Multipath TCP
INTERNET-DRAFT
Intended Status: Standard Track
Expires: August 18, 2014

Ramin Khalili
T-Labs/TU-Berlin
Nicolas Gast
Miroslav Popovic
Jean-Yves Le Boudec
EPFL-LCA2
February 14, 2014

Performance Issues with MPTCP
draft-khalili-mptcp-performance-issues-05

Abstract

We show, by measurements over a testbed and by mathematical analysis, that the current MPTCP suffers from two problems: (P1) Upgrading some TCP users to MPTCP can reduce the throughput of others without any benefit to the upgraded users; and (P2) MPTCP users can be excessively aggressive towards TCP users. We attribute these problems to the "Linked Increases" Algorithm (LIA) of MPTCP [4], and more specifically, to an excessive amount of traffic transmitted over congested paths. Our results show that these problems are important and can be mitigated. We believe that the design of the congestion control of MPTCP should be improved.

Status of this Memo

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

Table of Contents

1 Introduction

Regular TCP uses a window-based congestion-control mechanism to adjust the transmission rate of users [2]. It always provides a Pareto-optimal allocation of resources: it is impossible to increase the throughput of one user without decreasing the throughput of another or without increasing the congestion cost [5]. It also guarantees a fair allocation of bandwidth among the users but favors the connections with lower RTTs [6].

Various mechanisms have been used to build a multipath transport protocol compatible with the regular TCP. Inspired by utility maximization frameworks, [7, 8] propose a family of algorithms. These algorithms tend to use only the best paths available to users and are optimal in static settings where paths have similar RTTs. In practice, however, they suffer from several problems [9]. First, they might fail to quickly detect free capacity as they do not probe paths with high loss probabilities sufficiently. Second, they exhibit flappiness: When there are multiple good paths available to a user, the user will randomly flip its traffic between these paths. This is not desirable, specifically, when the achieved rate depends on RTTs, as with regular TCP.

Because of the issues just mentioned, the congestion control part of MPTCP does not follow the algorithms in [7, 8]. Instead, it follows an ad-hoc design based on the following goals [4]. (1) Improve throughput: a multipath TCP user should perform at least as well as a TCP user that uses the best path available to it. (2) Do no harm: a multipath TCP user should never take up more capacity from any of its paths than a regular TCP user. And (3) balance congestion: a multipath TCP algorithm should balance congestion in the network, subject to meeting the first two goals.

MPTCP compensates for different RTTs and solves many problems of multipath transport [10, 11]: It can effectively use the available bandwidth, it improves throughput and fairness compared to independent regular TCP flows in many scenarios, and it solves the flappiness problem.

We show, however, by measurements over our testbed and mathematical analysis, that MPTCP still suffers from the following problems:
(P1) Upgrading some regular TCP users to MPTCP can reduce the throughput of other users without any benefit to the upgraded users. This is a symptom of non-Pareto optimality.
(P2) MPTCP users can be excessively aggressive towards TCP users.
We attribute these problems to the "Linked Increases" Algorithm (LIA) of MPTCP [4] and specially to an excessive amount of traffic transmitted over congested paths.

These problems indicate that LIA fails to fully satisfy its design goals, especially goal number 3. The design of LIA forces a trade off between optimal resource pooling and responsiveness, it cannot provide both at the same time. Hence, to provide good responsiveness, LIA's current implementation must depart from Pareto-optimality, which leads to problems (P1) and (P2).

This document provides a number of examples and scenarios (Sections 4 to 6) in which MPTCP with LIA exhibits problems (P1) and (P2). Our results show that the identified problems with LIA are important. Moreover, we show in Section 7 that these problems are not due to the nature of a window-based multipath protocol, but rather to the design of LIA; it is possible to build an alternative to LIA that mitigates these problems and is as responsive and non-flappy as LIA. Hence, we believe that the design of the congestion control of MPTCP should be improved.

1.1 Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [1].

