19970123T153600Z
```

Pause and Teardown

- Pause
  - May contain Range header to specify when to pause
  - Server will terminate session after timeout period expires
- Teardown
  - Frees up resources on the server

Client -> Server

```
PAUSE rtsp://audio.example.com/twister/audio.en/lofi RTSP/1.0
CSeq: 887
Session: 12345678
Range: smpte=0:15:27
… …
TEARDOWN rtsp://audio.example.com/twister/audio.en/lofi RTSP/1.0
CSeq: 892
Session: 12345678
```

Server->Client

```
RTSP/1.0 200 OK
CSeq: 892
```

RTSP Reliability

- If using TCP message is sent just once
- If using UDP, RTSP will retransmit if not receive ACK
  - Timeout is initially set to 500 ms
  - Can re-compute timeout based on RTT like TCP
- Sequence no is not incremented for retransmission
  - Timestamp is used to overcome retransmission ambiguity

Why IP telephony (VoIP)

- Advantages: Cheaper
  - No inter-connect charges; 6-8 kb/s (packet) vs 64kb/s
  - Regulation costs
- New value-added features; conferencing
- Single network
- Still immature; latency major issue
- ITU-T: H.323 (set of protocols)
- SIP used to imitate a session between users. Simple, cheap. Limited, but popular

Voice-over-IP (VoIP)

- *VoIP end-end-delay requirement*: needed to maintain "conversational" aspect
  - higher delays noticeable, impair interactivity
  - < 150 msec: good
  - > 400 msec bad
  - includes application-level (packetization,playout), network delays
- *session initialization:* how does callee advertise IP address, port number, encoding algorithms?
- *value-added services:* call forwarding, screening, recording
- *emergency services:* 911

VoIP: packet loss, delay

- *network loss:* IP datagram lost due to network congestion (router buffer overflow)
- *delay loss:* IP datagram arrives too late for playout at receiver
  - delays: processing, queueing in network; end-system (sender, receiver) delays
  - typical maximum tolerable delay: 400 ms
