1 Review A

This paper proposed a transport protocol for low-priority service, TCP-LP, to use the excess bandwidth in the network. TCP-LP uses one-way packet delays for early congestion indications. I think this is a strong and fairly thorough paper, and I would recommend it for acceptance by Transactions on Networking. I have some rather minor feedback to the authors below.

The smoothing parameter:
If the smoothing parameter is a fixed constant, such as 1/8, then the average one-way delay might be averaged over multiple round trip times when the congestion window is small, but averaged over a small fraction of a round-trip time when the congestion window is large. Is this the behavior that is intended? If one-way delays are measured for every packet, it would seem reasonable to me for the smoothing parameter to be a function of the congestion window, so that the average delay is computed over a fixed time period in RTTs, regardless of the congestion window. (The same criticism applies to TCP's current computation of the smoothed round-trip time when timestamps are used.)

———

(*) This is an interesting point raised by the reviewer. It is indeed true that the one-way delay (or RTT) is averaged over a period that depends on the window size. While we have simply adopted the solution that is embedded in all TCP versions, we believe that the logic behind this solution is to simply use every possible one-way delay (or RTT) sample to better estimate the network state. On the other hand, it may be true that a large number of correlated delay samples (in large window scenarios) may "weaken" the intended smoothing effects. However, we believe that this is a general problem (that applies to *all* TCP versions, as noted by the reviewer), and is beyond the scope of our paper.

———

Difficult scenarios:
It would be useful to discuss difficult scenarios for TCP-LP or for other low-priority transport protocols based on delay as early congestion indications. Examples of such scenarios include connections over wireless links with variable delay; or connections that use multiple paths. (Of course, the worst that would happen in a difficult scenario would be that TCP-LP would be as aggressive as standard TCP, or would fail to make use of available bandwidth.)

———

(*) The reviewer is right in observing that TCP-LP would have reduced performance over wireless links in scenarios dominated by non-congestion-induced delay variations. However, this is also true for other TCP stacks. For example, it is well known that even TCP Sack can have significant problems in wireless scenarios due to frequent non-congestion-induced packet losses. We have added a comment in Section VII to bring out this point.

———

```
It is clear that the use of one-way packet delay in TCP-LP has
advantages of the use of two-way packet delay, but are there
any extra costs incurred by TCP-LP, relative to, say, TCP Nice?
```

———

(*) The first extra cost is a more complex protocol implementation (which includes time-stamping), and the second is a reduced clock granularity, which is dominated by the less accurate of the two end-points. Both of these effects are explained in Section VI.A. However, we did not observe any performance degradations due to the above effects.

———

```
I like it that the ns code and simulation scripts are
publically available.
```

2 Review B

```
This paper designs an TCP congestion control algorithm, called TCP-LP,
to give a low-priority service to some flows. The algorithm uses delay
as an indicator of congestion which allows it to react before
traditional TCP congestion control. Furthermore timeouts are used to
ensure that TCP-LP gives priority to TCP flows. Some analysis is given
to motivate why TCP-LP should be unintrusive to TCP flows.
Simulations and experiments are then presented to support the claims that
TCP-LP is unintrusive to TCP flows, does not significantly increase
delay for interactive traffic, and shares fairly between TCP-LP flows,
while remaining able to use available bandwidth

The algorithm design is well motivated and described. To the
best of my knowledge using delay to separate priorities is a nice
contribution here. While the authors recognise the salient points about
\delta and RTT ratios, the itt variable is set heuristically.
```

———

(*) The reasoning behind the *itt* variable setting is to achieve a compromise between two extremes: with a longer inference time-out timer, TCP-LP becomes more responsive to congestion, whereas a smaller inference time-out timer causes TCP-LP to switch

sooner to the more aggressive additive-increase phase. The above tradeoff is explained in a more detail in Section III.D.3. While setting the *itt* parameter to three round-trip times is indeed heuristic, we believe that making the *itt* parameter a function of the RTT is actually the key in achieving the above compromise.

———

However the performance evaluation has several details that should be improved:
- the scale parameter of the Pareto distribution should be stated on page 7.

———

(*) The shape parameter of Pareto distribution is 1.2. This is stated in Section IV.B.

———

- it would be good to know that the HTTP response time simulations use the same sample path of file sizes and think times; otherwise the result is weaker.

———

(*) Yes, the HTTP response time simulations use the same sample path of file sizes and think times. We have added an explanation to this effect in the paper in Section V.C.1.

———

- It is also difficult to understand the simulation setup for HTTP flows; is there only one user or multiple users? From the results I suspect there is one user on the 1.5Mb pipe; I think the results would differ if there were multiple users because then the 1.5 pipe would always have competing long lived TCP traffic.

