Real-Time "Conversational" Applications (RTP & SIP)

Prof. Anja Feldmann, Ph.D.

Balakrishnan Chandrasekaran, Ph.D.

(Based on slide deck of Computer Networking, 7th ed., Jim Kurose and Keith Ross.)

Agenda

- Real-Time Protocol (RTP)
  - Real-Time Control Protocol (RTCP)
- Session Initiation Protocol (SIP)

Real-Time Protocol (RTP)

RTP specifies packet structure for packets carrying audio and video data

- *RFC 3550*

RTP packet provides

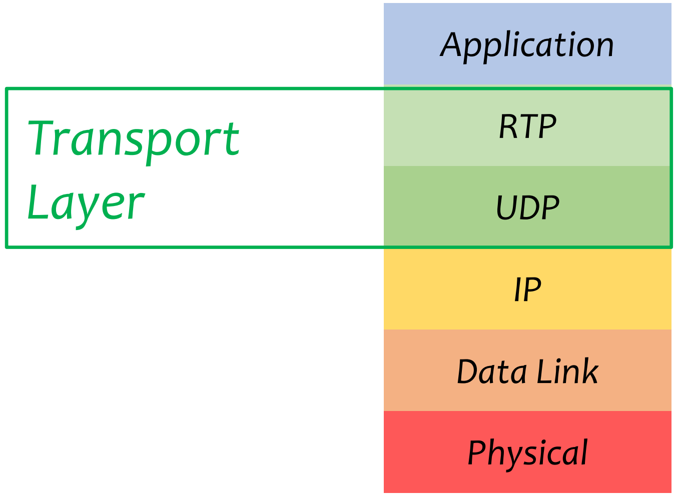
- Payload type identification
- Packet sequence numbering
- Time-stamping

- RTP runs in end systems
- RTP packets *encapsulated* in *UDP* segments
- *Interoperability*
  - If two VoIP applications run RTP, they may be able to work together

RTP: On top of UDP

RTP libraries provide transport-layer interface that ***extends*** UDP:

- Port numbers, IP addresses
- Payload type identification
- Packet sequence numbering
- Time-stamping



RTP: Example

Example: *Sending 64 Kbps PCM-encoded voice over RTP*

- Application collects encoded data in chunks, *e.g., every 20 ms = 160 bytes in a chunk*
- *Audio chunk + RTP header* form RTP packet, which is encapsulated in an UDP segment

- RTP header indicates ***type*** of audio ***encoding*** in each packet
  - Sender can change encoding during conference
- RTP header also contains sequence numbers, timestamps

RTP: QoS?

- RTP *does not* provide any mechanism to ensure *timely data delivery* or other QoS guarantees

- RTP encapsulation *only* seen at end systems (and *not* by intermediate routers)
  - Routers provide *best-effort service*, making no special effort to ensure that RTP packets arrive at destination in timely matter

RTP: Header

payload type	*sequence number*	*time stamp*	*Synchronization Source ID*	*Miscellaneous fields*

RTP: Header

payload type	*sequence number*	*time stamp*	*Synchronization Source ID*	*Miscellaneous fields*

Payload type (7 bits)

- Indicates type of encoding *currently being used*. If sender changes encoding during call, sender informs receiver via 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*

RTP: Header

payload type	sequence number	time stamp	Synchronization Source ID	Miscellaneous fields

Sequence number (16 bits)

- Increment by one for each RTP packet sent
- Detect packet loss, restore packet sequence

RTP: Header

payload type	sequence number	time stamp	Synchronization Source ID	Miscellaneous fields

Timestamp field (32 bits long)

- Sampling instant of first byte in this RTP data packet

- For audio, timestamp clock increments by one for each sampling period *(e.g., each 125 µs for 8 KHz sampling clock)*
- If application generates chunks of *160* encoded samples, timestamp increases by *160* for each RTP packet when source is active. Timestamp clock continues to increase at constant rate when source is inactive.

RTP: Header

payload type	sequence number	time stamp	Synchronization Source ID	Miscellaneous fields

SSRC field (32 bits long)

