Networking and protocols for real-time signal transmissions

by Hans-Peter Schwefel

- Mm1 Introduction & simple performance models (HPS)
- Mm2 Real-time Support in Wireless Technologies (HPS)
- Mm3 Transport Layer Aspects and Header Compression (HPS)
- Mm4 IP Quality of Service: Advanced Concepts (HPS)
- Mm5 Session Signalling and Application Layer/Codecs (SVA)

Note: Slide-set contains more material than covered in the lectures!

hps@kom.auc.dk
http://www.kom.auc.dk/~hps

Background: IP Protocol Stack

Network Layer (Layer 3):
- Internet Protocol (IPv4, IPv6)
- Packet (IP datagram) transmission between end-systems [hosts] (packet size up to 65535 bytes, often restricted by Layer 2 protocols)
- Routing using 32 bit addresses (v4)
- Unreliable, connectionless transmission: loss, duplication, reordering can occur

Transport Layer (Layer 4):
- Most frequent transport protocols
  Transmission Control Protocol TCP
  User Datagram Protocol UDP

Application Layer/Services (Layer 5-7)
- TCP based:
  HTTP (HyperText Transfer Protocol), FTP (File Transmission Protocol), SMTP (Simple Mail Transfer Protocol), Telnet, ...
- UDP based:
  DNS (Domain Name Service), Streaming media, VoIP, ...

![IP Protocol Stack Diagram]
Transport Layer Protocols

Goal: data transfer between application (processes) in end-systems

• support of multiplexing/de-multiplexing
  e.g. socket API

data stream/connection identified by:
  two IP addresses, protocol number, two port numbers

Overview: Transport Protocols

• User Datagram Protocol UDP (RFC 768)
  – Connectionless
  – Unreliable
  – No flow/congestion control

• Transmission Control Protocol TCP (RFC 793, 1122, 1323, 2018, 2581)
  – Connection-oriented (full duplex)
  – Reliable, in-order byte-stream delivery
  – Flow/congestion control

• Stream Control Transport Protocol SCTP
  – Connection oriented (full-duplex associations)
  – Reliable message delivery, support of multiple streams
  – Support of multi-homing, flow/congestion control

• Real-Time Transport Protocol RTP
  – Uses UDP
  – Provides: Time-stamps, sequence numbers
  – Supports: codecs, codec translation, mixing of multi-media streams
Transmission Control Protocol (TCP): Basics

- Point-to-point, bi-directional connections (between end-systems)
- Reliable, in-order transport of byte-stream using
  - Sequence Numbers
  - Acknowledgements
- Flow/Congestion Control:
  Prevent flooding of
  - Receiver
  - Intermediate Systems

Selected Header Fields
- Sequence number: number of first data byte transmitted in the segment
- ACK number: number of next byte expected in the reverse data flow
- Window size: number of bytes host is willing to accept in reverse data flow

<table>
<thead>
<tr>
<th>Source Port</th>
<th>Destination Port</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sequence Number</th>
<th>Acknowledgement Number</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HLEN</th>
<th>reserved</th>
<th>Window*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Checksum</th>
<th>Urgent Pointer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Options</th>
<th>Padding</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>User data ...</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

Source: RFC 793 (September 1981)

* Field for non-transmission error messages
Slow-Start & Congestion Avoidance

Flow/Congestion Control:
- #packets in network limited by receiver advertised window and congestion window (CWND)
- CWND sizes continuously increased until congestion indicated
- Two phases
  - slow-start: small initial cwnd, rapid increase
  - congestion avoidance: gradual increase
- Reaction to
  - Time-outs
  - Duplicate acknowledgments

Throughput = CWND*MSS / RTT
(for high-bandwidth links)

Illustration of flow control mechanism

TCP Reno trace

- S: Slow Start
- L: Packet Loss
- CA: Congestion Avoidance
- FR: Fast Recovery
- TO: Timeout

