ISSN No. 0976-5697

Volume 1, No. 4, Nov-Dec 2010

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

TCP-LP: Low-Priority Service via End-Point Congestion Control

P. Swarnalatha
Assistant Professor (Sr. Grade),
School of Computing Science and Engineering
VIT University, Vellore

H. Parveen Sultana
Assistant Professor (Sr. Grade),
School of Computing Science and Engineering
VIT University, Vellore -14

M. Pounambal*
Assistant Professor (Sr. Grade),
School of Computing Science and Engineering
VIT University, Vellore,India
mpounambal@vit.ac.in

Abstract: Transmission Control Protocol-Low Priority (TCP-LP), a algorithm goal is to utilize only the excess network bandwidth as compared to the "fair share" of bandwidth as targeted by TCP. Service prioritization among different traffic classes is an important goal for the Internet. Conventional approaches in solving this problem, considers the existing best-effort class as the low-priority class, and attempt to develop mechanisms that provide "better-than-best-effort" service. We explore the opposite approach, and devise a new distributed algorithm to realize a low-priority service (as compared to the existing best effort) from the network endpoints. To this end, we develop TCP Low Priority (TCP-LP), a distributed algorithm whose goal is to utilize only the excess network bandwidth as compared to the "fair share" of bandwidth as targeted by TCP. The key mechanisms unique to TCP-LP congestion control are the use of one-way packet delays for early congestion indications and a TCP-transparent congestion avoidance policy. The results of our simulation and Internet experiments show that TCP-LP is largely non-intrusive to TCP traffic. Both single and aggregate TCP-LP flows are able to successfully utilize excess network bandwidth; moreover, multiple TCP-LP flows share excess bandwidth fairly. Substantial amounts of excess bandwidth are available to the low-priority class, even in the presence of "greedy" TCP flows. Despite their low-priority nature, TCP-LP flows are able to utilize significant amounts of available bandwidth in a wide-area network environment.

Keywords: TCP Transmission Control Protocol, low priority, bandwidth

I. INTRODUCTION

In the world of globalization, the data is being the backbone of IT industries. The data transfer speed has been increased by the proposed method. TCP Low Priority (TCP-LP), a algorithm whose goal is to utilize only the excess network bandwidth as compared to the "fair share" of bandwidth[5] as targeted by TCP. Since TCP is the dominant protocol for best-effort traffic, we designed TCP-LP to realize a low-priority service as compared to the existing best effort service. The objective is for TCP-LP flows to utilize the bandwidth left unused by TCP flows in a non-intrusive[8], or TCP-transparent, fashion.

Moreover, TCP-LP is a distributed algorithm that is realized as a sender-side modification of the TCP protocol. One class of applications of TCP-LP is low-priority file transfer over the Internet. For network clients on low-speed access links, TCP-LP provides a mechanism to retain faster response times for interactive applications using TCP, while simultaneously making progress on background file transfers using TCP-LP. Similarly, in enterprise networks, TCP-LP enables large file backups to proceed without impeding interactive applications, a functionality that would otherwise require a multi-priority or separate network. In contrast, TCP-LP allows low priority applications to use all excess capacity while also remaining transparent to TCP flows.

A second class of applications of TCP-LP is inference of Available bandwidth for network monitoring [2], end-point admission control, and performance Optimization. Current techniques estimate available bandwidth by making statistical inferences on measurements of the delay or loss characteristics of a sequence of transmitted
Probe packets.

In contrast, TCP-LP is algorithmic with the goal of transmitting at the rate of the available bandwidth. Consequently, competing TCP-LP flows obtain their fair share of the available bandwidth, as opposed to probing flows which infer the total available bandwidth, overestimating the fraction actually available individually when many flows are simultaneously probing. Moreover, as the available bandwidth changes over time, TCP-LP provides a mechanism to continuously adapt to changing network conditions.

II. TRANSMISSION CONTROL PROTOCOL (TCP)

In the Internet protocol suite, TCP the intermediate layer between the Internet Protocol (IP) below it, and an application above it. Applications often need reliable pipe-like connections to each other, whereas the Internet Protocol does not provide such streams, rather only reliable packets. TCP does the task of the Transport Layer in the simplified OSI model of computer networks [5]. The Transmission Control Protocol (TCP) is one of the core protocols of the Internet Protocol suite. Using TCP, applications on networked hosts can create connections to one and in-order delivery of data from sender to receiver. TCP also distinguishes data for multiple connections [6] by current applications e.g. web server and e-mail server running on the same host. TCP supports many of Internet's most popular application protocols and resulting applications including the World Wide Web e-mail and Secure Shell.