- *loss tolerance:* depending on voice encoding, loss concealment, packet loss rates between 1% and 10% can be tolerated

Session Initiation Protocol (SIP)

- RFC 3261 – an application level protocol
  - A signaling protocol used for controlling multimedia communication sessions such as voice and video calls over IP
  - For create, modify and terminate two-party (unicast) or multiparty (multicast) sessions consisting of one or several media streams
  - An alternative to ITU's H.323, used for IP Telephony since 1994
- Application examples
  - Video conferencing
  - Streaming multimedia distribution
  - Instant messaging
  - Online games

SIP Services

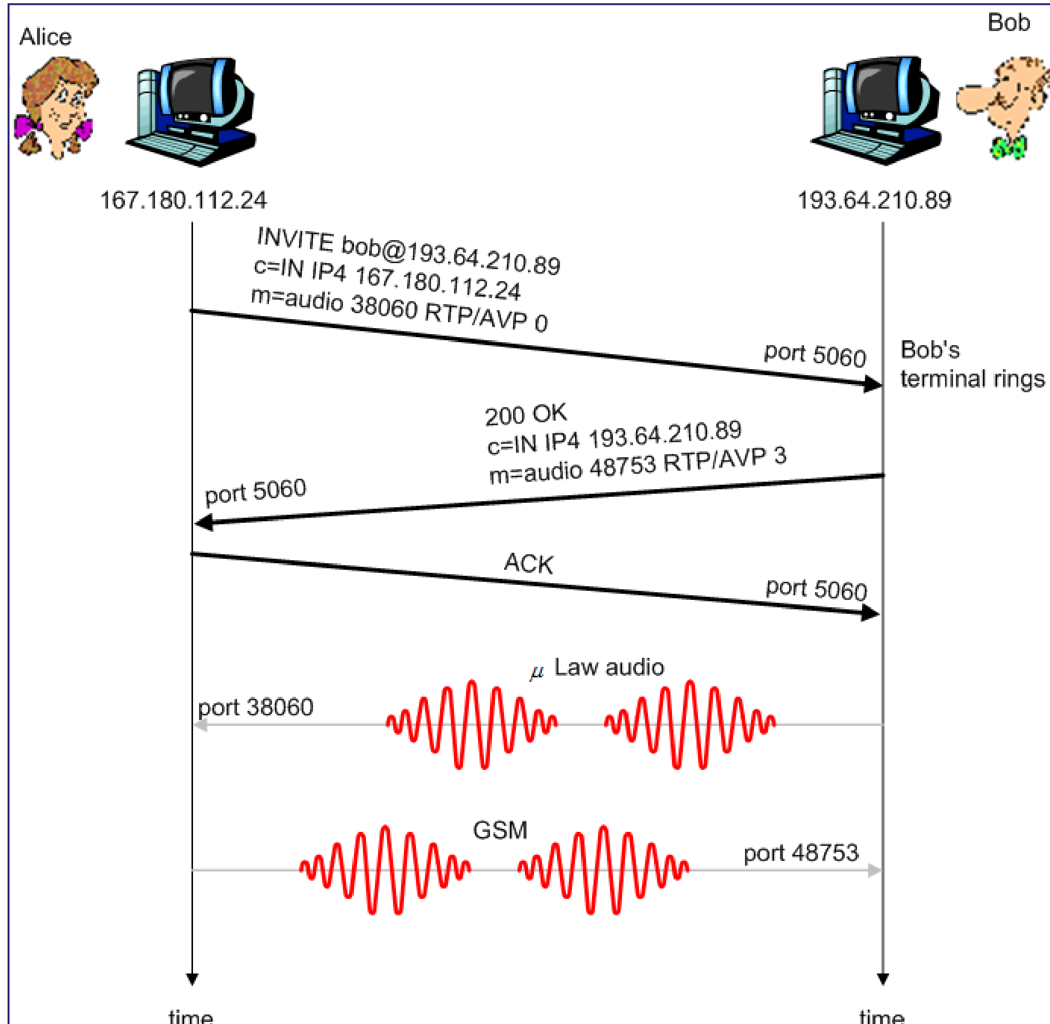
- Setting up a call
  - Determine current IP address of callee by SIP address
  - Let callee know caller wants to establish a call
  - Caller, callee agree on media type, encoding
- Call management
  - Add new media streams during call
  - Change encoding during call
  - Invite others, transfer and hold calls

SIP Addressing

- Uses Internet URLs
  - Uniform Resource Locators
  - Supports both Internet and PSTN addresses
  - General form is user@host
- Examples
  - sip:alan@wcom.com
  - sip:J.T. Kirk <kirk@starfleet.gov>
  - sip:+1-613-555-1212@wcom.com;user=phone
  - sip:guest@10.64.1.1
  - sip:790-7360@wcom.com;phone-context=VNET

Setting up a Call

- Alice's SIP agent *invite* msg, indicates her port, IP address, encoding she prefers to receive (PCM μlaw)
- Bob's agent 200 *OK* msg, indicates his port, IP address, preferred encoding (GSM)
- SIP msgs can be sent over TCP or UDP, here sent over RTP/UDP
- Default SIP port number is 5060



Other Possible Reply

- Rejecting a call
  - Bob' agent rejects with "600 busy", "503 unavailable", "302 gone", "401 unauthorized"

- Further negotiation
  - Bob replies with "606 Not Acceptable", listing his encoders
  - Alice can then send new "INVITE" message, advertising different encoder

A SIP Request Message

