University of Glasgow

School of Computing Science

60 YEARS OF COMPUTING AT GLASGOW

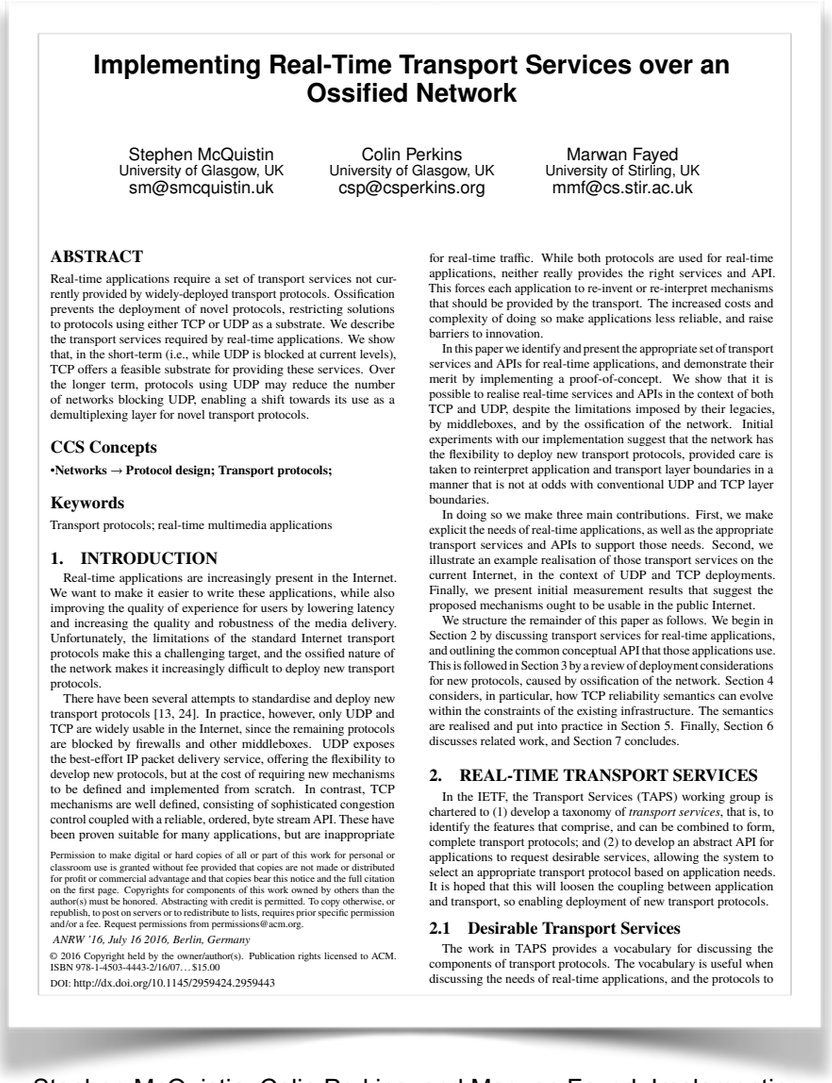
Transport Services for Low-Latency Real-Time Applications

Colin Perkins

Motivation and Goals

- To understand what transport services are desirable for low-latency real-time traffic
- To consider an appropriate abstract API for such services
- Derive from application requirements – not from existing transport protocols and APIs

- Basis for TCP Hollywood, but applicable to Post Sockets



Implementing Real-Time Transport Services over an Ossified Network

Stephen McQuistin
University of Glasgow, UK
sm@smcquistin.uk

Colin Perkins
University of Glasgow, UK
csp@csperkins.org

Marwan Fayed
University of Stirling, UK
mmf@cs.stir.ac.uk

ABSTRACT

Real-time applications require a set of transport services not currently provided by widely-deployed transport protocols. Ossification prevents the deployment of novel protocols, restricting solutions to protocols using either TCP or UDP as a substrate. We describe the transport services required by real-time applications. We show that, in the short-term (i.e., while UDP is blocked at current levels), TCP offers a feasible substrate for providing these services. Over the longer term, protocols using UDP may reduce the number of networks blocking UDP, enabling a shift towards its use as a demultiplexing layer for novel transport protocols.