A. Data Transfer in TCP

Application send streams of octets (8-bit bytes) to TCP for delivery through the network, and TCP divides the byte streams into appropriately sized segments usually delineated by the maximum transmission unit (MTU) size of the data link layer of the network the computer attached to. TCP then passes the resulting packets to the Internet Protocol, for delivery through a network to the TCP module of the entity at the other end. TCP checks to make sure no packets are lost by giving each packet a sequence number, which is also used to make sure that the data are delivered to the entity at the other end in the correct order. The TCP module at the far end sends back an acknowledgement for packets which have been successfully receiver, a timer at the sending TCP will cause a timeout if an acknowledgement is not received with a reasonable round-trip time RTT) and the data will be re-transmitted[11]. The TCP checks that no bytes are damaged by using a checksum; one is computer at the sender for each block of data before it is send, and checked at the receiver.

B. Congestion Control

The final part to TCP is congestion throttling. Acknowledgements for data send, or lack of acknowledgements, are used by senders to implicitly interpret network conditions between the TCP sender and receiver. Coupled with the timers, TCP senders and receivers can alter the behavior of the flow of data. This is generally referred to as flow control, congestion control and /or network congestion avoidance. TCP uses a number of mechanisms to achieve high performance and avoid congesting the network.

Enhancing TCP to reliably handle loss, minimize errors, manage congestion and go fast in very high-speed environments are ongoing areas of research and standards developments.

C. TCP-LP

In "TCP-LP: LOW PRIORITY SERVICE VIA END-POINT CONGESTION CONTROL" Service prioritization among different traffic classes is an important goal for the Internet. Conventional approaches in solving this problem, consider the existing best-effort class as the low-priority class, and attempt to develop mechanisms that provide "better-than-best-effort" service. In this paper, the opposite approach, and devise a new distributed algorithm to realize a low-priority service (as compared to the existing best effort) from the network endpoints. To this end, we develop TCP Low Priority (TCP-LP), a distributed algorithm whose goal is to utilize only the excess network bandwidth as compared to the "fair share" of bandwidth as targeted by TCP. The key mechanisms unique to TCP-LP congestion control are the use of one-way packet delays for early congestion indications and a TCP-transparent congestion avoidance policy. The result of the simulation and Internet experiments shows that:
(a) TCP-LP is largely non-intrusive to TCP traffic;
(b) Both single and aggregate TCPLP flows are able to successfully utilize excess network bandwidth; moreover, multiple TCP-LP flows share excess bandwidth fairly;
(c) Substantial amounts of excess bandwidth are available to the low-priority class, even in the presence of "greedy" TCP flows;
(d) the response times of web connections in the best-effort class decrease by up to 90% when long-lived bulk data transfers use TCP-LP rather than TCP;
(e) Despite their low-priority nature, TCP-LP flows are able to utilize significant amounts of available bandwidth in a wide-area network environment.

D. Existing Network

Motivated by the diversity of networked applications, a significant effort has been made to provide differentiation mechanisms in the Internet. However, despite the availability of simple and scalable solutions, deployment has not been forthcoming. A key reason is the heterogeneity of the Internet [3] itself: with vastly different link capacities, congestion levels, etc., a single mechanism is unlikely to be uniformly applicable to all network elements.

E. Idea on Proposed Network

TCP-LP (Low Priority), an end-point protocol will be devised **that achieves** two-class service prioritization without any support from the network. The key observation is that end-to-end differentiation can be achieved by having different end-host applications employ different congestion control algorithms as dictated by their performance objectives. Since TCP is the dominant protocol for best-effort traffic, we design TCP-LP to realize a low-priority service as compared to the existing best effort service. Namely, the objective is for TCP-LP flows to utilize the bandwidth left unused by TCP flows in a non-intrusive, or TCP-transparent, fashion. Moreover, TCP-LP is a distributed algorithm that is realized as a sender-side modification of the TCP protocol.

