Available online at www.sciencedirect.com
ScienceDirect
Computer Communications 30 (2007) 630–637
computer communications
www.elsevier.com/locate/comcom

Seamless transport service selection by deploying a middleware ☆

Sven Hessler *, Michael Welzl

University of Innsbruck, Institute of Computer Science, Technikerstr. 21a, 6020 Innsbruck, Austria

Available online 10 January 2006

Abstract

Despite the many research efforts at the transport layer (SCTP, DCCP, etc.), new innovations in that area hardly ever make it into the TCP/IP stacks of standard end systems. We believe that this is due to lack of a flexible interface to the application as well as a lack of transparency – a problem that could be solved by introducing a middleware above TCP/IP. In this paper, we present the architecture of our middleware, and show the importance of congestion awareness by simulations. In addition, we explain how to force congestion control on existing applications using our middleware, show the benefits of doing so with simulations, and finally discuss the impact of our middleware towards a better utilization of the network and a more suitable service for the user software.

Keywords: Congestion control; Middleware; DCCP; Protocols

1. Introduction

While many new Internet applications with specific requirements to the network emerge, ranging from peer-to-peer file-sharing tools to highly complex Grid computing software, the typical Internet end system provides only two different transport services:

(1) Unreliable datagram delivery (based on the User Datagram Protocol, UDP)
(2) Reliable in-order delivery of a consecutive data stream (based on the Transmission Control Protocol, TCP)

This has not changed much since the 1970s; given the many new developments at the application layer, it seems hard to believe that these two services – TCP, which realizes reliable in-order delivery of a consecutive byte stream, and UDP, which merely brings the IP "best effort" service to the transport layer – will suffice forever. Yet, although an immense amount of research on transport protocol enhancements for potential successors of TCP and UDP is carried out, the results of this research hardly seem to make it into the operating systems of our personal computer. We believe that one of the reason for this is the socket interface, which lacks the flexibility and transparency that would be needed to gradually enhance the transport layer without having to change the applications that use it.

We propose to add a middleware on top of TCP/IP, thereby providing a unique service oriented interface and realizing a certain degree of transparency. We believe that our middleware could enable the deployment of new transport services, no matter if they are realized as TCP alternatives or even as actual end-to-end QoS with strict guarantees.

The rest of this paper is organized as follows: Section 2 gives a survey of the state of the art of research on Internet transport protocols and their prospective successors; Section 3 illustrates how the network and the applications benefit from congestion awareness; in Section 4 we first give a detailed description of the design and the functions of our middleware and present simulations showing how existing congestion-ignorant applications could be forced to use TCP-friendly transport protocols. Section 5 summarizes our findings, discusses potential implications of our middleware, and presents further steps of research.

☆ This research is supported by the Austrian Science Fund (FWF).
* Corresponding author.
E-mail addresses: sven.hessler@uibk.ac.at (S. Hessler), michael.welzl@uibk.ac.at (M. Welzl).

0140-3664/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2005.11.015

2. State of the Art

According to an investigation in [1] TCP is still the prevalent transport protocol with a share of approximately 83%. Nevertheless the traffic composition changes all the time, and as the Internet grows, new applications arise. Following the traditional ones (file download and remote login) and the rise of the World Wide Web, an important application class that has emerged is real-time streaming media, encompassing live radio or TV transmissions, Internet telephones and video conferencing tools. Such applications differ from the typical earlier applications in that they have different requirements for the transport layer, e.g., they do not necessarily require reliability, but timely transmission of data.

2.1. Transport layer developments

While UDP merely adds identification of different communicating entities at a single site (port numbers) and data integrity (checksum) to the "best effort" service provided by the IP protocol, TCP encompasses a variety of functions:

Reliability. Connections are set up and torn down; missing segments are retransmitted. If the connections breaks, the application is informed.

Flow control. The receiver is protected against overload.

In-order delivery. Segments are ordered according to sequence numbers by the receiver.

Congestion control. The network is protected against overload by combining a variety of mechanisms, e.g.

- Ack-clocking helps to ensure network stability – roughly, in equilibrium, no packet enters the network unless a packet leaves the network.
- Rate control is used to reduce the sender's rate in response to congestion, following a rule called "Additive Increase, Multiplicative Decrease" (AIMD) for reasons of network stability and fairness.
- Round-trip-time estimation at the sender is carried out by monitoring received acknowledgments. This function is important to fine-tune the other mechanisms.

Congestion control consists of several intertwined algorithms; together, they make TCP a somewhat complex system with many setscrews to fine-tune it. This does not pose a problem in practice because its implementation is entirely left up to operating system designers and parameters are hardly changed.

As mentioned above, the most complex part of TCP is without a doubt its congestion control functionality. When it was added to TCP to protect the network from overload and ensure its stability [2], this appeared to be a sensible choice because TCP was the prevalent protocol, and congestion control in TCP is combined with its window-based flow control, which was already available in the protocol. In the case of UDP, implementing congestion control is entirely left up to application designers – and it is clearly necessary to have at least some kind of congestion control in order to maintain the stability of the Internet [3].

While applications using UDP should contain a congestion control mechanism that ensures fairness towards TCP ("TCP-friendliness"), the window-based stop-and-go behavior and heavy rate fluctuations of TCP are a poor match for real-time streaming media applications. Thus, a large amount of research on more suitable ("smoother") yet TCP-friendly congestion control has been carried out; some examples are TFRC [4], RAP [5], LDA+ [6] and Binomial Congestion Control [7]. An overview is given in [8].

2.1.1. The Datagram Congestion Control Protocol

The fact that the burden of implementing a congestion control mechanism is placed upon the application designer has only recently been acknowledged: the "Datagram Congestion Control Protocol" (DCCP), which is currently undergoing IETF standardization, is a means to support applications that do not require the reliable transport of TCP but should nevertheless perform (TCP-friendly) congestion control [9]. DCCP can be regarded as a framework for such mechanisms which provides a wealth of additional features, encompassing partial checksums – this makes it possible for an application to utilize corrupt data instead of not having it delivered at all (corrupt data are suitable for certain video and audio codecs). Combined with the "Data Checksum Option", partial checksums can also be used to circumvent the common problem of corruption being interpreted as a sign of congestion.

2.1.2. The Stream Control Transmission Protocol

The "Stream Control Transmission Protocol" (SCTP) is another new transport protocol which, like DCCP, is not yet commonly available in the Internet. Unlike DCCP, it is already standardized and ready for deployment [10]. SCTP was designed to transfer telephone signaling messages over IP networks, but is capable of broader applications. Among other things, it extends the TCP service model by separating reliability from in-order delivery. SCTP provides the following features [11]:

Multi-stream support – logically independent user messages can be delivered independently, which can alleviate the so-called "head-of-line-blocking-delay" caused by the strict sequence delivery of TCP. Nowadays, applications requiring this type of functionality typically utilize multiple TCP connections; this is similar to the SCTP feature from an application programmer's point of view, but different for the network as there is only one congestion control instance for multiple streams in the SCTP case.

Multi-homing support – with SCTP, a connection is not associated with a single IP address and a port number on each side but with a set of IP addresses and port numbers