Lecture 7: Real-time Streaming

Goals:
- Understand “real-time”
- Be aware of the problems of
  - Synchronization
  - Jitter
  and their solutions
- Understand the RTP architecture of senders, receivers, mixers and translators
- Understand some uses for the sister protocol RTCP.
- Understand how synchronization is achieved in RTP
- Know about SDP and RTSP

Outline:
- Real-time communication in general
- Synchronization
- RTP
  - Terminology
  - Header structure
  - Transport
  - Interleaving
- RTCP (Real-time Control Protocol)
- RTP Synchronization Model
- SDP – Session Description Protocol
- RTSP – Real-time Streaming Protocol

Real-time definition

What means “real-time”?
- Real-time communication: No noticeable delay
- Real-time application: An application able to have a sustained processing of a temporal data stream
  - Without having queues building up
  - With reasonable sustained quality level
But - there is of course always delay to some extent!
- Different communication scenarios may have different sensitiveness to delay.
Real-time Communication

Real-time communication involving humans typically fall into at least one of the following two categories:
- **Dialog situation** - when a user has some kind of an interaction with other user(s) or machines
  - Telephony
  - TV query shows where you may phone.
- **Moment sharing** – the feeling of me sharing a unique event with possibly other humans. Notion of being among the select few who first know what’s going on.
  - TV sports broadcast

Typical Design Decisions

When having real-time communication we usually
- **Skip retransmission**
  - The time it takes to send ACK:s/NACK:s upstream and wait for the lost packet may cost to much in delay.
- **Skip flow control**
  - Receiver should be a real-time application being able to cope with the (bounded) rate the sender transmits at.

Also if we have a broadcast scenario no or little upstream communication can be had.

Real-time Scenario

Transmitter Reference Clock

Receiver Reference Clock

Frames created regularly at frequency f

Network induces delay and jitter

Buffer smooths out the effect of jitter

Jitter makes the packets arrive irregularly

Typical Design Decisions

When having real-time communication we usually
- **Skip retransmission**
  - The time it takes to send ACK:s/NACK:s upstream and wait for the lost packet may cost to much in delay.
- **Skip flow control**
  - Receiver should be a real-time application being able to cope with the (bounded) rate the sender transmits at.

Also if we have a broadcast scenario no or little upstream communication can be had.

Technical Issues

There are two fundamental issues when designing a real-time communications system:
1) **Synchronization.**
   Sender and receiver needs to be synchronized to each other to avoid re-occurring buffer over-flows and under-runs
   - Essentially having the sender and receiver following a clock at exactly the same frequency.
2) **Jitter.**
   Jitter is the varying of the packet-delay around the mean delay. Jitter is usually caused by the network, but could also result from processing stages in the sender that varies in time.
   - The larger the jitter the larger the buffer at the receiver-side has to be.
Achieving Synchronization

1) Implicit synchronization
The receiver buffer level is used to control the speed of play-out. The most simplistic approach is to throw away suitable input at buffer over-flow and interpolate at buffer under-run

2) Global synchronization
Transmitter and receiver can be part of a global synchronization system like NTP (Network Time Protocol)

3) Point-to-point synchronization
Receiver need to now from the syntax of the received data stream when certain bits should have been sent in relation to other synchronization points. Using the arrival-times of these bits the receiver employs a smoothing algorithm like PLL (Phase Locked Loop) to slave a local clock to the transmitter's system clock.

Implicit Synchronization - Example

- Assume the receiving buffer is of size $N$ bytes
- Assume data is sent isochronously in packets of size $M < N$ bytes.

Point-to-point Synchronization

Example MPEG Systems

IETF Audio/Video Transport

The avt working group has defined the Real-Time Protocol (RTP) for multimedia transport (RFC3550)

- Collect common fields for synchronization, packet ordering and data format in one header.
- Different applications may reuse same RTP-stack with respect to the above mentioned parameters
- Thanks to the type identifier a library or a host may safely determine the format of incoming streams and provide the same service for various formats.
  - See translation and mixing
RTP does not...

...guarantee
- Bandwidth
- Low jitter

RTP is strictly an end-to-end protocol. Network guarantees on bandwidth, jitter etc is up to separate quality-of-service mechanisms.