F. Implementation Plan

First, we develop a reference model to formalize the two design objectives: TCP-LP transparency to TCP, and (TCP-like) fairness among multiple TCP-LP flows competing to share the excess bandwidth. The reference model consists of a two level hierarchical scheduler in which the first level provides TCP packets with strict priority over TCP-LP packets and the second level provides fairness among micro flows within each class. TCP-LP aims to achieve this behavior in networks with non differentiated (first-come-first-serve) service. Next, to approximate the reference model from a distributed end-point protocol, TCP-LP employs two new mechanisms. First, in order to provide TCP-transparent low-priority service, TCP-LP flows must detect oncoming congestion prior to TCP flows. Consequently, TCP-LP uses inferences of one-way packet delays as early indications of network congestion rather than packet losses as used by TCP. We develop a simple analytical model to show that due to the non-linear relationship between throughput and round-trip time, TCP-LP can maintain TCP-transparency even if TCP-LP flows have larger round-trip times than TCP flows. Moreover, a desirable consequence of early congestion inferences via one-way delay measurements is that they detect congestion only on the forward path (from the
Source to the destination) and prevent false early congestion indications from reverse cross-traffic.TCP-LP's second mechanism is a novel congestion avoidance
Policy with three objectives:

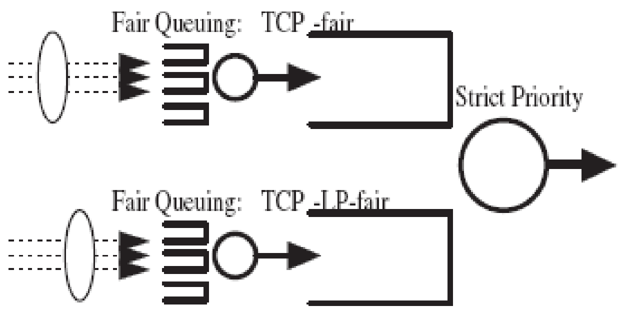
[a] Quickly back off in the presence of congestion from TCP flows,

[b] Quickly utilize the available excess bandwidth in the absence of sufficient TCP traffic, and

[c] Achieve fairness among TCP-LP flows.

To achieve these objectives, TCP-LP's congestion avoidance policy modifies the additive-increase multiplicative decrease policy of TCP via the addition of an inference phase and use of a modified back-off policy.

G. Reference Model



Two class hierarchical scheduling model

1. High-priority VS Low-priority class
2. Strict priority service

H. Algorithm Analysis

TCP-LP, a low-priority congestion control protocol that uses the excess bandwidth on an end-to-end path, versus the fair-rate utilized by TCP. We first devise a mechanism for early congestion indication via inferences of one-way packet delays. Next, we present TCP-LP's congestion avoidance policy to exploit available bandwidth while being sensitive to early congestion indicators. Then develop a simple queuing model to study the feasibility of TCP-transparent congestion control under heterogeneous round trip times. Finally, we provide guidelines for TCP-LP parameter settings.

I. Early Congestion Indication

To achieve low priority service in the presence of TCP traffic, it is necessary for TCP-LP to infer congestion earlier than TCP. In principle, the network could provide such early congestion indicators. For example, TCP-LP flows could use a type-of service bit to indicate low priority, and routers could use Early Congestion Notification (ECN) messages to inform TCPLP flows of lesser congestion levels than TCP flows. However, given the absence of such network support, we devise an endpoint realization of this functionality by using packet delays as early indicators for TCP-LP, as compared to packet drops used by TCP. In this way, TCP-LP and TCP implicitly coordinate in a distributed manner to provide the desired priority levels.

J. Delay Threshold

TCP-LP measures one-way packet delays and employs a simple delay threshold-based method for early inference of congestion. Denote di as the one-way delay of the packet with sequence number i, and dmin and dmax as the minimum and maximum one-way packet delays experienced throughout the As UDP flows are non-responsive, they would also be considered high priority and multiplexed with the TCP flows. Thus, dmin is an estimate of the one way propagation delay and dmax - dmin is an estimate of the maximum queuing delay. Next, denote as the delay smoothing parameter, and sdi as the smoothed one-way delay. An early indication of congestion is inferred by a TCP-LP flow whenever the smoothed one-way delay exceeds a threshold within the range of the minimum and maximum delay.