CCS Concepts

•**Networks** → **Protocol design; Transport protocols;**

Keywords

Transport protocols; real-time multimedia applications

1. INTRODUCTION

Real-time applications are increasingly present in the Internet. We want to make it easier to write these applications, while also improving the quality of experience for users by lowering latency and increasing the quality and robustness of the media delivery. Unfortunately, the limitations of the standard Internet transport protocols make this a challenging target, and the ossified nature of the network makes it increasingly difficult to deploy new transport protocols.

There have been several attempts to standardise and deploy new transport protocols [13, 24]. In practice, however, only UDP and TCP are widely usable in the Internet, since the remaining protocols are blocked by firewalls and other middleboxes. UDP exposes the best-effort IP packet delivery service, offering the flexibility to develop new protocols, but at the cost of requiring new mechanisms to be defined and implemented from scratch. In contrast, TCP mechanisms are well defined, consisting of sophisticated congestion control coupled with a reliable, ordered, byte stream API. These have been proven suitable for many applications, but are inappropriate for real-time traffic. While both protocols are used for real-time applications, neither really provides the right services and API. This forces each application to re-invent or re-interpret mechanisms that should be provided by the transport. The increased costs and complexity of doing so make applications less reliable, and raise barriers to innovation.

In this paper we identify and present the appropriate set of transport services and APIs for real-time applications, and demonstrate their merit by implementing a proof-of-concept. We show that it is possible to realise real-time services and APIs in the context of both TCP and UDP, despite the limitations imposed by their legacies, by middleboxes, and by the ossification of the network. Initial experiments with our implementation suggest that the network has the flexibility to deploy new transport protocols, provided care is taken to reinterpret application and transport layer boundaries in a manner that is not at odds with conventional UDP and TCP layer boundaries.

In doing so we make three main contributions. First, we make explicit the needs of real-time applications, as well as the appropriate transport services and APIs to support those needs. Second, we illustrate an example realisation of those transport services on the current Internet, in the context of UDP and TCP deployments. Finally, we present initial measurement results that suggest the proposed mechanisms ought to be usable in the public Internet.

We structure the remainder of this paper as follows. We begin in Section 2 by discussing transport services for real-time applications, and outlining the common conceptual API that those applications use. This is followed in Section 3 by a review of deployment considerations for new protocols, caused by ossification of the network. Section 4 considers, in particular, how TCP reliability semantics can evolve within the constraints of the existing infrastructure. The semantics are realised and put into practice in Section 5. Finally, Section 6 discusses related work, and Section 7 concludes.

2. REAL-TIME TRANSPORT SERVICES

In the IETF, the Transport Services (TAPS) working group is chartered to (1) develop a taxonomy of *transport services*, that is, to identify the features that comprise, and can be combined to form, complete transport protocols; and (2) to develop an abstract API for applications to request desirable services, allowing the system to select an appropriate transport protocol based on application needs. It is hoped that this will loosen the coupling between application and transport, so enabling deployment of new transport protocols.

2.1 Desirable Transport Services

The work in TAPS provides a vocabulary for discussing the components of transport protocols. The vocabulary is useful when discussing the needs of real-time applications, and the protocols to

ANRW '16, July 16 2016, Berlin, Germany

ISBN 978-1-4503-4443-2/16/07…$15.00

DOI: http://dx.doi.org/10.1145/2959424.2959443

Stephen McQuistin, Colin Perkins, and Marwan Fayed, Implementing Real-Time Transport Services over an Ossified Network, Proceedings of the ACM, IRTF, and ISOC Applied Networking Research Workshop, Berlin, Germany, July 2016. DOI:10.1145/2959424.2959443

Desirable Services for Real-time Applications

Timing	Expose playout deadlines and timing constraints to transport
Partial Reliability	Timeliness more critical than reliability; tune reliability for application
Dependencies	Some data not independently decodable → application knowledge; links to partial reliability
Messages	Application-level framing; make each packet usable
Multiple streams	Multiple transport flows → NAT traversal woes; demultiplex within flow
Multiple paths	Maybe → more critical to discover ICE candidates for NAT traversal
Congestion control	Close coupling with application desirable → elastic in what is sent, with constraints, but *must* send with predictable timing
Connections?	Useful for NAT keep-alive; application sessions outlive transport layer flows, coupled to higher layer signalling

Existing transport APIs don't satisfy these requirements