```
INVITE sip:bob@domain.com SIP/2.0
Via: SIP/2.0/UDP 167.180.112.24:5060
From: sip:alice@hereway.com
To: sip:bob@domain.com
Call-ID: a2e3a@pigeon.hereway.com
CSeq: 1 INVITE
Content-Type: application/sdp
Content-Length: 885
… … …
c=IN IP4 167.180.112.24
m=audio 38060 RTP/AVP 0
```

- Use HTTP message syntax
- Via: Shows route taken by request
- Call-ID: unique identifier generated by client
- CSeq: Command Sequence number, incremented for each successive request

A SIP Response Message

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP 167.180.112.24:5060
From: sip:alice@hereway.com
To: sip:bob@domain.com
Call-ID: a2e3a@pigeon.hereway.com
CSeq: 1 INVITE
```

- *Via*, *From*, *To*, *Call-ID*, and *CSeq* are copied exactly from Request
  - To and From are NOT swapped

Finding a Callee

- SIP address must be transformed to IP address of callee’s current host
  - Bob may move around, getting different IP addresses (using mobile devices)

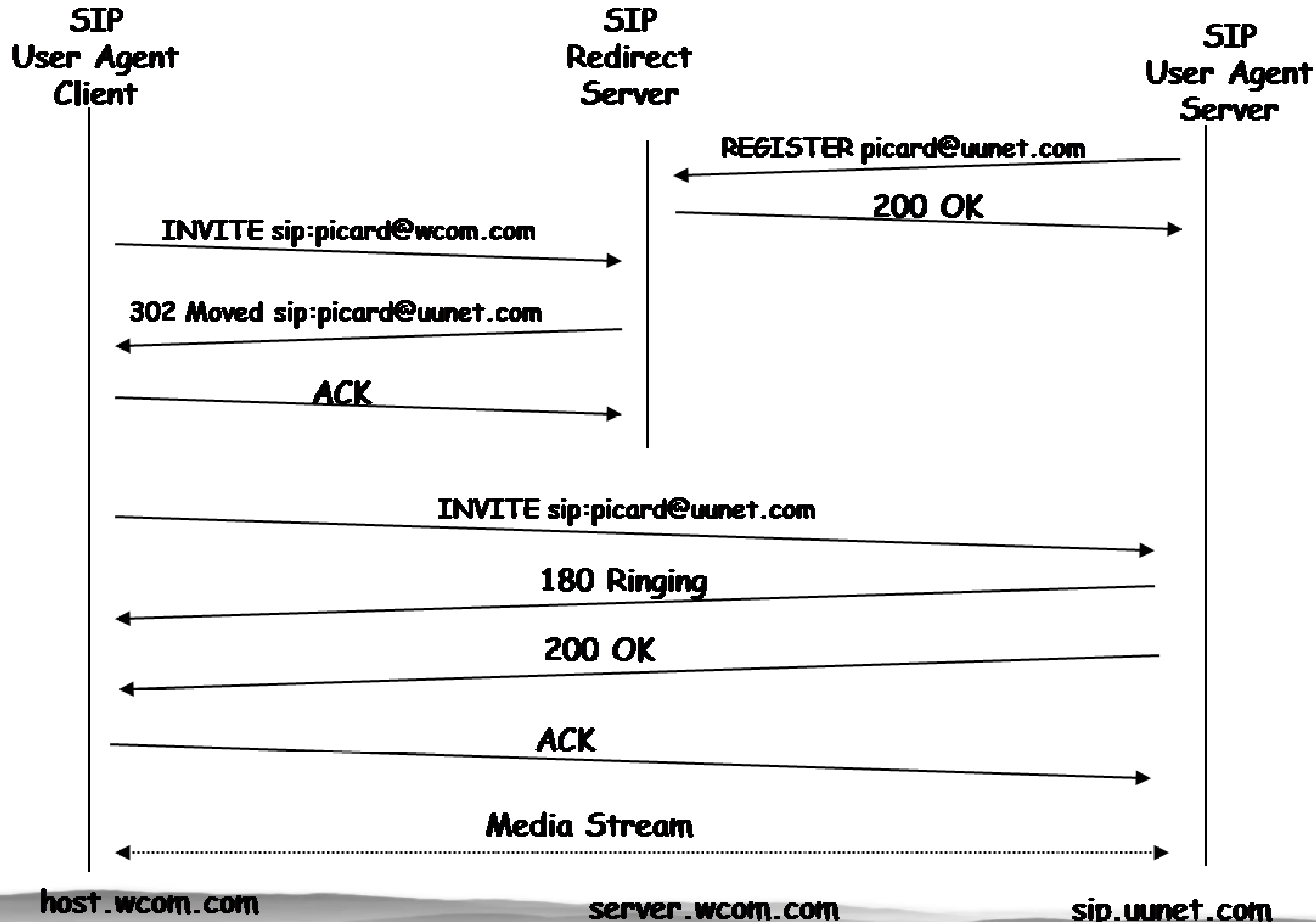
- Different SIP servers to handle this
  - Registrar: Accepts REGISTER requests from clients (caller or callee)
  - Redirect: Sends address of next hop towards callee back to caller
  - Proxy: Decides next hop and forwards request (as a broker)

