Real-time Protocol

Outline
  RTP
  RTCP
  Readings 5.4
  Why? - New Apps, New Protocols
Why RTP?

- Real-time means strong requirements for ‘timely’ delivery
- Internet - Not originally designed for Real-time
- Experiments as early as 1981
Real-time Requirements

• Synchronization of multimedia audio/video
• Allow similar apps to interoperate, e.g., 2 different conferencing apps
  – Need same encoding, compression, etc.
• Provide way for sender to tell receiver what scheme to use for voice, video, etc.
• Determine timing relationship among received data: jitter, timestamping
• Indicate packet loss - no recovery as in TCP
  – Allows sender to adjust to packet loss
• Indicate frame boundary
  – Application specific, e.g., talkspurt in audio
• Identity of senders beyond IP address
• Be efficient - short header
  – allows fast audio packet processing
Multimedia Apps

• Video, Audio, Data
• Interactive or Streaming
• Interactive
  – Vat - audio conferencing
  – IP Telephone
• Streaming
  – Real Audio
  – Video
RTP Details

RTP - Real-time Transport Protocol
multimedia data
- maintain the quality of traffic for real-time apps

RTCP - Real-time Transport Control Protocol
- periodically send control info between end points about data flow behaviors
- Not used to set up or manage connections
- Not used to directly control the way data is exchanged

Use consecutive ports, RTP on even port
- server ports assigned, 5004 and 5005
ALF - Application Level Framing

New Applications, New Requirements, & Old Protocols
Application ‘best’ understands its needs
  e.g., how to recover frames
  e.g., how to segment data for transmission, etc.
RTP Stack

- Rides on UDP
- UDP provides minimal functionality
  - Ports & port demultiplexing
- Transport on a Transport
  - Each provides support for other
  - Think of transport ‘extension headers’
Application on RTP

- Changing applications
- Application Classes
  - Profiles: provides info on understanding of RTP fields
  - Formats: how data following RTP header is to be interpreted, e.g., sequence of bytes or MPEG video
RTP Header

<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>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Extension header</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RTP payload</td>
</tr>
</tbody>
</table>
RTP Header

- First 12 bytes ALWAYS there
  - General app fields
- Optional header extensions
  - Specific app fields
- Version - 2 bits
- Padding - 1 bit
  - Padding count last byte of message
  - Message size known through UDP
  - No length in header
- Extensions - 1 bit
  - Yes or no
- Contributing sources - 4 bits
  - Count of the number of contributing sources
- Frame Mark - 1 bit
  - Yes/no, app specific
  - Provide synchronization events such as end of frame markers.
- Payload Field - 2 bits
  - App specific
  - Controls how the RTP header is used and therefore RTCP messages
RTP Header

- Sequence # - 16 bits
  - Detect missing or misordered packets
  - Left to app to decide what to do
- Timestamp - 32 bits
  - Timestamp of 1st byte of payload data
  - Ticks, monotonically increasing
  - Difference between timestamps important, granularity for jitter issues
  - Left to app to use
- SSRC - Synchronization source - 32 bits
  - Identifies source of an RTP stream
  - E.g., Allows multiple camera streams, from same IP Address
- CSRC - Contributing Source Identifiers
  - Contributing source Ids
  - RTP stream mixer
  - Can take many streams & send as a single stream
UDP/RTP Messages

![UDP/RTP Messages Diagram](image)
RTCP

• Control stream associated with data stream for a multimedia app.
• Feedback on performance of app and network
  – Rate-adaptive apps
  – Diagnose network
• Synchronize different media streams from a sender
  – Different streams may have different clocks
• Convey identity of a sender for display
RTCP Messages

• Sender Reports
  – Enable active senders to report transmission and reception stats

• Receiver Reports
  – Receivers that are not senders use to report reception stats

• Source Descriptions
  – Carry CNAMEs and other sender description info
  – Mikeatcs.hmc.edu would be included with the SSRC

• Application Specific control packets

• Transmission
  – Several RTCP packets can be packed into UDP packet
  – Actually, at least 2 are always sent:
    • Report packet
    • Source description packet

• Traffic control
  – Mechanisms for participants to scale back reporting frequency
  – Lots of rules, but basically, keep RTCP at about 5% of RTP traffic
RTCP Report Messages

• Sender Reports
  – Timestamp of time of day
  – RTP timestamp of report
  – Cumulative counts of packets and bytes sent since beginning
• Sender and Receiver Reports
  – Its SSRC
  – Fraction of data packets from this source lost since last report
  – Total number of packets lost since 1st transmission
  – Highest sequence number received from this source
  – Estimated interarrival jitter for the source
  – Last actual timestamp received via RTCP for this source
  – Delay since last sender report received via RTCP for this source
• Lots of catch all info, probably added for certain apps
RTP & RTCP Summary

Trying to add transport functionality that is directed at a set of applications
At the same time trying to include all the specific info that any app within the set of applications may desire
BEEP is another such protocol
SCTP - Stream Control Transmission Protocol is another