**RTP Terminology**

**RTP Session**
The set of communicating entities which may via RTP and RTCP know of each other’s identities.

**Synchronization Source**
The origin of a stream where RTP-packets are generated. Identified by a 32-bit number: SSRC.

**Translation**
The act of converting a stream to another format and/or rate.

**Mixing**
The act of adding two or more streams together and encode the result into a new stream (with new SSRC).

### RTP Oneway Unicast Example

- SSRC:1 sending a single media stream to SSRC:2
- There will be three streams.
- The RTP stream carries the media.
- Two RTCP (Real-Time Control Protocol) in each direction carries extra control information.
- It is recommended that the RTCP traffic only adds an extra 5% to the RTP bandwidth.

### RTP Translation Example

This server is a SIP-proxy and also redirects RTP session ports to go through the server instead (thereby passing the firewall). The server can then add:
- Privacy by encryption
- Translation services, i.e., transcoding of formats not understood by client
- Bit-rate conversion

Terminals only supporting low quality rates.
RTP Mixing Example

This machine is mixing together the separate video streams into one video stream which is multicast back to members.

SSRC: 2
SSRC: 4
SSRC: 5
SSRC: 10
CSRC: (2, 4, 5, 12)

Synchronization Source Identifier

- Every RTP packet is marked with a 32-bit number identifying a single timing and sequence number space.
- SSRC provides a transport layer independent “tag”. RTP may run over various protocols (IPv4, IPv6, ATM).
- SSRC must be unique among all hosts taking part in the RTP session.
- SSRC is picked randomly, but the sender must in case of a detected collision issue a BYE message over the control channel and change SSRC.

Contributing Source Identifiers

- When a mixer creates a new stream out of others it may include the SSRCs of up to 15 of the contributing sources.
- CC-field denotes the number of source identifiers in CSRC field.
- Using CSRC is a safety measure against loops in a RTP streaming session.

RTP Header

- The RTP header includes:
  - Timestamp
  - Synchronization source identifier (SSRC)
  - Contributing source identifiers (CSRC) (optional)
  - Payload
  - Padding
Payload

- The payload type (7 bits) indicate the format of the media put into the payload area. Compared to the number of codecs out there one realizes that assigning the payload type from the dynamic range might be common. It is then up to the session handling mechanism to inform parties about the type used.
- A flow can only carry one payload type. But the type can change.
- Each payload type has a reference clock frequency associated with it.

Payload Types (RFC3551)

<table>
<thead>
<tr>
<th>PT</th>
<th>A/V</th>
<th>Clock Freq</th>
<th>Channels</th>
<th>MIME-type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>audio</td>
<td>8000Hz</td>
<td>1</td>
<td>audio/gsm</td>
<td>13.2bps speech coder</td>
</tr>
<tr>
<td>10</td>
<td>audio</td>
<td>44100Hz</td>
<td>2</td>
<td>audio/16</td>
<td>two-channel CD-style</td>
</tr>
<tr>
<td>11</td>
<td>audio</td>
<td>44100Hz</td>
<td>1</td>
<td>audio/16</td>
<td>one-channel CD-style</td>
</tr>
<tr>
<td>14</td>
<td>audio</td>
<td>90KHz</td>
<td>special</td>
<td>audio/mpa</td>
<td>MPEG audio layer 1,2,3</td>
</tr>
<tr>
<td>26</td>
<td>video</td>
<td>90KHz</td>
<td>-</td>
<td>video/jpeg</td>
<td>Motion-JPEG</td>
</tr>
<tr>
<td>32</td>
<td>video</td>
<td>90KHz</td>
<td>-</td>
<td>video/mpv</td>
<td>MPEG-1,2 elementary stream</td>
</tr>
<tr>
<td>33</td>
<td>a/v</td>
<td>90KHz</td>
<td>special</td>
<td>video/mp2t</td>
<td>MPEG-2 transport stream</td>
</tr>
<tr>
<td>34</td>
<td>video</td>
<td>90KHz</td>
<td>-</td>
<td>video/h263</td>
<td>H.263 video telephony</td>
</tr>
<tr>
<td>35-71</td>
<td>unassigned</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>72-76</td>
<td>reserved</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>77-95</td>
<td>unassigned</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>96-127</td>
<td>dynamic</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Upcoming:
- AMR / AMR-WB speech codec used in 3G mobile system
- Motion-JPEG2000
- H.264 video codec (The new MPEG-4 AVC profile)
- DV audio/video
- MPEG-4 Elementary streams

