Multimedia

TELE3118 lecture notes
© by Tim Moors, May 25, 2011
Multimedia
Lecture outline

Jitter

Interactive media (e.g. VOIP)
• RTP, RTCP

Stored media (delay isn’t critical)
• RTSP
Resources

Relevant sections of Tanenbaum:

- 6.4.3 – RTP, RTCP
- 7.4.4 – impediments to RTP
- 7.4.3 – RTSP

- Kurose & Ross: (4th ed): Ch. 7.1-7.1.2, 7.2, 7.3-7.3.2, 7.4-7.4.2
Multimedia presentations

Are dispersed in time, e.g. view video; phone conversation
Consist of series of samples (eg video frame, audio amplitude)

- **Streaming**: Client presents early samples before receiving later ones.
  - Necessary for live source
  - Desirable for stored source:
    - Start presenting before have transferred all samples (important for bulky video)
    - Receiver needn’t store all samples => Less memory

- **Sensitive to delay** (variation): must be presented at right time
  - Absolute delay affects interaction between ends, e.g. satellite phone call vs landline; sluggish remote control
  - Delay variation also bad, both in presentation & for buffering

- **Loss tolerant**: OK to lose occasional sample, in network or discard if too late. Tolerance depends on compression level.
Transporting multimedia

- TCP seems wrong:
  - Don’t need perfect reliability
  - Don’t want delay from retransmission
  - Don’t want to slow down when network congested (though probably should)
- But Address Translators & Firewalls often block UDP
- Many pragmatic popular multimedia apps (e.g. Flash video, Skype) use proprietary techniques over TCP (often HTTP)

2 ways to stream media:
- **Pull server**: Receiver requests more when needed. Many requests sent to single sender (server).
- **Push server**: Receiver requests a flow at a specified rate. Sender (server) must remember state.
Jitter control

Packets crossing a network incur variable delays, e.g. MAC access delay, router queueing delay. **Jitter** measures delay variation.

Streaming media applications are sensitive to jitter, e.g. if a packet arrives later than normal, then the receiver may have nothing to play out for a while, leading to audio silence or flickering images.

Ways to address jitter:
- **In the network layer**: Providing guaranteed “Quality of Service” e.g. using Leaky Bucket regulators and scheduling (limit the possible jitter).
- **In the application layer**: Buffer received information to add a delay that compensates for the jitter.
Buffering to compensate for jitter

Client-side buffering, playout delay compensates for network-added delay, delay jitter

Slide from Kurose and Ross
Buffering to compensate for jitter

The receiver buffers information before playback, with duration of buffering compensating for variable transmission delay.

Packet 4 incurs more propagation delay (3 sec) => less playback delay (7 sec)

Packet 8 incurs excessive delay => discard it, or extend buffering and create gap in playback

Tanenbaum Fig 5-31
Outline
Protocol: RTP

**Full name:** Real-Time Protocol

**Purpose:** Transfer real-time content

**Layer:** Application

**Uses:** UDP

  **Identified by:** port number 5004

**Standards:** RFC 3550
## RTP packet format

Sender can change type during session, e.g. adapt to network conditions by changing codec type. Receiver can join multicast transmission at any point & immediately determine type of encoding. Allows playout of packets with correct timing, irrespective of delay variation when traversing network.

Multiple sources may share a UDP socket; e.g. separate audio & video sources. Part of same RTP stream to enable lip sync etc. Enables detection of lost packets => interpolate their values.

<table>
<thead>
<tr>
<th>V=2</th>
<th>P</th>
<th>X</th>
<th>CC</th>
<th>M</th>
<th>PT</th>
<th>sequence number</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>timestamp</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>synchronization source (SSRC) identifier</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>contributing source (CSRC) identifiers</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>....</td>
</tr>
</tbody>
</table>

### Minor fields:
- **Ver** = version number
- **P** = 1 => payload padded to multiple of 4 bytes, last byte=# of padding bytes
- **X** => extension header present
- **M** = Application-specific marker bit, e.g. mark start of video frame.
- **CC** = # of “contributing sources”, e.g. speakers in an audio conference; Contributing source identifiers @ end of packet.
Protocol: RTCP

Full name: RTP Control Protocol
Purpose: Report performance of RTP delivery s.t. sender knows how to adjust
Layer: Application
Uses: UDP

Identified by: port number 5004
As for RTP; merely different Payload Types

Standards: RFC 3550
RTCP - Continued

- For an RTP session there is typically a single multicast address; all RTP and RTCP packets belonging to the session use the multicast address.
- To limit traffic, each participant reduces their RTCP traffic as the number of conference participants increases.