Time in units

Congestion Window Size
**TCP and real-time applications**

Discussion: TCP properties

- Reliable transmissions may cause additional delays (time-out + additional retransmission)
  - Packet losses often less harmful for RT applications than retransmission delays
- Congestion control mechanisms control sending rate
  - Not acceptable for RT applications
- No support of multi-casting

Problems of TCP over wireless

- Large RTTs cause long connection set-up times and slow transmission rates in slow-start
- Packet losses due to wireless link properties trigger TCP congestion control mechanisms
- RTT variations (e.g. Due to Layer 2 retransmissions) could cause TCP time-outs

---

**Content**

1. Introduction
   - Transport Protocols
2. TCP and real-time transmissions
   - Overview TCP functionality
   - Discussion of real-time aspects
3. UDP & UDP-lite
   - Functionality, protocol header
   - Cross-layer aspects in UDP-lite
4. RTP & RTCP
   - Functionalities, entities, protocol headers
5. Introduction into Robust Header Compression
   - Header structure, compression, robustness
6. Summary/Conclusions/Outlook

Exercise
User Datagram Protocol UDP

- User Datagram Protocol UDP (RFC 768)
  - Connectionless
  - Unreliable
  - No flow/congestion control
- Functionalities
  - (De-)Multiplexing: Port Numbers
  - Simple additive checksum covering
    - UDP Header
    - UDP Payload
    - Pseudo IP Header
      - IP addresses & protocol type
      - Length of IP payload

When UDP checksums are indicating transmission errors → packet must be discarded at receiver

UDP-Lite (RFC3828)

Delivery of partially damaged packets may be beneficial for some applications → Basic idea: restrict checksum to certain important parts of packet

- Implemented in UDP-lite:
  - Uses own protocol type identifier (136)
  - Length-field replaced by checksum coverage:
    - number of bytes (starting from first UDP header byte) covered by checksum
      - 0 indicates full packet covered
      - Values 1-7 are not allowed (if checksum used, header must be covered)
      - Pseudo-IP header included (as for UDP)
    - Extended application interface
      - Sender: specify checksum coverage
      - Receiver: enable reception of partially corrupted packets
  - Checksum coverage may be used for L2 unequal error protection/detection
  - Note: partially damaged packets usually not useful in case of encryption/integrity protection (e.g. IPsec) or if Link-Layer discards packets!
Content

1. Introduction
   • Transport Protocols
2. TCP and real-time transmissions
   • Overview TCP functionality
   • Discussion of real-time aspects
3. UDP & UDP-lite
   • Functionality, protocol header
   • Cross-layer aspects in UDP-lite
4. RTP & RTCP
   • Functionalities, entities, protocol headers
5. Introduction into Robust Header Compression
   • Header structure, compression, robustness
6. Summary/Conclusions/Outlook

Exercise

Real-Time Transport Protocol, RTP (RFC3550)

- Supports multi-party multimedia conferences
- Uses UDP and multicasting capabilities of IP
- Provides Time-stamps, sequence numbers
- Supports:
  - Codec description (in separate profile description, e.g. RFC3551)
  - synchronization
  - mixing & codec translation of multi-media streams
- Header

```
<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Ver</td>
<td>P</td>
<td>X</td>
<td>CC</td>
<td>M</td>
<td>PTYPE</td>
<td>Sequence Number</td>
</tr>
</tbody>
</table>

- Timestamp
- Synchronization Source Identifier
- Contributing Source ID
  ...
```

VERsion=2
P=Padding
X=Extension Header
CC=Source Count
M=Marker
PayloadTYPE

Application \(\uparrow\) L5-7
RTP \(\uparrow\) L4
UDP \(\uparrow\) L3
IP L2
RTP details: Header Fields 1

- Version (2 bits)
  - currently 2
- P bit (1 bit)
  - 1 if zero padding follows the payload
- X bit (1 bit)
  - 1 if optional header extension is present
  - depends on application type
- CC: source count (4 bits)
  - maximum of 15 contributing sources is supported
- M: Marker bit (1 bit)
  - application dependent
  - mark events in data stream
  - e.g. frame start in video streams
- PTYPE: Payload Type (7 bits)
  - determines detailed interpretation of header and contents