K. Delay Measurement

TCP-LP obtains samples of one-way packet delays using the TCP timestamp option. Each TCP packet carries two four byte timestamp fields. A TCP-LP sender timestamps one of these fields with its current clock value when it sends a data packet. On the other side, the receiver echoes back this timestamp value and in addition timestamps the ACK packet with its own current time. In this way, the TCP-LP sender measures one-way packet delays. Note that the sender and receiver clocks do not have to be synchronized since we are only interested in the relative time difference. Moreover, a drift between the two clocks is not significant here as resets of dmin and dmax on timescales of minutes can be applied. Finally, we note that by using one-way packet delay measurements instead of round-trip times, cross-traffic in the reverse direction does not influence TCP-LP's inference of early congestion. Minimum and maximum one-way packet delays are initially estimated during the slow-start phase and are used after the first packet loss, i.e., in the congestion avoidance phase.

L. Congestion Avoidance Policy

[a] Objectives TCP-LP is an end-point algorithm that aims to emulate the functionality of the reference- scheduling model depicted in Figure.

[b] Consider for simplicity a scenario with one TCP-LP and one TCP flow. The reference strict priority scheduler serves TCP-LP packets only when there are no TCP packets in the system. However, whenever TCP packets arrive, the scheduler immediately begins service of higher priority TCP packets.

Similarly, after serving the last packet from the TCP class, the strict priority scheduler immediately starts serving TCP-LP packets. Note that it is impossible to exactly achieve this behavior from the network endpoints as TCP-LP operates on timescales of round-trip times, while the reference scheduling model operates on time-scales of packet transmission times. Thus, our goal is to develop a congestion control policy that is able to approximate the desired dynamic behavior.

M. Additive Increase Multiplicative Decrease

Additive Increase Multiplicative Decrease (AIMD) is the dominant algorithm for congestion avoidance and control in the Internet. The major goal of AIMD is to achieve fairness and efficiency in allocating resources. In the context of packet networks, AIMD attains its goal partially. We exploit here a property of AIMD-based data sources to share common knowledge, yet in a distributed manner; we use this as our departing point to achieve better efficiency and faster convergence to fairness. Our control model is based on the assumptions of the original AIMD algorithm; we show that both efficiency and fairness of AIMD can be improved.

TCP Congestion Control

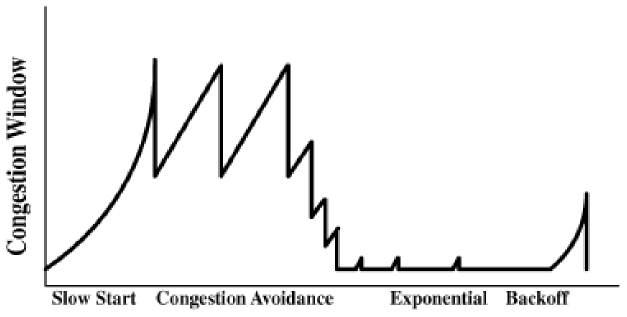


Figure .1 Behavior of TCP congestion control

Fig.1 shows a temporal view of the TCP/Reno congestion window behavior at different stages with points on the top indicating packet losses.1 Data transfer begins with the slow-start phase in which TCP increases its sending rate exponentially until it encounters the first loss or maximum window size. From this point on, TCP enters the congestion-avoidance phase and uses an additive-increase multiplicative-decrease policy to adapt to congestion. Losses are detected via either time-out from non receipt of an acknowledgment,. If loss occurs and less than three duplicate ACKs are received, TCP reduces its congestion window to one segment and waits for a period of retransmission time out (RTO), after which the packet is resent. In the case that another time out occurs before successfully retransmitting the packet, TCP enters the exponential-backoff phase and doubles RTO until the packet is successfully acknowledged. One objective of TCP congestion control is for each flow to transmit at its fair rate at its bottleneck link.

Finally, the minimum congestion window for TCP-LP flows in the inference phase is set to 1. In this way, TCP-LP flows conservatively ensure that an excess bandwidth of at least one packet per round-trip time is available before probing for additional Bandwidth.