1.2 Terminology

Regular TCP: The standard version of TCP that operates between a single pair of IP addresses and ports [2].

Multipath TCP: A modified version of the regular TCP that allows a user to spread its traffic across multiple paths.

MPTCP: The proposal for multipath TCP specified in [3].

LIA: The "Linked Increases" Algorithm of MPTCP (the congestion control of MPTCP) [4].

OLIA: The Opportunistic "Linked Increases" Algorithm for MPTCP proposed in [12].

RTT: The Round-Trip Time seen by a user.

MSS: The Maximum Segment Size that specifies the largest amount of data can be transmitted by a TCP packet.

AP: Access Point.

2 MPTCP's LIA

The design of the congestion control algorithm of MPTCP has been described in details by Mark Handly et al. at the 77th IETF meeting in Anaheim [13].

The actual implementation of the congestion control algorithm, called "Linked Increases" Algorithm (LIA), increases the window size w_r by a quantity proportional to w_r /w_tot for each ACK received on path r (see [13, slide 3]). w_tot is the sum of the congestion windows over all paths available to the user.

LIA is one of a family of multipath congestion control algorithms that can be indexed by a parameter 0<phi<2 (see [13, slide 9]). These algorithms results in transmitting a traffic proportional to (1/p_r)^{1/phi} a path that has a probability loss p_r:

* If phi is very close to zero, then only path with smallest loss rate will be used (e.g. if p2>p1then w2=0 and w1>0). It correspond to the fully coupled algorithm of [7] and is flappy [9].
* phi=2 corresponds to having uncoupled TCP flows on each of the path. It is very responsive and non-flappy, but does not balance congestion in the network.
* phi=1 correspond to LIA and is described in RFC6356 [4]. It provides a compromise between congestion balancing and responsiveness.

3 Testbed Setup

To investigate the behavior of MPTCP in practice, three testbed topologies are created representing scenarios in Sections 4, 5, and 6. We use server-client PCs that run MPTCP enabled Linux kernels. We use MPTCP for the Linux kernel 3.0.0 released in February 2012. In all our scenarios, laptop PCs are used as routers. We install "Click Modular Router" software [14] on the routers to implement topologies with different characteristics. Iperf is used to create multiple connections.

In our scenarios, we are able to implement links with configurable bandwidth and delay. We are also able to set the parameters of the RED queues following the structure in [15]. For a 10 Mbps link, we set the packet loss probability equal to 0 up to a queue size of 25 Maximum Segment Size (MSS). Then it grows linearly to the value 0.1 at 50 MSS. It again increases linearly up to 1 for 100 MSS. The parameters are proportionally adapted when the link capacity changes.

To verify that the problems observed are caused by the congestion-control algorithm of MPTCP and not by some unknown problems in our testbed, we perform a mathematical analysis of MPTCP. This analysis is based on the fix point analysis of MPTCP. As we will see, our mathematical results confirm our measurement results. The details of these mathematical analyses are available in [12].

4 Scenario A: MPTCP can penalize regular TCP users

Consider a network with two types of users. There are N1 users of type1, each with a high-speed private connection. These users access different files on a media streaming server. The server has a network connection with capacity limit of N1C1 Mbps. Type1 users can activate a second connection through a shared AP by using MPTCP. There are also N2 users of type2; they are connected to the shared AP. They download their contents from the Internet. The shared AP has a capacity of N2C2 Mbps.

We implement this scenario in a testbed similarly to Figure 1. Within router-PCs R1 and R2, we implement links with capacities N1C1 and N2C2 and RTTs of 150 ms (including queuing delay), modeling the bottleneck at the server side and the shared AP, respectively. High speed connections are used to implement private connections of type1 users.