RTP details: Header Fields 2

- Sequence Number (16 bits)
  - identifies RTP packets
  - initial sequence number chosen randomly for each session
- Timestamp (32 bits)
  - time at which the first octet of digitized data was sampled (relative)
  - random choice of initial time stamp
  - continuously incremented
  - clock granularity depends on application
- Synchronization Source Identifier (32 bits)
  - identifies the source of a stream
  - real source, mixer or translator
  - identification collisions resolved by protocol
- Contributing Source ID (variable size, CC x 32 bits)
  - list of source identifiers that contributed the samples mixed together by a mixer
**RTP: Functional Entities**

- **Mixer**
  - receives one or more RTP streams (contributing sources)
  - possibly changes data format (transcoding)
  - combines packets and forwards
  - Timing adjustments → mixer is new synchronizing source

- **Translator**
  - Forward RTP packets without changing synchronization
  - E.g. Encoding converters (without mixing), replicators from multicast to unicast

- **Monitor**
  - Receives RTCP packets, in particular receiver reports
  - Estimates QoS, fault-diagnosis, long-term statistics
  - Integrated in applications in session or separate application (3rd party monitor)

---

**RTP: Timestamps**

- **Wallclock time**
  - Absolute date and time
  - Using Network Time Protocol (NTP) format
    - 64 bit unsigned fixed-point number (32 bit integer part)
    - Representing seconds since 0h, Jan. 1, 1900
  - RTP uses only timestamp differences (no ambiguity as long as within 68 years)

- **RTP timestamps**
  - Measured in clock-ticks of certain resolution
  - Clock frequency depends on payload format (e.g. defined by sampling period)
  - Initial value should be random
  - Linking of RTP timestamps to reference (wallclock) time in RTCP Sender Reports
  - Synchronisation of sources (e.g. using NTP) recommended but not mandatory
Real-time transmission control protocol: RTCP

- Control Protocol for RTP flows
- Functionalities
  - Monitor QoS
  - Convey participant information
  - Rate adaptation for Receiver Reports (scalability, very few up to thousands of participants)
- Message Types
  - 200 Sender Report
  - 201 Receiver Report
  - 202 Source Description (including canonical name, CNAME)
  - 203 BYE
  - 204 Application specific

<table>
<thead>
<tr>
<th>P</th>
<th>RC</th>
<th>PT SR</th>
<th>N</th>
<th>sequence number</th>
<th>Header</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>32</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

RTCP Sender Report, see Sect. 6.4-6.7 of RFC 3550 for other packet types

---

Content

1. Introduction
   - Transport Protocols
2. TCP and real-time transmissions
   - Overview TCP functionality
   - Discussion of real-time aspects
3. UDP & UDP-lite
   - Functionality, protocol header
   - Cross-layer aspects in UDP-lite
4. RTP & RTCP
   - Functionalities, entities, protocol headers
5. Introduction into Robust Header Compression
   - Header structure, compression, robustness
6. Summary/Conclusions/Outlook

Exercise
**Protocol Enhancements: Motivation**

- Wireless links tend to show poor performance
  - Large delays
  - Low throughput
  - Bit errors / packet losses due to radio transmission
- Protocols in IP family not originally designed for such links
  - Increased volume due to headers
  - Deficiencies of TCP flow control
  - ... many more (e.g. applications HTTP \( \rightarrow \) WAP)
- Protocol Enhancements are required, examples
  - **Robust Header Compression (RoHC)**
  - Enhancements for Wireless TCP

**Header Compression (HC)**

- Motivation
  - IP voice packets: header 40/60Bytes, average payload 25Bytes
  - TCP ACK packets: header 40/60 Bytes, payload often 0Bytes
- Data in many header fields …
  - … hardly ever changes e.g. source/destination address within same IP flow
  - … or changes in a regular pattern
- Idea: reduce header length by compression, e.g.
  - differential encoding of fields
  - and/or variations of Huffman compression
- Compression can be applied to several protocol headers, e.g. RTP/UDP/II
**Header Structure: RTP/UDP/IP**