N. Behavior of TCP-LP congestion avoidance phase

Fig.2 illustrates a schematic view of TCP-LP's congestion window behavior at different stages, where points on the top mark early congestion indications and the inference timer period are labeled it. For example, with the first early congestion indicator, this flow enters the inference phase. It later successfully exits the inference phase into additive increase as no further early congestion indicators occur. On the other hand, the second early congestion indicator is followed by a second

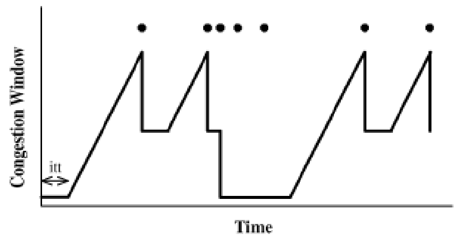
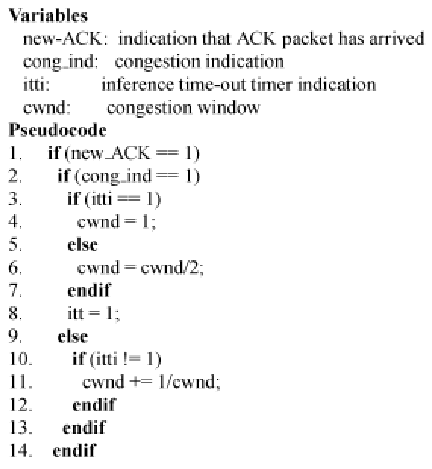


Figure. 2 Congestion Window

Indicator within the inference phase such that the congestion window is subsequently set to one.



Variables

```
new-ACK:  indication that ACK packet has arrived
cong_ind: congestion indication
itti:     inference time-out timer indication
cwnd:     congestion window
```

Pseudocode

```
if (new_ACK == 1)
    if (cong_ind == 1)
        if (itti == 1)
            cwnd = 1;
        else
            cwnd = cwnd/2;
        endif
        itt = 1;
    else
        if (itti != 1)
            cwnd += 1/cwnd;
        endif
    endif
endif
```

Figure. 3 AIMD Algorithm

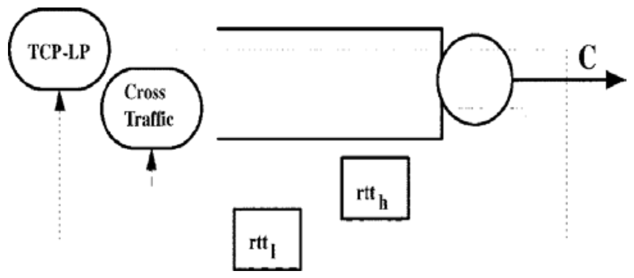
Fig. 3 shows the pseudo code for TCP-LP's Congestion avoidance policy. We denote cwnd as congestion window size and it's as the inference time-out timer state indicator. It is set to one when the timer is initiated and to zero when the timer expires.

O. Reacting to Early Congestion Indicators

TCP-LP must react quickly to early congestion indicators to achieve TCP-transparency. However, simply decreasing the congestion window promptly to zero packets after the receipt of an early congestion indication (as implied by the reference scheduling model) unnecessarily inhibits the throughput of TCP-LP flows. This is because a single early congestion indication cannot be considered as a reliable indication of network congestion given the complex dynamics of cross traffic. On the other hand, halving the congestion window of TCP-LP flows upon the congestion indication, as recommended for ECN flows would result in too slow a response to achieve TCP transparency. To compromise between the two extremes, TCP-LP employs the following algorithm. After receipt of the initial early congestion indication, TCP-LP halves its congestion window and enters an inference phase by starting an inference time-out timer. During this inference period, TCP-LP only observes responses from the network, without increasing its congestion window. If it receives another early congestion indication before the inference timer expires, this indicates the activity of cross traffic, and TCP-LP decreases its congestion window to one packet. Thus, with persistent congestion, it takes two round-trip times for a TCP-LP flow to decrease its window to 1. Otherwise, after expiration of the inference timer, TCP-LP enters the additive increase congestion avoidance phase and increases its congestion window by one per round-trip time (as with TCP flows in this phase). We observe that as with router-assisted early congestion indication consecutive packets from the same flow often experience similar network congestion state. Consequently, as suggested for ECN flows, TCP-LP also reacts to a congestion indication event at most once per round-trip time. Thus, in order to prevent TCP-LP from over-reacting to bursts of congestion indicated packets, TCP-LP ignores succeeding congestion indications if the source has reacted to a previous delay-based congestion indication or to a dropped packet in the last round-trip time. Finally, the minimum congestion window for TCP-LP flows in the inference phase is set to 1. In this way, TCP-LP flows conservatively ensure that an excess bandwidth of at least one packet per round-trip time is available before probing for additional bandwidth.