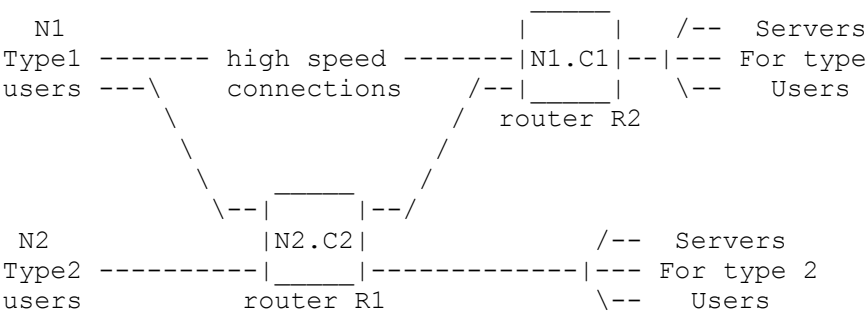

Figure 1: Testbed implementation of Scenario A: router R1 implements the bottleneck at the server side and router R2 implements the shared AP bottleneck.

When type1 users use only their private connection, each type1 user receives a rate of C1 and each type2 user receives a rate of C2. By upgrading type1 users to MPTCP, we observe, through measurement and by mathematical analysis, that the throughput of type2 users significantly decreases. However, type1 users receive the same throughput as before, because the bottleneck for their connections is at the server side. We report the throughput received by users after upgrading type1 users to MPTCP on Table 1 for C1=C2=1 Mbps. For each case, we take 5 measurements. In each case, the confidence intervals are very small (less than 0.01Mbps).

In [12 Section3.2], we provide a mathematical analysis of MPTCP that confirm our measurements: The predicted rate of type1 users is always 1 and the predicted rate for type2 users is, respectively, 0.74 when N1=N2=10 and 0.50 when N1=30 and N2=10.

		Type1 users are single path (measurement)	Type1 users are multipath: MPTCP (meas.)	Type1 users are multipath: optimal algorithm with p. cost (theory)	Type1 users are multipath: optimal algorithm w/out p. cost (theory)
N1=10	type1	0.98	0.96	1	1
N2=10	type2	0.98	0.70	0.94	1
N1=30	type1	0.98	0.98	1	1
N2=10	type2	0.98	0.44	0.8	1

meas.=measurements, p.=probing, w/out=without, Values are in Mbps.

Table 1: Throughput obtained by type1 and type2 users in Scenario A: upgrading type1 users to MPTCP decrease the throughput of type2 with no benefit for type1 users. The problem is much less critical using optimal algorithm with probing cost.

We observe that MPTCP exhibits problem (P1): upgrading type1 users to MPTCP penalizes type2 users without any gain for type1 users. As the number of type1 users increases, the throughput of type2 users decreases, but the throughput of type1 users does not change as it is limited by the capacity of the streaming server. For N1=N2, type2 users see a decrease of 30%. When N1=3N2, this decrease is 55%.

We compare the performance of MPTCP with two theoretical baselines. They serve as references to see how far from the optimum MPTCP with LIA is. We show in Section 7 that it is possible to replace LIA by an alternative that keeps the same non-flappiness and responsiveness and performs closer to the optimum.

The first baseline is an algorithm that provides theoretical optimal resource pooling in the network (as discussed in [7] and several other theoretical papers). We refer to it as "optimal algorithm without probing cost".

In practice, however, the value of the congestion windows are bounded below by 1 MSS. Hence, with a window-based congestion-control algorithm, a minimum probing traffic of 1 MSS per RTT will be sent over a path. We introduce a second theoretical baseline, called "optimal algorithm with probing cost"; it provides optimal resource pooling in the network given that a minimum probing traffic of 1 MSS per RTT is sent over each path.

We show the performance of these optimal algorithms in Table 1. Using an optimal algorithm with probing cost, the entire capacity of the shared AP is allocated to type2 users. Hence, all the users in the network receives a throughput of 1 Mbps. By using an optimal algorithm with probing cost, type1 users will send only 1MSS per RTT over the shared AP. Hence, we observe a decrease on the throughput of type2 users. However, the decrease is much less than what we observe using MPTCP. The performance of our proposed alternative to LIA is shown in Section 7, Table 4.