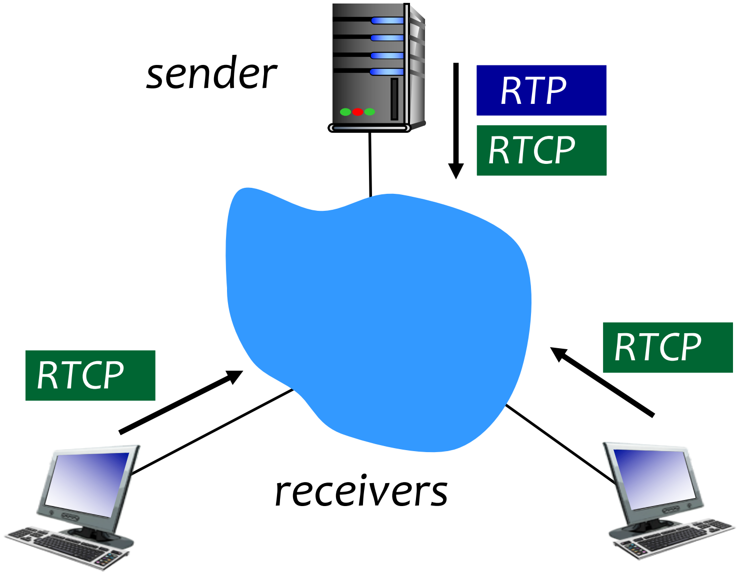
- Identifies source of RTP *stream*. Each stream in RTP session has a distinct SSRC.

Real-Time Control Protocol (RTCP)

Real-Time Control Protocol (RTCP)

- Works in conjunction with RTP
- Each *participant* in an *RTP session* periodically sends *RTCP control packets* to all other participants
- Each RTCP packet contains sender and/or receiver *reports*
  - Report statistics (# packets sent, # packets lost, interarrival jitter) is useful for application
- Feedback used to control performance
  - Sender may modify its transmissions based on feedback

RTCP: Multiple multicast senders



- Each RTP session
  - Typically a single multicast address
  - All RTP /RTCP packets belonging to session use multicast address
- RTP, RTCP packets distinguished from each other via distinct port numbers
- To limit traffic, each participant reduces RTCP traffic as number of conference participants increases

RTCP: Packet types

Receiver report packets

- Fraction of packets lost, last sequence number, average interarrival jitter

Sender report packets

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

Source description packets

- Sender's identification, SSRC of associated RTP stream
- Provide mapping between the SSRC and the user/host name

RTCP: Stream synchronization

- RTCP can *synchronize* different media streams within an RTP session
  - e.g., videoconferencing app: each sender generates one RTP stream for video, one for audio.
- Timestamps in RTP packets *tied to the video, audio sampling clocks*
  - Not tied to wall-clock time!
- Each RTCP *sender-report packet* contains (for the most recently generated packet in associated RTP stream):
  - Timestamp of RTP packet
  - Wall-clock time for when packet was created
- Receivers uses *association* to synchronize playout of audio, video

RTCP: Bandwidth scaling

RTCP attempts to limit its traffic to 5% of session bandwidth.

Example: one sender, sending video at *2 Mbps*

- RTCP attempts to *limit* RTCP traffic to *100 Kbps*
- RTCP allocates *75%* of the rate to receivers, and *25%* to the sender

- *75 Kbps* is *equally shared* among receivers:
  - With R receivers, each receiver gets to send RTCP traffic at *75/R Kbps*

- Sender can use 25 Kbps for RTCP

- Participant determines RTCP packet transmission period by calculating avg. RTCP packet size (across entire session) and dividing by allocated rate

Session Initiation Protocol (SIP)

SIP: Long-term vision

- All telephone calls, video conference calls take place over Internet
  - People identified by names or e-mail addresses, rather than by phone numbers
  - Can reach *callee* (if callee so desires), no matter where callee *roams*, no matter what IP device callee is currently using
- *RFC 3261*

SIP: Services

SIP provides mechanisms for ***call setup***

- For *caller* to let *callee* know she wants to *establish* a call
- So caller, callee can agree on media type and encoding
- To *end* call

Determine current IP address of callee

- maps *mnemonic* identifier to current IP address

Call management

- *Add new media streams* during call
- *Change encoding* during call
- *Invite* others
- *Transfer*, *hold* calls

