Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: October 26, 2011

M. Welzl
University of Oslo
D. Ros
Institut Telecom / Telecom Bretagne

April 24, 2011

A Survey of Lower-than-Best-Effort Transport Protocols
draft-ietf-ledbat-survey-07.txt

Abstract

This document provides a survey of transport protocols which are designed to have a smaller bandwidth and/or delay impact on standard TCP than standard TCP itself when they share a bottleneck with it. Such protocols could be used for delay-insensitive "background" traffic, as they provide what is sometimes called a "less than" (or "lower than") best-effort service.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on October 26, 2011.

Table of Contents

1. Introduction

This document presents a brief survey of proposals to attain a Less than Best Effort (LBE) service by means of end-host mechanisms. We loosely define a LBE service as a service which results in smaller bandwidth and/or delay impact on standard TCP than standard TCP itself, when sharing a bottleneck with it. We refer to systems that are designed to provide this service as LBE systems. With the exception of TCP Vegas, which we present for historical reasons, we exclude systems that have been noted to exhibit LBE behavior under some circumstances but were not designed for this purpose (e.g. RAPID [Kon09], [Aru10]).

Generally, LBE behavior can be achieved by reacting to queue growth earlier than standard TCP would, or by changing the congestion avoidance behavior of TCP without utilizing any additional implicit feedback. It is therefore assumed that readers are familiar with TCP congestion control [RFC5681]. Some mechanisms achieve an LBE behavior without modifying transport protocol standards (e.g., by changing the receiver window of standard TCP), whereas others leverage network-level mechanisms at the transport layer for LBE purposes. According to this classification, solutions have been categorized in this document as delay-based transport protocols, non-delay-based transport protocols, upper-layer approaches and network-assisted approaches. Some of the schemes in the first two categories could be implemented using TCP without changing its header format; this would facilitate their deployment in the Internet. The schemes in the third category are, by design, supposed to be especially easy to deploy, because they only describe a way in which existing transport protocols are used. Finally, mechanisms in the last category require changes to equipment along the path, which can greatly complicate their deployment.

This document is a product of the Low Extra Delay Background Transport (LEDBAT) Working Group. It aims at putting the congestion control algorithm that the working group has specified [Sha11] in the

context of the state of the art in LBE transport. This survey is not exhaustive, as this would not be possible or useful; the authors/ editors have selected key, well-known, or otherwise interesting techniques for inclusion at their discretion. There is also a substantial amount of work that is related to the LBE concept but not presenting a solution that can be installed in end hosts or expected to work over the Internet (e.g., there is a DiffServ-based, Lower-Effort service [RFC3662], and the IETF Congestion Exposure (CONEX) Working Group is developing a mechanism which can incentivize LEDBAT-like applications). Such work is outside the scope of this document.

2. Delay-based transport protocols

It is wrong to generally equate "little impact on standard TCP" with "small sending rate". Without Explicit Congestion Notification (ECN) support, standard TCP will normally increase its congestion window (and effective sending rate) until a queue overflows, causing one or more packets to be dropped and the effective rate to be reduced. A protocol which stops increasing the rate before this event happens can, in principle, achieve a better performance than standard TCP.

TCP Vegas [Bra94] is one of the first protocols that was known to have a smaller sending rate than standard TCP when both protocols share a bottleneck [Kur00] -- yet it was designed to achieve more, not less throughput than standard TCP. Indeed, when TCP Vegas is the only congestion control algorithm used by flows going through the bottleneck, its throughput is greater than the throughput of standard TCP. Depending on the bottleneck queue length, TCP Vegas itself can be starved by standard TCP flows. This can be remedied to some degree by the RED Active Queue Management mechanism [RFC2309]. Vegas linearly increases or decreases the sending rate, based on the difference between the expected throughput and the actual throughput. The estimation is based on RTT measurements.

The congestion avoidance behavior is the protocol's most important feature in terms of historical relevance as well as relevance in the context of this document (it has been shown that other elements of the protocol can sometimes play a greater role for its overall behavior [Hen00]). In congestion avoidance, once per RTT, TCP Vegas calculates the expected throughput as WindowSize / BaseRTT, where

WindowSize is the current congestion window and BaseRTT is the minimum of all measured RTTs. The expected throughput is then compared with the actual throughput measured by recent acknowledgements. If the actual throughput is smaller than the expected throughput minus a threshold called "beta", this is taken as a sign of congestion, causing the protocol to linearly decrease its rate. If the actual throughput is greater than the expected throughput minus a threshold called "alpha" (with alpha < beta), this is taken as a sign that the network is underutilized, causing the protocol to linearly increase its rate.