This performance problem of MPTCP can be explained by the fact that LIA does not fully balance congestion. For N1=N2, we observe through measurements that p1=0.009 and p2=0.02 (the probability of losses at routers R1 and R2). For N1=3N2, the value of p1 remains almost the same and p2 increases to p2=0.05. LIA excessively increases congestion on the shared AP, which is not in compliance with goal 3. In [12], we propose an alternative to LIA. Using this algorithm, we have p1=0.009 and p2=0.012 for N1=10 and 0.018 for N1=30. Hence, it is possible to provide a better congestion balancing in the network.

Because of some practical issues, we did not study larger number of connections in our testbed. However, we have mathematical results (using LIA's loss-throughput formula [12]) as well as simulation results (using a flow-level simulator) that confirm our observation for larger number of connections.

5 Scenario B: MPTCP can penalize other MPTCP users

Consider a multi-homing scenario as follows. We have four Internet Service Providers (ISPs), X, Y, Z, and T. Y is a local ISP in a small city, which connects to the Internet through Z. X, Z, and T are nation-wide service providers and are connected to each other through high speed links. X provides Internet services to users in the city and is a competitor of Y. They have access capacity limits of CX, CY, CZ, and CT. Z and T are hosts of different video streaming servers.

There are two types of users: Blue users download contents from servers in Z and Red users download from servers in ISP T. To increase their reliability, Blue users use multi-homing and are connected to both ISPs X and Y. Red users can connect either only to Y or to both X and Y .

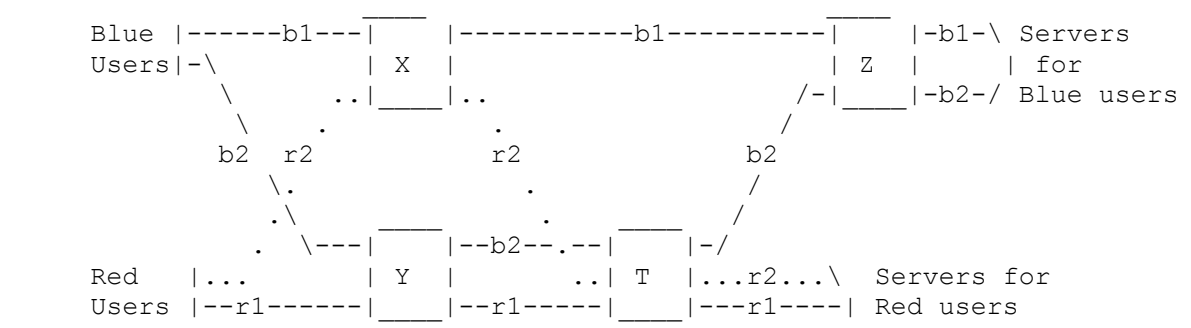

Figure 2. Testbed implementation of Scenario B. Blue users transmit over path b1 and b2. Red users use path r1, but can upgrade to MPTCP by establishing a second connection through path r2.

We implement the scenario in a testbed similar to Figure 2. b1 and b2 are the paths available to Blue users. Red users use the path r1, but can upgrade to MPTCP by establishing a second connection through path r2. The measurement results are reported in Table 2 for a setting with CX=27, CT=36, and CY=CZ=100, all in Mbps, where we have 15 Red and 15 Blue users. RTTs are around 150 ms (including queuing delay) over all paths. We also show the performance of theoretically optimal algorithms with and without probing cost.

We observe that when Red users only connect to ISP Y, the aggregate throughput of users is close to the cut-set bound, 63 Mbps. However, Blue users get a higher share of the network bandwidth. Now, let's consider that Red users upgrade to MPTCP by establishing a second connection through X (showed by pointed-line in Figure 2). Our results in Table 1 show that Red users do not receive any higher throughput. However, the average rate of Blue users drops by 20%,

which results in a drop of 13% in aggregate throughput.

