Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: September 4, 2009

M. Welzl
University of Innsbruck

March 3, 2009

A Survey of Lower-than-Best Effort Transport Protocols
draft-welzl-ledbat-survey-00.txt

Status of this Memo

Copyright Notice

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 low-priority "background" traffic, as they provide what is sometimes called a "less than" (or "lower than") best effort service.

Table of Contents

1. Introduction

As a starting point for the work in the LEDBAT group, this document presents a brief survey of efforts to attain a Less than Best Effort (LBE) service without help from routers. We loosely define a LBE service as a service which has smaller bandwidth and/or delay impact on standard TCP than standard TCP itself when sharing a bottleneck with it. We refer to systems that provide this service as Less than Best Effort (LBE) systems. 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. Some mechanisms achieve a LBE behavior at the application layer, e.g. by changing the receiver window of standard TCP, and 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. According to this classification, solutions have been categorized as delay-based transport protocols, non-delay-based transport protocols, application layer approaches and orthogonal work in this document.

The author wishes to emphasize that, in its present form, this document is only a starting point and not based on a thorough literature study. Many relevant references will be missing, and an apology goes to all authors of related work that has not been mentioned here.

2. Delay-based transport protocols

It is wrong to generally equate "little impact on standard TCP" with "small sending rate". Unless the sender's maximum window is limited for some reason, and in the absence of ECN support, standard TCP will normally increase its rate until a queue overflows, causing one or more packets to be dropped and the 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. In the absence of any other traffic, this is even true for TCP itself when its maximum send window is limited to the bandwidth*round-trip time (RTT) product.

TCP Vegas [Bra+94] is one of the first protocols that was known to have a smaller sending rate than standard TCP when both protocols share a bottleneck [Kur+00] -- yet it was designed to achieve more, not less throughput than standard TCP. Indeed, when it is the only protocol on the bottleneck, the throughput of TCP Vegas 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].

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 [Hen+00]). 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 (measured) throughput. If the actual throughput is smaller than the expected throughput minus a threshold, this is taken as a sign that the network is underutilized, causing the protocol to linearly increase its rate. If the actual throughput is greater than the expected throughput plus a threshold, this is taken as a sign of congestion, causing the protocol to linearly decrease 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 (which can be found in [Bra+94]) are not discussed here because of the historical "landmark" role that TCP Vegas has taken in the literature.

Transport protocols which were designed to be non-intrusive include

TCP-LP [Kuz+06], TCP Nice [Ven+02] and 4CP [Liu+07]. Using a simple analytical model, the authors of [Kuz+06] illustrate the feasibility of this endeavor by showing that, due to the non-linear relationship between throughput and RTT, it is possible to remain transparent to standard TCP even when the flows under consideration have a larger RTT than standard TCP flows.

TCP Nice [Ven+02] 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 [Ven+02] 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 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 halving the congestion window like standard TCP. It does so by halving the window at first in response to an early congestion indication, then initializing an "interference time-out timer", and maintaining the window size until this timer fires. If another early congestion indication appeared during this "interference phase", the window is then set to 1; otherwise, the window is maintained and TCP-LP continues to increase it 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.

With ns-2 simulations and real-life experiments using a Linux implementation, the authors of [Kuz+06] 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.

3. Non-delay-based transport protocols

4CP [Liu+07], 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. 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.

Some work was done on applying weights to congestion control mechanisms, allowing a flow to be as aggressive as a number of parallel TCP flows at the same time. This is usually motivated by the fact that users may want to assign different priorities to different flows. The first, and best known, such protocol is MulTCP [Cro+98], which emulates N TCPs in a rather simple fashion. An improved version of MulTCP is presented in [Hac+04], and there is also a variant where only one feedback loop is applied to control a larger traffic aggregate by the name of Probe-Aided (PA-)MulTCP [Kuo+08]. Another protocol, CP [Ott+04], applies the same concept to the TFRC protocol [RFC5348] in order to provide such fairness differentiation for multimedia flows.

The general assumption underlying all of the above work is that these protocols are "N-TCP-friendly", i.e. they are as TCP-friendly as N TCPs, where N is a positive (and possibly natural) number which is greater than or equal to 1. The MulTFRC [Dam+09] protocol, another extension of TFRC for multiple flows, is however able to support values between 0 and 1, making it applicable as a mechanism for a LBE service. Since it does not react to delay like the mechanisms above but adjusts its rate like TFRC, it 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 aggression is tunable at a fine granularity even when N is between 0 and 1.