TCP Vegas has been analyzed extensively. One of the most prominent properties of TCP Vegas is its fairness between multiple flows of the same kind, which does not penalize flows with large propagation delays in the same way as standard TCP. While it was not the first protocol that uses delay as a congestion indication, its predecessors (like CARD [Jai89], Tri-S [Wan91] or DUAL [Wan92]) are not discussed here because of the historical "landmark" role that TCP Vegas has taken in the literature.

Delay-based transport protocols which were designed to be non-intrusive include TCP Nice [Ven02] and TCP Low Priority (TCP-LP) [Kuz06]. TCP Nice [Ven02] follows the same basic approach as TCP Vegas but improves upon it in some aspects. Because of its moderate linear-decrease congestion response, TCP Vegas can affect standard TCP despite its ability to detect congestion early. TCP Nice removes this issue by halving the congestion window (at most once per RTT, like standard TCP) instead of linearly reducing it. To avoid being too conservative, this is only done if a fixed predefined fraction of delay-based incipient congestion signals appears within one RTT. Otherwise, TCP Nice falls back to the congestion avoidance rules of TCP Vegas if no packet was lost or standard TCP if a packet was lost. One more feature of TCP Nice is its ability to support a congestion window of less than one packet, by clocking out single packets over more than one RTT. With ns-2 simulations and real-life experiments using a Linux implementation, the authors of [Ven02] show that TCP Nice achieves its goal of efficiently utilizing spare capacity while being non-intrusive to standard TCP.

Other than TCP Vegas and TCP Nice, TCP-LP [Kuz06] uses only the one-way delay (OWD) instead of the RTT as an indicator of incipient congestion. This is done to avoid reacting to delay fluctuations

that are caused by reverse cross-traffic. Using the TCP Timestamps option [RFC1323], the OWD is determined as the difference between the receiver's Timestamp value in the ACK and the original Timestamp value that the receiver copied into the ACK. While the result of this subtraction can only precisely represent the OWD if clocks are synchronized, its absolute value is of no concern to TCP-LP and hence clock synchronization is unnecessary. Using a constant smoothing parameter, TCP-LP calculates an Exponentially Weighted Moving Average (EWMA) of the measured OWD and checks whether the result exceeds a threshold within the range of the minimum and maximum OWD that was seen during the connections's lifetime; if it does, this condition is interpreted as an "early congestion indication". The minimum and maximum OWD values are initialized during the slow-start phase.

Regarding its reaction to an early congestion indication, TCP-LP tries to strike a middle ground between the overly conservative choice of _immediately_ setting the congestion window to one packet, and the presumably too aggressive choice of simply halving the congestion window like standard TCP; TCP-LP tries to delay the former action by an additional RTT, to see if there is persistent congestion or not. It does so by halving the window at first in response to an early congestion indication, then initializing an "inference time-out timer", and maintaining the current congestion window until this timer fires. If another early congestion indication appeared during this "inference phase", the window is then set to 1; otherwise, the window is maintained and TCP-LP continues to increase it in the

standard Additive-Increase fashion. This method ensures that it takes at least two RTTs for a TCP-LP flow to decrease its window to 1, and, like standard TCP, TCP-LP reacts to congestion at most once per RTT.

Using a simple analytical model, the authors of TCP-LP [Kuz06] illustrate the feasibility of a delay-based LBE transport by showing that, due to the non-linear relationship between throughput and RTT, it is possible to avoid interfering with standard TCP traffic even when the flows under consideration have a larger RTT than standard TCP flows. With ns-2 simulations and real-life experiments using a Linux implementation, the authors of [Kuz06] show that TCP-LP is largely non-intrusive to TCP traffic while at the same time enabling it to utilize a large portion of the excess network bandwidth, which is fairly shared among competing TCP-LP flows. They also show that

using their protocol for bulk data transfers greatly reduces file transfer times of competing best-effort web traffic.

Sync-TCP [Wei05] follows a similar approach as TCP-LP, by adapting its reaction to congestion according to changes in the OWD. By comparing the estimated (average) forward queuing delay to the maximum observed delay, Sync-TCP adapts the AIMD parameters depending on the trend followed by the average delay over an observation window. Even though the authors of [Wei05] did not explicitly consider its use as an LBE protocol, Sync-TCP was designed to react early to incipient congestion, while grabbing available bandwidth more aggressively than a standard TCP in congestion-avoidance mode.