Timestamp

- 32-bit resolution
- Indicates the sampling instant of first octet of payload data.
- The mapping from transmitter’s reference clock to timestamp is defined by payload type.
- Initial value is random
- Clock resolution should be enough to support synchronization and jitter measurements based on timestamp alone.

Note that sender also transmits timestamps of “wall clock time” in RTCP sender reports. These are needed to synchronize two or more streams with each other.

Sequence Number

- Each packet has a 16-bit sequence number
- Increases with one for each new sent packet.
- Starts with a random value
- Used to detect packet loss. Need not be related to play-out order
- Basic RTP does not provide error correction
  - FEC extensions exist
  - limited ARQ profile exist
Marker bit

- Marks this packet as special in a sense defined by payload type.
  - May be set in speech formats for indicating talk starts again after a period of silence (or comfort noise)
  - Most video formats specify the marker bit to be set to “1” in the last packet for a frame. This gives a simple indication that it is time to display.

RTP Embedding

- RTP is designed to run over any packet switched network. On Internet we use UDP/IP.

```
<table>
<thead>
<tr>
<th>IP-header</th>
<th>UDP header</th>
<th>RTP header</th>
<th>payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>8</td>
<td>12</td>
<td></td>
</tr>
</tbody>
</table>
```

The overhead per RTP packet will be 40 bytes. This is sometimes seen as too excessive and several header compression techniques have been suggested:

- ROBust Header Compression (ROHC) – RFC3095
- Enhanced Compressed RTP (CRTP) - RFC3545
- IP Header Compression – RFC2507 (LuTH)

RTP Interleaving

How to mix audio and video?

- Same session, different SSRC
  - Does not support different QoS profiles
  - Impossible in a multicast scenario to join just the audio stream (for instance)
- Different sessions, same SSRC
  - works ok. SSRC:s totally unrelated
- Different sessions, different SSRC
  - works ok. SSRC:s totally unrelated

Real-time Control Protocol (RTCP)

Some of the functions of the sister-protocol RTCP are:

- Provide feedback of the reception quality
- Carry a persistent identifier for a RTP source.
- Carry NTP timestamps enabling inter-media synchronization.
- Keep track of number of participants so that RTCP can scale (i.e., lower the rate at which control messages is sent)
- Optional minimal session control.

**Note:** If RTP data is sent to the UDP-port P (should be even) RTCP messages should be sent on port P+1.
RTCP Compound Packets

Several RTCP packets are concatenated to form one compound packet. These will at least contain two individual packets (SR/RR and a SDES).

<table>
<thead>
<tr>
<th>Sender OR Receiver Report (SR/RR)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Additional Receiver Reports)</td>
</tr>
<tr>
<td>Source Description (SDES)</td>
</tr>
<tr>
<td>BYE or APP packet (optional)</td>
</tr>
</tbody>
</table>

There is also an encryption option which is skipped here.

RTCP Sender Report (SR)

- **SSRC of sender**
- NTP timestamp (most significant word)
- NTP timestamp (least significant word)
- RTP timestamp
- sender's packet count
- sender's octet count
- SSRC_1 (SSRC of first source)
- fraction lost
- cumulative number of packets lost
- extended highest sequence number received
- inter-arrival jitter
- last sender report NTP timestamp (middle 32 bits)
- delay since last sender report (65536Hz accuracy)

RTCP Receiver Report (RR)

- **SSRC of sender**
- NTP timestamp (most significant word)
- NTP timestamp (least significant word)
- RTP timestamp
- sender's packet count
- sender's octet count
- SSRC_1 (SSRC of first source)
- fraction lost
- cumulative number of packets lost
- extended highest sequence number received
- inter-arrival jitter
- last sender report NTP timestamp (middle 32 bits)
- delay since last sender report (65536Hz accuracy)

When participant in an RTP-session only receives flow we skip the sender information block. Note that a receiver need to have an SSRC anyway! A problem here is to which address and port do we send our RTCP messages! If it's a multicast session on UDP(IP#, port) we send them to UDP(IP#,port+1), but if it's a unicast session the RTCP upstream IP# and port has to be configured by a higher level session protocol.