Slide based on one from Kurose and Ross
Copyright © 25/05/11, Tim Moors
RTCP messages

1st 2 bytes as for RTP, but non-media Payload Types indicate type of message:

- **Sender Report**: Indicates
  - current time @ sender: Allows receivers to synchronise different streams
  - bytes/packets sent: => receiver can measure loss

- **Receiver Report**:
  - fraction of packets lost
  - highest sequence number received
  - interarrival jitter

RTCP aims to use only 5% of session bandwidth
=> Receiver Report rate $\propto 1/\#$ receivers
   (learned by multicasting Receiver Reports)
Voice over IP (VOIP)

The H323 protocol stack.

<table>
<thead>
<tr>
<th>Speech</th>
<th>Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>G.7xx</td>
<td>RTCP</td>
</tr>
<tr>
<td>RTP</td>
<td>H.225 (RAS)</td>
</tr>
<tr>
<td></td>
<td>Q.931 (Call signaling)</td>
</tr>
<tr>
<td></td>
<td>H.245 (Call control)</td>
</tr>
<tr>
<td>UDP</td>
<td>TCP</td>
</tr>
</tbody>
</table>

Session Initiation Protocol = alternative signalling/control protocol

Tanenbaum Fig. 7-65
Outline
Protocol: RTSP

Full name: Real-Time Streaming Protocol

Purpose: Controls the delivery of content from a server

Layer: Application

Uses: TCP or UDP

Identified by: port number 554

Standards: RFC 2326

(heavily based on HTTP RFC 2616)

www.rtsp.org (archived here)
RTSP Example

Scenario:
- metafile communicated to web browser
- browser launches media player
- player sets up an RTSP control connection, data connection to streaming server

Slide from Kurose and Ross
Metafile 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 messages = HTTP derivative

- HTTP header fields can be used with RTSP (e.g. Accept:, User-Agent:)
- Request: & Response: precede method & status
  e.g. Response: RTSP/1.0 200 OK\r\n
- New methods include:
  DESCRIBE: What content a stream carries & how formatted
  SETUP: Specify how to transport the stream (e.g. multicast, RTP & port #s)
  SET_PARAMETER: Further control of stream, e.g.
    SetDeliveryBandwidth:
  PLAY: Start streaming
  TEARDOWN: Stop streaming
  - HTTP OPTIONS method lists methods supported by other end

- Server tracks client’s state (session #s & client request sequence #s)
  - Unlike HTTP. Presumably because transfer is spread over time, to reduce receiver buffer or in case live source => server must relate control message (e.g. TEARDOWN) to session.
RTSP Exchange Example

C: SETUP rtsp://audio.example.com/twister/audio RTSP/1.0
   Transport: rtp/udp; compression; port=3056; mode=PLAY

S: RTSP/1.0 200 1 OK
   Session 4231

C: PLAY rtsp://audio.example.com/twister/audio.en/lofi
RTSP/1.0
   Session: 4231
   Range: npt=0-

C: PAUSE rtsp://audio.example.com/twister/audio.en/lofi
RTSP/1.0
   Session: 4231
   Range: npt=37

C: TEARDOWN rtsp://audio.example.com/twister/audio.en/lofi
RTSP/1.0
   Session: 4231

S: 200 3 OK

Slide from Kurose and Ross
Lecture summary

- **Receiver buffing** allows periodic playout of streaming media, despite network jitter
- **RTSP** gives receiver “remote control” of pre-recorded transmissions from server
- **RTP** carries content and timing info needed for playout
- **RTCP** conveys transmission stats (e.g. loss rate) used to guide choice of codec
- **Voice Over IP (VOIP):** Set up using SIP, carried using RTP