SIP Registrar

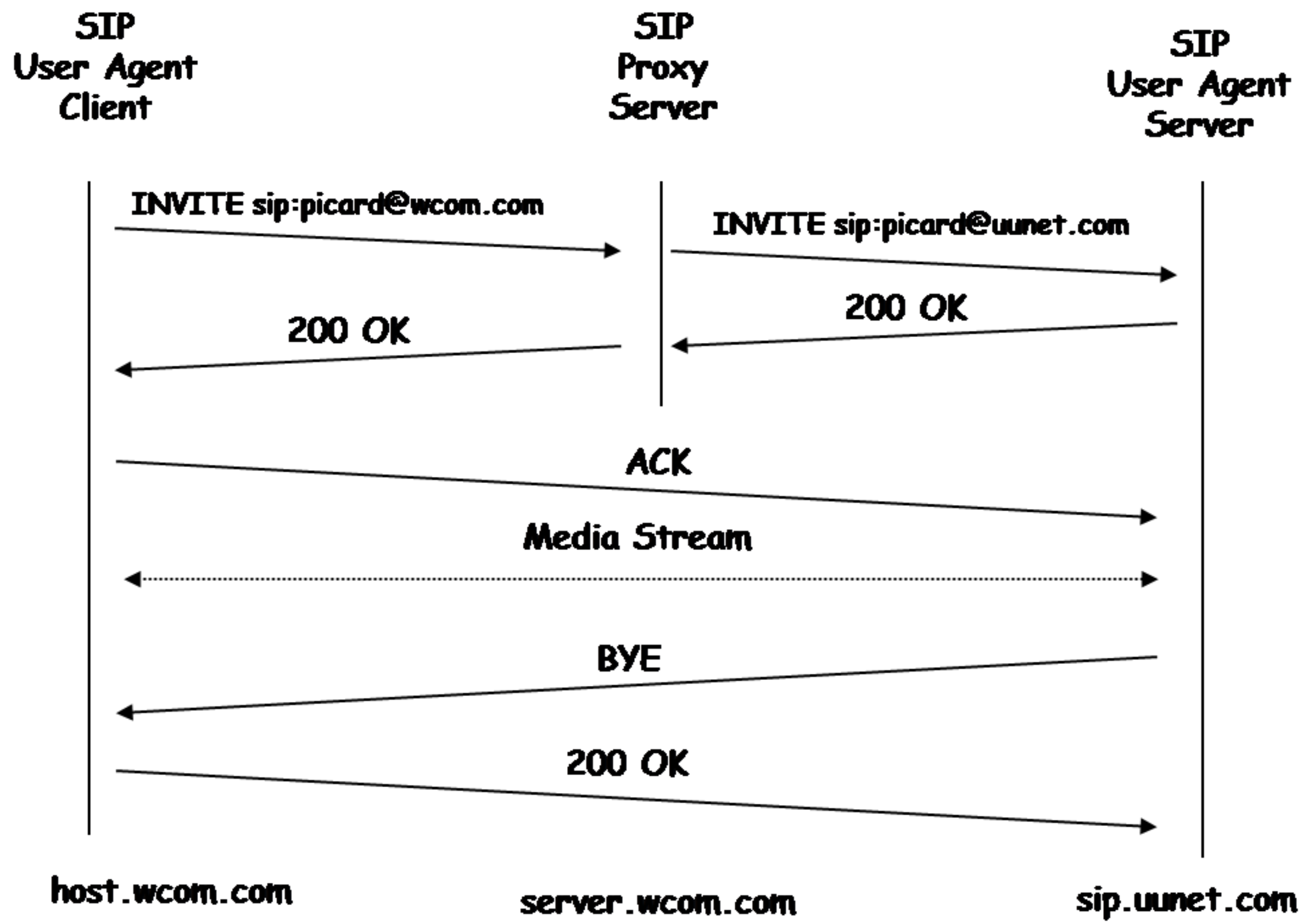
- When Bob starts SIP agent, agent sends SIP REGISTER message to Bob's registrar server
- Similar function needed by Instant Messaging

```
REGISTER sip:domain.com SIP/2.0
Via: SIP/2.0/UDP 193.64.210.89
From: sip:bob@domain.com
To: sip:bob@domain.com
Expires: 3600
```