RTCP Source Description (SDES)

- **SSRC / CSRC_1**
- SDES items
- SSRC / CSRC_2
- SDES items
- ...

SDES information blocks must be sent by every sender. It gives some textual information regarding the flow. The flow might possibly be the result of mixing in which case information for each mixed flow is present. An SDES item looks like

| item code | length | UTF-8 string |
### SDES Items

- **CNAME (1)**: Mandatory Item. Should uniquely identify the source. Name should be algorithmically derived from e.g., host IP. Flows from different RTP sessions but with the same CNAME indicates inter-media synchronization.
- **NAME (2)**: User Name. Ex: “John Doe, Bit Recycler”
- **EMAIL (3)**: Electronic mail address according to RFC2822.
- **PHONE (4)**: Phone number with a “+” replacing int. access code.
- **LOC (5)**: Geographic user location. Ex: “Room 473 1tr A-hus”.
- **TOOL (6)**: Name and optionally version of application generating the flow
- **NOTE (7)**: Transient message describing state of source.
- **PRIV (8)**: Experimental or application specific. Has another syntax for the “UTF-8 string” field.

### RTP Synchronization 1

**Global Synchronization**

Different RTP sessions on same host

**RTP Synchronization 2**

Inter-media synchronization in absence of global synchronization.

Receiver sees that CNAMEs are the same for SSRC:1 and 2. Receiver then knows these flows use the same reference clock and can achieve point-to-point synchronization with RC1 using timestamps from both flows. SSRC: 1 and 2 may be played out in absolute synch!

### Session Description Protocol (SDP)

- **RFC2327**
- **MIME-type: application/sdp**
- SDP conveys information needed to join one or more multimedia sessions
  - Session name and purpose
  - Time(s) when session is active
  - What media is involved
  - Information on how to receive those media (addresses, ports, formats etc.)
  - Optionally bandwidth requirements
  - Optionally how to contact person responsible
SDP Syntax (selected parts)

Information is in human readable form coded using UTF-8 encoded Unicode character set.

Session description:
v= (protocol version)
o= (owner/creator and session identifier).
   <username> <session id> <version> <network type> <address type> <address>
s= (session name)
i=* (session information)
   <more optional fields>
   <one or more time descriptions>
a=* (zero or more session attribute lines)

Time description:
t= (time the session is active)
r=* (zero or more repeat times)

Media description:
m= (media name and transport address)
i=* (media title)
c=* (connection info - optional if included at session-level)
b=* (bandwidth information)
k=* (encryption key)
a=* (zero or more media attribute lines)

SDP Example

v=0
o=mhandley 2890844526 2890842807 IN IP4 126.16.64.4
s=SDP Seminar
i=A Seminar on the session description protocol
u=http://www.cs.ucl.ac.uk/staff/M.Handley/sdp.03.ps
e=mjh@isi.edu (Mark Handley)
c=IN IP4 224.2.17.12/127
t=2873397496 2873404696
a=recvonly
m=audio 49170 RTP/AVP 11
m=video 51372 RTP/AVP 32
m=application 32416 udp wb
a=orient:portrait

Note that an SDP-session is not the same as an RTP session. The three medias above will flow into three parallel RTP sessions on different UDP ports but to the same (multicast) address.

Real-time Streaming Protocol (RTSP)

- RFC2326
- “Network remote control for multimedia servers”
- Mimics HTTP. Example of the rtp: URI:
  rtsp://videoserv.liu.se/filmtest/audio
- Well-know port 554.
- RTSP is state-ful while HTTP is state-less. An RTSP session typically contains state such as:
  - One of {Init, Ready, Playing, Recording}
  - PLAY-commands are queued at server
  - A PAUSE may be set to appear at a specific time for many URI:s at the same time!