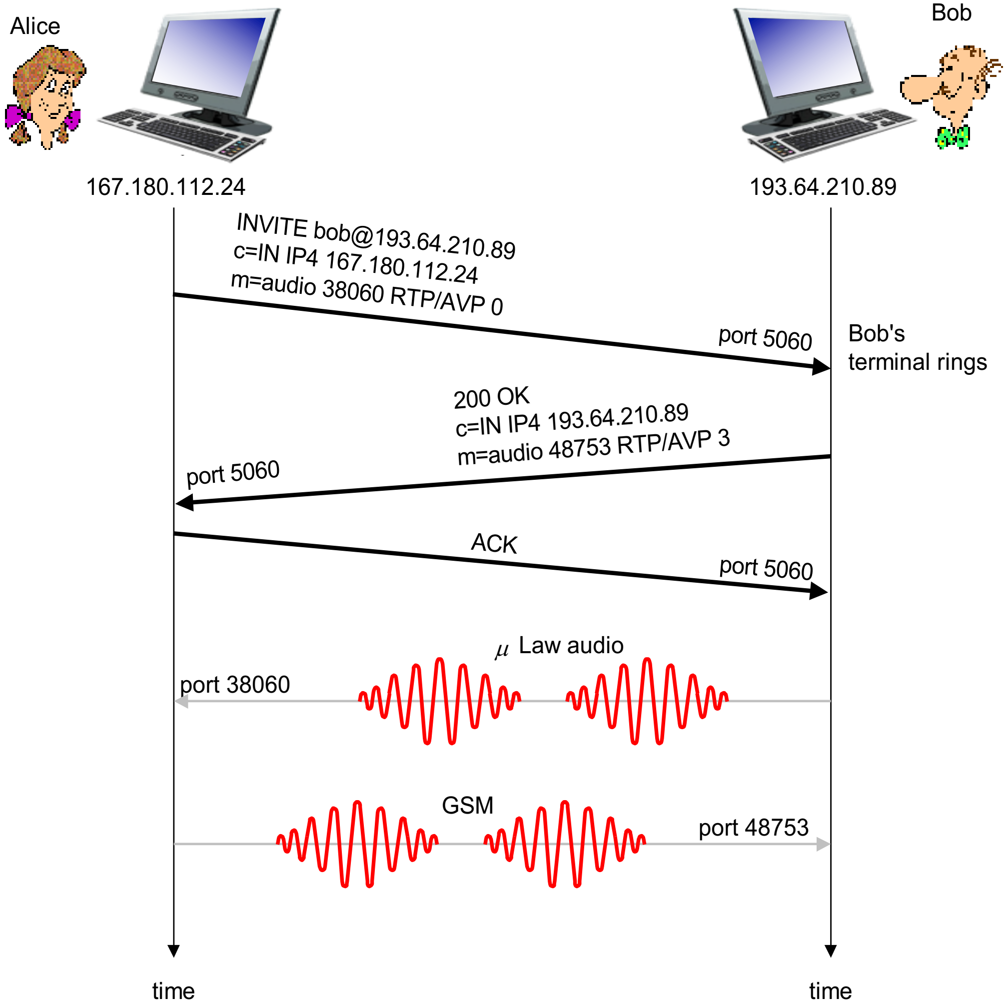
SIP Example: Calling a known IP addr.

- Alice's SIP *invite message* indicates her *port number, IP address, and encoding* she prefers to receive (*PCM**)

- Bob's *200 OK* message indicates his *port number, IP address, and preferred encoding* (GSM)

- SIP messages can be sent over TCP or UDP; here, it is sent over RTP/UDP

- Default SIP port number is 5060



SIP: Setting up a call

Codec negotiation

- Suppose Bob doesn't have the *PCM** encoder
- Bob will instead reply with *606 Not Acceptable Reply*, listing his encoders
- Alice can then send *new INVITE* message, advertising a different encoder

Rejecting a call

- Bob can reject with replies *"busy," "gone," "payment required," "forbidden"*

- Media can be sent over RTP or some other protocol

SIP: Example Message

We don't know Bob's IP address

- Intermediate SIP servers needed!

- Alice sends, receives SIP messages using SIP default port 5060

- Alice specifies in header that her SIP client sends, receives SIP messages over UDP

```
INVITE sip:bob@domain.com SIP/2.0
Via: SIP/2.0/UDP 167.180.112.24
From: sip:alice@hereway.com
To: sip:bob@domain.com
Call-ID: a2e3a@pigeon.hereway.com
Content-Type: application/sdp
Content-Length: 885

c=IN IP4 167.180.112.24
m=audio 38060 RTP/AVP 0
```

Notes:

- *HTTP message syntax*
- *SDP: Session description protocol*
- *Call-ID is unique for every call*

SIP: Name translation & user location

Caller wants to call callee, but only has callee's name or e-mail address!

Need to get IP address of callee's current host:

- User moves around
- *DHCP* protocol
- User has different IP devices (*e.g., PC, smartphone, and car device)*

Result can be based on:

- Time of day (work, home)
- Caller (don't want boss to call you at home)
- Status of callee (calls sent to voicemail when callee is already talking to someone)

SIP: Registrar

One function of *SIP server*: ***registrar***

- When Bob starts SIP client, client sends *SIP REGISTER* message to Bob's *registrar server*