Delay-based congestion control is also at the basis of proposals aiming at adapting TCP's congestion avoidance to very high-speed networks. Some of these proposals, like Compound TCP [Tan06][Sri08] and TCP Illinois [Liu08], are hybrid loss- and delay-based mechanisms, whereas others (e.g., NewVegas [Dev03], FAST TCP [Wei06] or CODE TCP [Cha10]) are variants of Vegas based primarily on delays.

2.1. Accuracy of delay-based congestion predictors

The accuracy of delay-based congestion predictors has been the subject of a good deal of research, see e.g. [Bia03], [Mar03], [Pra04], [Rew06], [McC08]. The main result of most of these studies is that delays (or, more precisely, round-trip times) are, in general, weakly correlated with congestion. There are several factors that may induce such a poor correlation:

- Bottleneck buffer size: in principle, a delay-based mechanism could be made "more than TCP friendly" _if_ buffers are "large enough", so that RTT fluctuations and/or deviations from the minimum RTT can be detected by the end-host with reasonable

accuracy. Otherwise, it may be hard to distinguish real delay variations from measurement noise.

- RTT measurement issues: in principle, RTT samples may suffer from poor resolution, due to timers which are too coarse-grained with respect to the scale of delay fluctuations. Also, a flow may obtain a very noisy estimate of RTTs due to undersampling, under some circumstances (e.g., the flow rate is much lower than the

link bandwidth). For TCP, other potential sources of measurement noise include: TCP segmentation offloading (TSO) and the use of delayed ACKs [Hay10]. A congested reverse path may also result in an erroneous assessment of the congestion state of the forward path. Finally, in the case of fast or short-distance links, the majority of the measured delay can in fact be due to processing in the involved hosts; typically, this processing delay is not of interest, and it can underlie fluctuations that are not related to the network at all.

- Level of statistical multiplexing and RTT sampling: it may be easy for an individual flow to "miss" loss/queue overflow events, especially if the number of flows sharing a bottleneck buffer is significant. This is nicely illustrated e.g. in Fig. 1 of [McC08].

- Impact of wireless links: several mechanisms that are typical of wireless links, like link-layer scheduling and error recovery, may induce strong delay fluctuations over short time scales [Gur04].

Interestingly, the results of Bhandarkar et al. [Bha07] seem to paint a slightly different picture, regarding the accuracy of delay-based congestion prediction. Bhandarkar et al. claim that it is possible to significantly improve prediction accuracy by adopting some simple techniques (smoothing of RTT samples, increasing the RTT sampling frequency). Nonetheless, they acknowledge that even with such techniques, it is not possible to eradicate detection errors. Their proposed delay-based congestion avoidance method, PERT (Probabilistic Early Response TCP), mitigates the impact of residual detection errors by means of a probabilistic response mechanism to congestion detection events.

2.2. Potential issues with delay-based congestion control for LBE transport

Whether a delay-based protocol behaves in its intended manner (e.g., it is "more than TCP friendly", or it grabs available bandwidth in a very aggressive manner) may depend on the accuracy issues listed in Section 2.1. Moreover, protocols like Vegas need to keep an estimate of the minimum ("base") delay; this makes such protocols highly

sensitive to eventual changes in the end-to-end route during the

lifetime of the flow [Mo99].

Regarding the issue of false positives/false negatives with a delay-based congestion detector, most studies focus on the loss of throughput coming from the erroneous detection of queue build-up and of alleviation of congestion. Arguably, for a LBE transport protocol it's better to err on the "more-than-TCP-friendly side", that is, to always yield to _perceived_ congestion whether it is "real" or not; however, failure to detect congestion (due to one of the above accuracy problems) would result in behavior that is not LBE. For instance, consider the case in which the bottleneck buffer is small, so that the contribution of queueing delay at the bottleneck to the global end-to-end delay is small. In such a case, a flow using a delay-based mechanism might end up consuming a good deal of bandwidth with respect to a competing standard TCP flow, unless it also incorporates a suitable reaction to loss.