	Red users are single-path			Red users are multipath		
	MPTCP (meas.)	Blue users use optimal algorithm with p. cost (theory)	Blue users use optimal algorithm w/out p. cost (theory)	MPTCP (meas.)	Blue and Red users use optimal algorithm with p. cost (theory)	Blue and Red users use optimal algorithm w/out p. cost (theory)
Red users	1.5	2.1	2.1	1.4	2.04	2.1
Blue users	2.5	2.1	2.1	2.0	2.04	2.1

meas.=measurements, p.=probing, w/out=without, Values are in Mbps.

Table 2: Throughput received by users before and after upgrading Red users to MPTCP. We have 15 Red and 15 Blue users. By upgrading Red users to MPTCP, the aggregate throughput of users decreases by 13% with no benefit for Red users.

In [12 Section 3.3], we also provide a mathematical analysis of MPTCP. Our mathematical results predict that by upgrading the Red user to MPTCP the rate of Blue users will be reduced by 21%. This results in 14% decrease in the aggregate throughput. Hence, our mathematical results confirm our observations from the measurement. Similar behavior is predicted for other values of CX and CT [12 Figure 4a].

Using an optimal algorithm without probing cost, Red users transmit only over path r1 and Blue users split their traffic over paths b1 and b2 to equalize the rate of blue and red users. Upgrading Red users to multipath does not change the allocation. Hence, we observe no decrease in the aggregate throughput and the rate of each user. By using an optimal algorithm with probing cost, the rate of Blue and Red users decreases by 3% when we upgrade the Red users to multipath users since red users are forced to send 1 MSS per RTT over path r2. This decrease is much less than what we observe using MPTCP with LIA. The performance of our proposed alternative to LIA is shown in Section 7, Table 5.

Similarly to Scenario A, the problem can be attributed to the excessive amount of traffic sent over the congested paths. This illustrates that MPTCP fails to balance the congestion in the network.

6 Scenario C: MPTCP is excessively aggressive towards TCP users

Consider a scenario with N1 multipath users, N2 single-path users, and two APs with capacities N1C1 and N2C2 Mbps. Multipath users connect to both APs and they share AP2 with single-path users. The users download their contents from the Internet. This scenario is implemented in a testbed similar to Figure 3.

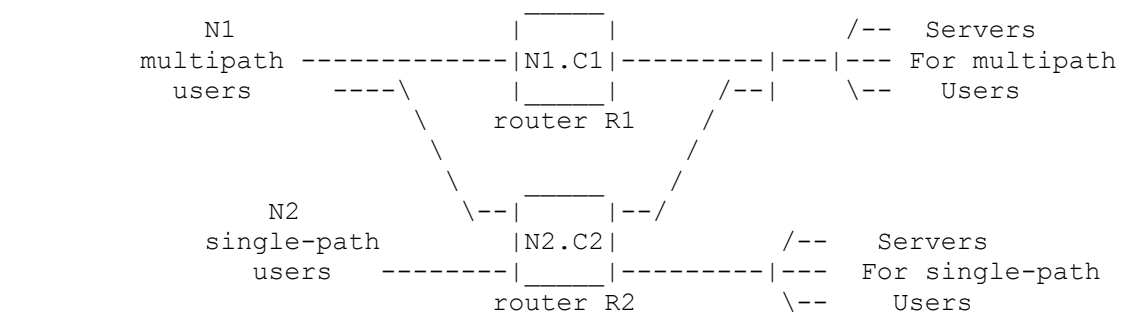

Figure 3: Testbed implementation of Scenario C: routers R1 and R2 implement AP1 and AP2 with capacities N1C1 and N2C2 Mbps.

If the allocation of rates is proportionally fair, multipath users will use AP2 only if C1<C2 and all users will receive the same throughput. When C1>C2, a fair multipath user will not transmit over AP2. However, using MPTCP with LIA, multpath users receive a disproportionately larger share of bandwidth.