Using Redirect Server



Using Proxy Server

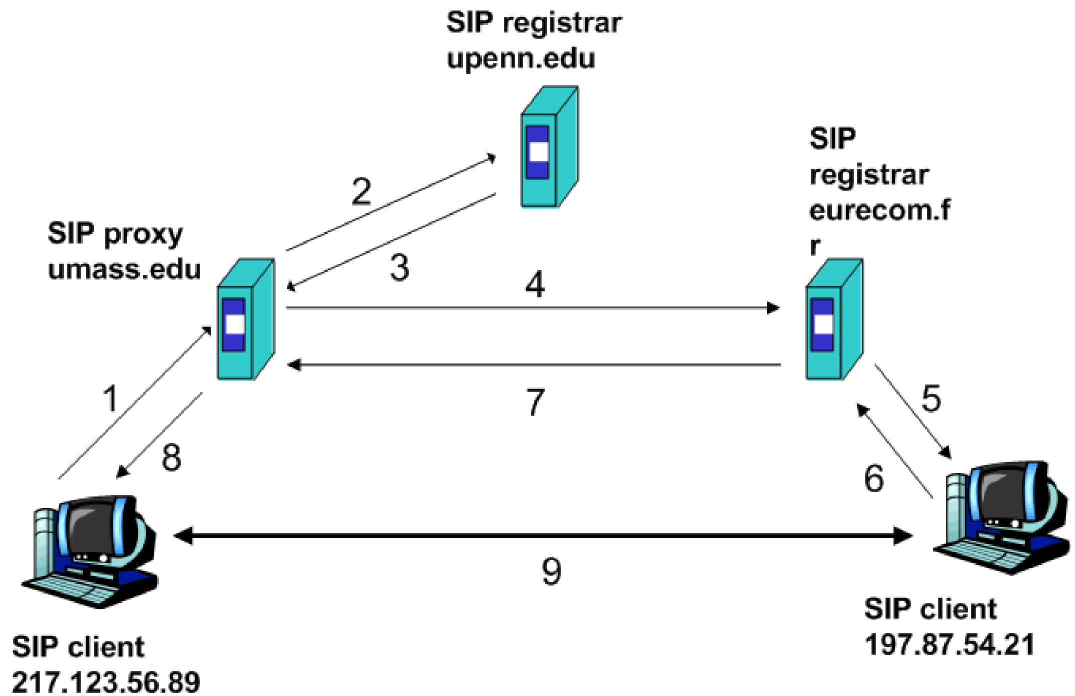


A More Complicated Example

Caller: Bob@umass.edu
Callee: Alice@upenn.edu

(1) Bob's agent sends INVITE message to umass SIP proxy
(2) Proxy forwards request to upenn registrar server

(3) upenn server returns redirect response, indicating that it should try Alice@eurecom.fr
(4) umass proxy sends INVITE to eurecom registrar



(5) eurecom registrar forwards INVITE to 197.87.54.21, which is running Alice's SIP agent
(6-8) SIP response sent back

(9) media sent directly between SIP agents

Summary

- Multimedia networking
- Multimedia applications
  - Streaming, stored audio, video
  - Conversational voice/video over IP
  - Streaming live audio, video
- Protocols
  - Real-time Transport Protocol (RTP)
  - Session Initiation Protocol (SIP)
  - Real Time Streaming Protocol (RTSP)

Homework

- 第7章：P1, P3, P5