A delay-based mechanism may also suffer from the so-called "latecomer advantage" (or latecomer unfairness) problem. Consider the case in which the bottleneck link is already (very) congested. In such a scenario, delay variations may be quite small, hence, it may be very difficult to tell an empty queue from a heavily-loaded queue, in terms of delay fluctuation. Therefore, a newly-arriving delay-based flow may start sending faster when there is already heavy congestion, eventually driving away loss-based flows [Sha05][Car10].

3. Non-delay-based transport protocols

There exist a few transport-layer proposals that achieve an LBE service without relying on delay as an indicator of congestion. In the algorithms discussed below the loss rate of the flow determines, either implicitly or explicitly, the sending rate (which is adapted so as to obtain a lower share of the available bandwidth than standard TCP); such mechanisms likely cause more queuing delay and react to congestion more slowly than delay-based ones.

4CP [Liu07], which stands for "Competitive and Considerate Congestion Control", is a protocol which provides a LBE service by changing the window control rules of standard TCP. A "virtual window" is maintained which, during a so-called "bad congestion phase" is reduced to less than a predefined minimum value of the actual congestion window. The congestion window is only increased again once the virtual window exceeds this minimum, and in this way the virtual window controls the duration during which the sender transmits with a fixed minimum rate. Whether the congestion state is "bad" or "good" depends on whether the loss event rate is above or

below a threshold (or target) value. The 4CP congestion avoidance algorithm allows for setting a target average window and avoids starvation of "background" flows while bounding the impact on "foreground" flows. Its performance was evaluated in ns-2 simulations and in real-life experiments with a kernel-level implementation in Microsoft Windows Vista.

The MulTFRC [Dam09] protocol is an extension of TCP-Friendly Rate Control (TFRC) [RFC5348] for multiple flows. MulTFRC takes the main idea of MulTCP [Cro98] and similar proposals (e.g., [Hac04], [Hac08], [Kuo08]) a step further. A single MulTCP flow tries to emulate (and be as friendly as) a number N > 1 of parallel TCP flows. By supporting values of N between 0 and 1, MulTFRC can be used as a mechanism for a LBE service. Since it does not react to delay like the protocols described in Section 2 but adjusts its rate like TFRC, MulTFRC can probably be expected to be more aggressive than mechanisms such as TCP Nice or TCP-LP. This also means that MulTFRC is less likely to be prone to starvation, as its aggressiveness is tunable at a fine granularity, even when N is between 0 and 1.

4. Upper-layer approaches

The proposals described in this section do not require modifying transport protocol standards. Most of them can be regarded as running "on top" of an existing transport, even though they may be implemented either at the application layer (i.e., in user-level processes), or in the kernel of the end hosts' operating system. Such "upper-layer" mechanisms may arguably be easier to deploy than transport-layer approaches, since they do not require any changes to the transport itself.

A simplistic, application-level approach to a background transport service may consist in scheduling automated transfers at times when the network is lightly loaded, as described in e.g. [Dyk02] for cooperative proxy caching. An issue with such a technique is that it may not necessarily be applicable to applications like peer-to-peer file transfer, since the notion of an "off-peak hour" is not meaningful when end-hosts may be located anywhere in the world.

The so-called Background Intelligent Transfer Service (BITS) [BITS] is implemented in several versions of Microsoft Windows. BITS uses a system of application-layer priority levels for file-transfer jobs, together with monitoring of bandwidth usage of the network interface (or, in more recent versions, of the network gateway connected to the end-host), so that, low-priority transfers at a given end-host give

way to both high-priority (foreground) transfers and traffic from interactive applications at the same host.

A different approach is taken in [Egg05] -- here, the priority of a flow is reduced via a generic idletime scheduling strategy in a host's operating system. While results presented in this paper show that the new scheduler can effectively shield regular tasks from low-priority ones (e.g., TCP from greedy UDP) with only a minor performance impact, it is an underlying assumption that all involved end hosts would use the idletime scheduler. In other words, it is not the focus of this work to protect a standard TCP flow which originates from any host where the presented scheduling scheme may not be implemented.

4.1. Receiver-oriented, flow-control based approaches

Some proposals for achieving an LBE behavior work by exploiting existing transport-layer features -- typically, at the "receiving" side. In particular, TCP's built-in flow control can be used as a means to achieve a low-priority transport service.