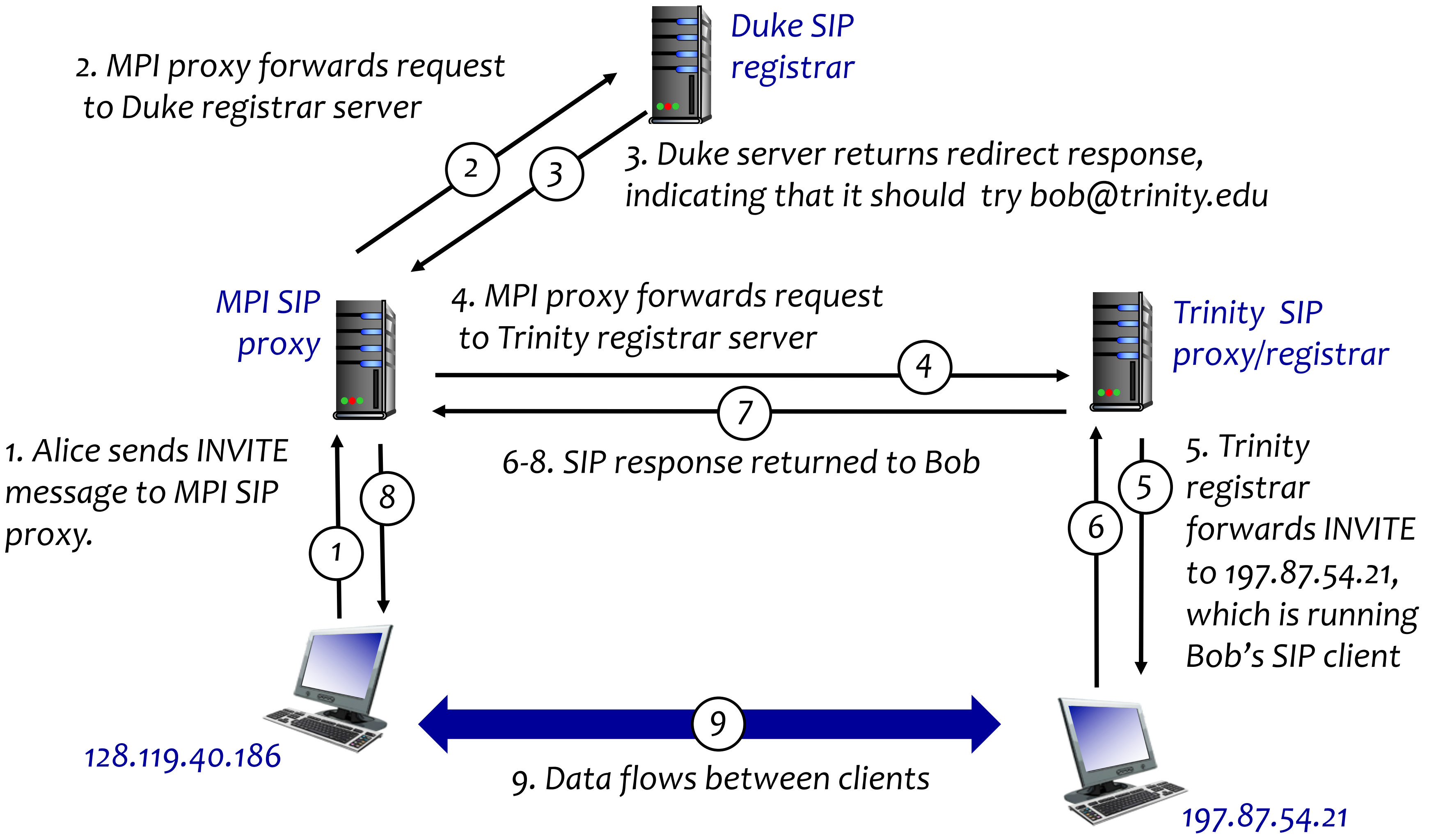
```
REGISTER sip:domain.com SIP/2.0
Via: SIP/2.0/UDP 193.64.210.89
From: sip:bob@domain.com
To: sip:bob@domain.com
Expires: 3600
```

SIP: Proxy

Another function of *SIP server*: ***Proxy***

- Alice sends *INVITE message* to her proxy server
  - Contains address sip:bob@domain.com
  - Proxy responsible for routing SIP messages to callee, possibly through multiple proxies
- Bob sends response back through same set of SIP proxies
- Proxy returns Bob’s SIP response message to Alice
  - Contains Bob’s IP address

SIP Example: alice@mpi.de calls bob@duke.edu



SIP: Comparison with H.323

H.323

- Another signaling protocol for real-time, interactive multimedia
- Complete, vertically integrated suite of protocols for multimedia conferencing: signaling, registration, admission control, transport, codecs

SIP

- Single component
- Works with RTP but does not mandate it. Can be combined with other protocols and services.

H.323

- Comes from the *ITU* (telephony)
- Has telephony flavor

SIP

- Comes from *IETF*
- Borrows much of its concepts from HTTP and has a Web flavor
- Uses the *KISS* principle

Summary

- Real-Time Protocol (RTP)
  - Packet structure, control protocol, stream synchronization
- Session Initiation Protocol (SIP)
  - registrar, proxies, call setup