P. Preserving TCP-transparency in Large Aggregation

A key goal of TCP-LP is to achieve non-intrusiveness to TCP flows. Thus, TCP-LP reduces its window size to one packet per RTT in the presence of TCP flows. However, in scenarios with many TCP-LP flows, it becomes increasingly possible for TCP-LP aggregates to impact TCP flows. For example, consider a scenario with a hundred TCP-LP flows competing with TCP flows on a 10Mb/s link. If the roundtrip time of the TCP-LP flows is 100ms and the packet size is 1500Bytes, then this TCP-LP aggregate utilizes 12% of the bandwidth, despite the fact that each flow sends only a single packet per RTT.5 To mitigate this problem, TCP-LP decreases the packet size to 64Bytes whenever the window size drops below 5 packets. In this way, TCP-LP significantly decreases its impact on TCP flows in high-aggregation regimes, yet it is still able to quickly react (after RTT) to changes in congestion. In the above example, a hundred TCP-LP flows would then utilize only 0.5% of the bandwidth in the presence of TCP flows.



Simplified model of heterogeneous RTT effects

Q. Comparison with Previous Work

TCP-LP uses early congestion indication (earlier than TCP) as a basis for achieving class differentiation. RIO (RED with in and out) in which routers apply different marking/dropping functions for different classes of flows, thereby providing service differentiation. While similar in philosophy to TCP-LP, TCP-LP develops an end-point realization of early congestion indication for the purpose of low-priority transfer. Consequently, TCP-LP is applicable over routers and switches that provide no active queue management or service differentiation.

TCP-LP relates to adaptive bandwidth allocation schemes that aim to minimize file-transmission times using file size-based service differentiation. In core routers and a packet classifier at the edge to distinguish between long- and short-lived TCP flows. TCP/SA Reno in which the AIMD parameters dynamically depend on the remaining file size. While TCP-LP also substantially improves file-transmission times in the best effort class, the key difference between TCP-LP and the above schemes is that it provides strict low-priority service, independent of the file size.

Next, as TCP-LP targets transmitting at the rate of available bandwidth, it is related to cross-traffic estimation algorithms which attempt to infer the available bandwidth via probing, provide algorithms for estimation of parameters of competing cross-traffic under multifractal and Poisson models of cross traffic. In contrast, TCP-LP provides an adaptive estimation of available bandwidth by continually monitoring one-way delays and dynamically tracking the excess capacity. Similarly, path load, a delay-based rate-adaptive probing scheme for estimating available bandwidth. The key difference between path load and TCP-LP is that the latter aims to utilize the available bandwidth, while the former only estimates it. Moreover, TCP-LP addresses the case of multiple flows simultaneously inferring the available bandwidth by providing each with a fair share (according to TCP fairness), an objective that is problematic to achieve with probes.
Finally, end-point admission control algorithms also use probes to detect if sufficient bandwidth is available for real-time flows .Unfortunately, such techniques have a “thrashing” problem when many users probe simultaneously and none can be admitted. While TCP-LP targets a low rather than high priority class, its basic ideas of adaptive and transparent bandwidth estimation could be applied to end-point admission control and alleviate the thrashing condition.

R. Application

One class of applications of TCP-LP is low-priority file transfer over the Internet. For network clients on low-speed access links, TCP-LP provides a mechanism to retain faster response times for interactive applications using TCP, while simultaneously making progress on background file transfers using TCPLP.

Similarly, in enterprise networks, TCP-LP enables large file backups to proceed without impeding interactive applications, a functionality that would otherwise require a multi-priority or separate network. Finally, institutions often rate-limit certain applications (e.g. peer-to-peer file sharing applications) such that they do not degrade the performance of other applications. In contrast, TCP-LP allows low priority applications to use all excess capacity while also remaining transparent to TCP flows.