The mechanism described in [Spr00] is an example of the above technique. Such mechanism controls the bandwidth by letting the receiver intelligently manipulate the receiver window of standard TCP. This is possible because the authors assume a client-server setting where the receiver's access link is typically the bottleneck. The scheme incorporates a delay-based calculation of the expected queue length at the bottleneck, which is quite similar to the calculation in the above delay-based protocols, e.g. TCP Vegas. Using a Linux implementation, where TCP flows are classified according to their application's needs, Spring et al. show in [Spr00] that a significant improvement in packet latency can be attained over an unmodified system, while maintaining good link utilization.

A similar method is employed by Mehra et al. [Meh03], where both the advertised receiver window and the delay in sending ACK messages are dynamically adapted to attain a given rate. As in [Spr00], Mehra et al. assume that the bottleneck is located at the receiver's access link. However, the latter also propose a bandwidth-sharing system, allowing to control the bandwidth allocated to different flows, as well as to allot a minimum rate to some flows.

Receiver window tuning is also done in [Key04], where choosing the right value for the window is phrased as an optimization problem. On this basis, two algorithms are presented, binary search -- which is faster than the other one at achieving a good operation point but fluctuates -- and stochastic optimization, which does not fluctuate but converges slower than binary search. These algorithms merely use the previous receiver window and the amount of data received during the previous control interval as input. According to [Key04], the encouraging simulation results suggest that such an application level

mechanism can work almost as well as a transport layer scheme like TCP-LP.

Another way of dealing with non-interactive flows, like e.g. web prefetching, is to rate-limit the transfer of such bursty traffic [Cro98b]. Note that one of the techniques used in [Cro98b] is, precisely, to have the downloading application adapt the TCP receiver window, so as to reduce the data rate to the minimum needed (thus, disturbing other flows as little as possible while respecting a deadline for the transfer of the data).

5. Network-assisted approaches

Network-layer mechanisms, like active queue management (AQM) and packet scheduling in routers, can be exploited by a transport protocol for achieving an LBE service. Such approaches may result in improved protection of non-LBE flows (e.g., when scheduling is used); besides, approaches using an explicit, AQM-based congestion signaling may arguably be more robust than, say, delay-based transports for detecting impending congestion. However, an obvious drawback of any network-assisted approach is that, in principle, they need modifications in both end-hosts and intermediate network nodes.

Harp [Kok04] realizes a LBE service by dissipating background traffic to less-utilized paths of the network, based on multipath routing and multipath congestion control. This is achieved without changing all routers, by using edge nodes as relays. According to the authors, these edge nodes should be gateways of organizations in order to align their scheme with usage incentives, but the technical solution would also work if Harp was only deployed in end hosts. It detects impending congestion by looking at delay, similar to TCP Nice

[Ven02], and manages to improve the utilization and fairness of TCP over pure single-path solutions without requiring any changes to the TCP itself.

Another technique is that used by protocols like Network-Friendly TCP (NF-TCP) [Aru10], where a bandwidth-estimation module integrated into the transport protocol allows to rapidly take advantage of free capacity. NF-TCP combines this with an early congestion detection based on Explicit Congestion Notification (ECN) [RFC3168] and RED [RFC2309]; when congestion starts building up, appropriate tuning of a RED queue allows to mark low-priority (i.e., NF-TCP) packets with a much higher probability than high-priority (i.e., standard TCP) packets, so low-priority flows yield up bandwidth before standard TCP flows. NF-TCP could be implemented by adapting the congestion control behavior of TCP without requiring to change the protocol on the wire -- with the only exception that NF-TCP-capable routers must

be able to somehow distinguish NF-TCP traffic from other TCP traffic.

In [Ven08], Venkataraman et al. propose a transport-layer approach to leverage an existing, network-layer LBE service based on priority queueing. Their transport protocol, which they call PLT (Priority-Layer Transport), splits a layer-4 connection into two flows, a high-priority one and a low-priority one. The high-priority flow is sent over the higher-priority queueing class (in principle, offering a best-effort service) using an AIMD, TCP-like congestion control mechanism. The low-priority flow, which is mapped to the LBE class, uses a non TCP-friendly congestion control algorithm. The goal of PLT is thus to maximize its aggregate throughput by exploiting unused capacity in an aggressive way, while protecting standard TCP flows carried by the best-effort class. Similar in spirit, [Ott03] proposes simple changes to only the AIMD parameters of TCP for use over a network-layer LBE service, so that such "filler" traffic may aggressively consume unused bandwidth. Note that [Ven08] also considers a mechanism for detecting the lack of priority queueing in the network, so that the non-TCP friendly flow may be inhibited. The PLT receiver monitors the loss rate of both flows; if the high-priority flow starts seeing losses while the low-priority one does not experience 100% loss, this is taken as an indication of the absence of strict priority queueing.