We implement the scenario in our testbed and measure the performance of MPTCP with LIA. We report the throughput received by single-path and multipath users in Table 3. We present the results for N2=10 and two values of N1=10 and 30, where C1=C2=1 Mbps. RTTs are around 150 ms (including queuing delay). We also present the performance of optimal (proportionally fair) algorithms.

As C1=C2, for any fairness criterion, multipath users should not use AP2. Our results show that, MPTCP users are disproportionately aggressive and exhibit problem (P2). Hence, single-path users receive a much smaller share than they should. For N1=N2, single-path users see a decrease of about 30% in their received throughput compared to a fair allocation. When N1=3N2, this decrease is around 55%.

These measurements are confirmed by our mathematical analysis as shown in [12 Section 3.4]. The predicted rate of type1 users is 1.3 for N1=N2=10 and is 1.17 when N1=30 and N2=10. For type2 users, the predicted rate is 0.7 when N1=N2=10 and 0.48 when N1=30 and N2=10.

		multipath users use MPTCP (measurement)	multipath users use optimal algorithm with p. cost (theory)	multipath users use optimal algorithm w/out p. cost (theory)
N1=10	multipath users	1.3	1.04	1
N2=10	single-path users	0.68	0.94	1
N1=30	multipath users	1.19	1.04	1
N2=10	single-path users	0.38	0.8	1

p.=probing, w/out=without, Values are in Mbps.

Table3: Throughput obtained by single-path and multipath users in Scenario C: MPTCP is excessively aggressive toward TCP users and performs far from how an optimal algorithm would perform.

An optimal algorithm with probing cost provide a proportional fairness among the users. By using an optimal algorithm with probing cost, single-path users receive a rate less than what a proportionally fair algorithm will provide them. However, as we observe, the problem is much less critical compared to the case we use MPTCP. The performance of our proposed alternative to LIA is shown in Section 7, Table 6.

These results clearly show that MPTCP suffers from fairness issues. The problem occurs because LIA fails to fully satisfy goal 3. As in Scenarios A and B, MPTCP sends an excessive amount of traffic over the congested paths.

7 Can the suboptimality of MPTCP with LIA be fixed?

We have shown in Section 4, 5, and 6 that MPTCP with LIA performs far behind an optimal algorithm. The question is, "is it possible to modify the congestion control algorithm of MPTCP to perform closer to the optimum". To answer this question, we implement a new congestion control algorithm for MPTCP, called Opportunistic "Linked Increases" Algorithm (OLIA). OLIA is described in [13] and its performance is analyzed in [12]. In this section, we show that in Scenarios A, B and C OLIA performs close to an optimal algorithm with probing cost. Moreover, as shown in [12, Sections 4.3 and 6.2], OLIA keeps the same non-flappiness and responsiveness as LIA.

Contrary to LIA, OLIA's design is not based on a trade-off between responsiveness and optimal resource pooling. OLIA couples the additive increases and uses unmodified TCP behavior in the case of a loss. The difference between LIA and OLIA is in the increase part. OLIA's increase part two has terms:

* The first term is an adaptation of the increase term of the optimal algorithm in [7]. This term is essential to provide congestion balancing and fairness.
* The second term guarantees responsiveness and non-flappiness of OLIA. By measuring the number of transmitted bytes since the last loss, it reacts to events within the current window and adapts to changes faster than the first term.

Because OLIA is rooted in the optimal algorithm of [7], it can provide fairness and congestion balancing. Because of the second term, it is responsive and non-flappy.

We implemented OLIA by modifying the congestion control part of the MPTCP implementation based on the Linux Kernel 3.0.0. For conciseness, we do not describe OLIA in this paper and refer to [12] for details about the algorithm and its implementation.

We study the performance of MPTCP with OLIA through measurements in Scenarios A, B, and C. The results are reported in Tables 4, 5 and 6. We compare the performance of our algorithm with MPTCP with LIA and with optimal algorithms. We observe that in all cases, MPTCP with OLIA provide a significant improvement over MPTCP with LIA. Moreover, it performs close to an optimal algorithm with probing cost.