- **Application**
  - RTP
    - UDP
      - IP
        - ROHC
          - Link Level

**Basic approach: HC and RoHC**

Identify and reduce redundancy in header fields:

- **Intra-packet redundancy**: between header fields of the same packet (e.g. RTP/UDP/IP)
- **Inter-packet redundancy**: between header parts between different packets within a certain flow

Compression context (Base) required, when eliminating inter-packet redundancy:

- Context transfer or re-establishment required in handover scenarios
- Packet loss/reordering can lead to de-synchronisation of compressor and decompressor → additional packet loss can result
- Additional methods for robustness introduced → RoHC
Classification of Header-Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Fixed</td>
</tr>
<tr>
<td>IP Header Length</td>
<td>Fixed</td>
</tr>
<tr>
<td>Type of service</td>
<td>Inferred from</td>
</tr>
<tr>
<td>Total length</td>
<td>Link layer</td>
</tr>
<tr>
<td>Identification</td>
<td>Fixed</td>
</tr>
<tr>
<td>Flags</td>
<td>Fixed</td>
</tr>
<tr>
<td>Fragment offset</td>
<td>Fixed</td>
</tr>
<tr>
<td>Time to Live</td>
<td>Fixed</td>
</tr>
<tr>
<td>Protocol</td>
<td>Fixed</td>
</tr>
<tr>
<td>Header checksum</td>
<td>Random</td>
</tr>
<tr>
<td>Source Address</td>
<td>Fixed</td>
</tr>
<tr>
<td>Destination Address</td>
<td>Fixed</td>
</tr>
</tbody>
</table>

IP Header Variable

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Port</td>
<td>Fixed</td>
</tr>
<tr>
<td>Destination Port</td>
<td>Fixed</td>
</tr>
<tr>
<td>Length</td>
<td>Inferred from</td>
</tr>
<tr>
<td>Check sum</td>
<td>Link layer</td>
</tr>
</tbody>
</table>

UDP Header Variable

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Fixed</td>
</tr>
<tr>
<td>Padding (P)</td>
<td>*</td>
</tr>
<tr>
<td>Extension (E)</td>
<td>*</td>
</tr>
<tr>
<td>CSRC Count (CC)</td>
<td>*</td>
</tr>
<tr>
<td>Marker</td>
<td>Random</td>
</tr>
<tr>
<td>Payload Type</td>
<td>Fixed</td>
</tr>
<tr>
<td>Sequence Number</td>
<td>Delta</td>
</tr>
<tr>
<td>Time stamp</td>
<td>Delta</td>
</tr>
<tr>
<td>SSRC</td>
<td>Fixed</td>
</tr>
<tr>
<td>CSRC</td>
<td>*</td>
</tr>
</tbody>
</table>

RTP Header Variable

Different Header Compression schemes

- Compressed TCP – Van Jacobsen RFC 1144
  - only for TCP/IP
  - for wired networks
- Perkins
  - improvement of CTCP
- IPHC [RFC 2507]
  - only for IP
  - no feedback
- RFC2508
**General Structure of Header Compressors**

- Two entities: compressor and decompressor
- Compressor sends initial base
- Base is used by compressor and decompressor
- Compressor removes redundancy
- Decompressor adds removed information
- Proposed solution differ in a possible feedback channel

![Diagram of General Structure of Header Compressors]

---

**Robustness: Perkins’ scheme**

- Robustness introduced by periodically repetition of full base information each N packets
- N packets define a frame
- All packets refer to the first packet of the frame
- Less compression due to higher delta values

![Diagram of Perkins’ scheme]


Robust Header Compression (RFC3095)