6. LEDBAT Considerations

The previous sections have shown that there is a large amount of work on attaining an LBE service, and that it is quite heterogeneous in nature. The algorithm developed by the LEDBAT working group [Sha11] can be classified as a delay-based mechanism, and is as such similar in spirit to the protocols presented in Section 2. It is, however, not a protocol -- how it is actually applied to the Internet, i.e., how to use existing or even new transport protocols together with the LEDBAT algorithm, is not defined by the LEDBAT Working Group. As it heavily relies on delay, the discussion in Section 2.1 and Section 2.2 applies to it. The performance of LEDBAT has been analyzed in comparison with some of the other work presented here in several articles, e.g. [Aru10], [Car10], [Sch10] but these analyses have to be examined with care: at the time of writing, LEDBAT was still a moving target.

7. Acknowledgements

The authors would like to thank Melissa Chavez, Dragana Damjanovic and Yinxia Zhao for reference pointers, as well as Jari Arkko, Mayutan Arumaithurai, Elwyn Davies, Wesley Eddy, Stephen Farrell,

Mirja Kuehlewind, Tina Tsou and Rolf Winter for their detailed reviews and suggestions.

8. IANA Considerations

This memo includes no request to IANA.

9. Security Considerations

This document introduces no new security considerations.

10. Changes from the previous version (TO BE REMOVED BY THE RFC EDITOR UPON COMPLETION)

- Fixed a broken sentence at the end of the introduction.

11. Informative References

[Aru10] Arumaithurai, M., Fu, X., and K. Ramakrishnan, "NF-TCP: A Network Friendly TCP Variant for Background Delay-Insensitive Applications", Technical Report No. IFI-TB-2010-05, Institute of Computer Science, University of Goettingen, Germany, September 2010, <http://www.net.informatik.uni-goettingen.de/publications/1718/NF-TCP-techreport.pdf>.

[BITS] Microsoft, "Windows Background Intelligent Transfer Service", <http://msdn.microsoft.com/library/bb968799(VS.85).aspx>.

[Bha07] Bhandarkar, S., Reddy, A., Zhang, Y., and D. Loguinov, "Emulating AQM from end hosts", Proceedings of ACM SIGCOMM 2007, 2007.

[Bia03] Biaz, S. and N. Vaidya, "Is the round-trip time correlated with the number of packets in flight?", Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement (IMC '03) , pages 273-278, 2003.

[Bra94] Brakmo, L., O'Malley, S., and L. Peterson, "TCP Vegas: New techniques for congestion detection and avoidance", Proceedings of SIGCOMM '94, pages 24-35, August 1994.

[Car10] Carofiglio, G., Muscariello, L., Rossi, D., and S. Valenti, "The quest for LEDBAT fairness", Proceedings of IEEE GLOBECOM 2010, December 2010.

[Cha10] Chan, Y., Lin, C., Chan, C., and C. Ho, "CODE TCP: A competitive delay-based TCP", Computer Communications , 33(9):1013-1029, June 2010.

[Cro98] Crowcroft, J. and P. Oechslin, "Differentiated end-to-end Internet services using a weighted proportional fair sharing TCP", ACM SIGCOMM Computer Communication Review vol. 28, no. 3 (July 1998), pp. 53-69, 1998.

[Cro98b] Crovella, M. and P. Barford, "The network effects of prefetching", Proceedings of IEEE INFOCOM 1998, April 1998.

[Dam09] Damjanovic, D. and M. Welzl, "MulTFRC: Providing Weighted Fairness for Multimedia Applications (and others too!)", ACM Computer Communication Review vol. 39, no. 3 (July 2009), 2009.

[Dev03] De Vendictis, A., Baiocchi, A., and M. Bonacci, "Analysis and enhancement of TCP Vegas congestion control in a mixed TCP Vegas and TCP Reno network scenario", Performance Evaluation , 53(3-4):225-253, 2003.

[Dyk02] Dykes, S. and K. Robbins, "Limitations and benefits of cooperative proxy caching", IEEE Journal on Selected Areas in Communications 20(7):1290-1304, September 2002.