	Type1 users are single path (measurement)	Type1 users are multipath: MPTCP w. OLIA [with LIA] (measurement)	Type1 users are multipath: optimal algorithm with p. cost (theory)	Type1 users are multipath: optimal algorithm w/out p. cost (theory)
N1=10	type1 0.98	0.98 [0.96]	1	1
N2=10	type2 0.98	0.86 [0.70]	0.94	1
N1=30	type1 0.98	0.98 [0.98]	1	1
N2=10	type2 0.98	0.75 [0.44]	0.8	1

p.=probing, w.=with, w/out=without, Values are in Mbps.

Table 4. Performance of MPTCP with OLIA compared to MPTCP with LIA in scenario A. We show the throughput obtained by users before and after upgrading type1 users to MPTCP. The values in brackets are the values for MPTCP with LIA (taken from table 1).

	Red users are single-path: Blue users use MPTCP with OLIA [with LIA] (meas.)	Red users are single-path: Blue users use optimal algorithm with p. cost (theory)	Red users are single-path: Blue users use optimal algorithm w/out p. cost (theory)	Red users are multipath: Blue and Red users use MPTCP with OLIA [with LIA] (meas.)	Red users are multipath: Blue and Red users use optimal algorithm with p. cost (theory)	Red users are multipath: Blue and Red users use optimal algorithm w/out p. cost (theory)
Red users	1.8 [1.5]	2.1	2.1	1.7 [1.4]	2.04	2.1
Blue users	2.2 [2.5]	2.1	2.1	2.2 [2.0]	2.04	2.1

meas.=measurements, p.=probing, w/out=without, Values are in Mbps.

Table 5. Performance of MPTCP with OLIA compared to MPTCP with LIA in scenario B. We show the throughput received by users before and after upgrading Red users to MPTCP. The values in brackets are the values for MPTCP with LIA (taken from Table 2).

		multipath users use MPTCP with OLIA [with LIA] (measurement)	multipath users use optimal algorithm with probing cost (theory)	multipath users use algorithm w/out probing cost (theory)
N1=10	multipath users	1.11 [1.30]	1.04	1
N2=10	single-path users	0.88 [0.68]	0.94	1
N1=10	multipath users	1.1 [1.19]	1.04	1
N2=10	single-path users	0.72 [0.38]	0.8	1

meas.=measurements, w/out=without, Values are in Mbps.

Table 6. Performance of MPTCP with OLIA compared to MPTCP with LIA in scenario C. We show the throughput obtained by single-path and multipath users. The values in brackets are the values for MPTCP with LIA (taken from Table 3).

The results show that it is possible to perform close to an optimal algorithm with probing cost by using a TCP-like algorithm. Moreover, we show in [12, Section 4.3 and Section 6.2] that MPTCP with OLIA is as responsive and non-flappy as MPTCP with LIA. This shows that it is possible to build a practical multi-path congestion control that works close to an optimal algorithm with probing cost.

In [17], Chen et al. study how MPTCP with LIA and OLIA performs in the wild with a common wireless environment, namely using both WiFi and Cellular simultaneously. Their results show that MPTCP with OLIA is very responsive to the changes in the environment and always outperforms MPTCP with LIA. Furthermore, using Experimental Design, Paasch et al. [8] show that MPTCP with OLIA satisfy the design goal of MPTCP in a very wide range of scenarios and always outperform MPTCP with LIA.

8 Conclusion

We have shown that MPTCP with LIA suffers from important performance issues. Moreover, it is possible to build an alternative to LIA that performs close to an optimal algorithm with probing cost while being as responsive and non-flappy as LIA. Hence, we believe that mptcp working group should revisit the congestion control part of MPTCP and that an alternative algorithm, such as OLIA [12], should be considered.

9 References

9.1 Normative References

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

[3] Ford, A., Raiciu, C., Greenhalgh, A., and M. Handley, "Architectural Guidelines for Multipath TCP Development", RFC 6182, March 2011.