- RTP/UDP/IP
- UDP/IP
- IP
- Uncompressed
- Current RoHC methods: 40 Bytes RTP/UDP/IP header \(\rightarrow\) on average 1 or 2 bytes

ROHC Modes

- Unidirectional (U)mode
- Optimistic (O)mode
- Reliable (R)mode
States of Compressor and Decompressor

IR STATE → FO STATE → SO STATE

NO CONTEXT → STATIC CONTEXT → FULL CONTEXT

Unidirectional

IR STATE → FO STATE → SO STATE

optimistic optimistic optimistic

timeout timeout/update timeout
**Decompressor**

SUCCESSFUL PACKET DECOMPRESSION

NO STATIC  NO DYNAMIC  SUCCESS  SUCCESS

NO CONTEXT  STATIC CONTEXT  FULL CONTEXT

REPEATED FAILURES  REPEATED FAILURES

---

**RoHC: Ongoing Work**

- Application of RoHC methods for SIP compression
  - Compression of whole SIP messages
  - Goal: Reduction of call-setup delay
    (SIP message up to several thousand Byte)
- Optimized use of Compression
  Trade-off: Data-Volume vs. Error Robustness on
  - Application Layer
  - RoHC Layer
  - Link-Layer
  ➔ Optimization across whole protocol stack required
Summary

1. Introduction
   - Transport Protocols

2. TCP and real-time transmissions
   - Overview TCP functionality
   - Discussion of real-time aspects

3. UDP & UDP-lite
   - Functionality, protocol header
   - Cross-layer aspects in UDP-lite

4. RTP & RTCP
   - Functionalities, entities, protocol headers

5. Introduction into Robust Header Compression
   - Header structure, compression, robustness

6. Summary/Conclusions/Outlook

Exercise

A VoIP stream shows an average (RTP) payload of 40 Bytes which is transported using RTP/UDP/IPv4 from one slave to another slave within a BT piconet using DH1 packets (1 slot, payload max 27 Bytes). The average inter-packet time is 20 ms.

1. Compute the average utilization (fraction of used timeslots) due to this VoIP stream, without header compression and when header compression is applied. Assume that header compression reduces the average header-length to 4 bytes (and use averages for your computation).

2. The BT link shows a bit-error rate (after utilizing FEC) of 1e-5. Assume independent errors. A simple RoHC mechanism is used that transmits first K full headers, and then (N-K) fully compressed headers (of assumed length 4 bytes). This sequence of N packets, called ‘cycle’, repeats afterwards. The decompressor is able to correctly reconstruct the full header from the compressed header of packet i, i> K, in the cycle if:
   - At least one of the K full headers at the beginning of the cycle was received correctly, AND
   - All fully compressed headers before packet i in the cycle were received correctly.

   For the following calculations, details of the access technology (BT) should not be considered.

   a) Compute and plot the average header length of the compression mechanism, AHL(N,K).
   b) Compute the probabilities, p_full and p_comp, that a full respectively compressed header is corrupted during transmission.
   c) Compute the probability p(i), that one of the K full headers in the beginning of the cycle was received correctly AND the i-th packet (i> K) is the first corrupted compressed header in the cycle.
   d) Compute the ‘additional packet loss’, APL, caused by correctly received compressed headers while incorrect base.
**Exercises (optional)**

**UDP-lite:**

As before: A VoIP stream shows an average (RTP) payload of 40Bytes which is transported using RTP/UDP/IPv4 from one slave to another slave within a BT piconet using DH1 packets (1 slot, payload max 27 Bytes). The average inter-packet time is 20ms.

1. Consider the case without HC: The BT link shows a raw bit error rate of 1e-4. Only consider BT payload (which has no FEC for DH1 packets) and assume that even erroneous packets are forwarded to higher layers. Compute the fraction of dropped data bits (due to wrong UDP checksums causing UDP packet drops). How is this fraction reduced when UDP-lite is applied with only protecting the minimal information.

**References: HC, RoHC**

4. C.Bormann, et.al., *Robust Header Compression (ROHC)*, RFC3095, July 2001