[Egg05] Eggert, L. and J. Touch, "Idletime Scheduling with Preemption Intervals", Proceedings of 20th ACM Symposium on Operating Systems Principles SOSP 2005, Brighton, United Kingdom, pp. 249/262, October 2005.

[Gur04] Gurtov, A. and S. Floyd, "Modeling wireless links for transport protocols", ACM SIGCOMM Computer Communications Review 34(2):85-96, April 2004.

[Hac04] Hacker, T., Noble, B., and B. Athey, "Improving Throughput and Maintaining Fairness using Parallel TCP", Proceedings of IEEE INFOCOM 2004, March 2004.

[Hac08] Hacker, T. and P. Smith, "Stochastic TCP: A Statistical Approach to Congestion Avoidance", Proceedings of PFLDnet 2008, March 2008.

[Hay10] Hayes, D., "Timing enhancements to the FreeBSD kernel to support delay and rate based TCP mechanisms", Technical Report 100219A, Centre for Advanced Internet Architectures, Swinburne University of Technology, February 2010.

[Hen00] Hengartner, U., Bolliger, J., and T. Gross, "TCP Vegas revisited", Proceedings of IEEE INFOCOM 2000, March 2000.

[Jai89] Jain, R., "A delay-based approach for congestion avoidance in interconnected heterogeneous computer networks", ACM Computer Communication Review , 19(5):56-71, October 1989.

[Key04] Key, P., Massoulie, L., and B. Wang, "Emulating Low-Priority Transport at the Application Layer: a Background Transfer Service", Proceedings of ACM SIGMETRICS 2004, January 2004.

[Kok04] Kokku, R., Bohra, A., Ganguly, S., and A. Venkataramani, "A Multipath Background Network Architecture", Proceedings of IEEE INFOCOM 2007, May 2007.

[Kon09] Konda, V. and J. Kaur, "RAPID: Shrinking the Congestion-control Timescale", Proceedings of IEEE INFOCOM 2009, April 2009.

[Kuo08] Kuo, F. and X. Fu, "Probe-Aided MulTCP: an aggregate congestion control mechanism", ACM SIGCOMM Computer Communication Review vol. 38, no. 1 (January 2008), pp. 17-28, 2008.

[Kur00] Kurata, K., Hasegawa, G., and M. Murata, "Fairness Comparisons Between TCP Reno and TCP Vegas for Future Deployment of TCP Vegas", Proceedings of INET 2000, July 2000.

[Kuz06] Kuzmanovic, A. and E. Knightly, "TCP-LP: low-priority service via end-point congestion control", IEEE/ACM Transactions on Networking (ToN) Volume 14, Issue 4, pp. 739-752., August 2006, <http://www.ece.rice.edu/networks/TCP-LP/>.

[Liu07] Liu, S., Vojnovic, M., and D. Gunawardena, "Competitive and Considerate Congestion Control for Bulk Data Transfers", Proceedings of IWQoS 2007, June 2007.

[Liu08] Liu, S., Basar, T., and R. Srikant, "TCP-Illinois: A loss- and delay-based congestion control algorithm for high-speed networks", Performance Evaluation , 65(6-7):417-440,

2008.

[Mar03] Martin, J., Nilsson, A., and I. Rhee, "Delay-based congestion avoidance for TCP", IEEE/ACM Transactions on Networking , 11(3):356-369, June 2003.

[McC08] McCullagh, G. and D. Leith, "Delay-based congestion control: Sampling and correlation issues revisited", Technical report, Hamilton Institute, 2008.

[Meh03] Mehra, P., Zakhor, A., and C. De Vleeschouwer, "Receiver-Driven Bandwidth Sharing for TCP", Proceedings of IEEE INFOCOM 2003, April 2003.

[Mo99] Mo, J., La, R., Anantharam, V., and J. Walrand, "Analysis and Comparison of TCP Reno and TCP Vegas", Proceedings of IEEE INFOCOM 1999, March 1999.

[Ott03] Ott, B., Warnky, T., and V. Liberatore, "Congestion control for low-priority filler traffic", SPIE QoS 2003 (Quality of Service over Next-Generation Internet), In Proc. SPIE, Vol. 5245, 154, Monterey (CA), USA, July 2003.

[Pra04] Prasad, R., Jain, M., and C. Dovrolis, "On the effectiveness of delay-based congestion avoidance", Proceedings of PFLDnet , 2004.

