Transport Services for Low-Latency Real-Time Applications

Stephen McQuistin and Colin Perkins
University of Glasgow

Marwan Fayed
University of Stirling

Motivation and Goals

- To understand what transport services are desirable for low-latency real-time applications
  - Streaming video
  - Interactive video conferencing
  - Augmented reality
  - Gaming
  - …
- To consider an appropriate abstract API for such services
- Derive from application requirements – not from existing transport protocols and APIs
  - Basis for TCP Hollywood (https://csperkins.org/research/tcp-hollywood/)
  - Concepts more general than that research project → relevant to TAPS?

What transport services do real-time applications need?

Timing

Partial Reliability

Dependencies

Messages

Multiple streams

Multiple paths

Congestion control

Connections?

- Timing is an essential characteristic – application data has a lifetime, after which it is not useful
  - 10s - 100s milliseconds for interactive applications
  - Maybe *O*(seconds) for non-interactive applications
- Transport protocols should not send data that will arrive too late to be useful

- Transport needs knowledge of
  - Data timing and lifetime/deadline for use
  - Estimated network transit time (or, at least, RTT)
  - Estimated jitter buffer duration at receiver

  to manage scheduling of data for transmission
- API must expose timing information

What transport services do real-time applications need?

- Timing
- **Partial Reliability**
- Dependencies
- Messages
- Multiple streams
- Multiple paths
- Congestion control
- Connections?

- Network is unreliable – “best effort” service

- Lost data recovered by FEC and/or retransmission
- Cannot guarantee delivery before a deadline
  - Might be able to estimate probably of delivery before deadline – but always $p < 1.0$
  - Potentially unbounded delay because retransmissions can be lost

- If deadlines are to be respected, transport has to offer a partial reliability mode
- API must expose that some data can be lost

What transport services do real-time applications need?

- Timing
- Partial Reliability
- **Dependencies**
- Messages
- Multiple streams
- Multiple paths
- Congestion control
- Connections?

- Partial reliability → some data will not be received
- If data items are not independently useful, must track dependencies
- Either:
  - Avoid wastefully sending data that depends on previously lost data
  - Send data that would miss its deadline, since needed to make use of later data
- API needs to allow data and dependencies to be identified

What transport services do real-time applications need?

- Timing
- Partial Reliability
- Dependencies
- **Messages**
- Multiple streams
- Multiple paths
- Congestion control
- Connections?

- Application-level framing – split data into packets on meaningful boundaries
- Named objects that form the basis for dependency tracking, reliability

- API and transport services must respect message boundaries

- Timing, message identity, and dependencies allow out-of-order delivery and processing – avoid HoL blocking

What transport services do real-time applications need?

- Timing
- Partial Reliability
- Dependencies
- Messages
- **Multiple streams**
- Multiple paths
- Congestion control
- Connections?

- Exposing messages boundaries in transport and API enables multi-streaming
  - Different streams of data multiplexed onto a single transport layer flow
  - Requires message boundaries be delineated, messages have identity that indicates what sub-flow they below to
- API and transport must expose sub-stream identity
- Optional – desirable for efficiency and reliability
  - Each additional flow increases risk of interference from firewall, NAT, or other middlebox
  - Sub-streams make multiple flows appear as one

What transport services do real-time applications need?

- Timing
- Partial Reliability
- Dependencies
- Messages
- Multiple streams
- **Multiple paths**
- Congestion control
- Connections?

- Devices increasingly have multiple interfaces and hence multiple paths between them
- Desirable to make use of these to balance load, reduce latency where possible

- Obvious extension, given multi-streaming and messages – build on MPTCP-style congestion control, etc.
    - Expose paths as first-class entity in API
    - Allows application to hint mapping sub-streams onto paths

What transport services do real-time applications need?

- Timing
- Partial Reliability
- Dependencies
- Messages
- Multiple streams
- Multiple paths
- **Congestion control**
- Connections?

- Essential to avoid network overload – algorithms should take into account data timing and lifetime
- API should expose detailed congestion metrics – applications are non-elastic in timing, but flexible with what they send
  - Scope for close partnership between applications and transport – it’s to the application’s benefit to cooperate

What transport services do real-time applications need?

Timing

Partial Reliability

Dependencies

Messages

Multiple streams

Multiple paths

Congestion control

Connections?

- Per-connection metadata useful for congestion control and in maintaining security association
- Connection set-up and teardown messages can help NAT/firewall traversal

- But, duration of many communication sessions can outlive a single connection

- API and transport services should expose long-lived metadata about endpoints, and ephemeral per-connection data

What transport services do real-time applications need?

- Timing
- Partial Reliability
- Dependencies
- Messages
- Multiple streams
- Multiple paths
- Congestion control
- Connections?

- Existing transport protocols do not provide these services – although some are close
- Existing APIs don’t expose the features required

- The draft sketches minimal extensions to Sockets API that expose many of these features – to fully enable this needs a radical API change
→ Post Sockets?

- Are these services identified/exposed in TAPS?
- Should TAPS be considering the API work needed to support these services?