9.2 Informative References

[4] Raiciu, C., Handley, M., and D. Wischik, "Coupled Congestion Control for Multipath Transport Protocols", RFC 6356, October 2011.

[5] F.P. Kelly. Mathematical modelling of the internet. Mathematics unlimited-2001 and beyond.

[6] F.P. Kelly, A.K. Maulloo, and D.K.H. Tan. Rate control for communication networks: shadow prices, proportional fairness and stability. Journal of the Operational Research society, 49, 1998.

[7] Kelly, F. and T. Voice, "Stability of end-to-end algorithms for joint routing and rate control", ACM SIGCOMM CCR vol. 35 num. 2, pp. 5-12, 2005.

[8] H. Han, S. Shakkottai, CV Hollot, R. Srikant, and D. Towsley. "Multi-path tcp: a joint congestion control and routing scheme to exploit path diversity in the internet", Trans. on Net., 14, 2006.

[9] D. Wischik, M. Handly, and C. Raiciu. "Control of multipath tcp and optimization of multipath routing in the internet", NetCOOP, 2009.

[10] D. Wischik, C. Raiciu, A. Greenhalgh, and M. Handly. "Design, implementation and evaluation of congestion control for multipath tcp", Usenix NSDI, 2011.

[11] C. Raiciu, S. Barre, C. Pluntke, A. Greenhalgh, D. Wischik, and M. Handly. "Improving datacenter performance and robustness with multipath tcp", ACM Sigcomm, 2011.

[12] R. Khalili, N. Gast, M. Popovic, U. Upadhyay, and J.-Y. Le Boudec. "MPTCP is not Pareto-optimality: Performance

issues and a possible solution", ACM CoNEXT 2012 (The extended version of this paper is published at IEEE/ACM Transaction of Networking, volume 5, issue 25, October 2013).

[13] M. Handly, D. Wischik, C. Raiciu, "Coupled Congestion Control for MPTCP", presented at 77th IETF meeting, Anaheim, California. <http://www.ietf.org/proceedings/77/slides/mptcp-9.pdf>.

[14] B. Chen J. Jannotti E. Kohler, R. Morris and M. F. Kaashoek. "The click modular router", Trans. Comput. Syst., 18, August 2000.

[15] S. Floyd and V. Jacobson. "Random early detection gateways for congestion avoidance", Trans. on Net., 1, August, 1993.

[16] R. Khalili, N. Gast, M. Popovic, J.-Y. Le Boudec, "Opportunistic Linked-Increases Congestion Control Algorithm for MPTCP", draft-khalili-mptcp-congestion-control-03.

[17] Chen Y.-C, Y.-S. Lim, R. J. Gibbens, E. M. Nahum, R. Khalili, and D. Towsley, "A Measurement-based Study of Multipath TCP Performance over Wireless Networks." ACM IMC 2013.

[18] C. Paasch, R. Khalili, and O. Bonaventure, "On the Benefits of Applying Experimental Design to Improve Multipath TCP." ACM CoNEXT 2013.

Authors' Addresses

Ramin Khalili
T-Labs/TU-Berlin
TEL 18, Ernst-Reuter-Platz 7
10587 Berlin
Germany

Phone: +49 30 8353 58276
EMail: ramin@net.t-labs.tu-berlin.de

Nicolas Gast
EPFL IC ISC LCA2
Station 14
CH-1015 Lausanne
Switzerland

Phone: +41 21 693 1254
EMail: nicolas.gast@epfl.ch

Miroslav Popovic
EPFL IC ISC LCA2
Station 14
CH-1015 Lausanne
Switzerland

Phone: +41 21 693 6466
EMail: miroslav.popovic@epfl.ch

Jean-Yves Le Boudec
EPFL IC ISC LCA2
Station 14
CH-1015 Lausanne
Switzerland

Phone: +41 21 693 6631
EMail: jean-yves.leboudec@epfl.ch