[RFC1323] Jacobson, V., Braden, B., and D. Borman, "TCP Extensions for High Performance", RFC 1323, May 1992.

[RFC2309] Braden, B., Clark, D., Crowcroft, J., Davie, B., Deering, S., Estrin, D., Floyd, S., Jacobson, V., Minshall, G., Partridge, C., Peterson, L., Ramakrishnan, K., Shenker, S., Wroclawski, J., and L. Zhang, "Recommendations on Queue Management and Congestion Avoidance in the Internet", RFC 2309, April 1998.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, September 2001.

[RFC3662] Bless, R., Nichols, K., and K. Wehrle, "A Lower Effort Per-Domain Behavior (PDB) for Differentiated Services", RFC 3662, December 2003.

[RFC5348] Floyd, S., Handley, M., Padhye, J., and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification",

RFC 5348, September 2008.

[RFC5681] Allman, M., Paxson, V., and E. Blanton, "TCP Congestion Control", RFC 5681, September 2009.

[Rew06] Rewaskar, S., Kaur, J., and D. Smith, "Why don't delay-based congestion estimators work in the real-world?", Technical report TR06-001, University of North Carolina at Chapel Hill, Dept. of Computer Science, January 2006.

[Sch10] Schneider, J., Wagner, J., Winter, R., and H. Kolbe, "Out of my Way -- Evaluating Low Extra Delay Background Transport in an ADSL Access Network", Proceedings of the 22nd International Teletraffic Congress ITC22, 2010.

[Sha05] Shalunov, S., Dunn, L., Gu, Y., Low, S., Rhee, I., Senger, S., Wydrowski, B., and L. Xu, "Design Space for a Bulk Transport Tool", Technical Report, Internet2 Transport Group, May 2005.

[Sha11] Shalunov, S., Hazel, G., Iyengar, J., and M. Kuehlewind, "Low Extra Delay Background Transport (LEDBAT)", draft-ietf-ledbat-congestion-04.txt (work in progress), March 2011.

[Spr00] Spring, N., Chesire, M., Berryman, M., Sahasranaman, V., Anderson, T., and B. Bershad, "Receiver based management of low bandwidth access links", Proceedings of IEEE INFOCOM 2000, pp. 245-254, vol.1, 2000.

[Sri08] Sridharan, M., Tan, K., Bansala, D., and D. Thaler, "Compound TCP: A new TCP congestion control for high-speed and long distance networks", Internet Draft draft-sridharan-tcpm-ctcp , work in progress, November 2008.

[Tan06] Tan, K., Song, J., Zhang, Q., and M. Sridharan, "A Compound TCP approach for high-speed and long distance networks", Proceedings of IEEE INFOCOM 2006, Barcelona, Spain, April 2008.

[Ven02] Venkataramani, A., Kokku, R., and M. Dahlin, "TCP Nice: a

mechanism for background transfers", Proceedings of OSDI '02, 2002.

[Ven08] Venkataraman, V., Francis, P., Kodialam, M., and T. Lakshman, "A priority-layered approach to transport for high bandwidth-delay product networks", Proceedings of ACM CoNEXT, Madrid, December 2008.

[Wan91] Wang, Z. and J. Crowcroft, "A new congestion control scheme: slow start and search (Tri-S)", ACM Computer Communication Review , 21(1):56-71, January 1991.

[Wan92] Wang, Z. and J. Crowcroft, "Eliminating periodic packet losses in the 4.3-Tahoe BSD TCP congestion control algorithm", ACM Computer Communication Review , 22(2): 9-16, January 1992.

[Wei05] Weigle, M., Jeffay, K., and F. Smith, "Delay-based early congestion detection and adaptation in TCP: impact on web performance", Computer Communications 28(8):837-850, May 2005.

[Wei06] Wei, D., Jin, C., Low, S., and S. Hegde, "FAST TCP: Motivation, architecture, algorithms, performance", IEEE/ACM Transactions on Networking , 14(6):1246-1259, December 2006.

Authors' Addresses

Michael Welzl
University of Oslo
Department of Informatics, PO Box 1080 Blindern
N-0316 Oslo,
Norway

Phone: +47 22 85 24 20
Email: michawe@ifi.uio.no

David Ros

Institut Telecom / Telecom Bretagne
Rue de la Chataigneraie, CS 17607
35576 Cesson Sevigne cedex,
France

Phone: +33 2 99 12 70 46
Email: david.ros@telecom-bretagne.eu