4. Application layer approaches

The mechanism described in [Spr+00] controls the bandwidth by letting the receiver intelligently manipulate the receiver window of standard

TCP. This is done 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, it is shown that a significant improvement in packet latency can be attained over an unmodified system while

maintaining good link utilization.

Receiver window tuning is also done in [Key+04], 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 [Key+04], the encouraging simulation results suggest that such an application level mechanism can work almost as well as a transport layer scheme like TCP-LP.

TODO: mention other rwnd tuning and different application layer work, e.g. from related work sections of [Egg+05] and [Kok+04] and intro of [Key+04].

5. Orthogonal work

Various suggestions have been published for realizing a LBE service by influencing the way packets are treated in routers. One example is the Persistent Class Based Queuing (P-CBQ) scheme presented in [Car+01], which is a variant of Class Based Queuing (CBQ) with per-flow accounting. RFC 3662 [RFC3662] defines a DiffServ per-domain behavior called "Lower Effort".

Harp [Kok+04] realizes a LBE service by dissipating background traffic to less-utilized paths of the network. This is achieved without changing 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 [Ven+02] and manages to improve utilization and fairness over pure single-path solutions.

An entirely different approach is taken in [Egg+05]: 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.

TODO: studies dealing with the precision of congestion prediction in end hosts (i.e. using delay to determine the onset of congestion) may be relevant in this document, and could be discussed here, e.g. [Bha+07] and the references therein.

6. Acknowledgements

The author would like to thank Dragana Damjanovic for reference pointers. Surely lots of other folks will help in one way or another later and I'll thank them all here.

7. IANA Considerations

This memo includes no request to IANA.

8. Security Considerations

This document introduces no new security considerations.

9. Informative References

[Bha+07] Bhandarkar, S., Reddy, A., Zhang, Y., and D. Loguinov, "Emulating AQM from end hosts", Proceedings of ACM SIGCOMM 2007, 2007.

[Bra+94] 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.

[Car+01] Carlberg, K., Gevros, P., and J. Crowcroft, "Lower than best effort: a design and implementation", Workshop on Data communication in Latin America and the Caribbean 2007, San Jose, Costa Rica, Pages: 244 - 265, 2001.

[Cro+98] 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.

[Dam+09] Damjanovic, D. and M. Welzl, "MulTFRC: Providing Weighted Fairness for Multimedia Applications (and others too!)", Work in progress ..., 2009.

[Egg+05] Eggert, L. and J. Touch, "A Lower Effort Per-Domain Behavior (PDB) for Differentiated Services", Proceedings of 20th ACM Symposium on Operating Systems Principles SOSP 2005, Brighton, United Kingdom, pp. 249/262, October 2005.

[Hac+04] Hacker, T., Noble, B., and B. Athey, "Improving Throughput and Maintaining Fairness using Parallel TCP", Proceedings of Infocom 2004, March 2004.

[Hen+00] Hengartner, U., Bolliger, J., and T. Gross, "TCP Vegas revisited", Proceedings of Infocom 2000, March 2000.

[Key+04] Key, P., MassouliA(C), L., and B. Wang, "Emulating Low-Priority Transport at the Application Layer: a Background Transfer Service", Proceedings of ACM SIGMETRICS 2004, January 2004.

[Kok+04] Kokku, R., Bohra, A., Ganguly, S., and A. Venkataramani, "A Multipath Background Network Architecture", Proceedings of Infocom 2007, May 2007.

[Kuo+08] 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.

[Kur+00] 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.

[Kuz+06] 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/>.

[Liu+07] Liu, S., Vojnovic, M., and D. Gunawardena, "Competitive and Considerate Congestion Control for Bulk Data

Transfers", Proceedings of IWQoS 2007, June 2007.

[Ott+04] Ott, D., Sparks, T., and K. Mayer-Patel, "Aggregate congestion control for distributed multimedia applications", Proceedings of Infocom 2004, March 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.

[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.

[Spr+00] Spring, N., Chesire, M., Berryman, M., Sahasranaman, V., Anderson, T., and B. Bershad, "Receiver based management of low bandwidth access links", Proceedings of Infocom 2000, pp. 245-254, vol.1, 2000.

[Ven+02] Venkataramani, A., Kokku, R., and M. Dahlin, "TCP Nice: a mechanism for background transfers", Proceedings of OSDI '02, 2002.

Author's Address

Michael Welzl
University of Innsbruck
Technikerstr. 21 A

Innsbruck, 6020
Austria

Phone: +43 512 507 6110
Email: michael.welzl@uibk.ac.at