———

(*) There are multiple web clients and servers, as explained in Section IV.B. On the other hand, there is one FTP connection that operates in the same direction as the web traffic. We improve the text in Section V.C to better explain this point.

———

- The binding of multihop congestion with RTT is poor; it is difficult to untangle the effects of RTT and multiple congested links.

———

(*) We completely agree with the reviewer that it is difficult to untangle the effects of RTT and multiple congestion links. Our claim is that a long-lived TCP-LP flow is able to preserve non-intrusiveness to the HTTP/TCP flows, despite the fact that the *average* control-loop time (the *average* RTT) increases. We have fixed the above problem by making appropriate changes in Section V.D.1.

———

- The RTT of the experimental testbed is not stated; this would help to understand the results.

———

(*) The minimum RTT in the test-bed experiments was 2 ms. We add this information in the text in Section VI.B.2.

———

My only problem with the performance evaluation is the claim that HTTP response times are significantly affected by TCP-LP. This will depend more on weather there is competing TCP traffic than whether TCP-LP is used. In a web traffic mix it is just as likely that there will be competing long-lived TCP flows. I admit its better but the claim seems stronger than it should be.

———

(*) The reviewer correctly points out that the HTTP response times will be affected if the long-lived background flows use TCP rather than TCP-LP. However, our statement is clearly limited to a scenario explained in the paper, where *all* long-lived background flows use TCP-LP. We believe that this is not an unrealistic scenario. For example, an end-point user with a clear end-point bottleneck (e.g., behind a dial-up or a DSL line) is able to completely control which application uses a particular TCP stack, and eventually use TCP-LP for long-lived background transfers.

———

Several little presentational points would improve the paper:

- The flow of paragraphs 1,2,3 in section III-B.2 is a little confusing. The first claims that reacting similarly to TCP-ECN is too slow. The second seems to suggest that it will react to each packet (clearly too fast). The third then seems to suggest that it will in fact react slower that TCP-ECN. Perhaps leading the reader more clearly would be nice.

———

(*) The source of the confusion seems to be the fact that we didn't clearly separate the questions (1) *how* and (2) *when* TCP-LP reacts to congestion. We have improved the presentation of this Section to address better the above issue. The first paragraph explains the problem, the second explains *how* TCP-LP reacts to congestion, and the third explains *when* does it react.

———

- pg5 seems to use Q \delta and \delta Q , its nice to have the same order for the same term.

———

(*) We have fixed the above problem.

———

- all the graphs are small (maybe they have to be), however fig 14, 17, 20 are just too cluttered. The X and + marks are fairly indistinguishable which makes them visually meaningless for driving any kind of point home. Perhaps splitting them into separate (larger) graphs, or even losing them and having error bars would be better.

———

(*) We have slightly increased the size of the figures. On the other hand, we were unable to split the above figures into separate figures as we would then violate the paper length constraints.

———

- web address in second paragraph spills into next column on page 11.

———

(*) We have fixed the above problem.

———

3 Review C

This paper proposed a protocol to utilize the unused bandwidth left by regular TCP traffic, while not intrusive to regular TCP traffic. This protocol is an interesting and useful protocol. For example, it can be used to background transport files.

The key idea is how to estimate the available bandwidth, or more exactly, how to determine whether there is available bandwidth or not. There are already some proposed methods to estimate the available bandwidth, which usually uses non-TCP traffic, and take at least multiple RTTs to get the result. Those bandwidth estimation techniques usually use the increasing pattern of delay as an indication that no available bandwidth is left. This paper propose to use the threshold to determine whether bandwidth is available or not, while using only TCP traffic.

Another important feature of this protocol is that it measures only the one-way delay instead of round-trip delay, since RTT is sensitive to the reverse traffic. To get the one-way delay, time stamp option is required, but it is not a big problem.

The authors give detailed analysis, simulation, and experiment result for this protocol, which prove that this is a good protocol.

```
Overall, I recommend that this paper is accepted to be
published. Even though I think there is a problem with this
protocol. In a multi-hop network, using a threshold between the
min and max delays to measure the available bandwidth may not be
a good way. For example, the max delay could be very large at
some time, and hence the average delay is below the threshold in
most cases. I did observe this happened in some of my
simulations.
```

———

(*) The reviewer correctly points out to a problem that may arise in multi-hop scenarios when the upper one-way delay bound may be inaccurately estimated due to cross-traffic on multiple hops. There are two issues with respect to the above problem. First, in such scenarios TCP-LP still behaves less aggressive than TCP, due to fact that TCP-LP applies a more responsive congestion control. Second, in our Linux implementation of TCP-LP, we reset the parameters d_{max} and d_{min} each three minutes (as explained in Section VI.A), because of a possible drift in clocks between sender and receiver. However, these resets may actually help TCP-LP to overcome the above problem in multi-hop network scenarios.

———