RTSP Client to Server Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SETUP</td>
<td>Agree on how a single media should be streamed. Addresses, protocols etc. (in an on-demand scenario also ports for upstream RTCP messages)</td>
</tr>
<tr>
<td>DESCRIBE</td>
<td>Request session description data</td>
</tr>
<tr>
<td>PAUSE</td>
<td>Pause the indicated URI</td>
</tr>
<tr>
<td>PLAY</td>
<td>Start play-back of indicated URI</td>
</tr>
<tr>
<td>RECORD</td>
<td>Tell server to start record from previously given invitation.</td>
</tr>
<tr>
<td>SET_PARAMETER</td>
<td>Generic method for setting stream params</td>
</tr>
<tr>
<td>GET_PARAMETER</td>
<td>Generic method for reading stream params</td>
</tr>
<tr>
<td>ANNOUNCE</td>
<td>Send session information for later recording</td>
</tr>
<tr>
<td>OPTIONS</td>
<td>Query URI for supported methods</td>
</tr>
<tr>
<td>TEARDOWN</td>
<td>Stop streaming and free resources</td>
</tr>
</tbody>
</table>
RTSP Session Setup

A client may connect to a RTSP server and query different URI:s via DESCRIBE-calls.

When client first issues a SETUP-call for a certain media URI an RTSP-session is initiated,

Client->S: SETUP rtsp://foo/twister/video RTSP/1.0
CSeq: 3
Transport: RTP/AVP;unicast;client_port=8002-8003

Server->C: RTSP/1.0 200 OK
CSeq: 3
Transport: RTP/AVP;unicast;client_port=8002-8003;
server_port=9004-9005
Session: 12345678

This is indicated by the Session: field in the server response. The client should in future requests include this in header. The session identifier should be a randomly created string with at least 8 characters.

SDP Pointing out RTSP

An SDP session description might point out media as a set of RTSP controlled resources.

v=0
o=- 2890844256 2890842807 IN IP4 204.34.34.32
s=I came from a web page
t=0 0
c=IN IP4 0.0.0.0
m=video 8002 RTP/AVP 31
a=control:rtsp://rtspserv.com/movie.aud
m=audio 8004 RTP/AVP 3
a=control:rtsp://rtspserv.com/movie.vid

Note that the m= sections does not convey any address/port information for how media is streamed. This is left to the SETUP-call. This kind of SDP-data could be given from a HTTP-request but can also be the result of a DESCRIBE-call for a specific RTSP URI.

RTSP Presentation

Each rtsp: URI correspond to a presentation. A presentation is one of two things

- A single media
- A container file consisting of one or more submedias

To find out which a client can

- Issue a SETUP call and prepare to get back an error “403 Aggregate Operation Not Allowed”.
- Issue a DESCRIBE call and study the SDP m= fields. Although note that DESCRIBE-support in an RTSP-server is not a “MUST”!

For playing an aggregated media a separate SETUP call per submedia is needed.

The RTSP-server could support aggregate control (PLAY, PAUSE etc.) of all submedias. One then issues the method on the container URI.

Streaming Through a Firewall

In general UDP-based traffic is tougher to handle for firewalls since there is no transport-level connection mechanism like TCP’s which the firewall could intercept. Several solutions exists however:

- A firewall might work on application-level and intercept SETUP-calls and open up for incoming flows.
- RTSP includes an option for embedding RTP/RTCP packets over TCP
- One could deploy an RTSP proxy beside the firewall and instruct the client to always go via the proxy. Such a proxy instructs the real RTSP-server to stream to the proxy server instead.
IETF Working Groups

avt – Audio/Video Transport
- RTP / RTCP
- Payload formats for various audio and video formats (μ-law/linear PCM, H.263, MPEG, GSM..)

mmusic – Multiparty Multimedia Session Control
- SDP
- RTSP
- SIP (earlier versions before SIP got a working group of its own)

Summary

- Real-time requires synchronization.
- Jitter affects buffer size.
- RTP and RTCP offers a common framework for handling
  - Synchronization
  - Packets arriving out of order
  - Source Identification
  - Loop detection
- SDP is a textual syntax for describing session setup parameters.
- RTSP is the “remote VCR” control protocol.