A second class of applications of TCP-LP is inference of available bandwidth for network monitoring, end-point admission control and performance optimization (e.g., to select a mirror server with the highest available bandwidth). Current techniques estimate available bandwidth by making statistical inferences on measurements of the delay or loss characteristics of a sequence of transmitted probe packets. In contrast, TCP-LP is algorithmic with the goal of transmitting at the rate of the available bandwidth. Consequently, competing TCP-LP flows obtain their fair share of the available bandwidth, as opposed to probing flows which infer the total available bandwidth, overestimating the fraction actually available individually when many flows are simultaneously probing. Moreover, as the available bandwidth changes over time, TCP-LP provides a mechanism to continuously adapt to changing network conditions.

S. Limitation

[a] Endpoint admission control certainly has its flaws. The Set-up delay is substantial, on the order of seconds, which may limit its appeal for certain applications.
[b] The utilization and loss rate can degrade somewhat under sufficiently high loads even with slow start probing.
[c] The quality of service is not predictable across settings.

T. Merits

[a] Security (Acknowledgement).
[b] Reliable (Sequence Number for every packet).
[c] The algorithm is adaptive it requires no priori traffic statistics.
[d] Effectively tracks changes in network conditions.
[e] Network simulator experiments revealed that Delphi gives accurate cross-traffic estimates for higher link utilization levels.

III. CONCLUSION

This paper presents TCP-LP, a protocol designed to achieve low-priority service (as compared to the existing best-effort class) from the network endpoints. TCP-LP allows low-priority applications such as bulk data transfer to utilize excess bandwidth without significantly perturbing non-TCP-LP flows. TCPLP is realized as a sender-side modification of the TCP congestion control protocol and requires no functionality from the network routers or any other protocol changes.

IV. REFERNCES

[1] L. Breslau, E. Knightly, S. Shenker, I. Stoica, and H. Zhang. “Endpoint admission control: Architectural issues and performance. In Proceedings of ACM SIGCOMM ’00, Stockholm, Sweden. Aug 2000.

[2] V. Ribeiro, M. Coates, R. Riedi, S. Sarvotham, B.Hendricks, and R. Baraniuk. Multifractal cross-traffic estimation. In Proceedings of ITC ’00, Monterey, CA. Sep 2000.

[3]S. Alouf, P. Nain, and D. Towsley. “Inferring network characteristics via moment-based estimators”. In Proceedings of IEEE INFOCOM Anchorage, Alaska. April 2001.

[4] S. Yang and G. de Veciana. “Size-based adaptive bandwidth allocation: Optimizing the average QoS for elastic flows”. In Proceedings of IEEE INFOCOM ’02, New York, NY. June 2002

[5] A. Venkataramani, R. Kokku, and M. Dahlin. TCP Nice: A mechanism for background transfers.In Proceedings of OSDI ’02, Boston, MA. Dec 2002.

[6] A. Bakre and B. R. Badrinath, “I-TCP: Indirect TCP for mobile hosts,” in *Proc. 15th Int. Conf. Distributed Computing Systems*, Vancouver, BC, Canada, June 1995, pp. 136–143.

[7] S. Floyd, “TCP and explicit congestion notification,”*ACM Comput. Commun. Rev.*, vol. 24, no. 5, pp. 10– 23, Oct. 1994.

[8] V. D. Park and M. S. Corson, “A highly adaptive distribu ted routing algorithm for mobile wireless networks,” in *Proc. IEEE INFOCOM*, Kobe, Japan, Apr. 1997, pp. 1405–1413.

[9] J. H. Salim and U. Ahmed, “Performance evaluation of explicit congestion notification (ECN) in IP networks,” RFC 2884, July 2000.

[10] K. K. Ramakrishnan and S. Floyd, “A proposal to add explicit congestion notification (ECN) to IP,”(Status: Experi mental), Jan. 1999.

[11] R. Yavatkar and N. Bhagawat, “Improving end-to- end performance of TCP over mobile internetworks,” in IEEE Workshop Mobile Computing Systems and Applications, Santa Cruz, CA, Dec. 1994, pp. 146–152.